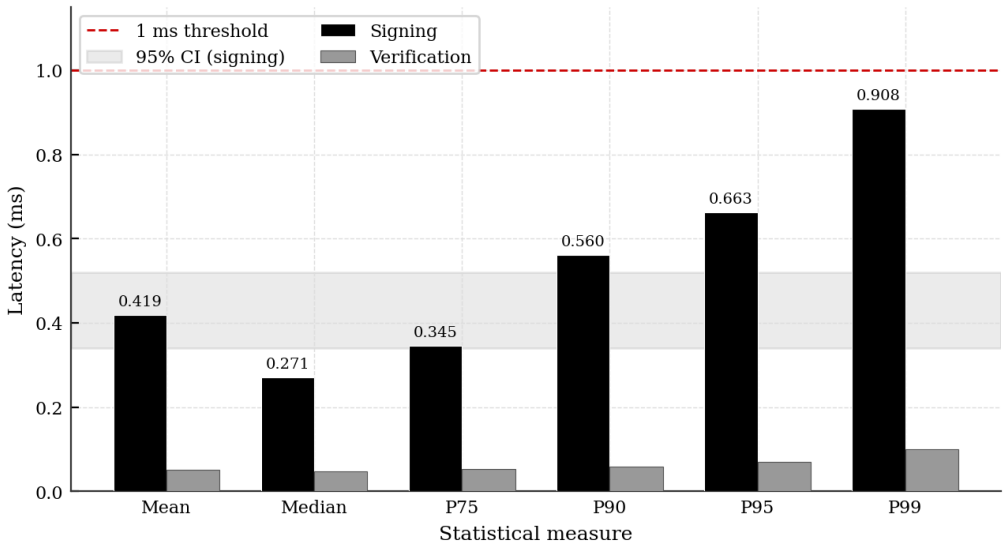
Supplementary Information for: Implementation and Evaluation of a Dilithium-Based Post-Quantum Blockchain Prototype

Ahmed Abdellatif[1,*], Eman K. Elsayed[1], Alaa Zaghloul[1], Khaled Abd El Salam[1,2]

[1]Department of Computer Science, College of Information Technology, Misr University for Science and Technology, 6th of October City, Giza, Egypt
[2]Department of Electrical Engineering, Suez Canal University, Ismailia, Egypt
*Corresponding author: ahmed.lotfy@must.edu.eg

This file follows the order of the main Results and provides the supporting distributions, scenario definitions, and reproducibility details cited at the relevant points in the article.

Original prototype baseline



Supplementary Figure S1: Legacy baseline latency. Signing and verification latency in the original $n = 1{,}000$ four-thread Dilithium2 run; the shaded band is the preserved 95% confidence interval and the dashed line marks 1 ms.

Supplementary Table S1: Dilithium2 latency statistics in the original single-machine prototype.

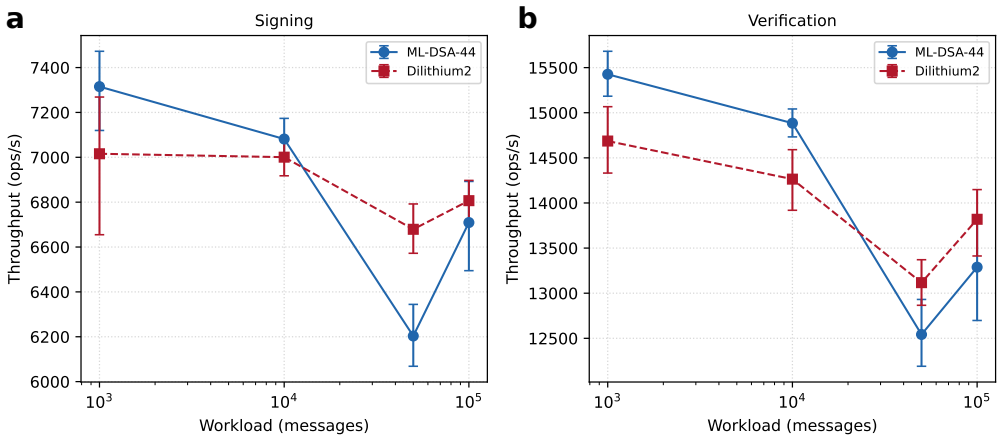
Statistic	Sign (ms)	Verify (ms)
Mean	0.4186	0.0507
Median	0.2712	0.0470
P75	0.4030	0.0504
P90	0.5605	0.0589
P95	0.6631	0.0690
P99	0.9080	0.0996
95% CI (mean)	[0.339, 0.519]	[0.050, 0.051]
Std. dev.	1.5283	0.0107

Supplementary Table S2: Original $n = 1{,}000$ scale-test results with four threads.

Metric	Value
Sign throughput	8,541.5 signatures/s
Verify throughput	18,714.4 verifications/s
P95 sign latency	0.663 ms
P95 verify latency	0.069 ms
Combined mean service time (sign + verify)	0.469 ms
Valid signatures	1,000 / 1,000

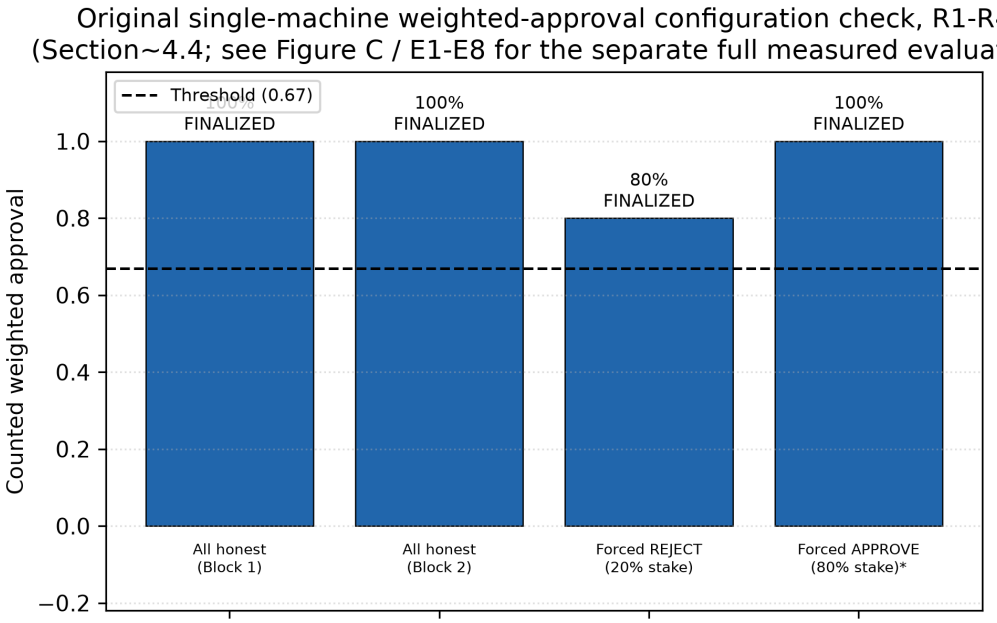
The combined mean service time is the sum of separately measured signing and verification means, not a separately timed end-to-end request path.

Workload scaling



Supplementary Figure S2: Single-worker workload scaling. (a) Signing throughput from 1,000 to 100,000 messages for ML-DSA-44 and legacy Dilithium2. (b) Verification throughput over the same workloads. Bars show 95% bootstrap confidence intervals.

Weighted-approval support



Supplementary Figure S3: Original R1–R4 weighted-approval checks.

Supplementary Table S3: Original R1–R4 weighted-approval configuration check.

Round	Scenario	Approval	Verdict
R1	All validators honest (Block 1)	100%	FINALIZED
R2	All validators honest (Block 2)	100%	FINALIZED
R3	1 faulty validator, forced REJECT (20% stake)	80%	FINALIZED
R4	Forced-decision check: A+B forced APPROVE (80% stake)	100%	FINALIZED

R4 uses a valid block and is therefore a configuration check, not evidence that an invalid block can be finalized. E5 in Supplementary Table S4 tests that boundary directly.

Supplementary Table S4: Measured E1–E8 weighted-approval configurations at the 0.67 acceptance threshold ($V_A = 0.50$, $V_B = 0.30$, $V_C = 0.20$).

Scenario	**Block**	V_A **message**	V_B **message**	V_C **message**	**Counted approval**	**Threshold**	**Outcome**
E1	Valid	APPROVE	APPROVE	APPROVE	1.00	MET	ACCEPT
E2	Mechanically invalid	REJECT	REJECT	REJECT	0.00	NOT MET	REJECT
E3	Valid	APPROVE	APPROVE	REJECT	0.80	MET	ACCEPT
E4	Mechanically invalid	REJECT	REJECT	APPROVE	0.20	NOT MET	REJECT
E5	Mechanically invalid	APPROVE	APPROVE	REJECT	0.80	MET	ACCEPT
E6	Valid	APPROVE	APPROVE	APPROVE	0.70	MET	ACCEPT
E7	Valid	APPROVE	REJECT	REJECT	0.50	NOT MET	REJECT
E8	Valid	APPROVE+REJECT	APPROVE	APPROVE	0.50	NOT MET	REJECT

Each scenario contains 20,000 measured rounds combined across both algorithms (10,000 per algorithm). In E5, V_A and V_B are forced to approve a block with a mechanically invalid stored Merkle root. In E6, V_B signs an approval bound to a different block hash and is excluded. E7 submits the same valid V_A approval twice, so it is counted once. In E8, V_A signs both APPROVE and REJECT and is excluded as equivocating. ML-DSA-44 and Dilithium2 produced identical logical outcomes in all scenarios.

Security-control support

Supplementary Table S5: Toy-scale cryptographic illustrations and signature-rejection checks.

Experiment	Target	Outcome
Toy factoring	RSA-32	Key recovered (toy)
Toy factoring	RSA-48	Key recovered (toy)
Toy factoring	RSA-64	Key recovered (toy)
BKZ-40	Dilithium2	Toy-scale reduction exercise
EUF-CMA $\times 4$	Dilithium2	All rejected

BKZ-40 is retained only as a toy-scale reduction exercise. It was not an attack on production Dilithium2 parameters and is not used to estimate a security margin.

Scope of the toy-scale illustration

The RSA-32/48/64 and BKZ-40 items are deliberately toy-scale demonstrations and are not used to estimate production security. The four forgery trials test implementation-level rejection of invalid signatures only. None of these exercises is treated as primary security evidence in the main article.

Supplementary Table S6: Outcomes of the original 550-input adversarial suite.

Attack	n	Blocked	Speed	Layer
Wrong-election-ID payload mutation	5	5	5,775/s	PQC (payload integrity)
Vote flood	500	500	16,336/s	PQC
Sig stripping	15	15	13,392/s	App
Bit corrupt	25	25	11,173/s	Hash+PQC
Sig swap	5	5	5,768/s	PQC
TOTAL	550	550	–	–

The row was originally labeled "Replay." Source inspection confirms that the test mutates the signed `election_id` before verification, so rejection reflects payload integrity rather than a dedicated replay or election-context control.

Supplementary Table S7: Phase F security-control outcomes by scenario.

Scenario	Test	Control origin	Expected	Observed	Correct handling	Rate
F0	Benign control	Legacy reused	ACCEPT	ACCEPT	20,000/20,000	100%
F1	Duplicate valid vote (identical resubmission)	Legacy reused	REJECT	REJECT	20,000/20,000	100%
F2	Conflicting valid votes from one voter	Legacy reused	REJECT	REJECT	20,000/20,000	100%
F3	Cross-voter signature reuse	Legacy reused	REJECT	REJECT	20,000/20,000	100%
F4	Signed timestamp modification	Legacy reused	REJECT	REJECT	20,000/20,000	100%
F5	Unmodified cross-election replay	Experimental safeguard	REJECT	REJECT	20,000/20,000	100%
F6	Polling-window / clock-skew violation	Experimental safeguard	REJECT	REJECT	20,000/20,000	100%
F7	Unauthorized voter	Experimental safeguard	REJECT	REJECT	20,000/20,000	100%
F8	Claimed-voter / key mismatch	Mixed	REJECT	REJECT	20,000/20,000	100%
F9	Isolated Merkle-root tampering	Legacy reused	REJECT	REJECT	20,000/20,000	100%
F10	Isolated previous-block-hash tampering	Legacy reused	REJECT	REJECT	20,000/20,000	100%

Each Phase F scenario contains 20,000 measured rounds combined across both algorithms. For F5–F7, source inspection showed that the corresponding safeguards were absent from the original implementation; legacy-path behavior is therefore characterized from code inspection rather than from separate measured legacy-only trials, and the reported rejection outcomes come from the added safeguards. F8 is the measured exception: the reproduced pattern is localized to the standalone legacy function `03_verify.py::verify_vote`, which used the caller-supplied voter identifier for public-key selection without explicitly comparing it with the identifier in the signed payload. This pattern accepted all 20,000 identity-mismatch attempts (0/20,000 detected), whereas the added identity-equality safeguard rejected all 20,000 corresponding attempts. The internal `04_blockchain.py::VotingChain._verify_vote` method derives the voter identifier from the signed payload and does not expose the identical caller-supplied mismatch interface. Rates apply only to the defined scenarios.

Supplementary Methods

Phase D2 concurrency validation

Phase D2 throughput, latency, speedup, and efficiency were recomputed from the raw repetition-level JSON records and checked against the final archived summaries.

Phase H defect discovery and revalidation

Phase H exposed one pre-fix deadline-handling error: a response arriving after the decision deadline was marked late but still entered the approval calculation in 1 of 8,000 measured rounds. After the response-absorption logic was corrected, 11 focused unit tests passed and a targeted 2,400-round re-measurement produced no outcome mismatches. Both the pre-fix data and the post-fix revalidation are preserved.

Bootstrap methodology

Confidence intervals for the extended evaluation use a percentile bootstrap with 2,000 resamples and seed 42 on repetition-level observations. The original $n = 1{,}000$ prototype baseline is the exception: because independent repetitions were not available, its preserved interval is an operation-level bootstrap within that run. Algorithms were resampled independently when their repetitions were not paired.