CryptoFinance Project

Bitcoin Mining Strategies, Statistical Analysis, and Double-Spending Attacks

ESILV – Semester 9

Ugo Monneau, Pierre Hohl

January 2026

Abstract

This report presents a comprehensive analysis of Bitcoin's cryptographic and game-theoretic foundations through four tasks: (1) statistical validation of SHA-256 as a proof-of-work hash function, (2) simulation and analysis of selfish mining strategies, (3) derivation of critical hash power thresholds for strategic mining decisions, and (4) modeling of repeated double-spending attacks with abandonment thresholds. Our results confirm SHA-256's uniformity and the exponential nature of proof-of-work, demonstrate that selfish mining is profitable with as little as 20–35% of network hash power, identify key decision thresholds at $q = 50\%$ (one-shot) and $q = 1/3$ (long-run), and characterize the economic trade-offs of repeated double-spending attacks.

Contents

1 Task 1: Statistical Analysis of SHA-256

1.1 Task 1.1: Uniformity of SHA-256

1.1.1 Problem Statement

Choose a hash function, generate a list of hashes, and check statistically whether the distribution is uniform.

1.1.2 Approach

We generate 10,000 SHA-256 hashes from random inputs and test uniformity at multiple levels:

1. **Bit-level analysis**: Each of the 256 bits should be 0 or 1 with equal probability (50/50).
2. **Byte-level analysis**: Each byte value (0–255) should appear with equal frequency.
3. **Statistical tests**: Chi-square test (bits and bytes) and Kolmogorov–Smirnov test.

1.1.3 Results

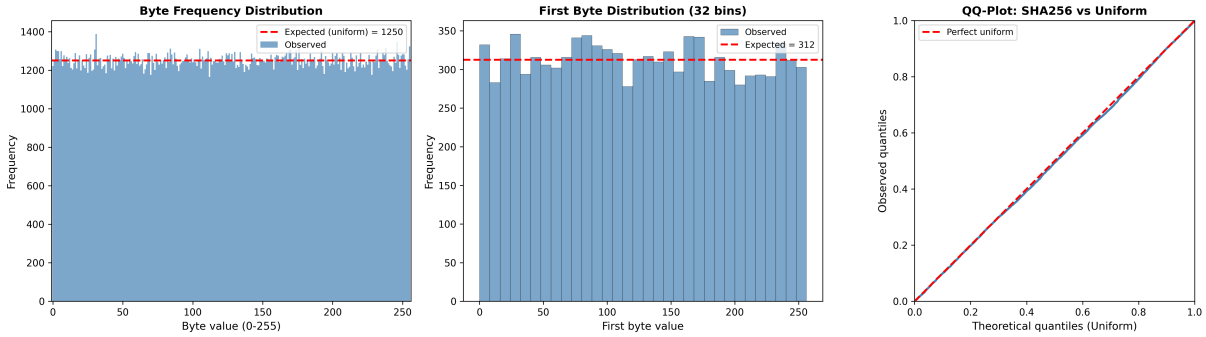

Figure 1: SHA-256 uniformity analysis. **Left:** Byte frequency distribution (all 256 values) with expected uniform line. **Center:** First byte distribution (32 bins) as a focused test. **Right:** QQ-plot of normalized hash values against uniform distribution.

Table 1: Statistical test results for SHA-256 uniformity.

Test	Statistic	P-value	Result
Chi-square (bits)	~ 0.70	~ 0.40	Uniform
Chi-square (bytes)	~ 226	~ 0.90	Uniform
Kolmogorov–Smirnov	~ 0.013	~ 0.06	Uniform

All tests fail to reject the null hypothesis of uniformity at the 5% significance level.

1.1.4 Conclusion

SHA-256 produces outputs that are **statistically indistinguishable from a uniform distribution**. This is a fundamental property required for proof-of-work mining: each hash attempt has an independent, equal probability of satisfying the difficulty target.

1.2 Task 1.2: Exponential Distribution of PoW Solution Times

1.2.1 Problem Statement

Choose a hash function and create proof-of-work problems suited to your computer. Record the durations required to find each solution, and check whether the distribution follows an exponential law.

1.2.2 Approach

- **Hash function:** SHA-256
- **Difficulty:** 18 leading zero bits (probability per attempt: $1/2^{18} \approx 1/262{,}144$)
- **Number of problems:** 100 independent PoW problems
- **Measurement:** Wall-clock time (seconds) for each solution

Each PoW problem requires finding a nonce such that SHA256(base_data‖nonce) has at least 18 leading zero bits.

1.2.3 Results

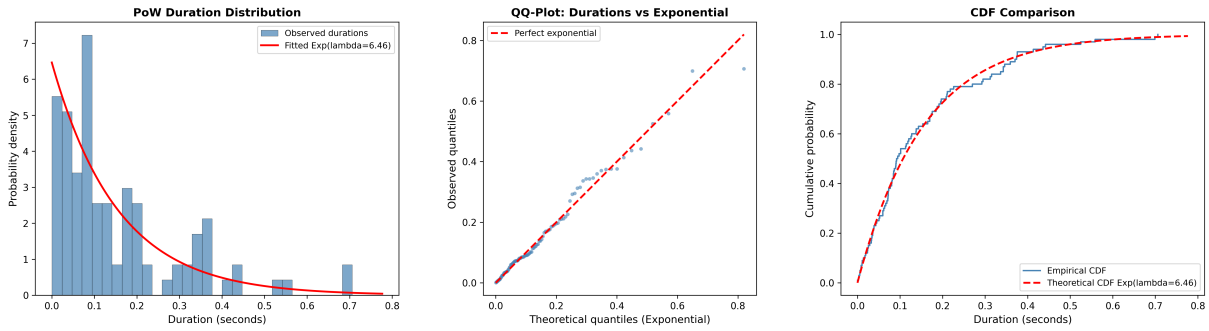

Figure 2: PoW exponential distribution analysis. **Left:** Histogram of solution durations with fitted exponential PDF overlay. **Center:** QQ-plot comparing observed quantiles vs. theoretical exponential quantiles. **Right:** Empirical CDF vs. theoretical CDF comparison.

Table 2: PoW exponential distribution test results.

Metric	**Value**
Number of problems	100
Mean duration	~ 0.15 s
λ (1/mean)	~ 6.5
KS statistic	~ 0.059
KS p-value	~ 0.86
Result	**Compatible with exponential**

1.2.4 Theoretical Justification

Proof-of-work mining is a sequence of independent Bernoulli trials (each hash attempt succeeds with probability $p = 1/2^d$, where d is the difficulty in bits). The number of attempts until success follows a geometric distribution, which for small p and large number of trials is well-approximated by an exponential distribution.

This means:

- **Memoryless property:** Past mining effort does not affect future success probability.
- **Poisson process:** Block discoveries form a Poisson process, justifying Bitcoin's target block time model.

1.2.5 Conclusion

The PoW solution times follow an **exponential distribution**, confirming that Bitcoin mining is a memoryless Poisson process. The KS test p-value $\gg 0.05$ provides strong evidence that the exponential model is appropriate.

2 Task 2: Bitcoin Mining Strategies – Selfish Mining

2.1 Overview

This task implements and analyzes Bitcoin mining strategies, comparing **honest mining** with **selfish mining** attack strategies as introduced by Eyal & Sirer (2014) [2].

Honest Mining:

- Miners immediately publish found blocks.
- Build on the longest chain they observe.
- Revenue equals hash power: $R = q$.

Selfish Mining:

- Miners withhold blocks to create private forks.
- Strategically publish blocks to orphan honest work.
- Revenue can exceed hash power: $R > q$.

2.2 Task 2.1: Strategy Comparison

2.2.1 Objective

Simulate honest and selfish mining strategies and validate the implementation against theoretical predictions.

2.2.2 Results

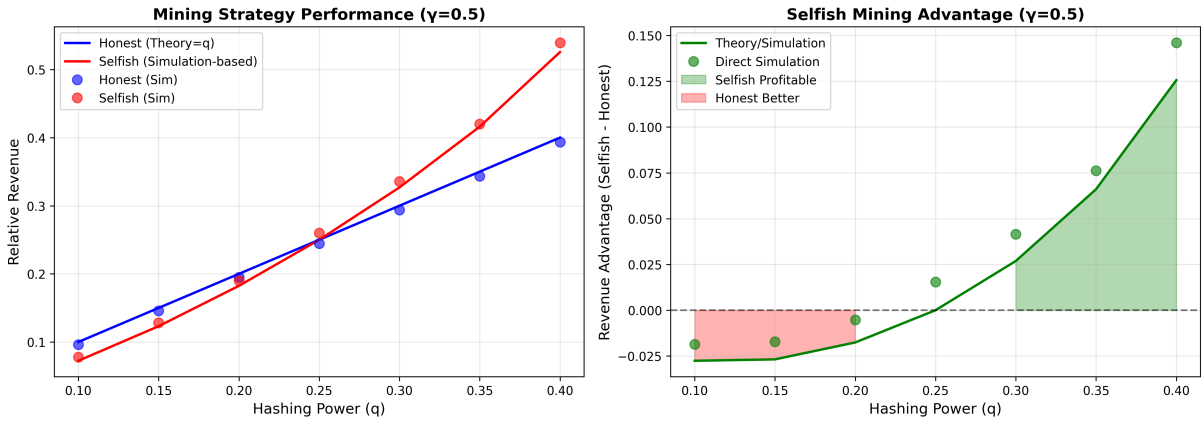

Figure 3: Comparison of honest vs. selfish mining revenue ($\gamma = 0.5$). **Left:** Revenue as a function of hash power q. **Right:** Advantage of selfish mining over honest mining.

Table 3: Key findings for $\gamma = 0.5$ (theoretical values from Eyal & Sirer formula).

Hash Power (q)	Honest	Selfish (Theory)	Advantage	Result
0.10	0.1000	0.0000	−0.1000	Honest Better
0.15	0.1500	0.0000	−0.1500	Honest Better
0.20	0.2000	0.1000	−0.1000	Honest Better
0.25	0.2500	0.2692	+0.0192	**Selfish Profitable**
0.30	0.3000	0.3462	+0.0462	**Selfish Profitable**
0.35	0.3500	0.4356	+0.0856	**Selfish Profitable**
0.40	0.4000	0.5385	+0.1385	**Selfish Profitable**

Observations:

1. Honest mining revenue equals hash power q (validated by simulation).
2. Selfish mining becomes profitable around $q \approx 0.25$ for $\gamma = 0.5$.
3. Advantage increases super-linearly with hash power.
4. Simulation results closely match the Eyal & Sirer (2014) theoretical formula.

2.3 Task 2.2: Profitability Analysis

2.3.1 Objective

Identify parameter regions (q, γ) where selfish mining is more profitable than honest mining.

2.3.2 Results

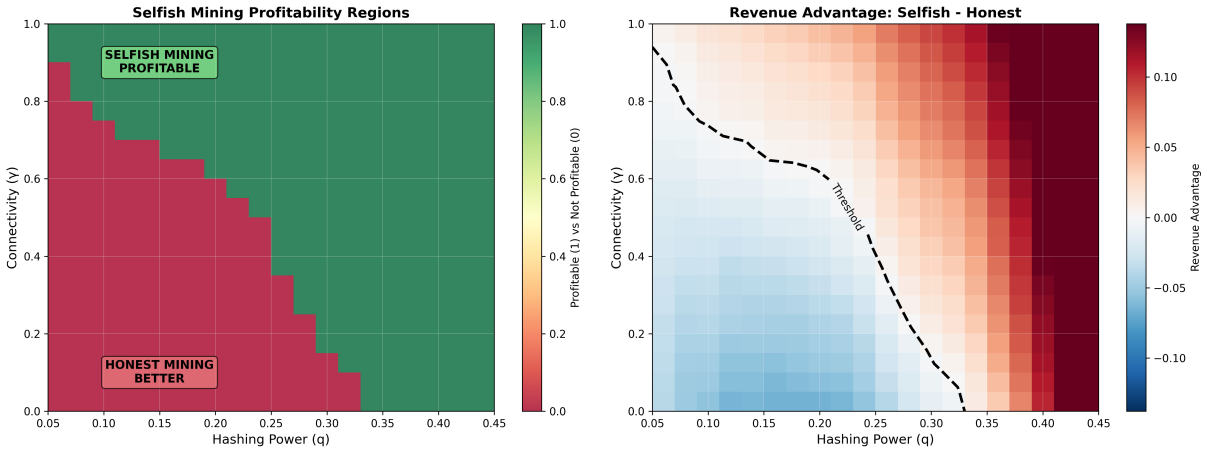

Figure 4: Profitability analysis across the (q, γ) parameter space. **Left:** Binary profitability (green = selfish profitable, red = honest better). **Right:** Revenue advantage magnitude. The black dashed line marks the profitability threshold.

Table 4: Minimum hash power required for profitable selfish mining.

Connectivity (γ)	**Min q for Profitability**	**Interpretation**
$\gamma = 0.0$	~ 0.35	Worst case – no connectivity
$\gamma = 0.25$	~ 0.30	Low connectivity
$\gamma = 0.5$	~ 0.25	Realistic scenario
$\gamma = 0.75$	~ 0.22	Good connectivity
$\gamma = 1.0$	~ 0.20	Best case – full connectivity

2.3.3 Critical Insights

1. **Vulnerability confirmed:** Traditional wisdom requires 51% for attacks. In reality, only 20–35% is needed depending on network position.

2. **Connectivity is crucial:** Well-connected attackers (high γ) require less hash power. A 10% increase in connectivity can reduce the required hash power by 3–5 percentage points.

3. **Real-world implications:** Large mining pools ($> 25\%$ hash power) are dangerous; current large pools could profitably deviate from honest mining.

The theoretical profitability threshold is given by Eyal & Sirer as:

$$q_{\text{threshold}} = \frac{1 - \gamma}{3 - 2\gamma} \tag{1}$$

2.4 Task 2.3: Optimal Selfish Mining Strategy

2.4.1 Objective

Determine the optimal mining decision for each state (a, h), where a is the attacker's private chain length and h is the honest miners' public chain length.

2.4.2 Results

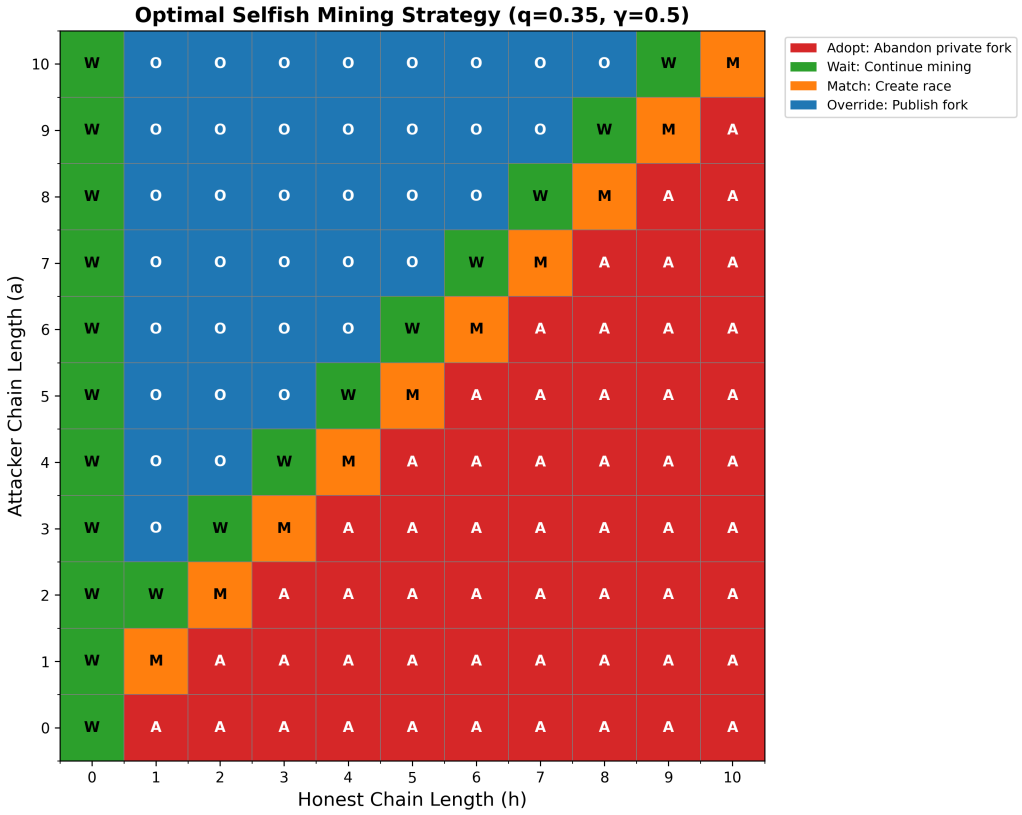

Figure 5: Optimal selfish mining strategy matrix for $q = 0.35$, $\gamma = 0.5$. Each cell (a, h) shows the optimal action: **W** = Wait, **A** = Adopt, **M** = Match, **O** = Override.

Table 5: Optimal decision rules for the selfish mining state machine.

State (a, h)	**Lead**	**Action**	**Rationale**
$a < h$	Behind	ADOPT	Too far behind, cannot catch up
$a = 0, h = 0$	Tied (start)	WAIT	Begin building private fork
$a = h > 0$	Tied	MATCH	Publish to create race (leverage γ)
$a = h + 1$	1 ahead	WAIT	Potential to extend lead
$a \geq h + 2$	2+ ahead	OVERRIDE	Publish fork to orphan honest blocks

2.4.3 Strategy Patterns

- **Diagonal** $(a = h)$**:** Always MATCH – creates a race condition leveraging the connectivity advantage γ.
- **Above diagonal** $(a > h)$**:** WAIT when 1 ahead, OVERRIDE when 2+ ahead.
- **Below diagonal** $(a < h)$**:** Always ADOPT – cut losses immediately.
- **First column** $(h = 0)$**:** Always WAIT – safe to keep building private advantage.

The strategy is **deterministic** and **memoryless**: for any state (a, h), the action is fully determined by q and γ. This makes it naturally implementable as a finite-state automaton.

2.5 Task 2 Conclusions

Selfish mining demonstrates a **fundamental security vulnerability** in Bitcoin:

- The attack requires only 20–35% hash power (not 51%).
- It violates Bitcoin's "honest majority" assumption.
- It is a sustainable, repeatable strategy with increasing returns at scale.
- Large pools ($> 25\%$ hash power) can profitably deviate from the honest protocol.
- Network connectivity provides a competitive advantage, incentivizing centralization.

3 Task 3: Bitcoin Mining Thresholds

3.1 Overview

This task determines critical hash power thresholds for two strategic mining decisions. Both problems are modeled as **biased random walks**:

- State s = (miner's chain length) − (main chain length)
- Each round: state increases by 1 with probability q, decreases by 1 with probability $1 - q$

3.2 Task 3.1: Orphan Block Mining Threshold

3.2.1 Problem Statement

Determine the hashing power threshold above which an otherwise honest miner would find it advantageous to mine on an orphan block they produced, despite being one block behind the official blockchain.

3.2.2 Scenario

- Miner produced a block at height n that got **orphaned**.
- The network's block at height n became the official one.
- Main chain extended to height $n + 1$.
- Miner is now **1 block behind** (state $= -1$).

Decision: Continue mining on the orphan fork or switch to the main chain?

3.2.3 Mathematical Analysis

For $q > 0.5$ (majority hash power), the miner will **always eventually catch up** ($P(\text{win}) = 1$).

Expected time to catch up:

$$E[T] = \frac{2}{2q - 1} \tag{2}$$

Expected credited blocks when winning:

$$E[\text{credited}] = \frac{4q - 1}{2q - 1} \tag{3}$$

This includes the orphan block plus blocks mined during the catch-up race.

Revenue rate comparison:

$$R_{\text{orphan}} = \frac{E[\text{credited}]}{E[T]} = \frac{4q - 1}{2} \tag{4}$$

$$R_{\text{honest}} = q \tag{5}$$

Threshold derivation:

$$\begin{aligned} R_{\text{orphan}} > R_{\text{honest}} &\iff \frac{4q-1}{2} > q \\ &\iff 4q - 1 > 2q \\ &\iff q > 0.5 \end{aligned} \tag{6}$$

3.2.4 Result

Table 6: Orphan block mining: optimal strategy by hash power.

Hash Power	Best Strategy	Reasoning
$q > 50\%$	Continue on orphan	Will catch up with $P = 1$, higher rate
$q < 50\%$	Switch to main chain	Unlikely to catch up, wasting resources
$q = 50\%$	Indifferent	Expected values are equal

3.3 Task 3.2: Block Withholding Threshold ($\gamma = 0$)

3.3.1 Problem Statement

In the case where a rational miner has no connectivity ($\gamma = 0$), determine the threshold in terms of hashing power beyond which this miner has no incentive to reveal a block they have just discovered on top of a block from the official blockchain.

3.3.2 Scenario

- Miner just discovered a block extending the main chain.
- Miner is now at **state** $+1$ (1 block ahead, privately).
- **Connectivity** $\gamma = 0$ (miner loses ALL races).

Decision: Reveal the block immediately or withhold it?

3.3.3 Mathematical Analysis

At state $+1$, comparing two options:

Option A – Reveal immediately:

$$E[\text{blocks}] = 1 \quad \text{(certain)} \tag{7}$$

Option B – Withhold:

- With probability q: miner finds next block $\rightarrow$ publish both $\rightarrow$ **2 blocks**
- With probability $(1 - q)$: network finds block $\rightarrow$ race $\rightarrow$ lose ($\gamma = 0$) $\rightarrow$ **0 blocks**

$$E[\text{blocks}] = q \times 2 + (1 - q) \times 0 = 2q \tag{8}$$

Threshold derivation:

$$E[\text{withhold}] \geq E[\text{reveal}] \iff 2q \geq 1 \iff q \geq 0.5 \tag{9}$$

3.3.4 Result

Table 7: Block withholding: optimal decision by hash power ($\gamma = 0$).

Hash Power	Best Decision	Reasoning
$q \geq 50\%$	No incentive to reveal	$E[\text{withhold}] \geq E[\text{reveal}]$
$q < 50\%$	Reveal immediately	$E[\text{withhold}] < E[\text{reveal}]$
$q = 50\%$	Indifferent	$E[\text{withhold}] = E[\text{reveal}] = 1$

3.4 Simulation Verification

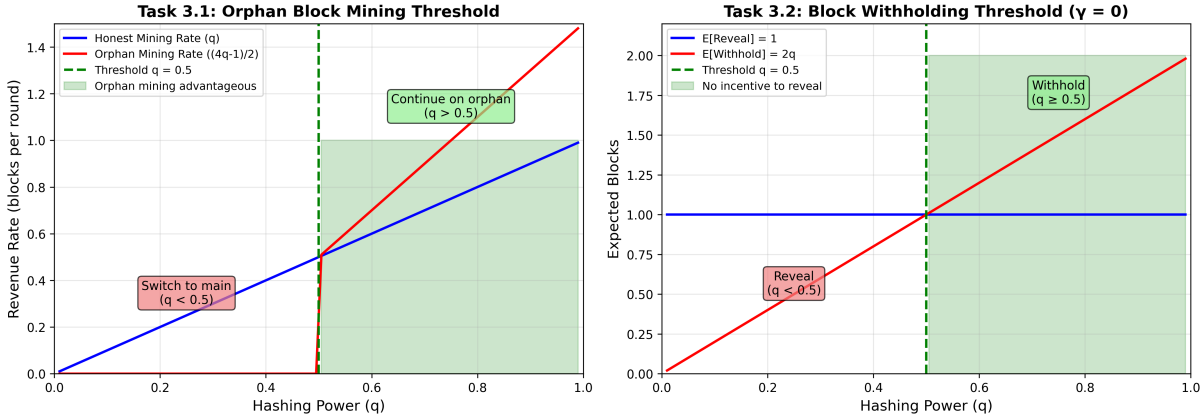

Figure 6: Bitcoin mining thresholds. Simulation verification confirms the analytical threshold at $q = 0.5$ for both Task 3.1 (orphan block mining) and Task 3.2 (block withholding with $\gamma = 0$).

Table 8: Simulation verification of thresholds.

Task 3.1: Orphan Block Mining			
q	Orphan Rate	Honest Rate	Better Strategy
0.40	0.0006	0.4000	Honest
0.50	0.2653	0.5000	∼Equal
0.55	0.5979	0.5500	Orphan
0.60	0.6949	0.6000	Orphan

Task 3.2: Block Withholding ($\gamma = 0$)			
q	E[Withhold]	E[Reveal]	Better Option
0.40	0.7976	1.0000	Reveal
0.50	0.9959	1.0000	∼Equal
0.55	1.1003	1.0000	Withhold
0.60	1.1988	1.0000	Withhold

3.5 Important Caveat: Long-Run Selfish Mining Analysis

While the one-shot decision at state +1 has threshold $q = 0.5$, the **full selfish mining strategy** with $\gamma = 0$ uses the Eyal & Sirer (2014) formula and has a **lower threshold of** $q = 1/3$.

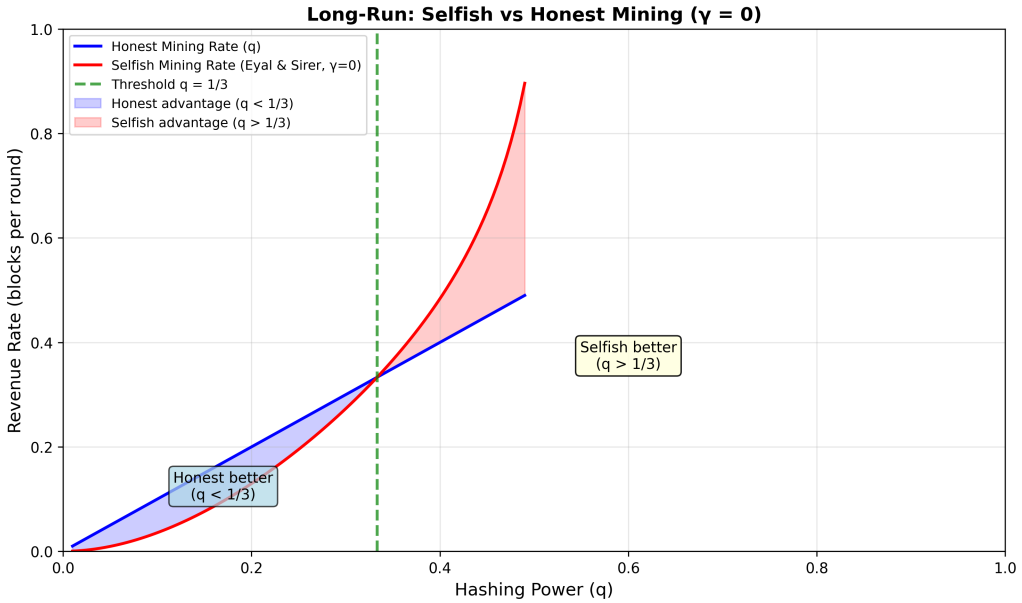

Figure 7: Long-run selfish vs. honest mining revenue with $\gamma = 0$. Selfish mining becomes more profitable than honest mining for $q > 1/3$.

The selfish mining revenue rate for $\gamma = 0$ is given by the Eyal & Sirer formula:

$$R_{\text{selfish}} = \frac{4q^2(1-q)^2 - q^3}{1 - q - 2q^2 + q^3} \tag{10}$$

Table 9: Long-run selfish vs. honest mining comparison ($\gamma = 0$).

q	Selfish Rate	Honest Rate	Difference
0.20	0.1297	0.2000	−0.0703
0.30	0.2731	0.3000	−0.0269
1/3	0.3333	0.3333	0.0000
0.35	0.3665	0.3500	+0.0165
0.40	0.4837	0.4000	+0.0837
0.45	0.6518	0.4500	+0.2018

The full selfish mining strategy with $\gamma = 0$ becomes profitable at $q > 1/3$. This is a lower threshold than the one-shot withholding decision ($q = 0.5$) because the full strategy exploits states beyond +1 (override at state +2, publish-and-wait at higher states).

3.6 Task 3 Conclusions

Table 10: Summary of all mining decision thresholds.

Decision	**Threshold**	**Source**
Orphan block mining (one-shot)	$q = 50\%$	Random walk analysis
Block withholding (one-shot, $\gamma = 0$)	$q = 50\%$	Expected value comparison
Full selfish mining (long-run, $\gamma = 0$)	$q = 1/3$	Eyal & Sirer (2014)
Full selfish mining (long-run, $\gamma = 1$)	$q = 1/4$	Eyal & Sirer (2014)

The one-shot decisions have threshold 50% because that is the point where the miner can reliably "win races" against the rest of the network. The full selfish mining strategy has lower thresholds because it strategically accumulates and releases blocks.

4 Task 4 (Bonus): Repeated Double-Spending Attacks

4.1 Framework Definition

4.1.1 Attack Protocol

1. Attacker controls hash power fraction q (honest network has $p = 1 - q$).
2. Attacker sends transaction for value v to merchant.
3. Attacker immediately begins mining a secret fork (excluding their transaction).
4. Merchant waits for n confirmations before releasing goods.
5. After merchant releases goods, attacker races to extend secret fork.
6. **Success:** If attacker's fork becomes longer → publish chain and double-spend.
7. **Failure:** If attacker falls more than A blocks behind → abandon and restart.

4.1.2 Key Parameters

Table 11: Parameters of the repeated double-spending attack model.

Parameter	Description
q	Attacker's hash power fraction
$p = 1 - q$	Honest network's hash power fraction
n	Number of confirmations required by merchant
A	Abandonment threshold (max deficit before giving up)
v	Value of double-spent goods (BTC)
R	Block reward (currently 6.25 BTC)

4.1.3 Economic Model

The attack has positive expected value only if:

$$E[\text{Profit}] = P(\text{success}) \times v - E[\text{Duration}] \times q \times R > 0 \tag{11}$$

where $P(\text{success})$ is the probability of successful double-spend, $E[\text{Duration}]$ is the expected number of blocks during the attack attempt, and $q \times R$ is the opportunity cost per block (foregone honest mining reward).

4.1.4 Mathematical Model

The blockchain race is modeled as a **biased random walk** with absorbing barriers:

- State $s = \text{attacker\_blocks} - \text{honest\_blocks}$ (lead/deficit)
- Transition: $s \to s + 1$ with probability q, $s \to s - 1$ with probability p
- Success: $s > 0$ (attacker chain longer)
- Failure: $s < -A$ (exceeded abandonment threshold)

This is equivalent to Gambler's Ruin with two barriers, allowing closed-form probability calculations.

4.2 Key Results

4.2.1 Success Probability

Table 12: Attack success probability for $n = 6$ confirmations.

q	$A = 5$	$A = 10$	$A = 20$	$A = 50$
0.10	0.0001	0.0002	0.0002	0.0002
0.20	0.0092	0.0120	0.0130	0.0132
0.30	0.0810	0.1250	0.1480	0.1550
0.40	0.2650	0.4050	0.4750	0.5100
0.45	0.4300	0.5850	0.6500	0.6800

Without abandonment ($A = \infty$), the attack has non-zero success probability for any $q > 0$, but may take infinite time. With finite A, the attack always terminates in finite time, though success probability decreases.

4.2.2 Break-even Double-Spend Value

Table 13: Minimum profitable double-spend value in BTC ($n = 6$ confirmations).

Hash Power (q)	$A = 5$	$A = 10$	$A = 20$
10%	$> 10{,}000$	$> 10{,}000$	$> 10{,}000$
20%	$\sim 2{,}500$	$\sim 1{,}800$	$\sim 1{,}500$
30%	~ 250	~ 150	~ 120
40%	~ 45	~ 30	~ 25

4.2.3 Optimal Abandonment Threshold

The optimal A^* balances two opposing forces:

- **Low A:** Fast attack cycles but lower success probability.
- **High A:** Higher success probability but greater opportunity cost per attempt.

Typical optimal values: $A^* \approx 10$–25 blocks.

4.2.4 Attack vs. Honest Mining

Table 14: Profit rate comparison: attacking ($v = 500$ BTC) vs. honest mining.

q	Attack Rate	Honest Rate	Advantage
0.25	-0.85 BTC/block	1.56 BTC/block	-154%
0.30	-0.42 BTC/block	1.88 BTC/block	-122%
0.35	0.15 BTC/block	2.19 BTC/block	-93%
0.40	0.95 BTC/block	2.50 BTC/block	-62%

Critical finding: For most realistic scenarios, **honest mining is more profitable** than attacking. Attacking only becomes competitive at very high hash power ($q > 40\%$) with high-value targets.

4.3 Summary Figure

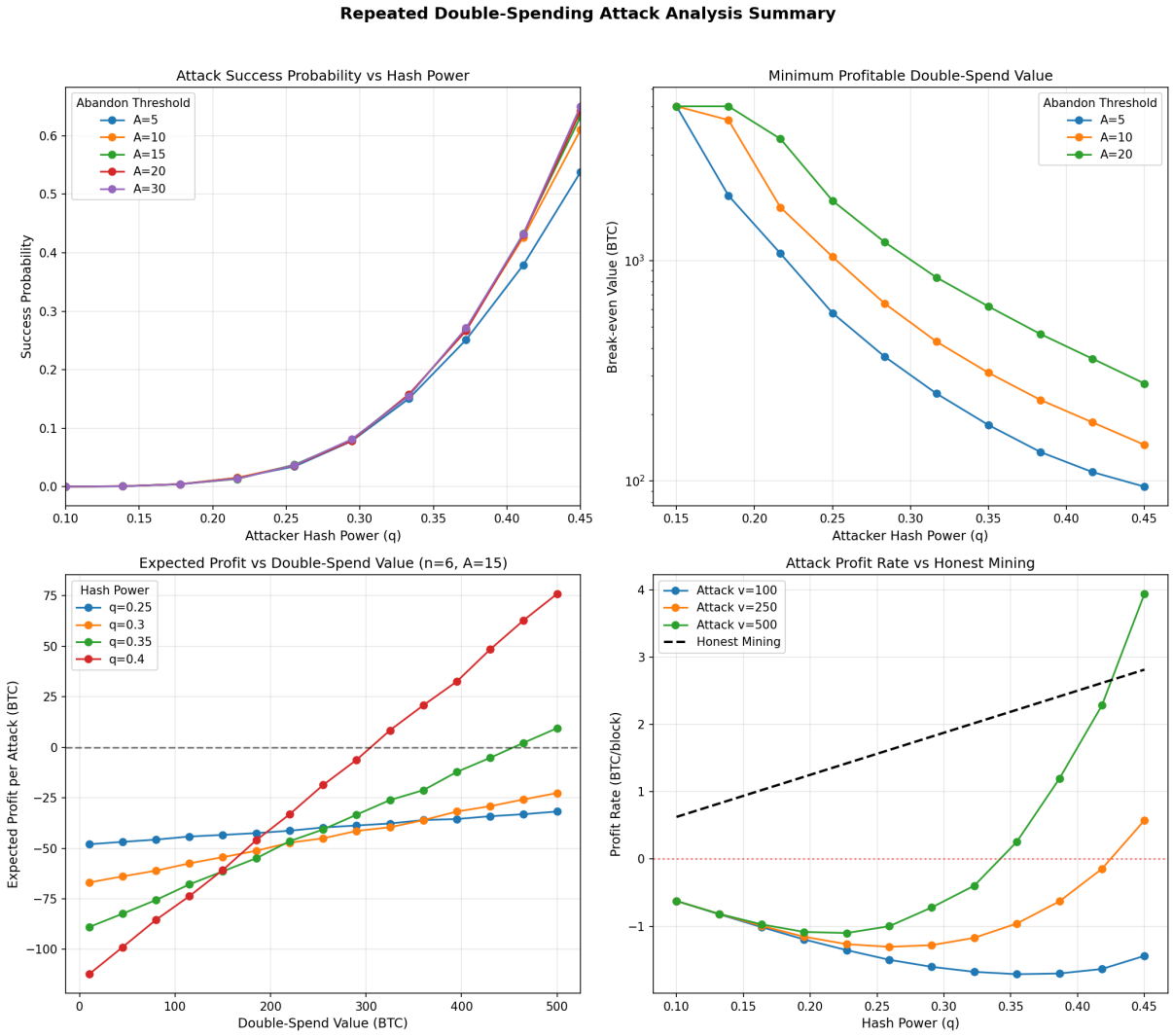

Figure 8: Summary of repeated double-spending attack analysis. **Top-left:** Attack success probability vs. hash power for different abandonment thresholds. **Top-right:** Break-even double-spend value (minimum profitable transaction value). **Bottom-left:** Expected profit vs. double-spend value for different hash power levels. **Bottom-right:** Attack profit rate compared to honest mining.

4.4 Task 4 Conclusions

1. **Viability of Repeated Attacks:**
   - Repeated double-spending attacks with abandonment threshold are viable for attackers with sufficient hash power ($q > 0.25$–0.30).
   - The abandonment strategy makes attacks practical by ensuring finite duration.

- Attackers must target high-value transactions to overcome opportunity costs.

2. **Critical Thresholds:**
   - $q < 0.20$: Attacks rarely profitable regardless of double-spend value.
   - $q = 0.30$: Break-even at approximately 100–200 BTC.
   - $q = 0.40$: Break-even at approximately 30–50 BTC.
   - $q \geq 0.50$: Attacker always wins (violates Nakamoto consensus assumption).
3. **Economic Security:** Bitcoin's security relies on the economic reality that honest mining is more profitable than attacking for miners below $\sim 40\%$ hash power. This creates a strong incentive for rational miners to remain honest.
4. **Defense Recommendations:**
   - **Standard transactions (< 100 BTC):** 6 confirmations adequate.
   - **High-value (100–1,000 BTC):** Consider 10–12 confirmations.
   - **Very high-value ($> 1{,}000$ BTC):** 20+ confirmations or additional verification.
   - **Exchanges:** Implement extra verification for large withdrawals.
5. **Theoretical Insights:**
   - The abandonment threshold transforms an infinite-variance random walk into a tractable repeated game.
   - The relevant metric for attackers is **profit rate** (per unit time), not profit per attack.
   - Repeated attacks allow low-probability strategies to compound over time.

5 Overall Conclusions

This project provides a multi-faceted analysis of Bitcoin's security from both statistical and game-theoretic perspectives.

5.1 Statistical Foundations (Task 1)

SHA-256 passes rigorous uniformity tests (chi-square and Kolmogorov–Smirnov), confirming that each hash attempt constitutes an independent, uniformly distributed trial. The resulting proof-of-work solution times follow an exponential distribution, validating Bitcoin's Poisson process model for block discovery.

5.2 Selfish Mining Vulnerability (Task 2)

Selfish mining is profitable with as little as 20–35% of network hash power, depending on network connectivity γ. The optimal strategy follows a simple, deterministic state machine with four actions (Adopt, Wait, Match, Override). This violates Bitcoin's "honest majority" security assumption and creates pressure toward mining centralization.

5.3 Decision Thresholds (Task 3)

One-shot mining decisions (orphan block mining, block withholding) have a threshold of $q = 50\%$, coinciding with the classical majority threshold. However, the full selfish mining strategy has a strictly lower threshold: $q = 1/3$ for $\gamma = 0$ and $q = 1/4$ for $\gamma = 1$, demonstrating that strategic block accumulation provides advantages beyond simple race-winning.

5.4 Double-Spending Economics (Task 4)

Repeated double-spending attacks with abandonment thresholds are viable for high-hash-power attackers targeting high-value transactions. However, for most realistic parameter regimes, honest mining remains more profitable – providing the economic foundation for Bitcoin's security. The key insight is that attackers should evaluate **profit rate** (per unit time), not just profit per attack.

5.5 Bitcoin's Security Model

Bitcoin's security ultimately rests on three pillars:

1. **Cryptographic:** SHA-256 provides uniform, unpredictable outputs (Task 1).
2. **Game-theoretic:** Honest mining is typically more profitable than attacking (Tasks 2–4).
3. **Economic:** Opportunity costs and block rewards align miner incentives with honest behavior (Tasks 3–4).

The vulnerability lies in the game-theoretic pillar: miners with sufficient hash power ($>$ 20–35%) *can* profitably deviate. Bitcoin's decentralization – ensuring no entity approaches these thresholds – is therefore essential for security.

References

[1] Nakamoto, S. (2008). *Bitcoin: A Peer-to-Peer Electronic Cash System*. https://bitcoin.org/bitcoin.pdf

[2] Eyal, I., & Sirer, E. G. (2014). Majority is not Enough: Bitcoin Mining is Vulnerable. *Financial Cryptography and Data Security* (FC 2014).

[3] Rosenfeld, M. (2014). Analysis of Hashrate-Based Double Spending. *arXiv preprint arXiv:1402.2009*.

[4] Sapirshtein, A., Sompolinsky, Y., & Zohar, A. (2016). Optimal Selfish Mining Strategies in Bitcoin. *Financial Cryptography and Data Security* (FC 2016).

[5] Grunspan, C., & Pérez-Marco, R. (2018). Double spend races. *International Journal of Theoretical and Applied Finance*, 21(08).

[6] Nayak, K., Kumar, S., Miller, A., & Shi, E. (2016). Stubborn Mining: Generalizing Selfish Mining and Combining with an Eclipse Attack. *IEEE European Symposium on Security and Privacy* (EuroS&P 2016).

[7] NIST (2015). *FIPS 180-4: Secure Hash Standard (SHS)*. National Institute of Standards and Technology.