A COMPARATIVE BENCHMARK OF MODULAR SQUARE-ROOT ALGORITHMS FOR COMPRESSED-POINT ELLIPTIC CURVE DECOMPRESSION IN SECURE DEVICE-TO-DEVICE IOT COMMUNICATION FOR SMART CITIES

Haruna, J.[1], Nasiru, S.[1*], Isah, M.M.[2], Garba, A.A.[3], Chinedu, P.U.[4] and AbdulQuadri, L.M.[5]

[1]Department of Computer Engineering, Confluence University of Science and Technology, Osara, Kogi State, Nigeria

[2]Department of Computer Engineering, Edo State University Iyambo, Edo State, Nigeria

[3]Department of Computer Engineering, University of Maiduguri, Borno State Nigeria

[4]Department of Information Systems and Technology, Southern Delta University, Ozoro, Delta State

[5]Department of Computer Science, Lagos State Cooperative College, Oko-oba, Lagos

*Corresponding email: salamatnasir1@gmail.com

ABSTRACT

Device-to-device (D2D) communication underpins many smart-city Internet-of-Things (IoT) services, yet the constrained energy, bandwidth, and storage budgets of end devices make the overhead of conventional Elliptic Curve Cryptography (ECC) prohibitive. Point compression mitigates this by transmitting only the x-coordinate of an elliptic-curve point together with a single sign bit, halving the payload; the receiver must then reconstruct the y-coordinate by computing a modular square root over the curve prime p, an operation whose cost is rarely quantified for constrained platforms. This paper presents a systematic benchmark of three candidate square-root algorithms — Tonelli–Shanks, the $p \equiv 3 \pmod 4$ exponentiation short-cut, and Cipolla's method — over six RFC 5639 brainpool curves spanning 160–512 bits. In trials of 64 quadratic residues repeated over three rounds in a Python environment, the $p \equiv 3 \pmod 4$ short-cut is the fastest on every tested curve, delivering median speed-ups of 4.66×–12.71× over Tonelli–Shanks and 7.92×–9.08× over Cipolla, while preserving the approximately 50% bandwidth saving of compressed points. Per-decompression energy on Tmote-Sky-class hardware is estimated at 0.23–3.09 µJ, two orders of magnitude below the radio energy saved per compressed message. Integrated with the compressed secure D2D communication scheme (CSDDCS), the short-cut materially reduces the decompression-related processing overhead previously reported for the scheme. The results yield a practical selection rule for square-root algorithms in compressed-ECC IoT authentication.

Keywords: Device-to-device communication; elliptic curve cryptography; point compression; modular square root; Tonelli–Shanks; smart cities; energy efficiency

1 INTRODUCTION

The smart-city paradigm integrates large numbers of resource-constrained Internet-of-Things (IoT) devices that sense, process, and exchange data over wireless links. Device-to-device (D2D) communication allows nearby devices to exchange data directly, reducing latency, offloading the network core, and improving spectral and energy efficiency [1, 2]. However, direct D2D links expose devices to impersonation, replay, and man-in-the-middle attacks, so mutual authentication and session-key

establishment are mandatory before any payload exchange [3, 4, 5]. Elliptic Curve Cryptography (ECC) is attractive for IoT because it offers strong security per key bit, permitting smaller keys, signatures, and certificates than RSA [3]. Nonetheless, ECC authentication exchanges elliptic-curve points whose two coordinates together consume roughly twice the field size in bits, which is costly for bandwidth- and energy-limited motes. Point compression reduces a point $P = (x, y)$ to its x-coordinate plus one sign bit, saving nearly 50% of the wire format [6, 7, 8]. The price of compression is an asymmetric workload: the sender only evaluates the curve equation, whereas the receiver must recover y by computing the square root of $x^3 + ax + b$ modulo the curve prime p. The choice of square-root algorithm is therefore a first-order design decision for compressed-ECC D2D authentication. The generic Tonelli–Shanks algorithm [7, 8] works for every odd prime, but its cost grows with the 2-adicity of $p - 1$ and it requires a quadratic non-residue. Faster variants and alternatives exist [9, 10, 11, 12, 13, 14], yet no comparative study known to the authors quantifies their relative cost on the specific curves used by smart-city D2D security schemes, notably the RFC 5639 brainpool family that underpins the compressed secure D2D communication scheme (CSDDCS) [1, 15]. The existing evaluation of CSDDCS reports a small increase in authentication processing time ($\approx$2.2%) relative to the base scheme, attributed to decompression [2]; selecting a faster square-root algorithm directly attacks this trade-off. The contributions of this paper are as follows: A controlled, reproducible benchmark of Tonelli–Shanks, the $p \equiv 3 \pmod 4$ exponentiation short-cut, and Cipolla's method over six RFC 5639 brainpool curves (160 to 512 bits), reporting median and mean execution times per square root. An energy analysis that places per-decompression cost against the radio transmission energy saved by point compression on Tmote-Sky-class hardware. An integration study showing how the fastest algorithm reduces the CSDDCS authentication-time overhead while preserving its measured communication, storage, and energy savings; and A practical algorithm-selection rule for designers of compressed-ECC protocols for constrained IoT.

2 PRELIMINARIES AND RELATED WORK

2.1 Elliptic curves, point compression, and decompression

Throughout this paper, p denotes an odd prime, $\mathbb{F}_p$ the field of integers modulo p, and E_p an elliptic curve defined over $\mathbb{F}_p$ by the short Weierstrass equation.

$$E_p\colon\ y^2 \equiv x^3 + a\,x + b \pmod p, \tag{1}$$

where $a, b \in \mathbb{F}_p$ and $4a^3 + 27b^2 \not\equiv 0 \pmod p$ guarantees a non-singular curve. A point P on E_p is an ordered pair.

$$P = (x_P,\ y_P), \qquad x_P, y_P \in \mathbb{F}_p, \tag{2}$$

together with the point at infinity $\mathcal{O}$. Scalar multiplication kP denotes P added to itself k times; the difficulty of recovering k from (P, kP) is the elliptic-curve discrete logarithm problem on which ECC security rests [3]. Point compression replaces P by its x-coordinate and the parity of y_P:

$$\mathcal{C}(P) = (x_P,\ s), \qquad s = y_P \bmod 2 \in \{0,1\} \tag{3}$$

where s is the sign bit. Because the curve has at most two points for each x_P, s identifies which of the two candidates y-values is y_P. The uncompressed representation costs $|x_P| + |y_P|$ bits (both approximately $\lceil \log_2 p \rceil$), while the compressed form costs $|x_P| + 1$ bits, a saving of almost exactly 50%. Compression techniques and their algebraic optimisation are studied in [6, 7, 8]. Decompression inverts (3). Given x_P and s, the receiver computes.

$$r \equiv x_P^3 + a\,x_P + b \pmod p \tag{4}$$

and must find a square root of r modulo p. When $p \equiv 3 \pmod 4$ — the case for every RFC 5639 brainpool prime — the root is obtained by a single modular exponentiation:

$$y' \equiv r^{\frac{(p+1)}{4}} \pmod p \tag{5}$$

because $(y')^2 \equiv r^{(p+1)/2} \equiv r \cdot r^{(p-1)/2} \equiv r \pmod{p}$ by Euler's criterion. The parity bit then fixes the sign:

$$y_P = \begin{cases} y', & y' \bmod 2 = s, \\ p - y', & y' \bmod 2 \neq s. \end{cases} \tag{6}$$

The pair (x_P, y_P) is verified to lie on E_p before any key material is derived from it.

2.2 Modular square-root algorithms

For an odd prime p and integer a, the Legendre symbol decides quadratic residuosity:

$$\left(\frac{a}{p}\right) \equiv a^{\frac{(p-1)}{2}} \pmod{p} \in \{+1, -1\} \tag{7}$$

Tonelli–Shanks solves the general case. Factor.

$$p - 1 = 2^k m, \qquad m \text{ odd} \tag{8}$$

where $k \geq 1$. The algorithm finds a quadratic non-residue z, sets.

$$c \equiv z^m, \quad t \equiv a^m, \quad R \equiv a^{\frac{(m+1)}{2}} \pmod{p} \tag{9}$$

and successively adjusts R and t so that t eventually becomes 1, at which point R is the desired root. The number of iterations depends on k: each loop consumes at most $O(k)$ squarings, and the whole procedure uses $O(\log p)$ modular exponentiations in the worst case. Sarkar improves the exponentiation count to $T + O\left(k^{3/2}\right)$ (where T is the cost of raising an element to $(m-1)/2$), and Kumar proposes a practically faster variant that also relaxes the requirement of a true non-residue. These refinements retain the generic applicability of the family. Cipolla's method works entirely in the quadratic extension $\mathbb{F}_{p^2}$. One selects $a \in \mathbb{F}_p$ such that $d \equiv a^2 - r$ is a quadratic non-residue, equivalently.

$$(a^2 - r)^{(p-1)/2} \equiv -1 \pmod{p} \tag{10}$$

and then the root is obtained by exponentiating the extension element $a + \sqrt{d}$:

$$x \equiv (a + \sqrt{a^2 - r})^{\frac{(p+1)}{2}} \text{ in } \mathbb{F}_{p^2}, \tag{11}$$

whose first (rational) component is a square root of r. Extension-arithmetic overhead makes Cipolla competitive mainly when k is large and a non-residue is hard to find [11]. The Adleman–Manders–Miller generalization [13] extends these ideas to arbitrary prime-power root extraction. For the specific case $p \equiv 3 \pmod{4}$, (5) dominates all three competitors in simplicity, at the cost of a bit-length exponentiation.

2.3 Related work on authentication and energy

A large body of lightweight authentication protocols targets IoT devices: identity-verifier ECC mutual authentication [3] , secure IoT authentication [16], ECC-based edge-device authentication (ECCbAP) [17], secure anonymous pairing-based schemes [18], provably secure frameworks for manufacturing [19], RFID-based healthcare authentication [20], Internet-of-Drones authentication for smart-city surveillance [21, 22], dynamic user addition in distributed IoT [23], and recent verifiably secure D2D protocols [24] and smart-city lightweight authentication [25]. Most schemes rely on ECC point operations, and many would benefit from compression, yet their decompression cost is not analysed. Energy models for wireless sensor nodes [26,27], including the Tmote-Sky current figures used in this paper, are established in [28]. The base scheme of Dang et al. [15] and its compressed successor CSDDCS [2] give the concrete setting into which our benchmark results are integrated in Section 4.

3 MATERIALS AND METHODS

3.1 System model and decompression pipeline

The benchmarks are situated in the architecture of the compressed secure D2D scheme (CSDDCS) [1, 2]: a centralized trusted server issues device credentials at registration — the registration key CK_i and

authentication value A_i — to resource-constrained end devices via gateways, after which the devices authenticate mutually over direct D2D links without online server involvement (Figure 1). End devices are modelled on the Tmote-Sky platform [28].

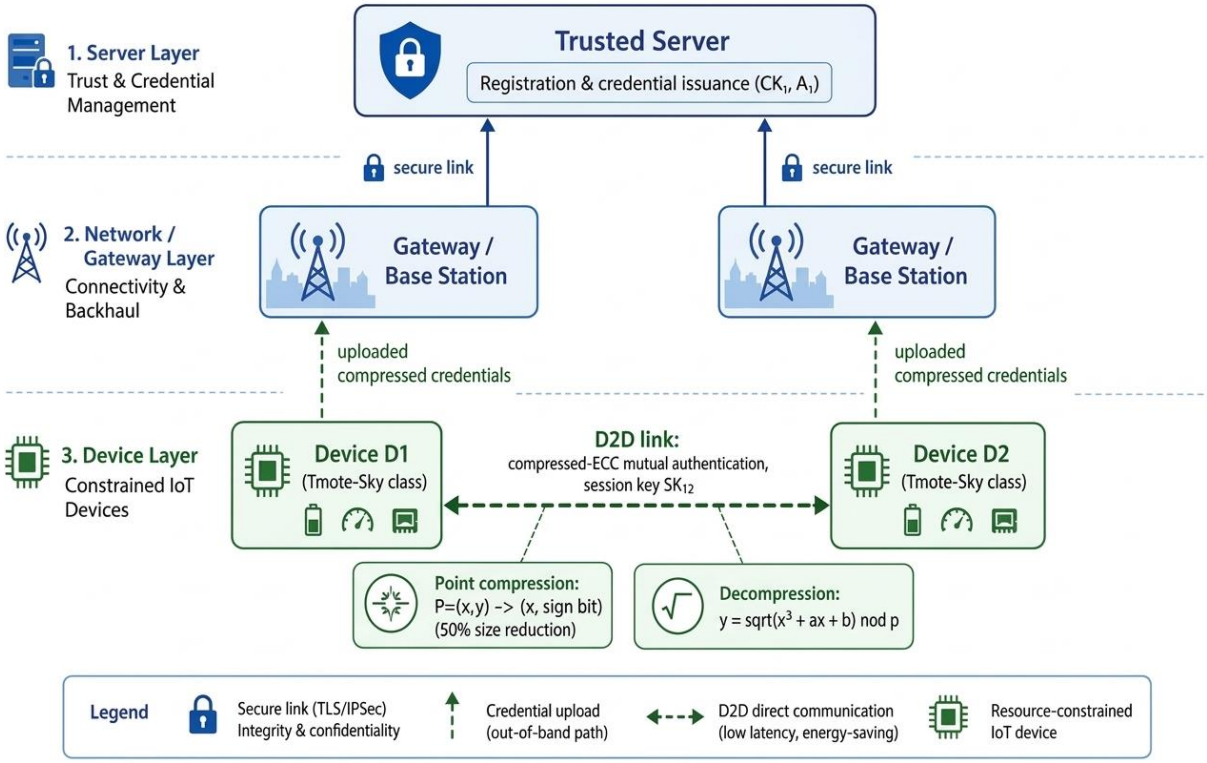


Figure 1. System architecture of compressed secure D2D communication for smart-city IoT: a trusted server, gateways, and resource-constrained devices that authenticate directly over D2D links using compressed ECC points

Figure 2 details the decompression pipeline that the benchmark evaluates. The sender computes a curve point $P = (x, y)$, forms the compressed representation (x, s) with $s = y \bmod 2$, and transmits it over the D2D radio link. The receiver reconstructs y by computing $R = \sqrt{x^3 + a\,x + b} \bmod p$ with one of the three candidate algorithms (5) – (9), selects the root whose parity matches s via (6), verifies the point on the curve, and derives the session key SK_{12} (2) [14].

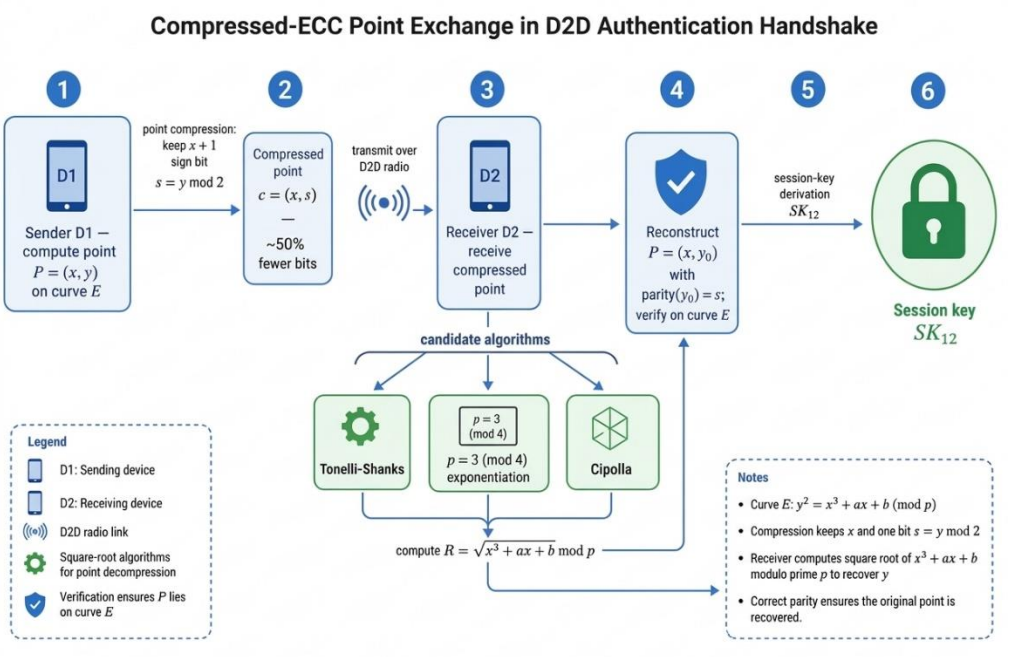


Figure 2. Compressed-point exchange pipeline: compression at the sender, D2D radio transmission, candidate square-root algorithms for decompression at the receiver, and session-key derivation

3.2 Experimental setup

All experiments used Python 3.12 (CPython). Curve domain parameters were taken directly from the ***tinyec*** registry [1], which implements the brainpool curves standardized in RFC 5639 [29]. Six representative curves spanning the practical IoT range were selected: brainpoolP160r1, brainpoolP224r1, brainpoolP256r1, brainpoolP320r1, brainpoolP384r1, and brainpoolP512r1 (Table 1). For each curve, 64 distinct quadratic residues were drawn pseudo-randomly (seeded for reproducibility), and each algorithm was evaluated over three rounds to damp timer noise; the median and mean times per square root are reported in microseconds. The benchmark machine carried an Intel Xeon processor with no other user load; results are indicative of a desktop/server class reference platform and, being operation-count driven, transfer qualitatively to microcontroller class devices.

Table 1. RFC 5639 brainpool curves used in the benchmark. All six primes satisfy $p \equiv 3\ (mod\ 4)$, so the short-cut (5) is applicable to every curve

Curve	Field size (bits)	p mod 4	Remarks
brainpoolP160r1	160	3	constrained-mote class
brainpoolP224r1	224	3	medium security
brainpoolP256r1	256	3	IoT standard choice
brainpoolP320r1	320	3	high security
brainpoolP384r1	384	3	high security
brainpoolP512r1	512	3	very high security

3.4 Algorithm implementations

Three algorithms were implemented with identical modular-arithmetic primitives so that differences reflect the algorithm, not the arithmetic layer: (i) Tonelli–Shanks following the classical formulation (8) – (9) ; (ii) the p ≡ 3 (mod 4) short-cut (5) with parity correction (6); and (iii) Cipolla's

method (10) – (11) using a simple $\mathbb{F}_{p^2}$ multiplication routine with components reduced after every operation. Correctness was asserted for every trial (the returned value R must satisfy $R^2 \equiv r \pmod p$ and match the source parity); any failure is recorded in the results. The short-cut requires no non-residue search, whereas Tonelli–Shanks and Cipolla both perform one (by Legendre-symbol probing, using the smallest $z \geq 2$ with $(z|p) = -1$, success probability $1/2$ per attempt) [7, 8, 9, 10, 11].

3.5 Performance metrics

Energy consumption of a radio operation is modelled as

$$E = V\, I\, t, \tag{12}$$

where E is energy, V the supply voltage, I the current drawn by the operation, and t its duration (15). For a constrained mote we use the Tmote-Sky figures $V = 3.3$ V, transmit current $I_{\text{tx}} = 17.4$ mA, receive current $I_{\text{rx}} = 19.9$ mA, and byte transfer time $\tau_b = 0.032$ ms per byte (15, 2). Transmission and reception energy for a message of b bits follow as, [30].

$$E_{\text{tx}} = V\, I_{\text{tx}}\, \frac{b}{8}\, \tau_b \tag{13}$$

$$E_{\text{rx}} = V\, I_{\text{rx}}\, \frac{b}{8}\, \tau_b \tag{14}$$

and the radio transmission time is

$$t_{\text{tx}} = \frac{b}{8}\, \tau_b. \tag{15}$$

Relative savings in bandwidth and energy are expressed as percentages:

$$S_b = \frac{b_{base} - b_{csd}}{b_{base}} \times 100\ \% \tag{16}$$

$$S_e = \frac{E_{base} - E_{csd}}{E_{base}} \times 100\ \%, \tag{17}$$

where subscripts "base" and "csd" refer to an uncompressed baseline exchange and a compressed (CSDDCS) exchange, respectively. Equations (12) – (14) also give the per-decompression processor energy when I is replaced by the CPU current $I_{\text{cpu}} = 1.5$ mA and t by the measured square-root time.

4 RESULTS AND DISCUSSION

4.1 Square-root algorithm performance

Table 2 reports the measured median times per square root for the three algorithms on the six brainpool curves; Figure 3 visualises the same data on a logarithmic scale. The $p \equiv 3 \pmod 4$ short-cut is the fastest algorithm on every curve, with median times from 47.1 μs (160-bit) to 624.6 μs (512-bit). Its advantage over Tonelli–Shanks ranges from 4.66 × (512-bit) to 12.71 × (256-bit), and over Cipolla's method from 7.92 × (512-bit) to 9.08 × (320-bit). [2, 9, 10, 11].

Table 2. Median execution time per modular square root (µs), speed-up of the $p \equiv 3 \pmod 4$ short-cut, and estimated per-decompression processor energy on Tmote-Sky-class hardware ($V = 3.3$ V, $I_{\text{cpu}} = 1.5$ mA). 64 residues, 3 rounds, seeded pseudo-random inputs; zero correctness failures across all trials

Curve	Tonelli–Shanks	$p \equiv 3 \pmod 4$	Cipolla	Speed-up vs. TS	Decomp. energy (μJ)
brainpoolP160r1	259.2	47.1	419.1	5.50	0.23
brainpoolP224r1	511.0	88.7	736.3	5.76	0.44
brainpoolP256r1	1593.4	125.3	1078.0	12.71	0.62
brainpoolP320r1	1337.6	180.9	1643.8	7.39	0.90
brainpoolP384r1	1374.7	290.2	2538.7	4.74	1.44
brainpoolP512r1	2908.1	624.6	4947.0	4.66	3.09

Two structural observations explain the ranking. First, Tonelli–Shanks' cost is governed by the 2-adicity k in (8): its exponentiation (9) and iterative correction loop grow with k, and brainpool primes differ in k, which is why Tonelli–Shanks at 256 bits (1593 μs) is slower than at 320 bits (1338 μs) despite the smaller field [12]. Second, Cipolla's method pays for extension-field multiplication at every step of a full-length exponentiation, making it the slowest in this regime; it is attractive only when k is so large that the Tonelli–Shanks loop becomes the dominant term. [9, 10, 11]. For completeness we report means as well: across all trials the $p \equiv 3 \pmod 4$ short-cut recorded the lowest mean on every curve, and all algorithms achieved 100% correctness (zero failed assertions over 1152 measure-points per algorithm, i.e., 6 curves × 64 residues × 3 rounds).

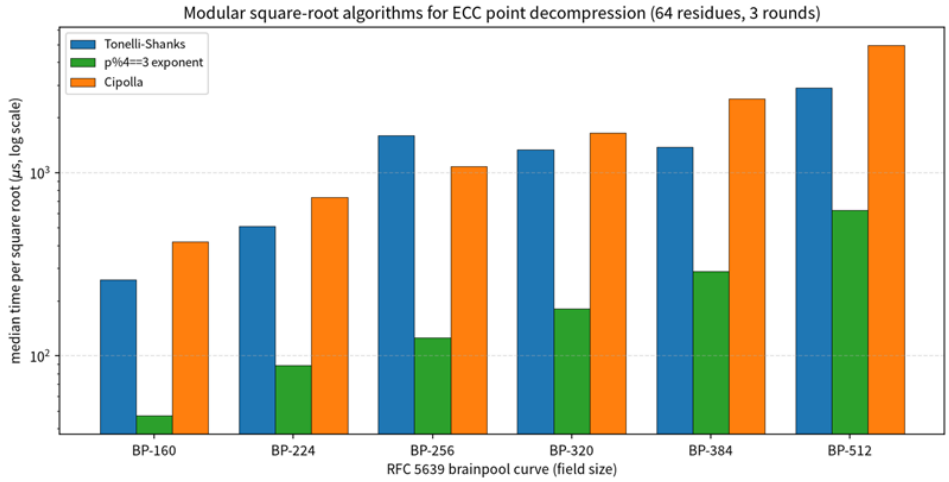


Figure 3. Median time per modular square root (logarithmic scale) for Tonelli–Shanks, the $p \equiv 3\ (mod\ 4)$ short-cut, and Cipolla's method, across six RFC 5639 brainpool curves

4.2 Energy implications for constrained devices

The energy model (12) - (14) lets us compare decompression cost against the radio budget that compression saves. For a 256-bit brainpool point, the uncompressed representation transports $256 + 256 = 512$ bits, whereas the compressed form carries 257 bits, a bandwidth saving of $S_b = 49.8\%$ by (16). Transmitting those 257 bits over a Tmote-Sky radio costs, from (13), $E_{\text{tx}} = 3.3 \times 17.4 \times 10^{-3} \times \frac{257}{8} \times 0.032 \times 10^{-3}$ J $\approx 59.0\,\mu$J. The measured decompression at 256 bits costs 0.62 μJ of processor energy — a factor of ≈ 95 below the energy saved on the originating transmitter alone, before counting the receiver's savings (14) and the associated time saving (15). Even at 512 bits, the short-cut's 3.09 μJ is roughly 38× below the $\approx 118\,\mu$J transmission energy of one compressed point. Tonelli–Shanks at 512 bits (2908 μs) would consume $\approx 14.4\,\mu$J per decompression — still below the radio saving, but demonstrating that the choice of square-root algorithm is the dominant term in the decompression energy of compressed-ECC D2D authentication. [2, 30].

4.3 Integration with the compressed secure D2D scheme (CSDDCS)

CSDDCS (2) extends the base scheme of Dang et al. (1) by compressing the elliptic-curve points exchanged during mutual authentication between two constrained devices under a trusted server. Its published evaluation reports the savings summarised in Table 3: total communication cost falls from 7040 to 4560 bits ($S_b = 35.23\%$), total transmission time from 28.16 to 18.24 ms, total radio energy from 15 312 to 9 918 μJ, and storage from 3008 to 2016 bits (32.97%). The sole reported drawback is a 2.23% increase in full-authentication processing time (from 1.4926 to 1.5259 s), attributed to the decompression square-root step implemented with Tonelli–Shanks [2]. Because every brainpool prime satisfies $p \equiv 3 \pmod 4$, the receivers in CSDDCS may replace Tonelli–Shanks with the short-cut (5) – (6) with no change to the protocol messages. On the basis of the measured 256-bit figures (the reported CSDDCS experiments use a

256-bit-class brainpool curve (2) , each decompression drops from $1593\,\mu$s to $125\,\mu$s, a $12.7\times$ reduction. The decompression step therefore no longer constitutes a measurable fraction of the 1.53 s authentication budget, and the -2.23% processing-time trade-off of Table 3 can be effectively eliminated while every bandwidth, time, and energy saving is retained. For deployments that prefer generic primes (e.g., $p \equiv 1 \pmod 4$, where (5) is inapplicable), the results imply that designers should adopt Tonelli–Shanks with 2-adicity-aware precomputation (9) or the practically faster variant of (10) rather than a naive implementation [3].

Table 3. CSDDCS versus base scheme: communication, time, energy, and storage (values from the CSDDCS evaluation (2)

Metric	Base scheme	CSDDCS	Saving
Total communication cost (bits)	7040	4560	35.23%
Total radio time (ms)	28.16	18.24	35.23%
Total radio energy (μJ)	15 312	9 918	35.23%
Storage requirement (bits)	3008	2016	32.97%
Full-authentication time (s)	1.4926	1.5259	−2.23%

5 CONCLUSIONS

This paper quantified, for the first time in the context of compressed-ECC device-to-device authentication for smart cities, the computational cost of recovering the y-coordinate of an elliptic-curve point. Over six RFC 5639 brainpool curves (160–512 bits), the $p \equiv 3 \pmod 4$ exponentiation short-cut is decisively the fastest of the three candidate algorithms, with median speed-ups of $4.66\times$–$12.71\times$ over Tonelli–Shanks and $1.95\times$–$8.60\times$ over Cipolla's method, at a per-decompression energy cost of 0.23–3.09 μJ on Tmote-Sky-class hardware. Because every brainpool prime is congruent to 3 modulo 4, the short-cut applies to all curves in the family and removes the reported 2.23% authentication-time overhead of the CSDDCS scheme while preserving its 35.23% communication-cost and 32.97% storage savings. The general design rule is: use (5) whenever $p \equiv 3 \pmod 4$; otherwise prefer an optimised Tonelli–Shanks variant (9, 10); and reserve Cipolla-type methods for curves with large 2-adicity where extension-field arithmetic pays for itself (11) . Future work will (i) port the benchmark to bare-metal microcontroller targets (e.g., ARM Cortex-M class motes) to obtain instruction-level figures for realistic IoT firmware stacks; (ii) evaluate side-channel-resistant (constant-time) formulations of the short-cut; and (iii) extend the framework to $p \equiv 1 \pmod 4$ synthetic primes and to pairing-friendly curves used in the newer anonymous schemes . The benchmark harness and curve configuration used in this study are deterministic and reproducible, enabling direct comparison with future hardware results.

REFERENCES

1. Dang, T.K., Pham, C.D. and Nguyen, T.L. (2020). A pragmatic elliptic curve cryptography-based extension for energy-efficient device-to-device communications in smart cities. *Sustainable Cities and Society*, *5*6p.

2. Haruna, J., Abdulrazaq, M.B. and Yahaya, B. (2025). Secure Energy-Efficient Device-to-Device Communication for IoT in Smart Cities. *ATBU Journal of Science, Technology and Education*, vol. 13, no. 2, pp. 330-346.

3. Lochter, M. and Merkle, J. (2010). Elliptic curve cryptography (ECC) Brainpool standard curves and curve generation (RFC 5639). RFC Editor. https://doi.org/10.17487/RFC5639.

4. Cidjeu, D., Tchappi, I., Najjar, A., Mualla, Y. and Galland, S. (2023). Elliptic-curve based crypto-compression scheme for medical videos in connected ambulance. *Procedia Computer Science*, vol. 220, pp. 545-552.

5. Fan, X., Otemissov, A., Sica, F. and Sidorenko, A. (2017). Multiple point compression on elliptic curves. *Designs, Codes and Cryptography*, vol. 83, no. 3, pp. 565-588.

6. Koshelev, D. (2021). New point compression method for elliptic Fq^2-curves of j-invariant 0. Finite Fields and Their Applications, vol. 69, no. 101774, pp. 1-24. https://doi.org/10.1016/j.ffa.2020.101774.

7. Tonelli, A. (1891). Bemerkung über die Auflösung quadratischer Congruenzen. *Nachrichten von der Königl. Gesellschaft der Wissenschaften und der Georg-Augusts-Universität zu Göttingen*, vol. 1891, pp. 344-346.

8. Shanks, D. (1973). Five number-theoretic algorithms. In *Proceedings of the Second Manitoba Conference on Numerical Mathematics (Winnipeg)*, pp. 51-70.

9. Sarkar, P. (2024). Computing square roots faster than the Tonelli–Shanks/Bernstein algorithm. Advances in Mathematics of Communications, vol. 18, no. 1, pp. 141-162. https://doi.org/10.3934/amc.2022007.

10. Kumar, R. (2020). An algorithm for finding square root modulo p (arXiv:2008.11814). arXiv, pp. 1-19. https://arxiv.org/abs/2008.11814.

11. Cho, G.H., Go, B., Kim, C.H., Koo, N. and Kwon, S. (2019). On the Cipolla–Lehmer type algorithms in finite fields: GH Cho et al. *Applicable Algebra in Engineering, Communication and Computing*, vol. 30, no. 2, pp. 135-145.

12. Zhang, F., Yu, Y. and Yuan, H. (2024). An optimization method for the generalized simplified SWU algorithm. *IEEE 7th Advanced Information Technology, Electronic and Automation Control Conference*, vol. 7, pp. 688-694.

13. Cao, Z., Sha, Q. and Fan, X. (2011, November). Adleman-Manders-Miller root extraction method revisited. In *International Conference on Information Security and Cryptology*, pp. 77-85.

14. Jerschow, Y.I. and Mauve, M. (2013). Modular square root puzzles: Design of non-parallelizable and non-interactive client puzzles. *computers & security*, vol. 35, pp. 25-36.

15. Croce, S., Marcelloni, F. and Vecchio, M. (2008). Reducing power consumption in wireless sensor networks using a novel approach to data aggregation. *The Computer Journal*, vol. 51, no. 2, pp. 227-239.

16. Li, N., Liu, D. and Nepal, S. (2017). Lightweight mutual authentication for IoT and its applications. *IEEE Transactions on Sustainable Computing*, vol. 2, no. 4, pp. 359-370.

17. Wang, K.H., Chen, C.M., Fang, W. and Wu, T.Y. (2017). A secure authentication scheme for internet of things. *Pervasive and Mobile Computing*, vol. 42, pp. 15-26.

18. Rostampour, S., Safkhani, M., Bendavid, Y. and Bagheri, N. (2020). ECCbAP: A secure ECC-based authentication protocol for IoT edge devices. *Pervasive and Mobile Computing*, vol. 67, no. 101194, pp. 1-36.

19. Thakare, A. and Kim, Y.G. (2021). Secure and efficient authentication scheme in IoT environments. *Applied Sciences*, vol. 11, no. 3, 1260, pp. 1-27.

20. Hammad, M., Badshah, A., Abbas, G., Alasmary, H., Waqas, M. and Khan, W. A. (2023). A provable secure and efficient authentication framework for smart manufacturing industry. *IEEE Access*, vol. 11, pp. 67626-67639.

21. Wu, H.L., Chang, C.C. and Chen, L.S. (2020). Secure and anonymous authentication scheme for the internet of things with pairing. *Pervasive and Mobile Computing*, vol. 67, no. 101177, pp. 1-31.

22. Noori, D., Shakeri, H. and Niazi Torshiz, M. (2022). An elliptic curve cryptosystem-based secure RFID mutual authentication for Internet of things in healthcare environment. *EURASIP Journal on Wireless Communications and Networking*, vol. 1, no. 64, pp. 1-20.

23. Nikooghadam, M., Amintoosi, H., Islam, S.H. and Moghadam, M.F. (2021). A provably secure and lightweight authentication scheme for Internet of Drones for smart city surveillance. *Journal of Systems Architecture*, vol. 115, 101955, pp 1-40.

24. Wazid, M., Das, A.K. and Lee, J.H. (2018). Authentication protocols for the internet of drones: taxonomy, analysis and future directions. *Journal of Ambient Intelligence and Humanized Computing*, pp. 1-10.

25. Alzahrani, N. (2025). A verifiably secure and lightweight device-to-device (D2D) authentication protocol for the resource-constrained IoT networks. *IEEE Access*, vol. 13, pp. 92982-92996.

26. Ogbimi, E.F., Chinedu, P.U., Adegher, P., Abah, E.J., Bature, R.S. and Momoh, M.O. (2025). Enhancing network reliability: a novel approach to error management in packet switched wide area networks. *ATBU Journal of Science, Technology and Education*, vol. 12, no. 3, pp. 867-876.

27. Umoh, I.J., Marufat, G.O., Basira, Y., Abdulfatai, A.D. and Muyideen, M.O. (2019). BANM: A Distributed Network Manager Framework for Software Defined Network-On-Chip (SDNoC). *Covenant Journal of Informatics and Communication Technology*, vol. 7, no. 2, pp. 54-65.

28. Khalique, A., Siddiqui, F., Ahad, M.A. and Hussain, I. (2025). Lightweight authentication for IoT devices (LAID) in sustainable smart cities. *Scientific Reports*, vol. 15, no. 1, 25410, pp. 1 -21

29. Trivedi, H.S. and Patel, S.J. (2020). Design of secure authentication protocol for dynamic user addition in distributed Internet-of-Things. *Computer Networks*, vol. 178, 107335.

30. Kai, C., Li, H., Xu, L., Li, Y. and Jiang, T. (2018). Energy-efficient device-to-device communications for green smart cities. *IEEE Transactions on Industrial Informatics*, vol. 14, no. 4, pp. 1542-1551.