Trust per Joule: The Thermoeconomics of Proof-of-Work Settlement

Matthew Long
The YonedaAI Collaboration
YonedaAI Research Collective
Chicago, IL
matthew@yonedaai.com · https://yonedaai.com

23 July 2026

Abstract

Proof-of-work settlement converts electricity, specialized hardware, network communication, and miner incentives into a particular kind of assurance. The conversion is real, but aggregate energy is not a sufficient statistic for its output. Two networks can consume the same number of joules while differing in honest hashrate, attacker-accessible hardware, ownership concentration, electricity prices, block propagation, reward exposure, or the value available from an attack. Their settlement risks and attack margins can therefore differ. This paper develops a scenario-conditioned accounting framework for that claim. It separates full-node validation from mining, defines a threat-model tuple, and reports assurance as a vector rather than a universal scalar. The vector includes confirmation depth, effective adversarial share, a stated catch-up model, a finite settlement window, attack cost, attack benefit, and expected protected loss. A constructive theorem shows that equal operational energy can coexist with different reorganization risk. A second construction shows that equal energy and hashrate can coexist with different economic attack margins. The numerical companion reproduces the catch-up approximation in the Bitcoin whitepaper, implements a finite-window random-walk calculation, builds confirmation frontiers, and tests monotonicity and equal-energy counterexamples. Bitcoin Core v31.1 is inspected at commit `9be056a8a72b624dae9623b2f7bded92c2a21c91`. That source confirms that proof of work is checked at the block header, while full nodes separately enforce transaction, script, unspent-output, reward, and other consensus rules. The proposed "trust per joule" terminology is therefore a conditional accounting shorthand. It is not a claim that energy creates truth, validates arbitrary transactions, or supplies one context-free measure of Bitcoin security.

Keywords: proof of work; Bitcoin; settlement assurance; energy accounting; confirmations; double spending; attack cost; validation.

Evidence status. Definitions and algebraic constructions are PROVED. Results that hold under a declared adversarial and stochastic model are CONDITIONAL. Reference calculations and tests are COMPUTATIONAL. Questions not identified by the declared data are OPEN. No live-network security estimate is presented.

Contents

1 The claim and its boundary

Proof of work is often described by pointing to electricity consumption. The description is incomplete. Electricity powers hashing, and hashing makes block production costly under the protocol's difficulty rule. Settlement assurance also depends on who can obtain compatible hardware, how much productive hashrate each side can deploy, how blocks propagate, which consensus rules independent nodes enforce, how long a recipient waits, what rewards an attacker can recover, and what value the attacker expects from success. A joule enters a physical process. Assurance emerges only through a protocol and a threat model.

The paper asks a narrow question: under what declarations can operational energy be related to economically meaningful settlement outcomes? The proposed answer is not a single coefficient. It is a scenario-conditioned frontier. For a scenario $\mathcal{S}$ and confirmation depth z, the frontier records

$$\mathcal{F}(\mathcal{S}, z) = \left(E_H, q_{\text{eff}}, \mathcal{R}_N, \mathcal{R}_T, C_A^{\text{gross}}, C_A^{\text{net}}, M_A, \mathbb{E}[L_P]\right).$$

The components are honest operational energy, effective adversarial share, a named recipient-risk approximation, finite-window reorganization risk, gross and net private attack cost, attack margin, and expected protected loss. The tuple prevents one quantity from borrowing the interpretation of another.

This treatment is thermoeconomic in a restrained sense. It follows physical energy through a conversion chain and then attaches explicit economic functionals to selected outputs. It does not identify energy with economic value. It also does not claim that Bitcoin's security reduces to a conventional heat-engine efficiency. Hashing is a protocol-selected signal whose economic meaning depends on validation, competition, incentives, and timing.

1.1 Contributions

The analysis makes five contributions.

1. It gives a source-grounded account of what Bitcoin Core checks when a node receives a block. The source separates proof-of-work verification, contextual block checks, transaction and script validation, and candidate chain selection.
2. It defines a scenario contract that names confirmations, settlement window, adversarial share, hardware access, energy intensity, propagation loss, electricity price, rental cost, rewards, attacker benefit, and protected loss.
3. It proves that aggregate energy is not sufficient to identify either hashrate-based reorganization risk or private attack margin.
4. It proposes a vector-valued assurance frontier and explains when a narrower avoided-loss-per-joule statistic can be formed.
5. It supplies a standard-library Python model and tests for the whitepaper approximation, finite-window catch-up, costs, monotonicity, confirmation frontiers, and equal-energy counterexamples.

1.2 Nonclaims

Several tempting claims are outside the result.

- The paper does not estimate Bitcoin's current total electricity use.
- It does not estimate the probability of a successful attack on the current network.
- It does not treat a transaction as consuming a pro rata share of network energy.
- It does not equate confirmations with legal finality or guaranteed irreversibility.
- It does not claim that more energy makes an invalid transaction valid.
- It does not derive censorship resistance, decentralization, monetary value, or social trust from a double-spend calculation.
- It does not claim that the Bitcoin whitepaper approximation is the exact probability for the deployed peer-to-peer network.
- It does not claim that private attack expenditure equals social cost.

These exclusions are part of the measurement result. Without them, "trust per joule" would acquire more meanings than its inputs can support.

2 Prior work and the proposed delta

2.1 Protocol security

Nakamoto describes a chain of proof of work and gives a Poisson approximation for the probability that an attacker catches up while a recipient waits for confirmations [1]. That calculation made confirmation depth legible, but it is an idealized race. Rosenfeld develops the underlying hashrate-based double-spend processes in more detail [3]. Grunspan and Pérez-Marco derive a corrected expression, prove exponential decay in confirmation count under their assumptions, and condition risk on observed validation time [4]. The present paper retains the whitepaper calculation only as a reproducible benchmark and labels it accordingly.

Bonneau and coauthors survey Bitcoin's technical security and economic incentives, organizing open problems across consensus, mining, networking, privacy, and related mechanisms [14]. Their breadth also shows why one double-spend calculation cannot serve as a total security score.

The Bitcoin Backbone analysis proves common-prefix and chain-quality properties in a formal execution model [2]. Its results are conditional on assumptions about honest mining power, rounds, message delivery, and the adversary. That structure supports the central position here: a security statement is indexed by a model, not by joules alone. Gervais and coauthors model how block interval, block size, mining power, and network delay affect proof-of-work security [7]. Decker and Wattenhofer measure block and transaction propagation in the Bitcoin network [8]. Croman and coauthors identify propagation and verification constraints in decentralized blockchain scaling [9].

Mining behavior is not exhausted by a majority threshold. Eyal and Sirer show how strategic block withholding can alter mining rewards [10]. Sapirshtein, Sompolinsky, and Zohar compute optimal selfish-mining policies for an explicit model [11]. These results do not

become energy coefficients. They show why behavior, connectivity, and reward rules belong in the threat scenario.

2.2 Economic security

Budish separates a miner zero-profit condition from an attack incentive constraint [5]. In his model, recurring payments to miners must be sufficiently large relative to the one-off value available from an attack. The important point for the present framework is structural: competitive expenditure and attack deterrence are related through assumptions about costs, rewards, and attacker benefit. Observed electricity expenditure does not reveal all three.

Carlsten and coauthors analyze instability when transaction fees replace the block subsidy as the main miner reward [6]. Their modeled strategies illustrate why the time profile and conditional recovery of rewards matter, not only total expenditure. Easley, O'Hara, and Basu study the joint evolution of fees and mining incentives [12]. Huberman, Leshno, and Moallemi analyze congestion and fees in a permissionless payment system [13]. These papers study different mechanisms, but all reject an account in which electricity alone determines economic security.

2.3 Energy estimates

The Cambridge Bitcoin Electricity Consumption Index describes a hybrid top-down method that combines network hashrate, an equipment basket, profitability assumptions, electricity price assumptions, and power usage effectiveness [15]. Its methodology emphasizes that actual electricity demand cannot be observed directly from the protocol and publishes lower, upper, and best estimates. This is appropriate measurement practice. Even a perfect estimate of operational electricity, however, would remain an input estimate. It would not identify attacker access, private benefit, or recipient exposure.

2.4 What is added here

The literature already contains energy estimates, double-spend models, formal protocol properties, network-delay analyses, and economic incentive constraints. The proposed delta is their accounting interface. It makes the required declarations explicit, proves an energy insufficiency result, keeps node validation distinct from mining, and connects the formal objects to an executable assurance frontier. This is a synthesis and measurement contribution, not a new consensus theorem.

3 What Bitcoin Core actually validates

Protocol interpretation should begin with the implementation rather than a metaphor. The source inspection in this section uses Bitcoin Core release v31.1. The signed annotated tag resolves to commit `9be056a8a72b624dae9623b2f7bded92c2a21c91` [16, 17]. Every source

link below is pinned to that commit. The release may later cease to be current, but the cited object will not change.

3.1 Header proof of work

In `src/pow.cpp`, `CheckProofOfWork` delegates to `CheckProofOfWorkImpl`. The implementation derives the target from the compact difficulty field, rejects negative, zero, overflowing, or above-limit targets, and rejects a header hash larger than the target [18]. This check is narrow. It establishes that the header meets the claimed work target under the configured consensus parameters.

3.2 Block and transaction validity

In `src/validation.cpp`, `CheckBlockHeader` invokes the proof-of-work check. The separate `CheckBlock` function performs context-independent validation. It checks the header, Merkle root, size limits, coinbase placement, transaction structure, and legacy signature-operation limits [19]. Later in block connection, `Consensus::CheckTxInputs` checks transaction inputs against the unspent-output view, sequence locks are enforced, script checks run, and the coinbase output is capped by subsidy plus fees [20]. The definition of `CheckTxInputs`, including missing or spent input, coinbase-maturity, value-range, and fee checks, is in `src/consensus/tx_verify.cpp` [21].

This separation matters economically. Miners search for headers and choose transactions to propose. Full nodes independently decide whether the resulting block satisfies consensus rules. A miner cannot spend someone else's output, create an excessive coinbase, or bypass a script merely by attaching more work to that invalid block. Nodes reject the block.

3.3 Work and chain candidates

The `GetBitsProof` function converts the encoded target into a per-block work value [23]. The candidate-selection function begins from the best work candidate and inspects the path to the active chain. Candidates with failed ancestors or missing data are not usable for activation [22]. The phrase "most-work chain" therefore needs its qualification: work selects among candidates that can pass the validation pipeline. Work is not a waiver from the rules.

3.4 Validation is not mining

The source supports a three-part distinction.

1. **Validity**: Does a block satisfy the consensus rules enforced by this node?
2. **Work ordering**: Among usable chain candidates, which has the greatest accumulated proof?
3. **Settlement policy**: Given a valid active chain and a threat model, how long should an economic actor wait before treating a payment as sufficiently unlikely to be reorganized?

Source location	Function or range	Semantics relevant here
`pow.cpp:140to170`	`CheckProofOfWork`, `DeriveTarget`	Target range and header hash are checked.
`validation.cpp: 3876to4030`	`CheckBlockHeader`, `CheckBlock`	Header work, Merkle structure, block limits, coinbase placement, and context-free transaction rules are checked.
`validation.cpp: 2528to2616`	block connection loop	Inputs, locks, signature operations, scripts, UTXO updates, and reward cap are enforced.
`validation.cpp: 3157to3217`	candidate selection	A high-work candidate with failed ancestors or missing block data is not usable.
`chain.cpp:121to150`	proof and equivalent time	Encoded difficulty is mapped to per-block proof and chainwork differences.

Table 1: Pinned Bitcoin Core v31.1 source audit. Line ranges refer to commit `9be056a8a72b624dae9623b2f7bded92c2a21c91`.

The first is a deterministic software decision relative to consensus rules and local data. The second is a consensus selection rule. The third is an economic risk decision. Energy participates directly in block production and indirectly in the second and third questions. It does not replace the first.

4 The scenario contract

Definition 4.1 (Assurance scenario)**.** An assurance scenario is the tuple

$$\mathcal{S} = (N, \tau, z, T, H_H, H_A, \eta_H, \eta_A, s_H, p_e, p_r, F_A, B_A, L_P, R_B, \Delta).$$

Here N identifies the network and consensus rules; τ is the measurement interval; z is confirmation depth; T is the post-confirmation settlement window; H_H and H_A are nominal honest and attacker-accessible hashrates; η_H and η_A are operational joules per terahash; s_H is a stylized honest stale fraction; p_e is electricity price; p_r is a hardware access or rental price; F_A is a fixed attack cost; B_A is attacker benefit; L_P is protected loss; R_B is reward value per accepted block; and Δ is the nominal target interval calibrated to the pre-shock nominal hashrate.

The tuple is deliberately redundant with respect to some derived quantities. That redundancy exposes assumptions. If an analyst supplies only energy without hashrate and hardware efficiency, the adversarial share is not identified. If the analyst supplies hashrate without attacker access, the relevant share is still not identified. If probability is given without protected loss or attacker benefit, neither expected loss nor incentive compatibility follows.

4.1 Physical boundary

For the numerical model, honest operational energy during T is

$$E_H(T) = H_H \eta_H T, \qquad E_A(T) = H_A \eta_A T.$$

The units are consistent when H is terahashes per second and η is joules per terahash. These quantities cover hashing equipment only in the reference fixture. Facility overhead, embodied energy, networking, full-node operation, manufacturing, and end-user devices are excluded. A study may add them, but must change the boundary label and recompute the denominator.

The network's ongoing energy is a shared security input. Allocating E_H/n to each of n transactions is an accounting convention, not a causal statement about the marginal electricity required for that transaction. Empty blocks still require proof of work. A marginal transaction can often be included without a proportional change in global hashrate. Per-transaction energy is therefore not used in the frontier.

4.2 Threat boundary

The adversary in the reference model attempts a private-chain catch-up or reorganization. It has a declared maximum hashrate H_A, begins at a stated deficit, and acts for a finite window T. Its objective is summarized by B_A, while L_P describes the loss borne by the protected party if the specified event succeeds.

This adversary is not an eclipse attacker, a routing attacker, a coercive state, a software supply-chain attacker, or a cartel pursuing censorship without a double spend. Those threats require different variables and outcome functionals. The tuple may be extended, but a result obtained for one threat must not be relabeled as total network security.

4.3 Time boundary

Three clocks appear.

1. The energy measurement interval τ determines the joule account.
2. Confirmation depth z records how much accepted chain growth a recipient observes before a settlement decision.
3. The response window T limits the post-confirmation event whose probability and private cost are evaluated.

An annual energy estimate, a six-confirmation rule, and a two-hour attack window are not naturally commensurate. Converting the annual estimate to a two-hour flow assumes that hashrate and efficiency are stationary over the interval. That assumption should be stated, not hidden in unit conversion.

5 Risk models

5.1 Effective adversarial share

The computational reference uses

$$q_{\text{eff}} = \frac{H_A}{H_A + H_H(1 - s_H)}.$$

The stale fraction is a compact sensitivity parameter for productive honest work lost to propagation or coordination effects. It is not a calibrated Bitcoin networking model. In the fixture, s_H is a sudden, uncompensated loss during a window shorter than the relevant difficulty response. The reference interval Δ is tied to nominal pre-shock H_H, so the productive honest rate slows to $(1 - s_H)/\Delta$ during the modeled window. This is not a persistent equilibrium stale rate. Under a difficulty-compensated equilibrium convention, the honest productive rate would instead remain $1/\Delta$, and both Equations (5.3) and (7.1) would require renormalization by $H_H(1 - s_H)$. The parameter shows why nominal hashrate alone is not enough: two networks with the same H_H and H_A can have different effective races if honest work reaches the competing tip at different rates.

For $q_{\text{eff}} < 1/2$, the gambler's-ruin probability that an attacker starting z blocks behind ever catches up is

$$G_z(q_{\text{eff}}) = \left(\frac{q_{\text{eff}}}{1 - q_{\text{eff}}}\right)^z.$$

For $q_{\text{eff}} \geq 1/2$, this eventual probability is one in the stylized unbounded random walk. The expression is not a prediction about the timing, profitability, detectability, or operational feasibility of an attack.

Lemma 5.1 (Monotonicity of the random-walk benchmark)**.** *For integer $z \geq 1$ and $0 < q < 1/2$, $G_z(q)$ is strictly increasing in q and strictly decreasing in z.*

Proof. The ratio $q/(1 - q)$ lies strictly between zero and one. Its derivative with respect to q is $1/(1 - q)^2 > 0$, so raising it to a positive integer power preserves strict increase in q. Multiplication by another factor $q/(1 - q) < 1$ gives $G_{z+1}(q) < G_z(q)$. □

PROVED. The result applies to Equation (5.1), not to every attack strategy or network condition.

5.2 The whitepaper approximation

The Bitcoin whitepaper assumes honest share $p = 1 - q$, adversarial share q, and a Poisson count of attacker progress while the recipient waits for z honest blocks. With $\lambda = zq/p$, its approximation is

$$\mathcal{R}_N(z, q) = 1 - \sum_{k=0}^{z} \frac{e^{-\lambda}\lambda^k}{k!}\left[1 - \left(\frac{q}{p}\right)^{z-k}\right], \qquad q < p.$$

Confirmations z	Whitepaper printed value	Reference code
1	0.2045873	0.204587274
2	0.0509779	0.050977893
3	0.0131722	0.013172242
4	0.0034552	0.003455243
5	0.0009137	0.000913682
6	0.0002428	0.000242803

Table 2: Reproduction of the $q = 0.1$ table in Section 11 of the Bitcoin whitepaper. Agreement is computational evidence about the implementation of Equation (5.2), not validation of its assumptions.

For $q \geq p$, it returns one. The companion code implements the same recurrence as the whitepaper sample and reproduces its values for $q = 0.1$ and $z = 1, \ldots, 6$.

Later analyses correct and refine the whitepaper calculation [3, 4]. For that reason the notation $\mathcal{R}_N$ names the approximation rather than calling it "the Bitcoin attack probability." It is retained because it is widely recognizable and because exact reproduction provides a useful test fixture.

5.3 A finite post-confirmation window

A recipient may also ask a different question: if an attacker is currently z blocks behind, what is the probability of hitting zero deficit within the next T seconds? Let the merged block-discovery process have mean

$$\mu(T) = \frac{T}{\Delta}\left[(1 - s_H) + \frac{H_A}{H_H}\right].$$

Conditional on n discoveries, a dynamic program advances the deficit by -1 with probability q_{eff} and $+1$ with probability $1 - q_{\text{eff}}$. State zero is absorbing success. If $h_n(z, q)$ is the hit probability within n steps, then

$$\mathcal{R}_T(z, q, \mu) = \sum_{n=0}^{\infty} e^{-\mu} \frac{\mu^n}{n!} h_n(z, q).$$

The implementation truncates the Poisson sum after

$$n_{\max} = \left\lceil \mu + 12\sqrt{\mu + 1} + 25 \right\rceil$$

and returns bounds. When the initial Poisson mass is representable, the lower bound assigns the genuine residual tail the latest computed hit probability; the upper bound assigns it probability one. If the initial mass underflows to zero, the implementation returns the uninformative but valid interval $[0, 1]$. Thus numerical truncation or underflow is not silently reported as exact.

Proposition 5.2 (Finite-window bound)**.** *For $0 \le q < 1/2$, fixed deficit $z \ge 1$, and finite mean μ,*

$$0 \le \mathcal{R}_T(z, q, \mu) \le G_z(q).$$

The quantity $\mathcal{R}_T$ is also nondecreasing as the window T, and hence μ, increases.

Proof. The event of hitting zero within a finite number of discoveries is a subset of the event of ever hitting zero, which gives the upper bound. Coupling two Poisson processes so that the discoveries in the shorter window are a prefix of those in the longer window shows event inclusion and monotonicity. □

CONDITIONAL. The proof is exact for the declared Poisson random-walk model. Whether that model is an adequate empirical account is a separate question.

5.4 Why two risks are reported

$\mathcal{R}_N$ and $\mathcal{R}_T$ answer different questions. In $\mathcal{R}_N$, the attacker may have made progress while the recipient waits for the honest chain to add z blocks. In $\mathcal{R}_T$, the attacker begins the post-confirmation window at a fixed deficit z. The numerical values should not match. Reporting both makes the temporal assumption visible and prevents a model switch from looking like a change in observed security.

6 Aggregate energy is not sufficient

Theorem 6.1 (Equal energy, unequal hashrate assurance)**.** *Fix a window $T > 0$, attacker-accessible hashrate $H_A > 0$, stale fraction $s_H \in [0, 1)$, and confirmation depth $z \ge 1$. There exist two honest systems with identical operational energy E_H but different effective adversarial shares and different random-walk catch-up probabilities.*

Proof. Choose positive efficiencies $\eta_1 < \eta_2$ and fix energy $E > 0$. Set

$$H_1 = \frac{E}{\eta_1 T}, \qquad H_2 = \frac{E}{\eta_2 T}.$$

Then $H_1 > H_2$ while $H_1\eta_1 T = H_2\eta_2 T = E$. From Equation (5.1),

$$\frac{H_A}{H_A + H_1(1 - s_H)} < \frac{H_A}{H_A + H_2(1 - s_H)}.$$

Choose H_A small enough that both shares lie below $1/2$. By lemma 5.1, their G_z values differ strictly. Thus equal operational energy does not identify the modeled catch-up risk. □

PROVED. The theorem is an identification counterexample. It does not say that efficiency improvements automatically reduce security. It says that the product $H_H\eta_H T$ loses information about its factors.

Corollary 6.2 (No universal risk-from-energy function)**.** *On the domain of scenarios in theorem 6.1, no function* $f : \mathbb{R}_{>0} \to [0, 1]$ *can satisfy* $G_z(q_{\text{eff}}) = f(E_H)$ *for every scenario.*

Proof. The two scenarios in the theorem have the same input to f and different required outputs. □

PROVED. Adding observed aggregate energy to a dataset cannot, by itself, recover the missing hardware and threat variables.

6.1 A second counterexample: equal energy, unequal attack margin

Suppose two scenarios have the same $E_H, H_H, H_A, \eta_A, T, z$, but electricity prices $p_e^{(1)} \neq p_e^{(2)}$. A stylized gross private attack cost is

$$C_A^{\text{gross}} = \frac{E_A}{3.6 \times 10^6} p_e + H_A p_r \frac{T}{3600} + F_A.$$

Here p_r is denominated in USD per unit of hashrate capacity per hour, USD/((TH/s) h). Thus $H_A p_r (T/3600)$ has units of USD. Holding all other terms fixed, the cost difference is

$$C_A^{(1)} - C_A^{(2)} = \frac{E_A}{3.6 \times 10^6} \left(p_e^{(1)} - p_e^{(2)} \right).$$

The two networks have identical honest energy and hashrate-based risk, yet the private cost of the specified attack differs. If attacker benefits lie between the two costs, their incentive margins have opposite signs.

PROVED for Equation (6.1). Whether electricity can be obtained at p_e, whether suitable hardware is rentable at p_r, and whether those prices remain fixed during an attack are CONDITIONAL.

6.2 Ownership and access

Network hashrate is not the same as attacker-accessible hashrate. An equipment fleet can be physically distributed yet coordinated by a small number of pool operators. Conversely, a large nominal pool share need not imply ownership or unilateral control of every participant's machines. Rentals, firmware access, legal compulsion, pool switching, and supply-chain constraints alter the amount of hardware that an attacker can direct over a given window. Aggregate energy reveals none of these relationships.

A mature empirical study would publish a range $[H_A^L, H_A^U]$ rather than one unsupported share. It would distinguish ownership, routine pool assignment, rapid redirection capability, and new-hardware acquisition. The assurance frontier can then be evaluated over that range.

7 Attack economics

7.1 Gross cost, recovered rewards, and benefit

An attacker may recover block rewards if its private chain becomes accepted. Let $n_A(T)$ be expected attacker blocks in the reference window:

$$n_A(T) = \frac{T}{\Delta}\frac{H_A}{H_H}.$$

Using the midpoint $\bar{R}_T$ of the finite-window probability bounds as a declared approximation, the companion model computes

$$\mathbb{E}[R_A] = \bar{R}_T\, n_A(T) R_B, \qquad C_A^{\text{net}} = C_A^{\text{gross}} - \mathbb{E}[R_A].$$

The attack margin is

$$M_A = C_A^{\text{net}} - B_A.$$

A positive M_A means modeled private cost exceeds modeled private benefit. A negative value means the fixture does not establish deterrence. Neither sign is a statement about legality, ethics, social welfare, or the behavior of a nonprofit attacker.

The reward-recovery expression is intentionally simple. A richer model would account for fees in the attacker's selected transactions, orphaned honest rewards, price impact, detection, capital depreciation, hedging, and the attacker's ability to short related assets. The simple expression remains useful because it prevents the common mistake of treating all mining expenditure as irrecoverable attack cost.

7.2 Budish's stock-flow distinction

Budish's economic limit can be read as a warning against confusing an equilibrium flow with a one-time deviation payoff [5]. Miners receive a recurring compensation flow. A double-spend or sabotage opportunity can create a stock benefit. Deterrence requires a relationship between the two under a specified attack duration and recovery assumption. Observing the flow cost alone does not identify the stock benefit.

The margin in Equation (7.1) is a local implementation of that logic, not a reproduction of Budish's full model. It names the attack window, private resources, conditional recovered rewards, and benefit. It does not assume that competitive honest mining necessarily earns zero economic profit, nor does it infer B_A from market capitalization.

7.3 Fee composition and dynamic incentives

Miner revenue contains subsidy and transaction fees. As the subsidy changes, fee timing can affect strategic behavior. Carlsten and coauthors show in a modeled fee-dominant regime that miners may face incentives to fork, withhold, or undercut [6]. Easley, O'Hara, and Basu analyze fee formation and miner participation as users and miners interact [12].

The frontier therefore asks for expected reward value per accepted attacker block rather than a timeless "security budget." A serious application would model the distribution of fees across candidate blocks, the maturity and spendability of rewards, and price response to a visible attack. One annual revenue number cannot capture these dynamics.

7.4 Private cost is not social cost

Electricity expenditure can be a transfer to producers, a use of scarce generation, and a source of external costs. Hardware rental is a payment to an owner and may reallocate machines from honest mining. A successful reorganization can impose losses on recipients, miners, exchanges, and holders. Adding these terms without an incidence model double counts some flows and misses others.

Accordingly, C_A is labeled private attack cost. A social-cost account would need a separate boundary and valuation functional, including generation externalities, displaced loads, hardware opportunity costs, and distributional effects. That extension belongs to environmental and welfare accounting, not the catch-up equation.

8 A vector assurance frontier

Definition 8.1 (Scenario-conditioned assurance frontier)**.** For fixed scenario declarations other than confirmation count, and a finite set Z of candidate depths, define

$$\mathcal{F}_{\mathcal{S}}(Z) = \{\mathcal{F}(\mathcal{S}, z) : z \in Z\},$$

where each point contains the components in Equation (1).

The frontier is a set of options, not a ranking. One recipient may value faster settlement. Another may face a larger protected loss. A third may have off-chain recourse that reduces loss on success. The frontier shows how declared risk measures change with z, while keeping these economic differences visible.

8.1 Conditional avoided loss per joule

A narrower scalar is possible when a decision maker declares a counterfactual. Let z_0 be a baseline confirmation policy and z_1 an alternative. Under a fixed loss-on-success approximation,

$$\Delta\mathbb{E}[L_P] = L_P\left[\mathcal{R}(z_0) - \mathcal{R}(z_1)\right].$$

If ΔE is the incremental energy causally attributable to changing the policy, then

$$\mathrm{ALPJ} = \frac{\Delta\mathbb{E}[L_P]}{\Delta E} \quad [\mathrm{USD}_{2026}/\mathrm{J}].$$

This ratio often cannot be estimated from a confirmation-policy change. A recipient who waits longer does not normally cause the global mining network to expend proportionally

more energy. In that case ΔE may be near zero under a marginal operational boundary even though elapsed network energy is large. Dividing avoided loss by all network energy during the wait would be an allocation rule, not incremental causation.

Proposition 8.2 (Conditional meaning of an avoided-loss ratio)**.** *Equation* (8.1) *supports a causal interpretation only if the energy increment and loss reduction refer to the same intervention, system boundary, time horizon, and counterfactual.*

Proof. If the numerator and denominator correspond to different interventions, their ratio is not the marginal rate of transformation of either intervention. Changing the boundary or counterfactual changes at least one term while leaving the displayed unit unchanged. The required declarations are therefore necessary for the stated causal interpretation. □

PROVED as an accounting statement. Identification of either increment from empirical data is CONDITIONAL.

8.2 Why no universal "trust" scalar appears

Reorganization probability, censorship delay, resistance to invalid state transitions, availability, and loss recovery are not interchangeable units. Weights could combine them, but the weights would encode a value functional. The present framework refuses to hide that choice. A study that needs a scalar must name the outcome, utility or loss mapping, and affected population.

This is also why the title uses "Trust per Joule" as a question and program label. In the actual accounting, "trust" is replaced by specified events and losses. The terminology is useful only when it directs attention to the conversion chain. It becomes misleading when treated as a natural unit.

9 Worked scenario

The computational fixture is synthetic. It is not a snapshot of Bitcoin. Its purpose is to expose dependencies and make the calculations reproducible. Table 3 gives the assumptions.

The effective attacker share is approximately 0.091743. Honest hashing uses 1.44×10^{13} J during the two-hour window. The attacker's hashing uses 1.8×10^{12} J. Equation (6.1) gives a gross private attack cost of 80,040 USD under the declared energy and hardware prices.

The attack margin remains negative at every displayed depth because the declared benefit is 100,000 USD and net cost approaches 80,040 USD. This does not mean success is likely at high z. It shows that probability and incentive margin are different outputs. A benefit-on-success model would replace B_A by a probability-weighted payoff. A sabotage model may treat benefit as partly independent of reorganization success. The fixture chooses a fixed benefit so that this modeling choice cannot remain implicit.

Input	Fixture value
Honest nominal hashrate H_H	100,000,000 TH/s
Attacker-accessible hashrate H_A	10,000,000 TH/s
Honest energy intensity η_H	20 J/TH
Attacker energy intensity η_A	25 J/TH
Honest stale fraction s_H	0.01
Settlement window T	7,200 s
Reference block interval Δ	600 s
Electricity price p_e	0.06 USD/kWh
Hardware access price p_r	0.000002 USD/((TH/s) h)
Fixed attack cost F_A	50,000 USD
Attacker benefit B_A	100,000 USD
Protected loss L_P	2,000,000 USD
Reward value R_B	200,000 USD/block

Table 3: Synthetic inputs used by the executable fixture. Monetary units are illustrative 2026 USD.

z	$\mathcal{R}_N$	finite-window risk	net cost USD	expected loss USD
1	1.8738×10^{-1}	1.0098×10^{-1}	55,804	201,969
3	1.0171×10^{-2}	1.0259×10^{-3}	79,794	2,052
6	1.4524×10^{-4}	9.9802×10^{-7}	80,040	1.996
12	3.2107×10^{-8}	4.5719×10^{-13}	80,040	9.14×10^{-7}

Table 4: Scenario-conditioned confirmation frontier. The two risk columns use different temporal experiments, so their values are not estimates of one latent quantity. Printed finite-window bounds coincide at this precision.

9.1 The equal-energy fixture

System A uses $H_H = 100{,}000{,}000$ TH/s at 20 J/TH. System B uses 50,000,000 TH/s at 40 J/TH. Over 7,200 seconds both consume

$$E_H = 1.44 \times 10^{13} \text{ J}.$$

With the same attacker hardware and stale fraction, their effective attacker shares are approximately 0.0917 and 0.1681. At six confirmations, the whitepaper approximation returns approximately

$$1.45 \times 10^{-4} \quad \text{and} \quad 5.21 \times 10^{-3}.$$

The latter is about 36 times the former despite equal honest energy. The calculation is COMPUTATIONAL; the general non-identification result is PROVED by theorem 6.1.

9.2 Propagation sensitivity

Holding nominal energy fixed while increasing s_H from zero to 0.15 raises q_{eff} in the fixture. Both modeled catch-up risks increase. The parameter does not claim that all stale blocks

result from propagation, or that an attacker is immune to its own delays. It serves as a controlled counterexample: communication can change productive work without changing the joule total.

10 Censorship, invalidity, and reorganization are different

10.1 Invalid state transitions

An invalid transaction fails consensus checks. The source audit in section 3 shows checks for input availability, scripts, lock constraints, and reward limits. An attacker with dominant hashrate can create a high-work chain that its own modified software calls valid. Unmodified full nodes need not accept it. The attacker may cause disruption by withholding valid blocks or by mining a valid alternative history, but proof of work does not convert an invalid spend into a valid one.

The assurance outcome here is therefore not "truth." It is resistance to a specified valid-chain history attack under a threat model. Calling the output truth would collapse rule enforcement, evidence, and economic finality.

10.2 Reorganization and double spending

A reorganization replaces a suffix of the active chain with a higher-work valid suffix. A double spend is one possible economic use: a payment appears in the replaced history while a conflicting valid transaction appears in the winning history. Not every reorganization contains a double spend, and not every loss from a reorganization is borne by one recipient.

The variable L_P must therefore name the protected position. An exchange deposit, sale of a physical good, derivative settlement, and cross-chain bridge have different loss mappings and recourse. Confirmation depth alone cannot standardize them.

10.3 Censorship

Censorship asks whether a valid transaction is excluded or delayed. A simple majority attacker can refuse to include it in blocks the attacker mines, but the transaction may still enter blocks produced by others. Sustained censorship can involve orphaning blocks that include the target, bribing miners, network partitioning, or policy coordination. Relevant outputs include inclusion delay, probability of inclusion before a deadline, fee escalation, and the fraction of reachable honest hashrate.

None of these is computed by Equation (5.2). A censorship-per- joule study would require a separate threat tuple, transaction propagation model, miner policy model, and outcome definition. Reusing double-spend risk as a censorship score would be a category error.

10.4 Availability and liveness

Block production can slow if hashrate leaves before difficulty adjusts. Network partitions can make local views diverge. Software bugs can interrupt validation. These are liveness and availability questions. They interact with energy and mining participation but are not equivalent to catch-up risk.

A broad assurance dashboard could add expected inclusion time, tail block interval, node reachability, client diversity, and recovery procedures. The present paper leaves those quantities OPEN.

11 From estimates to an empirical study

11.1 Energy and efficiency

Network hashrate is inferred from observed proof and block timing rather than measured at every machine. Hardware efficiency varies across models, firmware, temperature, voltage, maintenance state, and facility configuration. CBECI constructs an equipment basket and publishes a range [15]. An empirical assurance study should preserve that range:

$$\eta_H \in [\eta_H^L, \eta_H^U], \qquad E_H \in [E_H^L, E_H^U].$$

It should state whether reported joules are machine electricity, facility electricity with power usage effectiveness, or lifecycle energy.

11.2 Attacker hardware access

The hardest input may be H_A. A useful evidence table would separate:

- hardware physically owned by a candidate attacker;
- hashrate normally pointed to a pool but redirectable by machine owners;
- rentable compatible hardware with binding availability and duration;
- idled inventory that can be activated within the attack window;
- capacity that could be coerced, compromised, or subsidized;
- new capacity whose manufacturing lead time exceeds the window.

Each class has a different access probability and cost. Pool dashboards alone do not identify ownership or unilateral control. Manufacturer shipment claims do not identify deployment. A defensible study would publish a low, central, and high access scenario with provenance.

11.3 Prices and opportunity costs

Marginal electricity price may differ from an average tariff. Curtailable industrial loads can face interruptible contracts. A miner may own generation or have sunk prepaid energy. Hardware can have a resale value and an honest mining opportunity cost. The reference p_r combines access and time but is not adequate for every ownership structure.

The attack-cost range should therefore be formed from scenarios rather than false precision:

$$C_A \in \left[C_A(H_A^L, \eta_A^L, p_e^L, p_r^L), C_A(H_A^U, \eta_A^U, p_e^U, p_r^U)\right],$$

with dependence handled explicitly. Taking every lower or upper endpoint may produce impossible joint scenarios if, for example, the most efficient hardware is not available at the lowest rental price.

11.4 Propagation

Propagation is a distribution, not one stale fraction. Block size, topology, relay protocols, geographic distance, validation time, and adversarial routing all matter. Decker and Wattenhofer provide an early empirical account [8]; Gervais and coauthors show why delay enters a security model [7]. Current measurement would need current node observations and a sampling design. The fixture's s_H is only a sensitivity knob.

11.5 Benefit and protected loss

Attacker benefit can include a double-spent asset, short position, competing system gain, political objective, or extortion. Protected loss can include direct asset loss, hedging cost, service interruption, and recovery. These variables are partly private and counterfactual. Point estimates should not be invented from public market capitalization.

Partial identification is preferable. If

$$B_A \in [B_A^L, B_A^U], \qquad L_P \in [L_P^L, L_P^U],$$

then the frontier should report corresponding margin and expected-loss ranges. A deterrence statement that survives the sensitivity range requires $M_A > 0$ throughout the admissible benefit range, not merely at a convenient midpoint.

12 Measurement protocol

The following protocol turns the framework into an auditable study.

1. **Name the event.** State whether the outcome is a valid-chain reorganization, double spend, censorship interval, or availability failure.
2. **Pin the rules.** Identify the network, software version or consensus specification, and relevant validation semantics.
3. **Fix the clocks.** State the energy interval, confirmation rule, and attack or settlement window.
4. **Declare the physical boundary.** List hashing, facility, networking, embodied, and node energy inclusions and exclusions.
5. **Estimate productive rates.** Provide honest hashrate, attacker-accessible hashrate, efficiency ranges, and propagation assumptions.

6. **Name the stochastic model.** Provide the formula, starting state, network timing assumptions, and known limitations.
7. **Build the private cost model.** Separate electricity, hardware access, fixed costs, recovered rewards, and opportunity cost.
8. **Build the loss model.** Name the protected party, loss-on-success, recourse, and counterfactual.
9. **Report a frontier.** Vary confirmation depth and material uncertain inputs. Do not hide uncertainty in one score.
10. **Test non-identification.** Construct equal-energy scenarios with different hashrate, access, price, propagation, or benefit.
11. **Separate evidence statuses.** Mark algebra, model-conditional results, computations, empirical estimates, and open questions.

12.1 Minimum reporting table

Field	Required declaration	Failure if omitted
Network and rules	chain, release or specification, validity checks	"security" may refer to incompatible protocols
Event	reorganization, double spend, censorship, liveness	probability has no defined outcome
Time	τ, z, T, Δ	energy and risk cover different windows
Physical inputs	$H_H, H_A, \eta_H, \eta_A, s_H$	energy cannot identify race dynamics
Economic inputs	$p_e, p_r, F_A, R_B, B_A, L_P$	cost, deterrence, and loss are confused
Uncertainty	ranges, dependence, estimator	point output appears more precise than evidence
Counterfactual	baseline settlement policy or attack-free state	avoided loss is undefined

Table 5: Minimum scenario disclosure for an assurance-per-energy study.

13 Computational artifact

The companion module `src/joule_standard/pow_assurance.py` uses only the Python standard library. It contains:

- `gambler_ruin_eventual_probability`;
- `nakamoto_catch_up_probability`;
- `finite_horizon_catch_up_probability`;
- a frozen `ProofOfWorkScenario` data class;
- `assess_scenario`; and

- `assurance_frontier`.

Inputs carry units in their names. Validation rejects negative costs, invalid shares, nonpositive physical rates, and noninteger confirmation depths. Probability truncation returns lower and upper bounds. Monetary loss bounds use a separate type so that dollars cannot be validated as probabilities.

13.1 Test claims

The dedicated test file checks ten behaviors.

1. The whitepaper $q = 0.1$ table is reproduced to printed precision.
2. Catch-up risk falls strictly with depth for $q < 1/2$.
3. Catch-up risk rises strictly with adversarial share in the tested range.
4. Finite-window risk rises with the window and does not exceed the eventual random-walk probability.
5. Poisson underflow returns the conservative interval $[0, 1]$.
6. Gross attack cost rises with duration and accessed hashrate.
7. Both reported risks fall along the tested confirmation frontier.
8. Equal honest energy can yield different effective attacker shares and risks.
9. Propagation loss can change risk without changing honest energy.
10. Invalid physical and economic inputs are rejected.

Passing tests are COMPUTATIONAL evidence that the code implements these fixtures. They do not establish that the fixture describes a live network. The distinction is essential: software correctness cannot supply missing empirical premises.

13.2 Numerical method

The whitepaper approximation uses a recurrence for Poisson mass rather than factorials. The finite-window calculation conditions on the number of merged block discoveries. At each step it stores the probability mass at each positive deficit and accumulates probability when the attacker moves from deficit one to zero. The outer Poisson mixture is then bounded using its residual tail.

The reference method is suitable for modest settlement windows. It rejects a Poisson cutoff above 10,000 events rather than allocating an unbounded dynamic program. Production risk software should use numerically specialized distributions, verify extreme-tail accuracy, and compare independent implementations.

14 Interpretation and decision use

14.1 Merchant or exchange policy

A recipient can use a scenario frontier to compare confirmation depths for a named deposit size and recourse structure. The decision is not "how much energy did this transaction use?"

It is "given these network and attacker assumptions, how does waiting change modeled loss and operational delay?" Energy enters as one network input. The recipient's delay cost belongs on the same decision table.

A practical policy might partition withdrawals by exposure, require more confirmations for unusual deposits, impose velocity limits, and use fraud signals. Those controls alter L_P or the attacker's realizable B_A. They can improve the position without changing network energy. That fact alone shows why energy cannot be the complete security control.

14.2 Miner and protocol analysis

For miners, the frontier can organize sensitivity to electricity prices, hardware efficiency, fees, and propagation. It does not prescribe a mining equilibrium. Strategic entry, pool choice, withholding, and capital investment require equilibrium models such as those studied in the mining literature [10, 11, 6].

For protocol analysis, the framework helps prevent boundary errors. A change to block relay may alter stale rates and effective race dynamics without changing device efficiency. A change to reward composition may alter attack opportunity cost without changing hashrate immediately. A validation-rule change may alter which histories are admissible while leaving proof-of-work energy unchanged.

14.3 Energy policy

An electricity regulator may care about grid congestion, emissions, demand flexibility, and local economic activity. Those outcomes require a grid and social boundary. A protocol assurance frontier cannot decide whether a marginal megawatt-hour has higher social value in mining, manufacturing, storage, or another use. That allocation problem belongs to a broader marginal-value-of- energy analysis.

The framework can still improve the debate. It replaces an unsupported claim that all mining electricity is "security" with an explicit chain from energy to hashrate, competitive share, valid-chain selection, event probability, and loss. A policy analyst can then dispute a particular assumption rather than a metaphor.

15 Failure modes in trust-per-joule accounting

15.1 Dividing by transaction count

Annual network electricity divided by annual on-chain transactions produces an allocation average. It does not estimate the marginal energy of a transaction, the energy of one confirmation, or the energy that protects one asset. The network jointly orders blocks and maintains a history. Transaction batching, empty space, layer-two settlement, and demand changes break the causal reading.

15.2 Using market capitalization as protected value

Market capitalization is price times circulating supply. It is not the amount an attacker can steal, the loss caused by a reorganization, or a stock of value settled during one window. A visible attack may change price, making the quantity endogenous. Protected loss should be tied to affected positions and a counterfactual.

15.3 Calling all miner revenue a security budget

Revenue is a miner incentive and a resource available to support competition. Its relationship to attack deterrence depends on entry, costs, reward recovery, hardware access, and attacker benefit. Some attack strategies earn rewards. Some adversaries value disruption rather than profit. Revenue is therefore an input to an incentive model, not a measured quantity of assurance.

15.4 Assuming fifty percent is a complete threshold

An unbounded catch-up model has a qualitative change at one-half adversarial share. Other attacks may become profitable below one-half, particularly when communication advantage or strategic mining is present [10, 11]. A finite attack with more than one-half share is not instantaneous or free. The threshold is meaningful inside a specific model, not a universal switch for every security property.

15.5 Ignoring full nodes

Treating miners as sole arbiters of validity makes work appear able to authorize arbitrary state transitions. The source audit falsifies that description for the inspected Bitcoin Core release. Full nodes check blocks and transactions. Economic actors decide which software and rules they run. Mining and validation interact, but they are not the same function.

15.6 Reporting a point without its scenario

A probability such as 10^{-6} is uninterpretable without the event, adversarial share, starting state, timing, propagation assumptions, and model. A cost such as 80,040 USD is uninterpretable without hardware quantity, duration, electricity, rental terms, reward recovery, and fixed cost. The scenario tuple is not supplementary metadata. It is part of the result.

16 Limitations

Limitation 16.1 (Stylized block process). Independent exponential block discovery and a constant effective share omit difficulty changes, nonstationary participation, strategic publication, time-varying connectivity, and detection responses.

Limitation 16.2 (Simplified propagation). The honest stale fraction compresses a network process into one scalar. It does not model topology, relay asymmetry, attacker delay, validation latency, or block-size dependence.

Limitation 16.3 (Simplified economics). The cost model is partial equilibrium. It holds input prices fixed, treats hardware access as divisible, and omits financing, capital depreciation, liquidity, legal exposure, price impact, hedging, and endogenous miner entry.

Limitation 16.4 (Reward recovery). Expected recovered reward uses the midpoint of numerical probability bounds and an expected attacker-block count. It omits the joint distribution of attack duration, blocks, fees, and success. It is a fixture, not a valuation model.

Limitation 16.5 (No live calibration). Every worked number is synthetic. The paper offers no claim about present Bitcoin attacker share, energy, security, transaction value, or attack cost.

Limitation 16.6 (Narrow assurance event). The numerical model covers one family of private-chain catch-up events. It does not measure eclipse attacks, software compromise, key theft, governance, censorship, legal coercion, data availability, or recovery.

Limitation 16.7 (Operational energy boundary). The fixture excludes facility overhead, embodied energy, full nodes, network equipment, and end-user systems. It cannot support lifecycle or social-cost claims.

17 Research agenda

Several extensions would materially improve the framework.

17.1 Calibrated access distributions

Research should estimate distributions over attacker-accessible hashrate for different windows. The unit of observation must separate hardware ownership, pool assignment, firmware control, rental contracts, and manufacturing lead time. Confidential commercial data may make partial identification more credible than point estimation.

17.2 Network-aware finite settlement

A network model could replace s_H with measured relay distributions and validation times. It should allow asymmetric information, topology, eclipse risk, and strategic release. Outputs should distinguish common-prefix failure, temporary local disagreement, and economic double spend.

17.3 Joint payoff and price dynamics

Attack benefit, reward recovery, and asset price can be jointly endogenous. An attacker may hedge, short, or profit in another market. Recipients may halt withdrawals after observing unusual blocks. A dynamic game would connect these responses to the physical resource race.

17.4 Censorship frontier

A parallel paper should define censorship assurance in terms of inclusion probability by a deadline, reachable honest share, fees, and miner policy. It should not inherit the double-spend formula. The energy denominator can be shared only if both studies use the same physical boundary and time interval.

17.5 Empirical loss prevention

The economically realized output is avoided loss, not an abstract probability. Estimating it requires incident data, near misses, recourse, and selection effects. Systems with stronger controls may observe fewer attacks because attackers self-select away. A causal design must distinguish deterrence from failed or unobserved attempts.

17.6 Grid interaction

Mining can be geographically flexible and interruptible in some settings. Coupling the assurance frontier to nodal electricity prices and grid services would show when marginal operation is privately and socially valuable. Such a model must count displaced generation, emissions, and reliability effects at the relevant location and hour.

18 Conclusion

Proof of work is an energy-to-assurance process only after the intervening mechanism is named. Bitcoin Core checks proof at the header, validates blocks and transactions under separate consensus rules, and selects high-work usable chain candidates. A recipient then applies a settlement policy under an economic threat model. Joules participate in this chain but do not summarize it.

The formal counterexample is decisive. Equal operational energy can arise from different combinations of hashrate and hardware efficiency. With the same attacker-accessible hardware, those combinations produce different adversarial shares and different modeled catch-up risks. Equal energy can also coexist with different electricity prices, rental markets, ownership structures, propagation, attacker benefits, and protected losses. Aggregate energy is therefore not sufficient for settlement security.

The useful object is a scenario-conditioned frontier. It reports energy, confirmations, adversarial share, named risks, costs, margins, and expected loss without collapsing them into a universal trust score. A scalar value per joule is defensible only after a specific outcome, counterfactual, attribution rule, and energy boundary are fixed. That discipline makes the energy conversion legible while preserving what the arithmetic cannot decide.

Symbol	Meaning	Unit or range
N	network and consensus-rule identifier	text or source pin
τ	physical energy measurement interval	seconds or dated interval
z	accepted confirmation depth	nonnegative integer
T	post-confirmation settlement window	seconds
H_H	nominal honest hashrate	TH/s
H_A	attacker-accessible hashrate	TH/s
η_H, η_A	operational hardware efficiency	J/TH
s_H	stylized unproductive honest-work fraction	$[0, 1)$
q_{eff}	effective adversarial share in the race	$[0, 1]$
Δ	nominal pre-shock target interval	seconds/block
$\mu(T)$	expected merged discoveries in T	blocks
E_H, E_A	honest and attacker hashing energy	J
p_e	attacker electricity price	USD/kWh
p_r	hardware access price	USD/((TH/s) h)
F_A	fixed private attack cost	USD
R_B	reward value per accepted attacker block	USD/block
B_A	attacker benefit under the scenario	USD
L_P	protected loss if the event succeeds	USD
C_A^{gross}	gross private attack cost	USD
C_A^{net}	cost net of modeled reward recovery	USD
M_A	$C_A^{\text{net}} - B_A$	USD
G_z	eventual random-walk catch-up probability	$[0, 1]$
$\mathcal{R}_N$	whitepaper Poisson approximation	$[0, 1]$
$\mathcal{R}_T$	fixed-deficit finite-window hit probability	$[0, 1]$

Table 6: Symbols are local to this paper. Monetary amounts require a declared price date and currency convention in empirical use.

A Symbol and unit register

B Derivation and checks for the synthetic fixture

B.1 Energy

For the honest side,

$$E_H = (100{,}000{,}000 \text{ TH/s})(20 \text{ J/TH})(7{,}200 \text{ s}) = 1.44 \times 10^{13} \text{ J}.$$

For the attacker,

$$E_A = (10{,}000{,}000)(25)(7{,}200) = 1.8 \times 10^{12} \text{ J}.$$

Unit cancellation leaves joules in both cases.

B.2 Effective share and expected discoveries

Productive honest rate is

$$H_H(1 - s_H) = 100{,}000{,}000(0.99) = 99{,}000{,}000 \text{ TH/s}.$$

Thus

$$q_{\text{eff}} = \frac{10}{10 + 99} = 0.091743119\ldots$$

when rates are expressed in millions of TH/s. The expected merged discoveries in two hours are

$$\mu = \frac{7{,}200}{600}\left(0.99 + \frac{10}{100}\right) = 13.08.$$

B.3 Gross cost

Attacker electricity use is

$$\frac{1.8 \times 10^{12}}{3.6 \times 10^{6}} = 500{,}000 \text{ kWh}.$$

At 0.06 USD/kWh, electricity cost is 30,000 USD. Hardware access is

$$(10{,}000{,}000 \text{ TH/s})\left(\frac{0.000002 \text{ USD}}{(\text{TH/s})\,\text{h}}\right)(2 \text{ h}) = 40 \text{ USD}.$$

Adding the 50,000 USD fixed cost gives 80,040 USD.

The low hardware-access price is intentionally artificial. It makes each term visible and should not be interpreted as a market quote. A live study would need evidence that the specified quantity can be controlled at the specified price for the full interval.

C Source-audit reproducibility

C.1 Release pin procedure

The source pin can be reproduced without trusting a moving branch.

1. Open the official Bitcoin Core 31.1 release page.
2. Resolve Git tag `v31.1` through the GitHub repository API.
3. Inspect the annotated tag object and its signature-verification field.
4. Follow the tag object's target to the commit whose identifier is

   `9be056a8a72b624dae9623b2f7bded92c2a21c91`.

5. Open raw or rendered files at that commit, preserving line anchors.

At the time of inspection, the annotated tag object identifier was

`bfa6a4b79cd4c1562fd32857e3147763efae37fb`.

The paper cites the commit because source files live at the commit object. The tag object records the release reference and signature metadata.

C.2 Interpretation checklist

The source lines justify the following limited statements.

- A block header's hash is compared with a derived target.
- A block undergoes checks beyond header proof of work.
- Context-dependent input and script checks occur during block connection.
- The coinbase cannot exceed subsidy plus fees under the checked rules.
- Candidate-chain work ordering does not make known-invalid ancestry usable.

They do not, by themselves, prove peer-to-peer liveness, economic decentralization, absence of implementation bugs, or a numerical attack probability. Those claims require other evidence.

D Artifact-to-claim map

Claim	Evidence status	Artifact
Equal energy does not identify modeled risk	PROVED	theorem 6.1
Equal-energy numerical witness	COMPUTATIONAL	equal-energy test in `test_pow_assurance.py`
Whitepaper table reproduction	COMPUTATIONAL	whitepaper-table test in `test_pow_assurance.py`
Risk monotonicity in fixture	COMPUTATIONAL	monotonicity test in `test_pow_assurance.py`
Finite-window truncation is bounded	COMPUTATIONAL	`ProbabilityBounds` and finite-window test
Full nodes enforce rules beyond work	source evidence	Pinned Bitcoin Core lines in table 1
Live network attack probability	OPEN	no claim or artifact
Live attacker benefit and hardware access	OPEN	no claim or artifact

Table 7: The map prevents executable fixtures from being presented as empirical measurement or protocol proof.

References

[1] Satoshi Nakamoto. Bitcoin: A Peer-to-Peer Electronic Cash System. 2008. https://bitcoincore.org/bitcoin.pdf.

[2] Juan Garay, Aggelos Kiayias, and Nikos Leonardos. The Bitcoin Backbone Protocol: Analysis and Applications. In *Advances in Cryptology, EUROCRYPT 2015*, pages 281 to

310, 2015. Cryptology ePrint Archive, Report 2014/765. https://eprint.iacr.org/2014/765.

[3] Meni Rosenfeld. Analysis of Hashrate-Based Double Spending. arXiv:1402.2009, 2014. https://arxiv.org/abs/1402.2009.

[4] Cyril Grunspan and Ricardo Pérez-Marco. Double Spend Races. *International Journal of Theoretical and Applied Finance*, 21(8), 2018. Preprint: https://arxiv.org/abs/1702.02867.

[5] Eric Budish. The Economic Limits of Bitcoin and the Blockchain. NBER Working Paper 24717, 2018. https://www.nber.org/papers/w24717. doi:10.3386/w24717.

[6] Miles Carlsten, Harry Kalodner, S. Matthew Weinberg, and Arvind Narayanan. On the Instability of Bitcoin Without the Block Reward. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 154 to 167, 2016. https://doi.org/10.1145/2976749.2978408.

[7] Arthur Gervais, Ghassan O. Karame, Karl Wüst, Vasileios Glykantzis, Hubert Ritzdorf, and Srdjan Capkun. On the Security and Performance of Proof of Work Blockchains. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 3 to 16, 2016. https://doi.org/10.1145/2976749.2978341.

[8] Christian Decker and Roger Wattenhofer. Information Propagation in the Bitcoin Network. In *IEEE P2P 2013 Proceedings*, pages 1 to 10, 2013. https://doi.org/10.1109/P2P.2013.6688704.

[9] Kyle Croman, Christian Decker, Ittay Eyal, Adem Efe Gencer, Ari Juels, Ahmed Kosba, Andrew Miller, Prateek Saxena, Elaine Shi, Emin Gün Sirer, Dawn Song, and Roger Wattenhofer. On Scaling Decentralized Blockchains. In *Financial Cryptography and Data Security*, pages 106 to 125, 2016. https://doi.org/10.1007/978-3-662-53357-4_8.

[10] Ittay Eyal and Emin Gün Sirer. Majority Is Not Enough: Bitcoin Mining Is Vulnerable. In *Financial Cryptography and Data Security*, pages 436 to 454, 2014. https://doi.org/10.1007/978-3-662-45472-5_28.

[11] Ayelet Sapirshtein, Yonatan Sompolinsky, and Aviv Zohar. Optimal Selfish Mining Strategies in Bitcoin. In *Financial Cryptography and Data Security*, pages 515 to 532, 2016. https://doi.org/10.1007/978-3-662-54970-4_30.

[12] David Easley, Maureen O'Hara, and Soumya Basu. From Mining to Markets: The Evolution of Bitcoin Transaction Fees. *Journal of Financial Economics*, 134(1):91 to 109, 2019. https://doi.org/10.1016/j.jfineco.2019.03.004.

[13] Gur Huberman, Jacob D. Leshno, and Ciamac Moallemi. Monopoly Without a Monopolist: An Economic Analysis of the Bitcoin Payment System. *Review of Economic Studies*, 88(6):3011 to 3040, 2021. https://doi.org/10.1093/restud/rdab014.

[14] Joseph Bonneau, Andrew Miller, Jeremy Clark, Arvind Narayanan, Joshua A. Kroll, and Edward W. Felten. SoK: Research Perspectives and Challenges for Bitcoin and Cryptocurrencies. In *2015 IEEE Symposium on Security and Privacy*, pages 104 to 121, 2015. https://doi.org/10.1109/SP.2015.14.

[15] Cambridge Centre for Alternative Finance. Cambridge Blockchain Network Sustainability Index: CBECI Methodology. Accessed 23 July 2026. https://ccaf.io/cbnsi/cbeci/methodology.

[16] Bitcoin Core. Bitcoin Core 31.1. Release page, 8 July 2026. https://bitcoincore.org/en/releases/31.1/.

[17] Bitcoin Core. Signed tag v31.1 and target commit. GitHub repository objects, inspected 23 July 2026. https://github.com/bitcoin/bitcoin/releases/tag/v31.1.

[18] Bitcoin Core. `src/pow.cpp`, lines 140 to 170. Commit `9be056a8a72b624dae9623b2f7bded92c2a21c91`. Pinned source.

[19] Bitcoin Core. `src/validation.cpp`, lines 3876 to 4030. Commit `9be056a8a72b624dae9623b2f7bded92c2a21c91`. Pinned source.

[20] Bitcoin Core. `src/validation.cpp`, lines 2528 to 2616. Commit `9be056a8a72b624dae9623b2f7bded92c2a21c91`. Pinned source.

[21] Bitcoin Core. `src/consensus/tx_verify.cpp`, lines 164 to 214. Commit `9be056a8a72b624dae9623b2f7bded92c2a21c91`. Pinned source.

[22] Bitcoin Core. `src/validation.cpp`, lines 3157 to 3217. Commit `9be056a8a72b624dae9623b2f7bded92c2a21c91`. Pinned source.

[23] Bitcoin Core. `src/chain.cpp`, lines 121 to 150. Commit `9be056a8a72b624dae9623b2f7bded92c2a21c91`. Pinned source.