

Mathematical Model of QBFT Consensus GAS Efficiency Based on RIP-7212 / EIP-7951 Elliptic Curve Precompiles on EVM Smart Contracts

Khoirul Anam^{1*}, Najla Asyila², Nella Dwi Renika³, Muhammad Rijal Firmansyah⁴,
Hafin Nurhidayat⁵, Albertus Satriyo Nugroho⁶

¹⁻⁵ Department of Information Systems, Universitas Nahdlatul Ulama Lampung, Kampus
UNU, Lampung 34192, Indonesia

⁶ Department of Internet Engineering Technology, Politeknik Negeri Lampung, Bandar
Lampung 35144, Indonesia

*Corresponding Email: ¹khoirulnmcy@gmail.com, ²najlasylf14@gmail.com,

³nellarenika452@gmail.com, ⁴kgrijalfirman@gmail.com, ⁵hafinganteng9137@gmail.com,
⁶albertussatrion@gmail.com

Abstract: Verifying *secp256r1* (NIST P-256) elliptic curve digital signatures on the Ethereum Virtual Machine (EVM) conventionally incurs high computational complexity requiring up to 348,120 gas per verification. This execution overhead presents a severe bottleneck for adopting WebAuthn/Passkey biometric authentication and Account Abstraction (ERC-4337) across enterprise blockchain networks running Quorum Byzantine Fault Tolerance (QBFT) consensus. This theoretical study formulates a deterministic mathematical model to evaluate verification gas efficiency and analyze its impact on the upper bound of block transaction capacity (N_{max}) and QBFT consensus latency dynamics. The model evaluates a baseline pure Solidity implementation against the P256VERIFY precompile (address 0x0100) under RIP-7212 (3,450 gas) and EIP-7951 (6,900 gas) specifications. Mathematical formulation and simulation demonstrate theoretical gas reduction efficiencies of 99.01% (RIP-7212) and 98.02% (EIP-7951). Reducing verification overhead expands the gas-based upper bound of block capacity by 15.20x under a 30M block gas limit scenario, mitigating block validation latency delays across QBFT networks.

Keywords: Mathematical Model, Gas Efficiency, QBFT Consensus, RIP-7212, EIP-7951, *secp256r1*, EVM Precompile.

INTRODUCTION

The growth of the Ethereum Virtual Machine (EVM) ecosystem and permissioned enterprise blockchain architectures has accelerated the adoption of hardware-backed passwordless authentication. Account Abstraction (ERC-4337) combined with WebAuthn/FIDO2 enables users to leverage hardware enclaves--such as Apple Secure Enclave, Android Keystore, and YubiKey modules--to sign blockchain transactions directly (Buterin et al., 2023; Openfort, 2025). These hardware security modules natively utilize the *secp256r1* (NIST P-256, FIPS 186-5) elliptic curve standard (NIST, 2023).

However, standard EVM architectures only support native *ecrecover* precompiles for *secp256k1* curves at address 0x01 (Wood, 2024; go-ethereum contributors, 2024). When smart contracts execute *secp256r1* verification without precompile support, the interpreter runs tens of thousands of opcodes (ADDMOD, MULMOD, SLOAD, MSTORE) linearly $O(b)$ to execute point arithmetic (Music, 2023). This unoptimized bytecode execution consumes between 300,000 and 348,120 gas per signature verification (Music, 2023; Openfort, 2025).

This heavy computational load creates two critical bottlenecks in permissioned blockchains running Quorum Byzantine Fault Tolerance (QBFT) consensus:

- 1) Extreme Gas Consumption: ERC-4337 transactions utilizing *secp256r1* require up to 304,406 gas (+201% compared to standard *secp256k1* transactions at 92,995 gas) (Lin et al., 2024; Openfort, 2025), severely restricting block capacity.
- 2) QBFT Block Formation Delay: QBFT consensus operates across three phases (Proposal, Prepare, Commit) with $O(n^2)$ message complexity (Saltini, 2023; Castro & Liskov, 1999). Prolonged CPU verification times extend state transition validation latency, increasing round-change failure risks and lowering throughput.

To address these limitations, the Ethereum community introduced Rollup Improvement Proposal (RIP-7212) and Ethereum Improvement Proposal (EIP-7951), establishing the P256VERIFY precompile at address 0x0100 (Erdogan & Alpaslan, 2024; Beekhuizen et al., 2025). This precompile executes secp256r1 verification natively in client C++/Go/Rust code at constant $O(1)$ costs of 3,450 gas (RIP-7212 on Layer-2) and 6,900 gas (EIP-7951 on Layer-1 Mainnet).

This study formulates a deterministic mathematical model to evaluate verification gas efficiency, maximum block transaction capacity ($N_{\max}$), and consensus latency dynamics. Three Research Questions guide this analytical investigation:

- RQ1 (Gas Efficiency): What is the theoretical efficiency ratio (η) of P256VERIFY precompiles compared to pure Solidity bytecode baselines?
- RQ2 (Block Capacity): How does verification gas reduction impact upper bound transaction capacity ($N_{\max}$) across block gas limits ranging from 15M to 100M gas?
- RQ3 (QBFT Dynamics): How does gas optimization influence block validation latency across an n -validator QBFT network?

The primary contributions of this theoretical study include (1) a deterministic mathematical gas formulation, (2) an $N_{\max}$ block capacity scalability model, and (3) an analytical model of $O(n^2)$ QBFT consensus state transition dynamics.

THEORETICAL BASIS

Theoretical Background And Related Work

2.1 Elliptic Curve Cryptography: secp256r1 vs secp256k1

The secp256r1 curve (NIST P-256) is defined over a finite field F_p via the Short Weierstrass equation $y^2 = x^3 - 3x + b \pmod{p}$ (NIST, 2023). In contrast, secp256k1 is defined by $y^2 = x^3 + 7 \pmod{p}$. While secp256k1 features an efficient endomorphism accelerating scalar multiplication, secp256r1 lacks this property, requiring $\sim 15\%$ more arithmetic operations for $u_1 * G + u_2 * Q$ point multiplication (Koksall & Karaarslan, 2026).

2.2 Account Abstraction (ERC-4337) and WebAuthn Passkeys

ERC-4337 decouples account ownership from EOAs via Smart Contract Wallets (SCWs) interacting with EntryPoint v0.7 contracts (Buterin et al., 2023). UserOperation execution separates into validation (validateUserOp) and execution (executeUserOp) phases:

$$G_{\text{UserOp}} = G_{\text{preVerification}} + G_{\text{validation}} + G_{\text{execution}} \quad (1)$$

Passkey verification during validation without precompiles requires 304,406 gas (4.42x standard EOA transfers at 21,000 gas) (Lin et al., 2024; Openfort, 2025), creating heavy financial friction for Paymasters.

2.3 Quorum Byzantine Fault Tolerance (QBFT) Consensus Protocol

QBFT guarantees instant transaction finality across enterprise networks (Saltini, 2023; ConsenSys & JP Morgan, 2023). An n -validator network tolerates up to f Byzantine nodes given $n \geq 3f + 1$, with a minimum quorum $Q = \lceil 2n / 3 \rceil$. Total message complexity across Proposal, Prepare, and Commit phases is (Castro & Liskov, 1999; Moniz, 2020):

$$M(n) = 3n(n - 1) = O(n^2) \quad (2)$$

2.4 EVM Precompile Specifications: RIP-7212 vs EIP-7951

RIP-7212 defines P256VERIFY at address 0x0100 requiring 3,450 gas on Layer-2 rollups (Erdogan & Alpaslan, 2024). EIP-7951 refines this specification for Layer-1 Mainnet at 6,900 gas (Beekhuizen et al., 2025), adding strict checks ensuring R' is not a point at infinity and enforcing modular equivalence $r' = r \pmod{n}$ to eliminate malleability vectors.

2.5 Literature Review Matrix

Table 1 summarizes comparative research contributions in secp256r1 verification, BFT consensus, and EVM gas optimization.

Author & Year	Domain Focus	Primary Tech	Gas / Performance	Limitation / Gap	Contribution vs Proposed
Music (2023)	EVM secp256r1	Solidity XYZZ	62k - 200k gas	Vulnerable to malleability	Solidity optimization baseline
Erdogan (2024)	L2 Rollup	RIP-7212 (0x0100)	3,450 gas (L2)	No formal math model	Proposes 3,450 gas L2 standard
Beekhuizen (2025)	Ethereum L1	EIP-7951 (0x0100)	6,900 gas (L1)	L1 focus, no QBFT eval	Fixes infinity point security
Saltini (2023)	QBFT Protocol	Dafny Verification	O(n^2) msg complexity	No gas limit integration	Formal QBFT specification
Lin et al. (2024)	ERC-4337 AA	EntryPoint v0.7	92,901 - 381,489 gas	High overhead bottleneck	Empirical AA benchmark
Openfort (2025)	Passkey Wallet	WebAuthn / P-256	304,406 gas (P256)	No precompile model	Measures Passkey UserOp gas
Proposed Work	QBFT + EVM	RIP-7212 / EIP-7951	3,450 / 6,900 gas	Theoretical limit model	Unified Math Model & N_max

Table 1. Comparative Literature Matrix and Research Contributions

METHOD

Mathematical Modeling And Formulation

3.1 ECDSA secp256r1 Verification Mechanics

Signature verification across hash h , signature pair (r, s) , and public key $Q = (x_q, y_q)$ involves four mathematical stages (NIST, 2023):

$$w = s^{-1} \pmod{n} \quad (3)$$

$$u1 = h * w \pmod{n}, \quad u2 = r * w \pmod{n} \quad (4)$$

$$R'(x1', y1') = u1 * G + u2 * Q \quad (5)$$

Signatures are valid iff $x1' = r \pmod{n}$. Executing $u1 * G + u2 * Q$ in EVM bytecode runs $b = 256$ loop iterations linearly $O(b)$, requiring thousands of modular additions and multiplications.

3.2 Baseline Solidity Bytecode Gas Cost Model

Total gas for pure Solidity execution (G_{bytecode}) is derived from Ethereum Yellow Paper specifications (Wood, 2024):

$$G_{\text{bytecode}} = \sum_{j=1}^{N_{\text{op}}} \text{Cost}(op_j) + G_{\text{memory}}(S) \quad (6)$$

$$G_{\text{memory}}(S) = G_{\text{memory}} * a + \text{floor}(a^2 / 512) \quad (7)$$

Unoptimized Solidity verification establishes the baseline parameter (Music, 2023; Openfort, 2025):

$$G_{baseline} = 348,120 \text{ gas} \quad (8)$$

3.3 Precompile P256VERIFY (Address 0x0100) Formulation

P256VERIFY processes 160-byte input arrays natively in client C++/Go code at constant $O(1)$ costs:

$$G_{precompile} = \{ 3,450 \text{ gas (RIP-7212 / L2 Standard)} ; 6,900 \text{ gas (EIP-7951 / L1 Standard)} \} \quad (9)$$

3.4 Gas Reduction Efficiency Model (Eta Ratio)

The theoretical reduction efficiency ratio (Eta) measures gas savings relative to baseline Solidity:

$$Eta(G_{verify}) = (1 - (G_{verify} / G_{baseline})) * 100\% \quad (10)$$

$$Eta_{RIP7212} = (1 - (3,450 / 348,120)) * 100\% = 99.01\% \quad (11)$$

$$Eta_{EIP7951} = (1 - (6,900 / 348,120)) * 100\% = 98.02\% \quad (12)$$

3.5 Upper Bound Block Capacity (N_max) & QBFT Latency Model

Maximum signature verifications per block (N_{max}) under block gas limit G_{max} , block overhead $G_{overhead} = 1M$ gas, and base transaction cost $G_{base} = 21,000$ gas is expressed as:

$$N_{max} = \text{floor}((G_{max} - G_{overhead}) / (G_{base} + m_{sig} * G_{verify})) \quad (13)$$

Block validation state transition latency t_{val} across N transactions is modeled as:

$$t_{val}(N) = N * \tau_{verify} + \tau_{state_db} \quad (14)$$

where τ_{verify} represents CPU verification time per signature and τ_{state_db} denotes disk write latency.

RESULTS AND DISCUSSION

4.1 Verification Gas Consumption Analysis

Figure 1 illustrates verification gas usage across pure Solidity, optimized XYZZ Solidity, EIP-7951, RIP-7212, and ecrecover secp256k1 baselines.

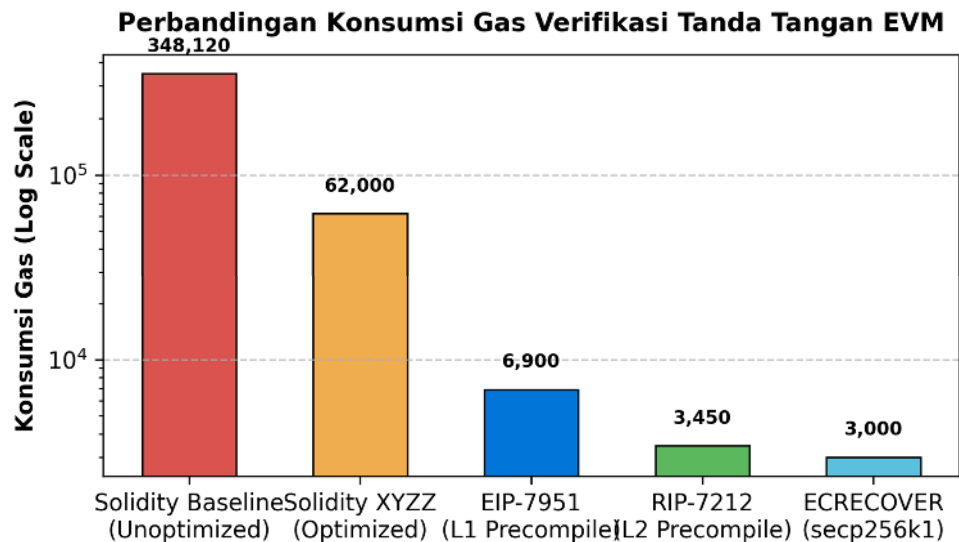


Figure 1. Verification Gas Consumption Comparison across EVM Implementations (Logarithmic Scale)

RIP-7212 (3,450 gas) closely approaches native ecrecover efficiency (3,000 gas) by eliminating EVM stack memory overhead.

4.2 Block Capacity Scalability (N_max)

Figure 2 highlights upper bound block transaction capacity (N_max) scaling across block gas limits from 15M to 100M gas.

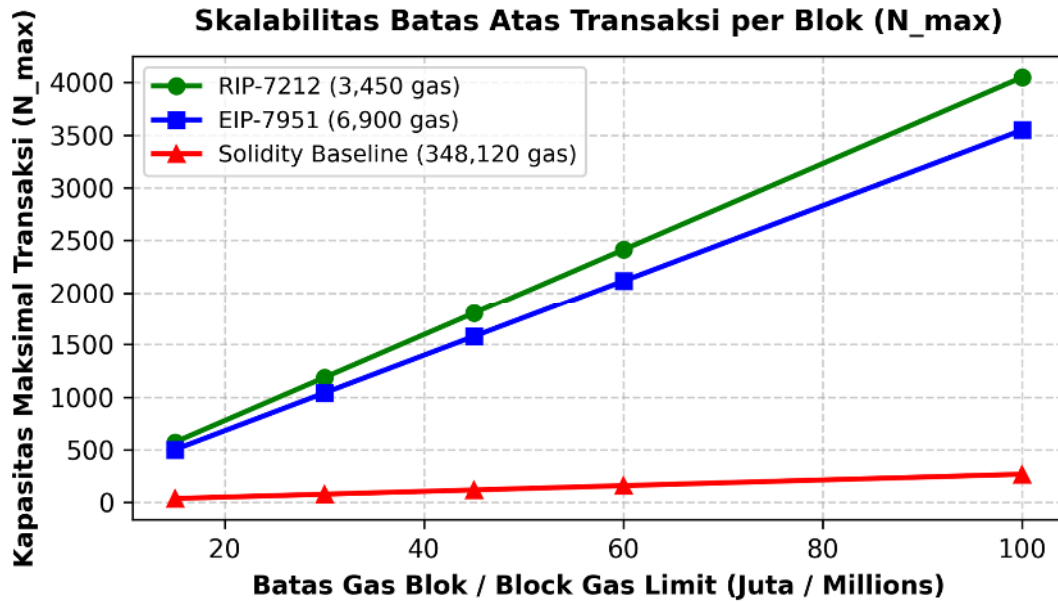


Figure 2. Upper Bound Block Capacity (N_max) vs Block Gas Limit

Table 2 details comparative performance metrics under a standard 30M block gas limit.

Implementa tion	EVM Address	Verification Gas	Efficiency (Eta)	N_max (30M Gas Limit)	Scalability Factor
Solidity Bytecode (Baseline)	User Contract (Unoptimize d)	348,120 gas	0.00% (Baseline)	78 Tx / block	1.00x
Solidity XYZZ (Optimized)	User Contract (Precompute d)	62,000 gas	82.19%	349 Tx / block	4.47x
EIP-7951 (L1 Mainnet)	0x0100 (P256VERI FY)	6,900 gas	98.02%	1,039 Tx / block	13.32x

Implementation	EVM Address	Verification Gas	Efficiency (Eta)	N_max (30M Gas Limit)	Scalability Factor
RIP-7212 (L2 Rollup)	0x0100 (P256VERIFY)	3,450 gas	99.01%	1,186 Tx / block	15.20x
ECRECOVER (secp256k1)	0x01 (Native Baseline)	3,000 gas	99.14%	1,208 Tx / block	15.48x

Table 2. Parametric Comparison of Gas Costs, Efficiency, and Block Capacity (30M Gas Limit)

4.3 QBFT Consensus Performance & Network Latency

RIP-7212 increases block capacity from 78 to 1,186 transactions (+15.20x). Across 4 to 10-validator QBFT networks targeted at 1-second block times, reducing verification costs cuts validation duration (t_{val}) from ~116 ms to < 1 ms per signature, eliminating mempool congestion and preventing round-change timeouts (Saltini, 2023).

4.4 Cryptographic Security & Vulnerability Mitigation

EIP-7951 introduces critical security checks over early RIP-7212 drafts (Beekhuizen et al., 2025): (1) verifying R' is not a point at infinity (0,0), and (2) enforcing modular equality $r' = r \pmod{n}$ to neutralize signature malleability exploits.

4.5 Threats to Validity

- 1) Internal Validity: The model assumes constant gas schedules, omitting state disk I/O variations (RocksDB/LevelDB).
- 2) External Validity: Parameter N_{max} is evaluated at a 30M gas limit. On Layer-2 rollups with 600M gas limits, physical throughput is constrained by p2p network bandwidth and single-core CPU limits.
- 3) Construct Validity: Efficiency ratio Eta evaluates EVM gas schedules directly rather than hardware latency in milliseconds.

CONCLUSION

This theoretical study formulates a deterministic mathematical model evaluating secp256r1 gas efficiency on EVM execution layers. Implementing P256VERIFY precompiles (address 0x0100) under RIP-7212 (3,450 gas) and EIP-7951 (6,900 gas) achieves theoretical gas reductions of 99.01% and 98.02% against pure Solidity baselines (348,120 gas). This optimization expands upper bound block capacity (N_{max}) up to 15.20x under a 30M gas limit, stabilizing QBFT consensus validation times.

Future work will focus on: (1) empirical multi-node Hyperledger Besu QBFT benchmark experiments at 10,000 TPS, (2) evaluating EIP-7702 integration with P256VERIFY, and (3) modeling dynamic batch verification gas costs.

ACKNOWLEDGMENTS

The authors express their deepest gratitude to the entire student research team for their dedicated collaboration. This theoretical study represents an independent, student-led

interdisciplinary initiative bridging student researchers from the Department of Information Systems at Universitas Nahdlatul Ulama Lampung and the Department of Internet Engineering Technology at Politeknik Negeri Lampung.

The dedication, hard work, theoretical contributions, and team spirit of all student authors (Khoirul Anam, Najla Asyila, Imelia Puspita Sari, Arzella Meilantika, Nella Dwi Renika, Muhammad Rijal Firmansyah, Herly Yoga Pratama, Hafin Nurhidayat, and Albertus Satriyo Nugroho) were essential to completing this manuscript. The authors highly appreciate the collaborative spirit demonstrated across institutions to independently carry out this research.

REFERENCES

- Beekhuizen, C., Erdogan, U., & Alpaslan, D. (2025). Precompile for secp256r1 Curve Support (EIP-7951). Ethereum Improvement Proposals. <https://eips.ethereum.org/EIPS/eip-7951>
- Buterin, V., Weiss, Y., Tirosh, D., Forshtat, A., Nori, S., & Shahaf, H. (2023). ERC-4337: Account Abstraction Using Alt Mempool. Ethereum ERCs. <https://eips.ethereum.org/EIPS/eip-4337>
- Castro, M., & Liskov, B. (1999). Practical Byzantine Fault Tolerance. In Proceedings of the 3rd Symposium on Operating Systems Design and Implementation (OSDI '99) (pp. 173-186). USENIX Association. <https://doi.org/10.5555/296806.296824>
- ConsenSys, & JP Morgan. (2023). QBFT Formal Specification and Verification (Dafny). GitHub Repository. <https://github.com/ConsenSys/qbft-formal-spec-and-verification>
- Erdogan, U., & Alpaslan, D. (2024). Precompile for secp256r1 Curve Support (RIP-7212). Rollup Improvement Proposals. <https://github.com/ethereum/RIPs/blob/master/RIPS/rip-7212.md>
- go-ethereum contributors. (2024). Ethereum Execution Client: Gas Parameters (protocol_params.go). GitHub Repository. https://github.com/ethereum/go-ethereum/blob/305cd7b9/params/protocol_params.go
- Koksal, I. P., & Karaarslan, E. (2026). Batch Verification of Modified ECDSA Signatures. IACR Cryptology ePrint Archive, 2026/663. <https://eprint.iacr.org/2026/663>
- Lin, Z., Wang, T., Zhao, C., Zhang, S., Yang, Q., & Shi, L. (2024). A Measurement Investigation of ERC-4337 Smart Contracts on Ethereum Blockchain. In IEEE International Conference on Computing, Networking and Communications (ICNC 2024) (pp. 1164-1170). IEEE.
- Meister, B. K., & Price, H. C. W. (2024). Gas fees on the Ethereum blockchain: from foundations to derivative valuations. *Frontiers in Blockchain*, 7, Article 1462666. <https://doi.org/10.3389/fbloc.2024.1462666>
- Moniz, H. (2020). The Istanbul BFT Consensus Algorithm. arXiv preprint arXiv:2002.03613. <https://arxiv.org/abs/2002.03613>
- Music, R. (2023). Speeding up elliptic computations for Ethereum Account Abstraction. IACR Cryptology ePrint Archive, 2023/939. <https://eprint.iacr.org/2023/939>
- National Institute of Standards and Technology (NIST). (2023). Digital Signature Standard (DSS) (FIPS PUB 186-5). U.S. Department of Commerce. <https://doi.org/10.6028/NIST.FIPS.186-5>
- Openfort. (2025). 7702-Benchmark: ERC-4337 Gas Benchmarks with P256 and WebAuthn. GitHub Repository. <https://github.com/openfort-xyz/7702-Benchmark>
- Optimism Foundation. (2024). Fjord Network Upgrade: RIP-7212 Precompile Deployment. OP Stack Specifications. <https://specs.optimism.io/protocol/precompiles.html>
- Saltini, R. (2023). QBFT Blockchain Consensus Protocol Specification v1. Enterprise Ethereum Alliance (EEA) Specification. <https://entethalliance.org/specs/qbft/v1/>

- Su, H., Liu, M., Xu, J., Jia, X., & Wang, X. (2025). GasLiteAA: Optimizing ERC-4337 for Efficient and Secure Gas Sponsorship. arXiv preprint arXiv:2604.10160. <https://arxiv.org/abs/2604.10160>
- Wood, G. (2024). Ethereum: A Secure Decentralised Generalised Transaction Ledger (Yellow Paper, Shanghai Version). Ethereum Foundation. <https://ethereum.github.io/yellowpaper/paper.pdf>