ECC RESEARCH PROGRAM · P2.3

Cheon's Divisor-Case Attack, Reproduced and Measured

A validated quarter-power scaling baseline and an honest report on the unreached non-divisor generalization and generic lower bound

math.st [1] **@math__street** [1]

with GPT 5.6 Sol · Claude Fable 5

[1]Independent research collective, math.st · July 2026

ABSTRACT

Cheon's algorithm solves the discrete-logarithm-with-auxiliary-inputs problem — recover x from g, g^x, g^{x^d} in a cyclic group of prime order n — in $O\big(\log n \cdot \big(\sqrt{(n-1)/d} + \sqrt{d}\big)\big)$ group operations whenever $d \mid n-1$, a quarter-power speedup over the generic square-root search that underlies the security analysis of the strong Diffie–Hellman assumption. The research target of this problem was to **generalize** that attack beyond the divisor case — to non-divisor exponents and structured polynomial auxiliary inputs — and to prove a matching generic-group lower bound. We report plainly that this target was **not** reached: no non-divisor speedup, no invariant for general polynomial families, and no generalized lower bound were obtained. What we did obtain, and what this paper documents, is a faithful, auditable reproduction of the known divisor-case algorithm and a controlled measurement of its scaling. We give the two-stage multiplicative-orbit construction exactly as implemented, prove its correctness, and separate its two natural cost metrics — oracle exponentiations and primitive double-and-add operations — the second of which carries Cheon's published $\log n$ factor. Across a predeclared seeded sweep of eight prime orders from 70,913 to 1.76×10^{13} with a divisor $d \approx \sqrt{n}$, all 328 recoveries verified against ground truth and the log–log slope of median oracle exponentiations was 0.2500 with a 95% bootstrap confidence interval $[0.2461, 0.2535]$, tightly consistent with the $n^{1/4}$ prediction. We then analyze exactly why the construction does not survive the removal of the divisor hypothesis: its correctness rests on two multiplicative orbits of **exact** orders $(n-1)/d$ and d that exist only when $d \mid n-1$, and substituting a nearby non-divisor destroys both search sets rather than perturbing them. We close with the falsifiable resumption plan the notes prescribe. The contribution is a validated baseline and an honest negative report, not a solution.

Keywords. discrete logarithm with auxiliary inputs · Cheon's algorithm · strong Diffie–Hellman · generic group model · baby-step giant-step · empirical scaling

1 Introduction

Let $G = \langle g \rangle$ be a cyclic group of prime order n, written multiplicatively, and let $x \in \mathbb{Z}_n$. The **discrete logarithm problem with auxiliary inputs** asks one to recover x from g together with a list of powers $g^{f_1(x)}, \ldots, g^{f_k(x)}$ for fixed public polynomials $f_i \in \mathbb{Z}[X]$. The interest of the problem

is that the extra powers can make recovery strictly easier than the $\Theta(\sqrt{n})$ generic square-root search that a bare instance (g, g^x) forces [1], [2]. This is not a curiosity: the strong Diffie–Hellman assumption of Boneh and Boyen [3] hands an adversary exactly such a ladder of powers g^{x^i}, and Cheon [4], [5] showed that this ladder can be turned against it; the algorithm has since been analyzed and refined further [6].

The sharpest known instance is the single-power case $f_1(X) = X$, $f_2(X) = X^d$, i.e. the input $\left(g, g^x, g^{x^d}\right)$. CITED Cheon's Theorem 1 [4] shows that when the exponent d **divides** $n - 1$, one recovers x in

$$O\left(\log n \cdot \left(\sqrt{(n-1)/d} + \sqrt{d}\right)\right) \text{ group operations,} \tag{1}$$

using $O\left(\max\left\{\sqrt{(n-1)/d}, \sqrt{d}\right\}\right)$ memory. Choosing a divisor $d \approx \sqrt{n}$ balances the two square roots and yields cost $\sim n^{1/4}$, a genuine quarter-power attack. The result is the reason a strong-Diffie–Hellman parameter n is chosen so that neither $n - 1$ nor $n + 1$ has a divisor near $\sqrt{n}$ that an attacker could exploit.

1.1 The research target, and what this paper is

The problem posed for P2.3 set a deliberately ambitious target: to move **past** the divisor case. Concretely, to express the recovery complexity through algebraic invariants of the polynomial family (f_i); to extend the special-case attack to non-divisor exponents $d \nmid (n \pm 1)$ and to structured families such as $\{x, x^2 + x, x^d\}$; and to seek a **matching generic-group lower bound** that would certify the quarter-power exponent as optimal. We state the outcome without softening it.

> **Honest outcome.** The overall research target **failed**. This session produced no non-divisor speedup, no invariant for general polynomial auxiliary inputs, and no generalized generic-group lower bound. What it produced, and what this paper reports, is a validated reproduction of the **known** $d \mid n - 1$ algorithm together with a controlled empirical confirmation of its $n^{1/4}$ scaling, plus a precise analysis of why the construction does not extend off the divisor set. The paper is a baseline-and-obstruction report, not a generalization.

This framing is deliberate and, in this research program, expected: a faithful negative report that pins down **where** an approach stops is a legitimate result and is more useful to a successor than an overstated one. Every mathematical claim below carries the epistemic tag it was recorded with — CITED for facts imported from the literature, EMPIRICAL: range for measured findings with their validity range, and CONJECTURE for the working hypotheses that a resumption would test.

1.2 Contributions

We contribute (i) the two-stage multiplicative-orbit recovery exactly as implemented, with an explicit correctness proof (§3); (ii) a clean separation of the two cost metrics — oracle exponentiations versus primitive group operations — and the reason Cheon's $\log n$ factor lives only in the second (§4); (iii) a predeclared, seeded empirical scaling study over two sweeps with fitted exponents and bootstrap confidence intervals, all recoveries verified (§5, Figure 1, Table 1); (iv) a structural analysis of the divisor obstruction that explains why substituting a non-divisor exponent destroys rather than perturbs the attack (§6, Figure 4); and (v) the falsifiable resumption plan the research notes prescribe (§7).

2 Setting and notation

Fix a prime $n > 3$ and a cyclic group $G = \langle g \rangle$ of order n. Because n is prime, the residue ring $\mathbb{Z}_n = \mathbb{Z}/n\mathbb{Z}$ is the field $\mathbb{F}_n$, and its unit group $\mathbb{F}_n^\times$ is cyclic of order $n-1$; fix a generator (primitive root) ζ of $\mathbb{F}_n^\times$. The secret is $x \in \mathbb{Z}_n$; we treat $x \neq 0$ as the generic case and handle $x = 0$ (equivalently $g^x = g^0$) as an immediate special case. For a fixed divisor $d \mid n-1$ write

$$q := (n-1)/d, \quad \text{so} \quad n - 1 = d \cdot q. \tag{2}$$

The attack is given the three group elements g, g^x, g^{x^d} and the public data (n, d); it must output x.

We deliberately keep the group interface **opaque**. The algorithm may call a single oracle, `scalar_mul`$(s, P) = P^s$ (equivalently $[s]P$ in additive notation), and compare returned handles for equality; it never sees an encoding it can compute with directly. This is the honest generic-group interface in which a lower bound would have to be proved, and it is what the implementation's simulator enforces. We distinguish two cost counters on this interface:

> **Definition 1** (the two cost metrics). An **oracle exponentiation** is one call `scalar_mul`(s, P), charged as one unit regardless of s. A **primitive group operation** is one squaring or one multiplication inside a double-and-add evaluation of that call; a call with scalar s costs $\lfloor \log_2 s \rfloor + 1 + \nu(s)$ primitive operations, where $\nu(s)$ is the number of 1-bits of s (one doubling per bit, one extra multiply per set bit). The exponentiation count is the representation-independent query measure; the primitive count is the concrete work and carries an extra $\Theta(\log n)$ factor.

This distinction is not pedantry: Cheon's bound Equation 1 counts **primitive** group operations and therefore includes the $\log n$ factor, whereas the balanced square-root structure $\sqrt{q} + \sqrt{d}$ is a statement about **exponentiations**. We report both and keep them separate throughout, following the research invariant that the two must never be conflated.

3 The divisor-case algorithm and its correctness

The implementation reduces the whole recovery to a generic subroutine — a baby-step giant-step (BSGS) search [7] inside a single **multiplicative orbit** of the field $\mathbb{F}_n^\times$ — invoked twice.

> **Definition 2** (multiplicative-orbit BSGS). Given a group element T, a field offset $a \in \mathbb{F}_n^\times$, a field multiplier $\mu \in \mathbb{F}_n^\times$ of multiplicative order ℓ, and the target promise that $T = g^{a \cdot \mu^k}$ for some $0 \le k < \ell$, the routine finds that k. Put $w = \lceil \sqrt{\ell} \rceil$. It builds a table of the **baby** handles $g^{a\mu^j} =$ scalar_mul$(a\mu^j, g)$ for $0 \le j < w$, then walks the **giant** steps $T \cdot g^{-\mu^{wi}}$ for $i = 0, 1, \ldots$ until a handle matches a table entry, returning $k = iw + j$. It uses $2w + O(1)$ oracle exponentiations and $w + O(1)$ table probes; a guard rejects inputs for which $\mu^\ell \neq 1$, i.e. for which the stated orbit order is wrong.

The correctness of the whole attack is the statement that the two invocations below have exactly the promised orbit structure. This is where — and the **only** where — the hypothesis $d \mid n-1$ is spent.

> **Proposition 3** (two-stage recovery). PROVED Let n be prime, $x \in \mathbb{Z}_n$ with $x \neq 0$, $d \mid n-1$, $q = (n-1)/d$, and ζ a primitive root modulo n. Then:
> (i) x^d lies in the order-q subgroup $\langle \zeta^d \rangle \subseteq \mathbb{F}_n^\times$, and there is a unique $k_1 \in \{0, \ldots, q-1\}$ with $x^d = (\zeta^d)^{k_1}$;

(ii) $x = \zeta^{k_1} \cdot (\zeta^q)^{k_2}$ for a unique $k_2 \in \{0, \ldots, d-1\}$, where ζ^q has order d and generates the group μ_d of d-th roots of unity in $\mathbb{F}_n^\times$.

Consequently, running the orbit-BSGS first with $(T, a, \mu, \ell) = \left(g^{x^d}, 1, \zeta^d, q\right)$ and then with $\left(g^x, \zeta^{k_1}, \zeta^q, d\right)$ recovers k_1 and k_2, and $\zeta^{k_1}(\zeta^q)^{k_2} = x$.

Proof. Write $x = \zeta^t$ with $t \in \{0, \ldots, n-2\}$; this is possible since $\mathbb{F}_n^\times = \langle\zeta\rangle$ and $x \neq 0$. Then $x^d = \zeta^{td} = \left(\zeta^d\right)^t$, and because ζ^d has order

$$(n-1)/\gcd(d, n-1) = (n-1)/d = q \tag{3}$$

(using $d \mid n-1$), we have $x^d = \left(\zeta^d\right)^{t \bmod q}$; set $k_1 = t \bmod q$, which is the unique exponent in $\{0, \ldots, q-1\}$, proving (i). For (ii), note

$$x/\zeta^{k_1} = \zeta^{t-(t \bmod q)} = \zeta^{q\lfloor t/q \rfloor} = (\zeta^q)^{\lfloor t/q \rfloor}. \tag{4}$$

Since $(\zeta^q)^d = \zeta^{qd} = \zeta^{n-1} = 1$ and ζ^q has order exactly $(n-1)/\gcd(q, n-1) = (n-1)/q = d$, the element ζ^q generates μ_d and $x/\zeta^{k_1} \in \mu_d$. Because $t < n-1 = qd$, the quotient $\lfloor t/q \rfloor$ lies in $\{0, \ldots, d-1\}$, so $k_2 := \lfloor t/q \rfloor \bmod d = \lfloor t/q \rfloor$ is the unique such exponent, proving (ii). Finally

$$\zeta^{k_1}(\zeta^q)^{k_2} = \zeta^{(t \bmod q) + q\lfloor t/q \rfloor} = \zeta^t = x, \tag{5}$$

using $t = (t \bmod q) + q\lfloor t/q \rfloor$. Each orbit-BSGS finds its exponent by the standard meet-in-the-middle over a cyclic orbit of the stated exact order, matching handles through the `scalar_mul` oracle; the divisor hypothesis is exactly what makes those two orders equal to q and d. □

The implementation performs one final **verification** exponentiation, checking $g^{\text{recovered}} = g^x$ and raising if it fails; this makes every reported recovery self-certifying. The subroutine's orbit guard ($\mu^\ell \neq 1$) is what causes the attack to **reject**, rather than silently misbehave, when handed a non-divisor d — a fact we return to in §6.

Remark 4 (what each stage searches). PROVED Stage 1 searches the q-element subgroup of d-th powers to pin down x^d; stage 2 searches the d-element coset of d-th roots of unity to pin down which d-th root of x^d equals x, disambiguated by the extra input g^x. The two searches are independent and their sizes multiply the **field** structure $n - 1 = qd$, not the group order n. This is the whole source of the speedup.

4 Cost model and the two metrics

Summing the subroutine costs of Figure 4's two stages gives the exponentiation cost directly.

Proposition 5 (exponentiation cost). PROVED The two-stage recovery uses

$$C(n, d) = 2\lceil\sqrt{q}\rceil + 2\left\lceil\sqrt{d}\right\rceil + O(1) = 2\left(\sqrt{(n-1)/d} + \sqrt{d}\right) + O(1) \tag{6}$$

oracle exponentiations and $O\left(\max\left\{\sqrt{q}, \sqrt{d}\right\}\right)$ memory. Over divisors $d \mid n-1$, Equation 6 is minimized near $d \approx \sqrt{n}$, where $q \approx \sqrt{n}$ and $C(n, d) = \Theta\left(n^{1/4}\right)$.

Multiplying each exponentiation by its double-and-add length recovers Cheon's primitive-operation bound Equation 1: since every scalar is reduced modulo n, a call costs $\Theta(\log n)$ primitive operations, so the primitive count is $\Theta\left(\log n \cdot \left(\sqrt{q} + \sqrt{d}\right)\right)$. The two metrics therefore scale with **different** exponents in n along a sweep that holds $d \approx \sqrt{n}$: the exponentiation count as $n^{1/4}$

and the primitive count as $n^{1/4} \log n$, which over a bounded range of n presents as a slightly steeper effective slope. Keeping the metrics apart is thus not cosmetic — it is what makes the clean quarter-power statement true of the query measure while the concrete work retains its logarithmic overhead. Figure 2 shows both series on one log–log axis.

5 Empirical scaling

EMPIRICAL: 328 verified trials, 70913 <= n <= 1.76e13 We reproduced the attack under the opaque oracle and measured its cost with a predeclared, seeded experiment. Primes were generated in the structured form $n = d \cdot e + 1$ with $d = 2^m$ and e the least odd integer $\geq d + 1$ making n prime, so that $d = 2^m$ is a divisor of $n - 1$ sitting a hair below $\sqrt{n}$; this is the regime in which the quarter-power is expected. Two sweeps were run from a single seed (2303): a fast four-size smoke sweep ($m \in \{4, 5, 6, 7\}$, 3 trials per size) and the main eight-size sweep ($m \in \{8, 10, \ldots, 22\}$, 41 trials per size, 2000 within-size bootstrap resamples).

Table 1: Main sweep (`hb8-22_t41_s2303`). Per-size medians over 41 seeded trials: oracle exponentiations, primitive group operations, the theoretical shape $\sqrt{q} + \sqrt{d}$, and the residual of the fitted power law on median exponentiations. All 328 recoveries verified against their seeded secrets.

m	$\lceil \log_2 n \rceil$	$d = 2^m$	q	med. exp.	med. grp. ops	$\sqrt{q} + \sqrt{d}$	log resid.
8	17	256	277	47	890	32.6	−0.035
10	21	1024	1027	102	3110	64.0	+0.066
12	25	4096	4117	185	6306	128.2	−0.033
14	29	16384	16393	384	15732	256.0	+0.005
16	33	65536	65541	752	34955	512.0	−0.016
18	37	262144	262189	1552	81418	1024.0	+0.016
20	41	1048576	1048581	3082	176926	2048.0	+0.009
22	45	4194304	4194309	6037	382626	4096.0	−0.012

The fitted log–log slope of median oracle exponentiations is the headline number.

Measured scaling. EMPIRICAL: 328 trials, 8 sizes, seed 2303 On the main sweep the median oracle-exponentiation count fits a power law in n with slope $\mathbf{0.2500}$ and 95% within-size bootstrap confidence interval $[0.2461, 0.2535]$ — squarely on the $n^{1/4}$ prediction and inside the predeclared target band $[0.20, 0.30]$. Every one of the 328 recoveries verified against ground truth.

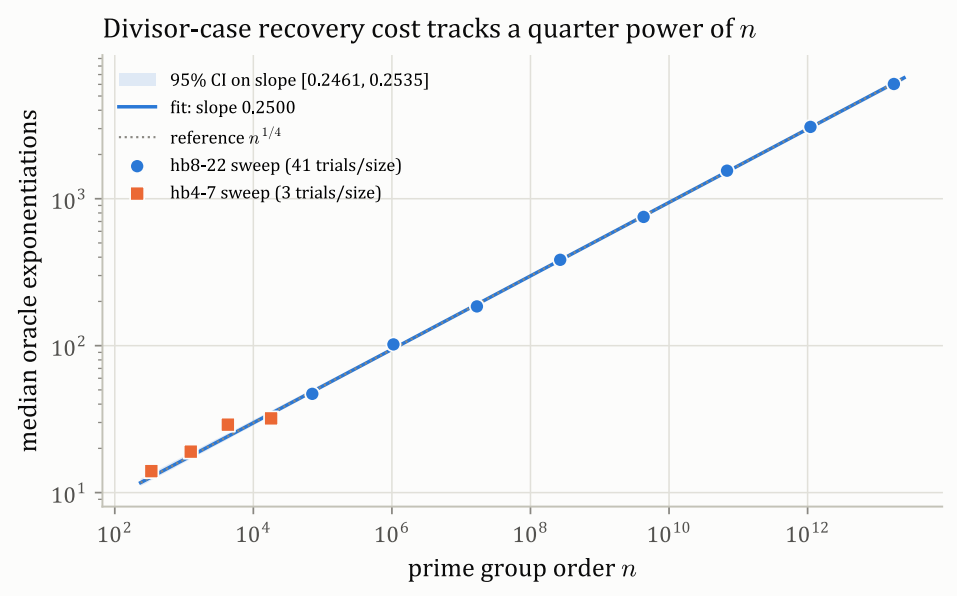

Figure 1: EMPIRICAL: two seeded sweeps, seed 2303 Median oracle exponentiations against prime order n on log–log axes. Filled circles: main sweep ($m = 8, \ldots, 22$; 41 trials/size). Squares: smoke sweep ($m = 4, \ldots, 7$; 3 trials/size). The band is the 95% bootstrap confidence interval on the fitted slope; the dotted line is the exact $n^{1/4}$ reference. Data: `run_scaling_summary_hb8-22_t41_s2303_20260713.csv` and the smoke counterpart, with fits in the paired `_fit_*.json`.

The two sweeps agree on the exponent within their statistics; the smaller smoke sweep is noisier, as expected from three trials per size (Table 2). The residual column of Table 1 shows no systematic drift with size — the single power law is an adequate description over the whole tested range — and the measured counts sit at a near-constant multiple ≈ 1.5 of the theoretical shape $\sqrt{q} + \sqrt{d}$, the multiple being below the worst-case 2 because the giant-step walk terminates at the match rather than exhausting the orbit. Figure 3 displays both facts.

Table 2: Fit summary for the two sweeps. Slopes and intervals are on median oracle exponentiations; the main-sweep interval is the tighter one and pins the exponent at a quarter power. Both intervals intersect the predeclared band and every recovery in both sweeps verified. Source: the two `run_scaling_fit_*.json`.

Sweep	sizes	trials/size	total	fitted slope	95% CI	in $[0.20, 0.30]$?
`hb4-7` (smoke)	4	3	12	0.2186	$[0.2024, 0.2649]$	yes
`hb8-22` (main)	8	41	328	0.2500	$[0.2461, 0.2535]$	yes

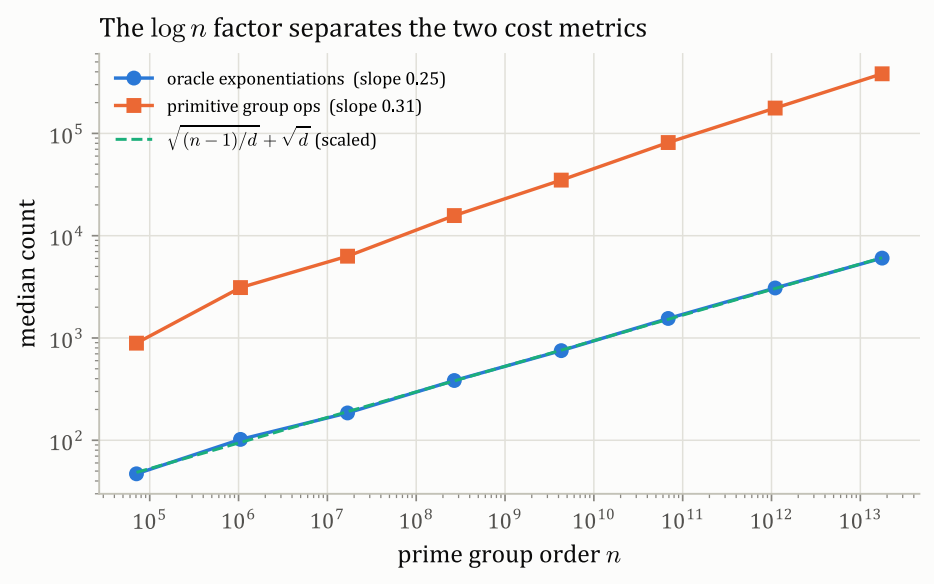

Figure 2: EMPIRICAL: main sweep, seed 2303 The two cost metrics on one log–log axis. Oracle exponentiations (slope ≈ 0.25) realize the clean quarter power; primitive double-and-add group operations (steeper effective slope) carry Cheon's extra $\log n$ factor. The dashed line is the theoretical shape $\sqrt{(n-1)/d}+\sqrt{d}$ scaled onto the exponentiation series. Data: `run_scaling_summary_hb8-22_t41_s2303_20260713.csv`.

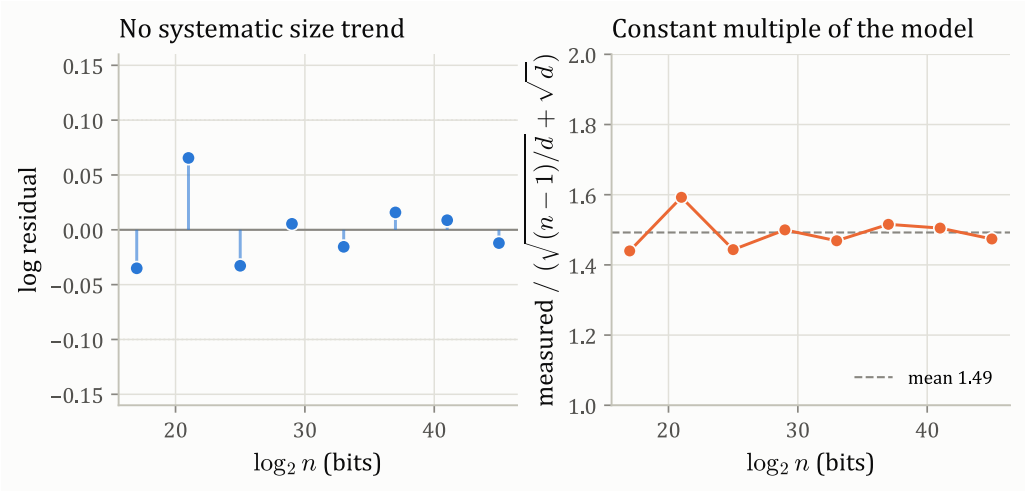

Figure 3: EMPIRICAL: main sweep, seed 2303 Left: log-residuals of the fitted power law against $\log_2 n$, all within ± 0.07 and without a size trend, supporting a single power-law description. Right: the ratio of measured median exponentiations to the theoretical shape $\sqrt{q}+\sqrt{d}$, a near-constant ≈ 1.5 across four orders of magnitude in n. Data: same summary CSV.

Two exhaustive known-answer checks anchor the sweep at small scale. EMPIRICAL: exhaustive n in {17,19,31} In the opaque simulator the implementation recovered **every** secret $x \in \{0, \ldots, n-1\}$ for $(n, d) \in \{(17, 4), (19, 3), (31, 5)\}$, and on the concrete order-19 short-Weierstrass elliptic-curve group it recovered every secret for $d \in \{3, 6\}$. These fix the construction against ground truth before any asymptotic claim is made.

6 The divisor obstruction: why the attack does not generalize

The empirical section confirms the **known** case. The research target was to leave it. We now record precisely why the construction, as it stands, does not survive the removal of the divisor hypothesis — the analysis that a resumption must start from.

The correctness proof of §3 spends the hypothesis $d \mid n-1$ in exactly two places, and nowhere else:

1. **Stage-1 orbit.** The map $y \mapsto y^d$ on $\mathbb{F}_n^\times$ has image the subgroup of d-th powers, of order $(n-1)/\gcd(d, n-1)$. When $d \mid n-1$ this is exactly $q = (n-1)/d$, so x^d lies in a cyclic orbit of the **known** size q that the BSGS enumerates. When $d \nmid n-1$, the image has order $(n-1)/\gcd(d, n-1) > q$, the parameter q is no longer an orbit order, and the subroutine's guard $\mu^q \neq 1$ fires.
2. **Stage-2 coset.** The d-th roots of a fixed value form a coset of $\mu_{\gcd(d,n-1)}$, the group of $\gcd(d, n-1)$-th roots of unity. When $d \mid n-1$ this group has order exactly d and is generated by ζ^q, so the correct root of x^d is one of d candidates that stage 2 enumerates. When $d \nmid n-1$ there are only $\gcd(d, n-1) < d$ candidates and ζ^q has the wrong order, so the second search set is malformed as well.

Both search sets are therefore not merely **perturbed** by moving d off the divisor lattice — they **disappear**, because the exact factorization $n-1 = qd$ that defines them no longer holds. Figure 4 makes this visible: the two orbits that the two stages ride are features of the divisor factorization, and there is no nearby integer one can substitute for d to keep them. The implementation encodes this honestly: its known-answer suite includes a test that `cheon_recover` **raises** on $(n, d) = (19, 4)$, since $4 \nmid 18$.

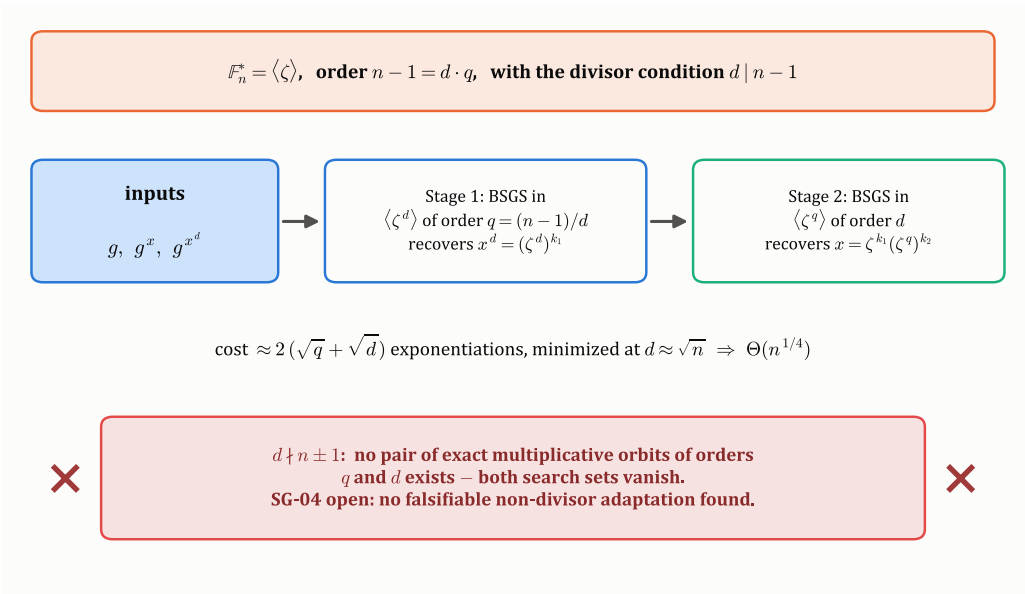

Figure 4: Structure of the two-stage recovery and the divisor obstruction. The two BSGS searches ride the exact multiplicative orbits $\langle\zeta^d\rangle$ (order q) and $\langle\zeta^q\rangle$ (order d), whose orders multiply to $n-1 = qd$ — available only when $d \mid n-1$. Off the divisor set the two orbits do not exist, so a bare substitution of a non-divisor exponent has no search space to run in.

This yields the working hypothesis under which the notes leave SG-04.

> **Remark 6** (the replacement-search-set hypothesis). CONJECTURE The divisor condition supplies the two exact multiplicative orbits that the implementation rides; a **useful** non-divisor adaptation must therefore identify a genuine replacement search set — for instance by exploiting the $\gcd(d, n-1)$ structure, or by importing Cheon's separate $n+1$ construction, which uses a longer supplied power ladder — rather than merely substituting a nearby integer for d. This hypothesis is **refuted** by exhibiting an algorithm that, using the same inputs for some $d \nmid (n \pm 1)$, achieves measured cost below generic square-root search on a controlled infinite family. No such algorithm was found this session; the hypothesis stands untested, not established.

We also did not attempt the matching generic-group lower bound (SG-07) beyond noting its shape: it would need a collision-space argument in an opaque model of the kind Shoup and Nechaev use for the plain discrete logarithm [1], [2], here adapted to §2′s oracle, that charges the exponentiation oracle and shows no generic algorithm can beat $n^{1/4}$ using the auxiliary power — a statement about the **field** structure $n - 1 = qd$ seen only through group handles. Formalizing that model and proving such a bound remains entirely open, and we make no claim about it here.

7 Open questions and resumption plan

The sub-goals decompose the gap between this baseline and the target. We record them as the notes prescribe, in the order a resumption should take them.

Table 3: Sub-goal ledger. The first two are complete and constitute this paper's baseline; SG-03 through SG-07 are the unreached generalization and lower-bound targets, listed in resumption order.

Sub-goal	Status	What it requires
SG-01 known-case recovery	PROVED done	two-stage recovery, exhaustively validated (§3, §5)
SG-02 quarter-power fit	EMPIRICAL: 328 trials done	slope 0.2500, CI $[0.2461, 0.2535]$ (§5)
SG-03 full divisor sweep	open	for fixed n, sweep **all** $d \mid n-1$ and compare counts with $\sqrt{n/d} + \sqrt{d}$ across the divisor lattice, not only $d \approx \sqrt{n}$
SG-04 non-divisor adaptation	open	define an executable search set for $d \nmid (n \pm 1)$ with a nearby divisor-case live control; measure whether it degrades or fails
SG-05 structured families	open	test polynomial families beyond monomials, beginning with $\{x, x^2 + x, x^d\}$
SG-06 algebraic invariants	open	state candidate invariants of (f_i) and falsify them against all measurements
SG-07 generic lower bound	open	formalize the opaque collision space of §2 and attempt an $n^{1/4}$ bound

The single most informative next step is SG-03: a full divisor sweep at fixed n would show whether the measured cost tracks $\sqrt{n/d} + \sqrt{d}$ across the **entire** divisor lattice or only near the balance point, and it would furnish the live divisor-case control against which any SG-04 non-divisor adaptation must be measured. Only after that control exists can a non-divisor claim be made falsifiable rather than anecdotal.

8 Conclusion

We set out to generalize Cheon's divisor-case discrete-logarithm attack beyond $d \mid n \pm 1$ and to prove a matching generic-group lower bound. We did neither. What we can stand behind is narrower and, we hope, cleanly stated: a faithful, self-certifying reproduction of the two-stage $d \mid n-1$ algorithm (Proposition 2), a careful separation of its two cost metrics with Cheon's $\log n$ factor placed exactly where it belongs (§4), and a predeclared seeded measurement in which all 328 recoveries verified and the oracle-exponentiation count scaled as $n^{0.2500}$ with a 95% confidence interval $[0.2461, 0.2535]$ — the quarter power, confirmed (§5). We then pinned the obstruction: the construction rides two multiplicative orbits of exact orders q and d that exist only because $n-1 = qd$, so a non-divisor exponent leaves it with no search space rather than a harder one (§6). That is the honest state of P2.3 — a validated baseline and a precise account of where the road stops, handed forward with the divisor sweep of SG-03 marked as the next step.

Reproducibility

The recovery is `code/cheon.py` (the two-stage `cheon_recover`, the `multiplicative_orbit_bsgs` subroutine, and the opaque `OpaquePrimeOrderGroup` simulator whose counters separate oracle exponentiations from primitive group operations). The scaling driver is `code/run_scaling.py`; the predeclared main run is reproduced with default arguments (`--half-bits 8,10,12,14,16,18,20,22 --trials 41 --seed 2303 --bootstrap 2000`) and the smoke run with `--smoke`. These write the row data, per-size summaries, and bootstrap fits used here: `run_scaling_hb8-22_t41_s2303_20260713.csv` and its `summary/fit` companions, and the paired `hb4-7_t3_s2303` files. Exhaustive known-answer tests are in `code/tests/test_cheon.py` (orders $17, 19, 31$ in the simulator and the concrete order-19 elliptic-curve group). All group orders stay within the scaffold's 60-bit ceiling. Every mathematical claim above carries one of the epistemic tags PROVED, CITED, EMPIRICAL: range, CONJECTURE exactly as recorded in the research log; untagged sentences are exposition, not claims. The overall research target is recorded as **failed**: this paper reproduces the known divisor case and does not solve the generalization.

References

[1] V. Shoup, "Lower Bounds for Discrete Logarithms and Related Problems," in *Advances in Cryptology — EUROCRYPT 1997*, in LNCS, vol. 1233. Springer, 1997, pp. 256–266.

[2] V. I. Nechaev, "Complexity of a Determinate Algorithm for the Discrete Logarithm," *Mathematical Notes*, vol. 55, no. 2, pp. 165–172, 1994, doi: 10.1007/BF02113297.

[3] D. Boneh and X. Boyen, "Short Signatures Without Random Oracles," in *Advances in Cryptology — EUROCRYPT 2004*, in LNCS, vol. 3027. Springer, 2004, pp. 56–73. doi: 10.1007/978-3-540-24676-3_4.

[4] J. H. Cheon, "Security Analysis of the Strong Diffie-Hellman Problem," in *Advances in Cryptology — EUROCRYPT 2006*, in LNCS, vol. 4004. Springer, 2006, pp. 1–11. doi: 10.1007/11761679_1.

[5] J. H. Cheon, "Discrete Logarithm Problems with Auxiliary Inputs," *Journal of Cryptology*, vol. 23, no. 3, pp. 457–476, 2010, doi: 10.1007/s00145-009-9047-0.

[6] S. Kozaki, T. Kutsuma, and K. Matsuo, "Remarks on Cheon's Algorithms for Pairing-Related Problems," in *Pairing-Based Cryptography — Pairing 2007*, in LNCS, vol. 4575. Springer, 2007, pp. 302–316. doi: 10.1007/978-3-540-73489-5_17.

[7] D. Shanks, "Class Number, a Theory of Factorization, and Genera," *Proc. Symp. Pure Math.*, vol. 20, pp. 415–440, 1971.