

Hashrate-Tiered Fees and PPLNS Settlement for Home-Miner Pools on a Sharded Proof-of-Work Blockchain: Design, Deployment and Simulation on SCDO

Ying Willam ZHANG (ORCID 0009-0002-1526-3332), SCDO Laboratory, 9Y9 PTY LTD — corresponding author

Preprint, October 2026. Manuscript prepared for *Ledger*.

Abstract

Miners who use one home graphics card or desktop CPU earn very irregular income when they mine alone: a small device can go several hours without finding a block. We present two mining pools that are running in production on the SCDO sharded proof-of-work (PoW) blockchain and that pay miners automatically on chain under Pay-Per-Last-N-Shares (PPLNS) accounting with hashrate-tiered fees of 5 %, 3 % and 1 %. The first pool serves GPU miners on SCDO Shard0, an EVM-compatible Ethash shard (chain ID 5680). The second, zpool, is a purpose-built Go pool for CPU miners on SCDO Classic Shard1, whose zpow algorithm rewards finding a large determinant of a 30×30 matrix derived from the block header. Because the work behind a zpow share does not grow linearly with share difficulty, zpool weights every share by its empirically measured expected number of hashes. We report production records, including 28,823 blocks found by the Shard0 pool and 587 automatic payouts, with sample transactions that anyone can check on chain. We also report a share-level Monte-Carlo simulation whose parameters come from the live pool configurations. Across 30 simulated days on each shard, every miner's share of gross rewards stayed within 0.3 % of its share of hashrate. For the smallest simulated home miner, the coefficient of variation (CV) of hourly income fell from 1.44 when mining alone to 0.075 in the pool on Shard0, and from 0.86 to 0.098 on Classic Shard1. Hours without income fell from 61.7 % and 25.6 % to zero. Under the tier schedule, net income never decreases as hashrate rises, and splitting one miner into several addresses can never lower its fee rate. Half of the pool-fee revenue is passed on to full-node operators.

Keywords: mining pools; PPLNS; tiered fees; home mining; proof of work; sharded blockchain; zpow; Ethash

1. Introduction

Proof-of-work blockchains pay block rewards to whoever finds a valid block.[1] A miner's expected income is proportional to its share of network hashrate, but the timing of that income is random: block discoveries follow a Poisson process, and a miner holding a small share of the network waits a long and highly variable time between its own blocks. For a person mining with one consumer graphics card or one desktop CPU, solo mining can mean several hours without any income, followed by a single block reward. Mining pools solve this by combining many miners' work and sharing each block found among the contributors according to the work they submitted.[2] The pool reward function, the fee policy, the length of the PPLNS window and the mechanics of payout together decide how fair, how predictable and how transparent the pool is for its members.

SCDO is a sharded PoW platform. Its Shard0 is an EVM-compatible PoW shard (chain ID 5680) that uses Ethash, the algorithm of pre-Merge Ethereum, and on which other SCDO systems are deployed.[3] The SCDO Classic shards run the go-scdo node and the zpow algorithm, which derives a pseudo-random 30×30 matrix

from the block header and nonce and accepts the nonce when the matrix determinant meets the target. The two shards therefore suit two different kinds of home hardware: Shard0 suits consumer GPUs, and the Classic shards can be mined with ordinary CPUs using the zminer client described below. This setting allows one pool design to be deployed on two PoW algorithms within one platform.

This paper describes the design, deployment and simulation-based evaluation of the two SCDO pools. Both are production systems: they accept shares, find blocks and send on-chain transactions. At the time of writing, their production data come mainly from hardware operated by the SCDO project itself, so we evaluate behaviour with many independent miners through a reproducible simulation whose parameters are taken from the live configurations.

The paper makes four contributions:

1. **One tiered-fee PPLNS design on two PoW algorithms.** We deploy the same fee schedule (5 %, 3 %, 1 % by hashrate) and PPLNS settlement on Ethash (Shard0) and on zpow (Classic Shard1). The schedule keeps net income non-decreasing in hashrate, and splitting one miner into several addresses can never lower its fee rate.
2. **Expected-hash share weighting for zpow.** Because the work behind a zpow share does not grow linearly with share difficulty, we measured the tail of the determinant distribution on real Shard1 block headers (1.2 million samples). The resulting lookup table assigns each share a weight equal to the number of hashes it is expected to cost, so miners at different variable-difficulty (vardiff) levels are paid on an equal basis.
3. **Transparent, automatic on-chain settlement with hot-wallet controls.** Payouts run every 10 minutes from 1 SCDO, the pool pays the network fee, and every block and payment is public. The Classic pool caps its hot wallet at 50 SCDO, sweeps fees and any excess to a Treasury, and holds back new payout rounds until earlier transactions are included in a block.
4. **A reproducible fairness simulation based on the live configurations.** A share-level Monte-Carlo model, run for 30 simulated days per scenario, quantifies proportionality, income variability and how fees fall on each tier.

Section 2 reviews pool reward schemes. Section 3 presents the design, Section 4 the deployment and production records, and Section 5 the simulation. Section 6 discusses configuration choices, Section 7 concludes, and Section 8 contains the declarations.

2. Background and Related Work

A pool's reward function decides how each block reward found by the pool is divided among its miners. Rosenfeld's analysis of pooled-mining reward systems describes the main families.[2] Under the proportional scheme (PROP), each block reward is divided in proportion to the shares submitted during the current round. PROP is vulnerable to pool hopping: a miner gains by joining a pool early in a round and leaving when the round grows long. Pay-per-share (PPS) pays a fixed expected amount for every valid share. This removes almost all variance for the miner but moves the risk of block-finding luck to the operator, who must hold reserves. Pay-per-last-N-shares (PPLNS) pays each block over a sliding window of the most recent shares, whatever round they belong to. This removes the main pool-hopping incentive of PROP without the operator risk of PPS.[2] The window can be counted in shares or in total share difficulty. The two SCDO pools use both

kinds of window: a fixed amount of difficulty on Shard0, and a multiple of the current block work on Classic Shard1.

Schrijvers, Bonneau, Boneh and Roughgarden study pool reward functions as mechanisms and ask when honest, immediate reporting of shares is a miner's best strategy.[4] Lewenberg, Bachrach, Sompolinsky, Zohar and Rosenschein model pools as a cooperative game and study how miners can split rewards and move between pools.[5] Eyal and Sirer show that a sufficiently large pool can gain more than its fair share by selfish mining, which links pool size to the security of the chain.[6] Romiti, Judmayer, Zamyatin and Haslhofer measure the share-level behaviour and concentration of real Bitcoin pools.[7] Most of this literature concerns Bitcoin. Our work contributes a deployed design for a sharded platform with two PoW algorithms, one of which (zpow) needs its own share-weighting method.

The Shard0 pool is built on the open-etc-pool-friends codebase family, represented by the public repository at <https://github.com/yuriy0803/open-etc-pool-friends>,[8] with SCDO additions: tiered fees, a standard PPLNS window and a block-timestamp gate. Ethash is specified in the Ethereum Yellow Paper.[9] The Classic pool, zpool, follows the same architecture but is new code written for zpow and the go-scdo node interface (scdo_getWork and scdo_submitWork).

The fee policy is the other half of a pool's design. Here the fee depends on the miner's measured hashrate, and larger miners pay a lower rate (a volume discount). Section 3.2 shows that the schedule never makes a miner's net income fall as its hashrate grows, and that it gives no reason to split one miner into several identities.

3. System Design

3.1 Chains and pools

Table 1 summarises the two deployments. All values come from the pools' public interfaces, the SCDO block explorer (scdoscan.io) and the pool configuration and source code.

Table 1: The two SCDO pools.

Item	Shard0 pool	Classic Shard1 pool
Chain	SCDO Shard0, EVM-compatible, chain ID 5680	SCDO Classic Shard1, go-scdo node
PoW algorithm	Ethash (as pre-Merge Ethereum)	zpow (determinant of a 30×30 matrix derived from the header)
Block reward	2 SCDO	3 SCDO
Mean block interval (measured)	11.54 s	28.7 s (blocks 9,262,000 to 9,262,700)
Pool software	open-etc-pool-friends with SCDO additions	zpool v0.4.0 (Go), with the zminer CPU miner
Miner connection	Stratum, <code>stratum+tcp://82.223.19.88:3333</code> ; GPU miners such as Rigel and lolMiner	Newline-delimited JSON-RPC on TCP 3341 (<code>login</code> , <code>getjob</code> , <code>submit</code> , server-pushed jobs); HTTP <code>getWork</code> on port 8341
Miner identity	The miner's own 0x address, optionally <code>0x...r.igname</code>	The miner's Shard1 address (<code>1501...</code>); logins from other shards are refused
Public data	Pool web page with every miner, block and payout; no IP addresses shown	<code>/api/stats</code> , <code>/api/blocks</code> , <code>/api/payments</code> , <code>/api/miner/<address></code>

Miners need only a public address; no account, password or key is shared with the pool. Since 2026-09-29 the SCDO project’s own rigs have mined on the Shard0 pool, and the Classic Shard1 pool has made live on-chain payouts since 2026-10-03 18:23 (UTC+8).

3.2 Hashrate-tiered fees

Both pools use three fee tiers chosen by the miner’s measured hashrate (Table 2). The rate falls as hashrate rises.

Table 2: Fee tiers.

Tier	Shard0 (Ethash)	Classic Shard1 (zpow)
Small	below 100 MH/s: 5 %	below 5 kH/s: 5 %
Medium	100 to 500 MH/s: 3 %	5 to 50 kH/s: 3 %
Large	500 MH/s and above: 1 %	50 kH/s and above: 1 %
Hashrate used	average over the last 24 h of accepted share difficulty, re-evaluated for every block	average over the last 10 min, taken when the block matures

On Shard0, a new miner is placed in the small tier until 24 hours of history exist. The thresholds and rates are configuration values. The Classic thresholds were set on 2026-10-03 to match the hashrate of zpow CPU mining, so that a desktop CPU running about three or more zminer threads (at the measured per-thread rate) reaches the medium tier.

For each matured block, the Classic pool first divides the block reward among the miners in proportion to their PPLNS weights, then deducts each miner’s own tier fee from that miner’s part. Rounding remainders go to the Treasury. Fees go to the operator’s Treasury address on each shard (Shard0: 0xf65977d69f6a67e52d9d2ec5b9c4a67efa08e49b; Shard1: 15012ee61a554900ded2f765395c899ee17d02dc91).

Part of the fee revenue goes back into network infrastructure. Under the published SCDO node-reward rules (v1.1), 50 % of pool-fee revenue funds a reward pool for full-node operators, and the other 50 % goes to the Treasury. This leaves consensus unchanged and creates no new coins. Node rewards are settled hourly and paid on chain once a node has earned 1 SCDO; they have been paid live since 2026-10-03 19:12 (UTC+8). Node rewards will be analysed in a separate paper.

The schedule has two properties that hold by construction:

1. **Net income never falls as hashrate rises.** Let $f(h)$ be the fee rate for hashrate h . The net reward rate $h(1 - f(h))$ is non-decreasing in h : within a tier it grows linearly, and at each threshold the rate drops, so net income jumps up (Figure 3).
2. **Splitting does not pay.** If one miner divides its hashrate across several addresses, each address has less hashrate than the whole and therefore falls into the same tier or a higher-fee one. Splitting can never reduce the fee rate, so the schedule creates no reason to fragment identities.

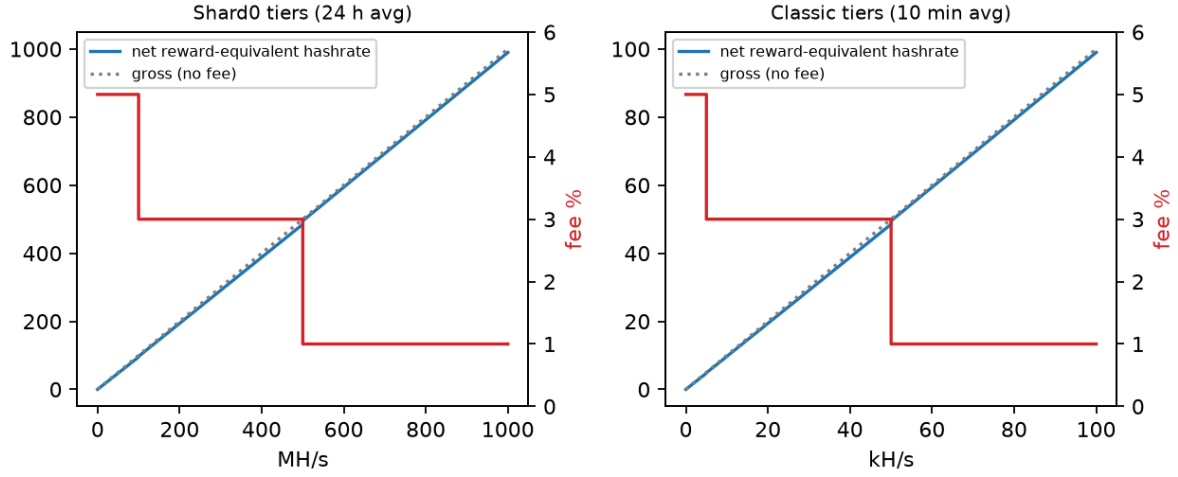


Figure 3: Fee tiers (right axis, red) and the net reward-equivalent hashrate $h \cdot (1 - f(h))$ (left axis) for Shard0 (left) and Classic Shard1 (right). Net income is non-decreasing in hashrate.

3.3 PPLNS settlement

Let R be the reward of a matured block, w_i the weight of miner i 's shares in the PPLNS window, $W = \sum w_i$ the total window weight, and f_i miner i 's fee rate. Miner i is credited

$$P_i = R \cdot (w_i / W) \cdot (1 - f_i),$$

and the pool fee for the block is $\sum_i R \cdot (w_i / W) \cdot f_i$ plus rounding remainders.

The two pools define the window differently.

- **Shard0: a fixed amount of work.** The window is 1,000 units of 10^6 share difficulty, i.e. 10^9 in total. With the current network difficulty of about 6.39×10^8 , this is about 1.56 blocks' worth of work.
- **Classic Shard1: a window that scales with difficulty.** The window is twice the expected work of the current block (`pplns_factor` = 2), so it keeps the same size relative to a block as difficulty changes.

3.4 Expected-hash share weighting for zpow

On an Ethash chain, the difficulty of a share equals the expected number of hashes behind it, so counting difficulty measures work. For zpow this does not hold: a share is valid when the determinant of the header-derived matrix reaches a target, and how fast the probability of reaching a target falls depends on the tail of the determinant distribution. If the pool weighted shares by nominal difficulty, miners whom vardiff had placed at different levels would be paid unequally.

zpool therefore weights each share by the number of hashes it is expected to cost. We sampled the determinant distribution 1.2 million times on real Shard1 block headers and built an eight-level table that maps share difficulty to expected hashes. Examples: difficulty 246,345 corresponds to 100 expected hashes, 656,260 to 1,000, 1,324,893 to 10,000, 2,461,372 to 100,000 and 3,244,828 to 300,000. Between levels the pool interpolates on a logarithmic scale; the same function gives the expected work of a block (about 177,340 hashes at the recent block difficulty of 2,869,924). Vardiff aims for one share every 10 seconds per miner, and jobs are refreshed every 20 seconds. Because shares are weighted by expected work, a miner's PPLNS weight does not depend on the vardiff level it was assigned.

The pool recomputes the zpow check for every submitted share. It rejects stale or unknown jobs (error 21), duplicate nonces (22) and shares below the share target (23). It hands out a job only when the block header's creator is the pool address, so every block found pays the pool and is then shared out by PPLNS.

3.5 Maturity, payouts and hot-wallet controls

Table 3: Settlement and payout parameters.

Parameter	Shard0	Classic Shard1
Maturity before crediting	64 blocks	12 confirmations
Payout cycle	every 10 minutes (median gap between the last 50 payouts: 610 s)	every 600 s
Minimum payout	1 SCDO	1 SCDO
Network fee	paid by the pool (recorded transaction cost to the miner: 0)	paid by the pool (gas price 10 Wen, 21,000 gas)

The Classic pool adds several controls on its hot wallet:

1. **Hot-wallet cap.** The pool wallet holds at most 50 SCDO beyond what it owes miners. Fees and any excess are swept to the shard Treasury. Funds set aside for node rewards are kept in the hot wallet and are never swept.
2. **One round at a time.** Every transaction sent is recorded as pending until it appears in a block. While any transaction is pending, the pool starts no new payout or sweep round, so nonces cannot collide and a restart cannot pay twice. A transaction not included within 30 minutes raises an alert and is not resent automatically.
3. **Debit before send.** A miner's balance is debited before the transfer is sent and restored if sending fails.
4. **Key handling and portability.** The signing key never enters the code repository, and the pool refuses to start live mode unless the key matches the configured pool address. One install script deploys the pool, which makes it easy to move between servers.
5. **Small footprint.** The pool process uses about 0.1 to 0.2 % CPU and about 18 MB of memory.

4. Deployment and Production Records

All figures in this section were read from the public interfaces and the block explorer on 2026-10-03 at 21:07 (UTC+8). The production data so far come mainly from mining hardware operated by the SCDO project, so the behaviour of many independent miners is studied by simulation in Section 5.

4.1 Shard0 pool

The Shard0 pool runs open-etc-pool-friends with the SCDO additions described above.[8] Miners connect with standard Ethash software to `stratum+tcp://82.223.19.88:3333`. The pool web page shows every miner, every block and every payout, linked to the explorer, and never shows IP addresses.

- **Blocks.** The pool has found 28,823 blocks, of which 28,754 had matured. Over the last 64, 128 and 256 blocks the orphan and uncle rates were zero, and the pool's luck was 1.028, 1.100 and 1.027.
- **Payouts.** The pool has made 587 automatic payouts. In the latest 50, the median gap was 610 s and amounts ranged from 2.75 to 126.00 SCDO (median 92.0 SCDO, total 3,996.04 SCDO).

- **Miners.** The active hardware is two home graphics cards operated by the project: an RTX 5060 Ti (about 51.7 MH/s) and a GTX 940MX (about 1.8 MH/s). The pool hashrate has been about 53 to 58 MH/s, with a 24-hour average of 54.2 MH/s for the project address `0x42c4854a...c0e4`. That address has received 57,339.75 SCDO in 574 payouts. A second miner address (`0xb51199e0...4381`) found 68 blocks and received 128.13 SCDO in 5 payouts; its last share was on 2026-09-29 at 13:42 (UTC+8). The fee address (the Shard0 Treasury) has received 40.25 SCDO in 8 payouts.
- **On-chain check.** Payout transaction `0xfd625825fb013d212eb412dca0fa727a5559e3515cc1dd20bf480aaf129d75b3` (Shard0 block 49,065, 2026-10-03 20:57 UTC+8) transferred 106.400399 SCDO from the pool wallet `0x675e8639...8d24` to the miner address `0x42c4...c0e4`, with receipt status 1.

4.2 Classic Shard1 pool (zpool)

zpool v0.4.0 is written in Go. It shares the general architecture of open-etc-pool-friends but its own code covers zpow verification, expected-hash weighting, the difficulty-scaled PPLNS window and the hot-wallet controls. Miners connect with zminer, a multi-threaded command-line CPU miner for Linux and Windows. zminer reaches about 2,500 H/s on one thread and about 2,130 H/s per thread with four threads on our test CPU. At the recent block difficulty and the measured 28.7 s block interval, Shard1's network hashrate is about 6.2 kH/s, so a single four-thread home CPU is a large share of the network.

- **Test run.** On 2026-10-03 from 17:50 to 18:03 (UTC+8), one CPU miner submitted 174 accepted shares, 6 stale shares and no invalid shares. It found 15 block solutions; the 13 that joined the main chain show the pool address as creator and were credited through PPLNS at the 5 % small-miner rate.
- **On-chain check.** Transaction `0xc481b390fd8eeef9e7420bb7d9be5bed21cc79ccab9578e7496363a964c6afa4` (Shard1 block 9,262,425) settled the test period by transferring 38.99 SCDO from the pool hot wallet `1501e99d...0651` to the Shard1 Treasury `15012ee61a554900ded2f765395c899ee17d02dc91`.
- **Live mode.** Live payouts have been enabled since 2026-10-03 18:23 (UTC+8); the pool runs a payout cycle every 10 minutes. The pools are in early deployment, and external-miner data will be reported in a future version.

5. Fairness Simulation

5.1 Method

We simulate the share stream exactly. Every share represents the same amount of work. A share belongs to miner i with probability h_i/H , where h_i is the miner's hashrate and H the total, and it solves a block with probability q , the ratio of share work to block work. When a block is found, its reward is divided in proportion to the shares in the most recent PPLNS window, and each miner's tier fee is deducted. We assume difficulty keeps the mean block interval at the measured value (11.54 s on Shard0, 28.7 s on Classic Shard1). The solo baseline is the same miner mining alone against the same difficulty. Its blocks then arrive as a Poisson process, and the CV of its income over a period with λ expected blocks is $1/\sqrt{\lambda}$.

The model leaves out orphan and uncle blocks, network latency and stale shares. It keeps hashrates constant and fixes each miner's tier from its nominal hashrate (the live pools re-evaluate tiers from 24-hour or 10-minute averages). On Classic Shard1, all shares are given one size of 10,000 expected hashes, which is one of the real vardiff levels. The code is written in Python with NumPy 2.5.3 and Matplotlib 3.11.2 and uses fixed random seeds. `python3 run_all.py && python3 plots.py` reproduces every number and figure in about 20 seconds.

5.2 Scenarios

Parameters come from the live configurations: block rewards, block intervals, fee tiers, PPLNS windows ($1,000 \times 10^6$ on Shard0, twice the block work on Classic Shard1) and the 1 SCDO payout threshold. The rates of individual devices are measured. **The miner populations in S0-B and C1 are assumed scenarios, not observations.**

Table 4: Simulation scenarios (30 simulated days each).

Scenario	Miners	Total hashrate	Window / block work	Blocks
S0-A (measured devices, Shard0)	RTX 5060 Ti 51.7 MH/s and GTX 940MX 1.8 MH/s, treated as two miners	53.5 MH/s	1.62	224,898
S0-B (community, Shard0)	50 home GPUs (log-normal, median 30 MH/s, $\sigma = 0.8$, limited to 1.8–99 MH/s; drawn values 6.3–99 MH/s), 4 medium miners (150, 250, 350, 450 MH/s) and 1 farm (1,000 MH/s)	4.07 GH/s	0.021	224,990
C1 (community, Classic Shard1)	20 home CPUs (six with 1 thread, six with 2, five with 4, three with 8, at 2,130 H/s per thread) and 1 farm of 30 threads (63.9 kH/s)	196 kH/s	2.0	89,946

5.3 Results

Table 5: Simulation results.

Metric	S0-B (Shard0)	C1 (Classic Shard1)
Gross reward share ÷ hashrate share, all miners	0.9978 – 1.0030	0.9978 – 1.0018
Smallest home miner	6.3 MH/s GPU	1-thread CPU, 2.13 kH/s
Hourly income CV, pool vs solo	0.075 vs 1.44	0.098 vs 0.86
Hours with no income, pool vs solo	0 % vs 61.7 %	0 % vs 25.6 %
Mean time between own blocks if solo	2.07 h	0.73 h
Time to reach the 1 SCDO payout threshold in the pool	1.08 h	0.26 h
Small tier: hashrate share → net reward share	45.90 % → 45.14 %	19.57 % → 19.11 %
Medium tier: hashrate share → net reward share	29.51 % → 29.65 %	47.83 % → 47.70 %
Large tier: hashrate share → net reward share	24.59 % → 25.21 %	32.61 % → 33.19 %

Metric	S0-B (Shard0)	C1 (Classic Shard1)
Pool fees as a share of block rewards	3.43 %	2.74 %

Proportionality. On both shards, each miner’s share of gross rewards was within 0.3 % of its share of hashrate (Figure 1). This held for both windows tested: the Shard0 window of a fixed amount of work and the Classic window scaled to the block, and for both difficulty-weighted Ethash shares and expected-hash-weighted zpow shares.

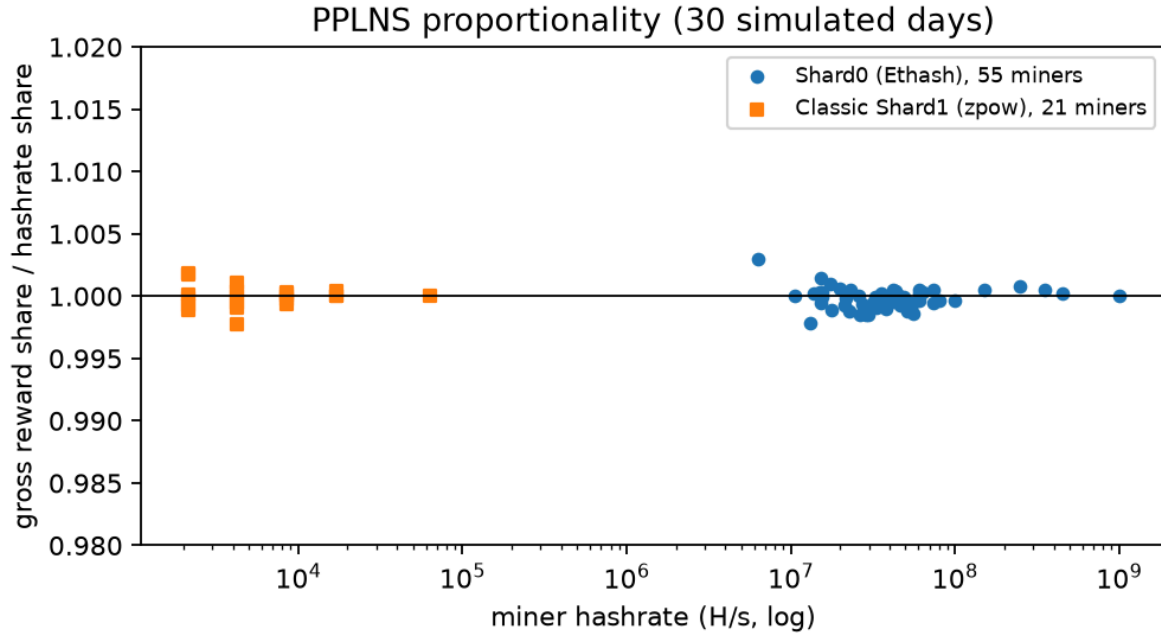


Figure 1: Each miner’s gross reward share divided by its hashrate share, plotted against hashrate, for S0-B (Shard0) and C1 (Classic Shard1) over 30 simulated days. All values lie close to 1.

Income stability. Joining the pool makes home miners’ income far steadier (Figure 2). For the smallest simulated GPU on Shard0 (6.3 MH/s), the CV of hourly income fell from 1.44 when mining alone to 0.075 in the pool, a factor of about 19. For a single-thread CPU on Classic Shard1, it fell from 0.86 to 0.098, about ninefold. Solo, these miners would have no income in 61.7 % and 25.6 % of hours; in the pool, no hour was without income. In S0-A, the measured devices on today’s Shard0 network, the 1.8 MH/s GTX 940MX had an hourly CV of 0.057 in the pool against 0.31 solo. Inside the pool, every miner’s CV approaches the pool’s own floor of about $1/\sqrt{B_h}$, where B_h is the number of blocks the pool finds per hour (about 312 on Shard0, giving 0.057). In other words, once in the pool even the smallest home miner gets nearly the stability of the whole pool. In the pool, the smallest home miners reached the 1 SCDO payout threshold in about an hour on Shard0 and in about a quarter of an hour on Classic Shard1 (Figure 4).

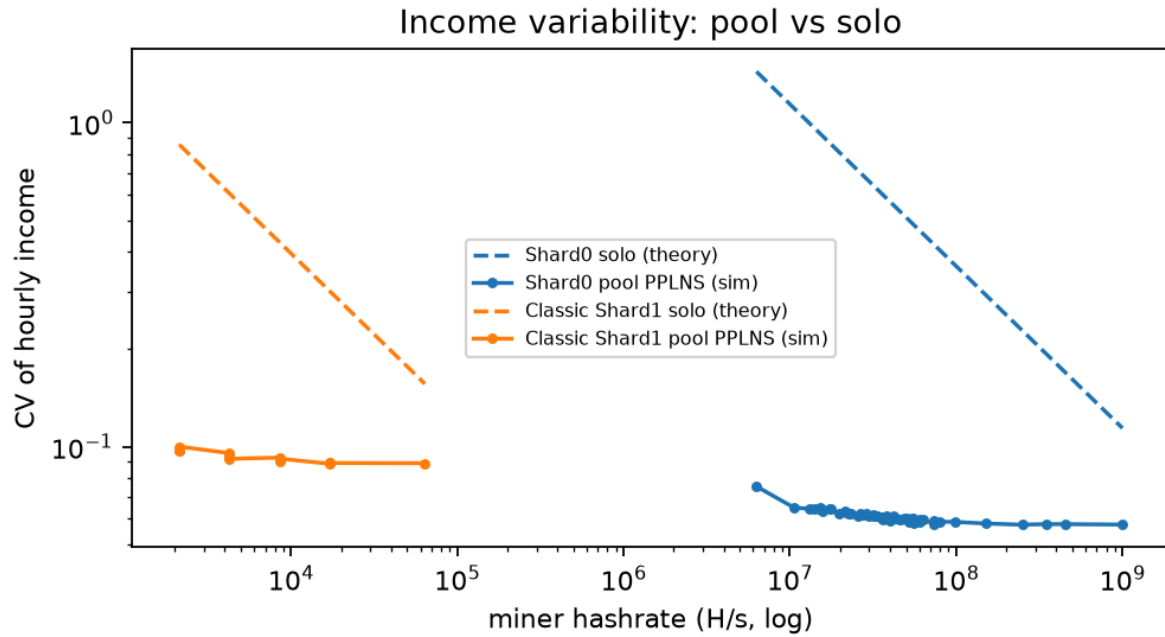


Figure 2: CV of hourly income against miner hashrate: pool (simulated, solid) and solo (theoretical, dashed), for Shard0 and Classic Shard1. Both axes logarithmic.

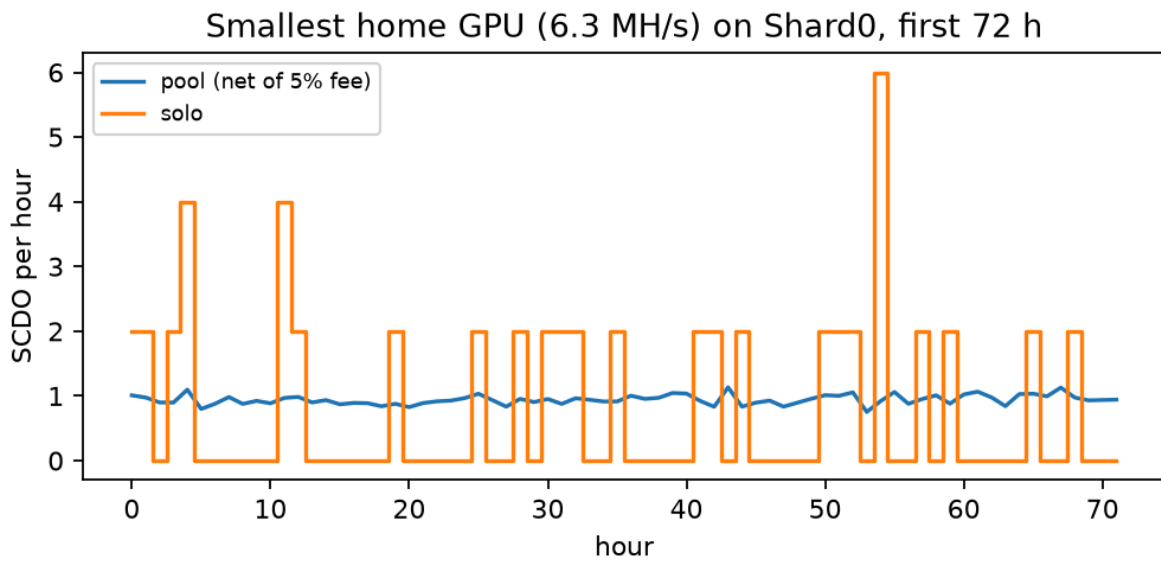


Figure 4: Hourly income of the smallest home GPU in S0-B (6.3 MH/s) over the first 72 simulated hours: pool income after the 5 % fee, and one simulated solo run.

Fees by tier. Small miners pay the highest rate, 5 %, and in exchange receive the large drop in income variability shown above. Large miners pay 1 %. As a result, each tier's share of net rewards differs from its hashrate share by less than one percentage point: the small tier receives 45.14 % of net rewards for 45.90 % of hashrate on Shard0 (19.11 % for 19.57 % on Classic), and the large tier receives 25.21 % for 24.59 % (33.19 % for 32.61 %). Total fees came to 3.43 % of block rewards on Shard0 and 2.74 % on Classic Shard1, and under the node-reward rules half of that revenue goes to full-node operators.

6. Discussion

The trade-off for home miners. For a single home GPU or CPU, the main obstacle to mining is unpredictable income, not its expected value. The pools turn hours-long gaps between rewards into steady payouts every ten minutes, with no network fee, no account and no key handed over. The cost to a small miner is the 5 % top rate. The volume discount brings larger hashrate into the same pool, which raises the pool's block rate. Since a pool member's income stability is set by the pool's block rate, larger miners joining also make home miners' income steadier. The two properties in Section 3.2 mean that the schedule rewards growing a miner's hashrate and never rewards splitting it.

Adjustable settings. Fee rates and tier thresholds are configuration values, so operators can follow changes in network hashrate without changing code. The Classic pool did this on its first day, when its thresholds were set for zpow CPU hashrates.

Window length. The Classic window scales with block difficulty (twice the block work), so it stays the same size relative to a block as the network grows. The Shard0 window is a fixed 10^9 of share difficulty, about 1.6 blocks of work on the current network; in the larger S0-B community it would cover 0.021 of a block. In both cases the simulation shows proportional rewards. A Shard0 window that scales with difficulty, as on Classic, is a natural configuration option as the network grows.

Scope. The simulation assumes constant hashrates and leaves out miner churn, stale shares and reorganisations. The production records so far come mainly from operator-run hardware. The pools are in early deployment, and external-miner data will be reported in a future version. The zpow expected-hash table rests on 1.2 million samples and can be refined with more.

7. Conclusion and Future Work

We have described two pools in production on the SCDO sharded PoW platform that combine PPLNS settlement with hashrate-tiered fees: one for Ethash GPUs on Shard0 and one, zpool, for zpow CPUs on Classic Shard1. zpool weights every share by its measured expected hash count, so miners are paid fairly across vardiff levels despite zpow's non-linear relation between difficulty and work. Both pools pay automatically on chain every ten minutes from 1 SCDO and publish every block and payout. The fee schedule keeps net income non-decreasing in hashrate and gives no reason to split identities, and half of fee revenue goes to full-node operators. A simulation based on the live configurations shows proportional rewards to within 0.3 %, and for the smallest home miners about a ninefold to nineteenfold reduction in the variability of hourly income, with no hours left without income.

Future work:

1. Report production records with independent miners on both pools in a future version as deployment expands.
2. Offer a Shard0 PPLNS window that scales with difficulty, as on Classic Shard1.
3. Extend the simulation to miner churn, stale shares and reorganisations.

8. Declarations

Use of AI tools. The initial English draft was generated by Doubao (ByteDance), a large language model chat assistant, from a structured writing brief prepared by the author. An AI research assistant verified the facts against the pools' public interfaces, the block explorer and the source code; wrote and ran the simulation and

produced the figures; checked the references against their sources; and revised the text. The author reviewed, edited and fact-checked the whole manuscript and takes full responsibility for its content.

Conflict of interest. The author is affiliated with SCDO Laboratory, 9Y9 PTY LTD, which operates both pools and the Treasury addresses described here. The author holds SCDO coins. Pool fees go to the operator's Treasury, half of them to the node-reward pool. The production data in Section 4 come mainly from mining hardware operated by the project.

Funding. This work received no external funding.

Data and code availability. The simulation code, its parameters, the results and a snapshot of the live data are included as supplementary files with this record. All pool records can be checked through the pools' public interfaces and the block explorer at <https://scdoscan.io>.

Author contributions. YWZ is the sole author, designed and operates the pools, and is responsible for the analysis and the manuscript (100 %).

References

- [1] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," 2008. <https://bitcoin.org/bitcoin.pdf>
- [2] M. Rosenfeld, "Analysis of Bitcoin Pooled Mining Reward Systems," arXiv:1112.4980, 2011. <https://arxiv.org/abs/1112.4980>
- [3] Y. W. Zhang, "SCDO Agent Ledger: On-Chain Identity, Work Attestation and Settlement for Autonomous AI Agents on a Sharded PoW Blockchain," Zenodo preprint, 2026. <https://doi.org/10.5281/zenodo.23119810>
- [4] O. Schrijvers, J. Bonneau, D. Boneh and T. Roughgarden, "Incentive Compatibility of Bitcoin Mining Pool Reward Functions," in *Financial Cryptography and Data Security (FC 2016)*, LNCS 9603, Springer, 2017, pp. 477–498. https://doi.org/10.1007/978-3-662-54970-4_28
- [5] Y. Lewenberg, Y. Bachrach, Y. Sompolinsky, A. Zohar and J. S. Rosenschein, "Bitcoin Mining Pools: A Cooperative Game Theoretic Analysis," in *Proc. AAMAS 2015*, pp. 919–927. <https://www.ifaamas.org/Proceedings/aamas2015/aamas/p919.pdf>
- [6] I. Eyal and E. G. Sirer, "Majority Is Not Enough: Bitcoin Mining Is Vulnerable," in *Financial Cryptography and Data Security (FC 2014)*, LNCS 8437, Springer, 2014, pp. 436–454. https://doi.org/10.1007/978-3-662-45472-5_28
- [7] M. Romiti, A. Judmayer, A. Zamyatin and B. Haslhofer, "A Deep Dive into Bitcoin Mining Pools: An Empirical Analysis of Mining Shares," Workshop on the Economics of Information Security (WEIS 2019); arXiv:1905.05999. <https://arxiv.org/abs/1905.05999>
- [8] open-etc-pool-friends, open-source mining pool software codebase family. <https://github.com/yuriy0803/open-etc-pool-friends>
- [9] G. Wood, "Ethereum: A Secure Decentralised Generalised Transaction Ledger" (Yellow Paper). <https://ethereum.github.io/yellowpaper/paper.pdf>