Metadata of the chapter that will be visualized in SpringerLink

Book Title	Advances in Information and Communication	
Series Title		
Chapter Title	ADESS: A Proof-of-Work Protocol to Deter Double-Spend Attacks	
Copyright Year	2024	
Copyright HolderName	The Author(s), under exclusive license to Springer Nature Switzerland AG	
Corresponding Author	Family Name	**Aronoff**
	Particle	
	Given Name	**Daniel**
	Prefix	
	Suffix	
	Role	
	Division	
	Organization	Massachusetts Institute of Technology
	Address	Cambridge, MA, 02139, USA
	Email	daronoff@mit.edu
	URL	https://danieljaronoff.io
Author	Family Name	**Ardis**
	Particle	
	Given Name	**Isaac**
	Prefix	
	Suffix	
	Role	
	Division	
	Organization	Ethereum Classic Cooperative
	Address	New York, USA
	Email	

Abstract	A principal vulnerability of a proof-of-work (“PoW”) blockchain is that an attacker can re-write the history of transactions by forking a previously published block and building a new chain segment containing a different sequence of transactions. If the attacker’s chain has the most cumulative mining puzzle difficulty, nodes will recognize it as canonical. We propose a modification to PoW protocols, called *ADESS*, that contains two novel features. The first modification enables a node to identify the attacker chain by comparing the temporal sequence of blocks on competing chains. The second modification penalizes the attacker by requiring it to apply exponentially increasing hashrate in order to make its chain canonical. We demonstrate two things; (i) the expected cost of carrying out a double-spend attack is weakly higher under *ADESS* compared to the current PoW protocols and (ii) for any value of transaction, there is a penalty setting in *ADESS* that renders the expected profit of a double-spend attack negative.

Keywords (separated by '-')	Proof-of-Work - Blockchain - Consensus - Double-spend attacks

ADESS: A Proof-of-Work Protocol to Deter Double-Spend Attacks

Daniel Aronoff[1(✉)] and Isaac Ardis[2]

[1] Massachusetts Institute of Technology, Cambridge, MA 02139, USA
daronoff@mit.edu
[2] Ethereum Classic Cooperative, New York, USA
https://danieljaronoff.io

Abstract. A principal vulnerability of a proof-of-work ("PoW") blockchain is that an attacker can re-write the history of transactions by forking a previously published block and building a new chain segment containing a different sequence of transactions. If the attacker's chain has the most cumulative mining puzzle difficulty, nodes will recognize it as canonical. We propose a modification to PoW protocols, called *ADESS*, that contains two novel features. The first modification enables a node to identify the attacker chain by comparing the temporal sequence of blocks on competing chains. The second modification penalizes the attacker by requiring it to apply exponentially increasing hashrate in order to make its chain canonical. We demonstrate two things; (i) the expected cost of carrying out a double-spend attack is weakly higher under *ADESS* compared to the current PoW protocols and (ii) for any value of transaction, there is a penalty setting in *ADESS* that renders the expected profit of a double-spend attack negative.

Keywords: Proof-of-Work · Blockchain · Consensus · Double-spend attacks

1 Introduction

The protocol currently used in Bitcoin, Ethereum Classic and most other PoW cryptocurrencies instructs nodes to recognize as canonical the chain with the highest score, which is defined as cumulative mining puzzle difficulty (also referred to as "work"). A principal vulnerability of a blockchain is that an attacker can re-write the history of transactions by forking a previously published block and building a new chain segment containing a different sequence of transactions. If the attacker chain (denoted $\mathcal{A}$) has more cumulative mining puzzle difficulty compared to the incumbent canonical chain (denoted $\mathcal{IC}$), the PoW protocol instructs nodes to recognize the attacker chain as canonical. One motivation for forking a blockchain is to carry out a double-spend attack, whereby the attacker negates a transfer of its tokens that are recorded on chain $\mathcal{IC}$. The safety threat posed by double-spending is not just hypothetical. There have been instances of sizable attacks that have succeeded. For example, from

K. Arai (Ed.): FICC 2024, LNNS 920, pp. 1–27, 2024.
https://doi.org/10.1007/978-3-031-53963-3_11

2018 to 2020 there were several double spend attacks in Ethereum Classic and Bitcoin Gold[1]. The double-spend vector of attack raises questions about the security of transactions on a PoW blockchain.

We propose a modification to the protocols introduced by Satoshi Nakamoto in the Bitcoin White Paper (Nakamoto [9]) and by Gavin Wood in the Ethereum "Yellow Paper" (Wood [12]), which we refer to as the "Nakamoto" protocol. We name our protocol modification Absolute Discontinuous Exponential Subjective Scoring ("*ADESS*"), which is designed to increase security against double-spend attacks by raising the cost to make a fork chain canonical without significantly compromising other key dimensions of security and performance of the network and without using any new oracles. *ADESS* introduces two key modifications to current PoW protocols. One is a criteria for identifying an attacker chain which, unlike current *PoW* protocols, requires nodes to observe the temporal sequence of blocks. The other is an altered criteria for scoring chains after the attacker has been identified. *ADESS* is most effective in raising the cost of an attack when the mining puzzle difficulty is adjusted between short intervals of blocks.

1.1 The Two *ADESS* Modifications

The first *ADESS* modification is a criteria for identification of an attacker chain based on behavior associated with a double-spend attack. The intended victim, Bob, must believe that the transaction sending him tokens is appended to the canonical chain before he conveys an exchange item to the sender, Alice. When he observes the transaction appended to a block, Bob will wait until a few more blocks have been appended to the chain to confirm that the chain remains canonical before conveying the item to Alice. At the same time Alice does not want Bob to know that she is building her own chain, so she will not broadcast her chain until after she has received the item from Bob. We use Alice's delay in broadcasting her chain to form a criteria for identification. Roughly, when comparing two chains with a common ancestor block (the "fork-block"), a penalty is assigned to the chain that was last to broadcast a minimum number of successive blocks, starting from the fork-block.

The second *ADESS* modification is to penalize the identified attacker chain. When chain $\mathcal{A}$ is identified the criteria for choosing the canonical chain is changed. Chain $\mathcal{IC}$ and chain $\mathcal{A}$ are assigned scores based on the number of post-fork blocks, with a discount applied to each block in chain $\mathcal{A}$. This requires chain $\mathcal{A}$ to grow at a faster rate than chain $\mathcal{IC}$ to become canonical. To speed up the growth rate, an attacker must increase the amount of hashrate, i.e. puzzle solution guesses per unit of time, on chain $\mathcal{A}$ in expectation. The elevated growth rate causes an increase in mining puzzle difficulty, which requires the attacker to increase hashrate further after each puzzle adjustment in order to maintain the elevated chain growth rate. This process leads to an exponential increase in

[1] For Ethereum Classic see Andrew Singer [11] and James Lovejoy [6]. For data on double-spend attacks on PoW cryptocurrencies prior to 2020, see the MIT Digital Currency Initiative 51% reorg tracker [7]

the hashrate an attacker must apply to exceed its penalty growth rate. Hashrate consumes electricity, so the attacker's cost increases exponentially over time. The extent to which mining expenditure exceeds the block reward is a sunk cost which the attacker cannot recover even if its chain becomes canonical and it receives block rewards.

The purpose of *ADESS* is to confront a double-spend attacker with the prospect, in terms of ex-ante expected value, of an exponentially increasing non-recoverable cost to carry out its attack.

1.2 Roadmap

The rest of the paper is organized as follows. In Sect. 2 we review the Nakamoto protocol for validating transactions in PoW blockchains and we describe a prototypical double-spend attack. Then we characterize the claim made by and Budish [1] and Gervais et al. [3] that a double-spend attack can be low cost, followed by the rebuttal of Moroz et al.'s [8] that retaliation by an intended victim induces a war of attrition between attacker and victim in which the outcome is uncertain. We conclude that Nakamoto neither promotes, nor discourages, double-spend attacks. Section 3 begins with a statement of the intended goal of *ADESS* and then describes the two parts of the *ADESS* protocol modification. The first part is the rule for assigning a penalty to a fork chain. The second part is the operation of the penalty. Section 3.4 states Proposition 1, which demonstrates that the cost of carrying out a double-spend attack under *ADESS* is weakly higher compared to Nakamoto. In Sect. 4 we present a baseline model for evaluating the properties of *ADESS* and characterize the attacker's decision problem. Section 5 states Theorem 1, which demonstrates that the *ADESS* penalty can be tuned to render a double-spend attack ex-ante unprofitable for any value of transaction. Section 6 examines the implications, in terms of blockchain performance, of relaxing two of the four constraints that were placed on the baseline model; partial adjustment of mining puzzle difficulty and network latency. In Sect. 7 we compare the performance of *ADESS* and Nakamoto along two security related dimensions; malicious attacks and unobserved forks. In the Appendix, we examine the implications of relaxing the remaining constraints that were placed on the baseline model; multiple chains and non-constant blockchain growth rates.

2 Related Work: The Double - Spend Vulnerability

In this section we first describe the mechanics of a double-spend attack. This serves as a foundation for a review of contributions to the literature on the vulnerability of a *PoW* blockchains to double-spend attacks, which provide the background and motivation for our proposal to modify the Nakamoto protocol.

2.1 A Double - Spend Attack Under the Nakamoto Protocol

We employ the following framework. The first block of a blockchain is called the "genesis block" and is assigned the number 0. The number assigned to the child

block is the parent block number plus 1. A blockchain has a tree structure in which the genesis block is the root and chain segments form branches that can fan out as the chain grows. A fork-block is the common ancestor block of two or more post- fork chain segments (which we refer to as "chains"). Block #98 in Fig. 1 is the common ancestor of chains $\mathcal{IC}$ and $\mathcal{A}$. Two blocks on different post-fork chains are assigned the same number if each block is the same number of blocks away from the Genesis block. In Fig. 1 each chain has a block #100. We assume no latency. Each node receives broadcasts instantaneously.

A double-spend attack is an operation whereby a node can negate the transfer of tokens it sent to a counter-party after receiving the exchange item from the counter-party. Here is how an Alice node double-spends a Bob node. Alice sends a token to a counterparty, Bob, which is appended to block #100 on chain $\mathcal{IC}$. Alice secretly builds a chain forking from an earlier block, #98, applies more hashrate to her chain $\mathcal{A}$ compared to chain $\mathcal{IC}$, in order to ensure that her chain grows faster in expectation, and appends to a block in chain $\mathcal{A}$ a transaction whereby she sends all of the tokens in the wallet from which she paid Bob, to other wallets, possibly owned by her. After Alice receives the exchange item from Bob - which may take place only after Bob observes several child blocks - provided chain $\mathcal{A}$ has more blocks, she broadcasts chain $\mathcal{A}$. Thereupon, in accordance with the Nakamoto protocol, the other nodes accept Alice's chain as the true, or 'canonical' chain. Because Alice emptied her wallet on an earlier block, her transfer to Bob is no longer valid.[2] Alice has succeeded in retrieving her tokens after collecting the exchange item from Bob. The double-spend attack is displayed in Fig. 1. Note that, in order for the attacker chain $\mathcal{A}$ to become canonical, it must have more cumulative mining puzzle difficulty at some future time, starting from block #98, compared to the incumbent chain $\mathcal{IC}$.[3]

2.2 Evaluating Nakamoto's Claim of Double-Spend Deterrence

In the 2008 Bitcoin White Paper (Nakamoto) [9], Satoshi Nakamoto recognized the vulnerability of the protocol to the possibility of a double-spend attack. Nakamoto offered two arguments in support of the idea that the vulnerability is limited. The first argument is that the cost of to acquiring the hashrate required to carry out a double-spend attack is a deterrent.[4] The second argument is that a miner with sufficient hashrate to carry out a double-spend attack would not risk the depreciation in the exchange value of the currency that a double-spend

[2] Alice's transfer to Bob remains in the pool of transactions that could be appended by a miner to a block in chain $\mathcal{A}$. However, since her original wallet is empty, the transaction would be invalid, as there are no tokens in her wallet to send to Bob.

[3] The ranking of chains in terms of mining puzzle difficulty does not necessarily match the ranking in terms of hashrate. The reason is that the rate of solving puzzles is stochastic. A lucky string of quickly solved puzzles on one chain will generate more blocks - and more cumulative puzzle difficulty - on that chain compared to another chain with more hashrate but less good luck in solving puzzles.

[4] Nakamoto [9] Sect. 11, pp. 6–7.

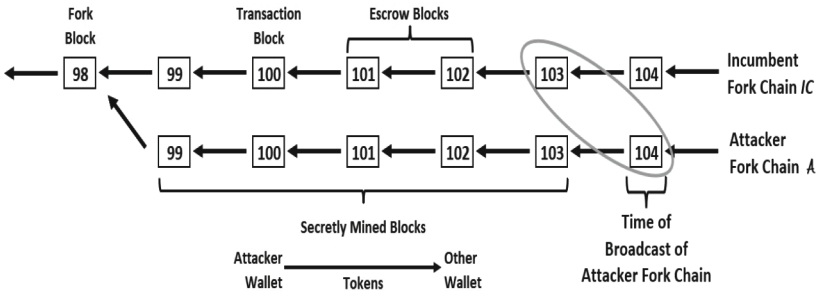

Fig. 1. A double-spend attack

attack might cause[5]. Budish [1] and Gervais et al. [3] cast doubt on the former claim by showing that a double-spend attack on Bitcoin could be profitable for a relatively modest value transaction.[6]

We construct an example to elucidate this point. To become canonical chain $\mathcal{A}$'s work must exceed chain $\mathcal{IC}$'s work at some time. When chain $\mathcal{A}$ becomes canonical, the attacker retains the double-spend transaction v and it will receive the block rewards on chain $\mathcal{A}$. Equation 1 is the ex-ante expected profit of a double-spend attack where chain $\mathcal{A}$ has N post-fork blocks at the time it becomes canonical. It reflects an attacker strategy of forking the head (i.e. rightmost block) of the canonical chain when it sends the transaction into the transaction pool and applying the same hashrate to chain $\mathcal{A}$ as is applied to chain $\mathcal{IC}$ (normalized to 1 unit of hashrate per block) until the victim conveys the exchange item - at block $N-1$. In expectation this implies that chain $\mathcal{A}$ will grow at the same rate and accumulate the same mining puzzle difficulty through post-fork block $N-1$ (which is normalized to 1 block per unit of time). After block $N-1$ the attacker adds an additional ϵ of hashrate to ensure that chain $\mathcal{A}$ has greater cumulative mining puzzle difficulty in expectation when it is broadcast at block N. We study this strategy because it can be implemented on any PoW blockchain regardless of the frequency or rate of adjustment of mining puzzle difficulty.

The notation is as follows. p_B - assumed constant for all blocks - is the block reward paid to the miner who is first to solve the puzzle. c is the attacker's cost, in dollars, of a unit of hashrate - defined as the cost of operating one mining computer for 1 unit of time. $\delta \in (0,1]$ is the discount rate per unit of time. The cost c is incurred at each block and the revenue is realized at the end, when chain $\mathcal{A}$ is broadcast. The crypto to dollar exchange rate is 1. There is no latency in the network.

[5] Nakamoto [9] Sect. 6, p. 4 states "[A miner] ought to find it more profitable to play by the rules, such rules favor him with more new coins than everyone else combined, than to undermine the system and the validity of his own wealth."

[6] Budish [1]) also pointed out that a short seller of Bitcoin could profit from a disruption that caused the exchange value to decline.

Attacker ex ante expected profit =

$$\mathrm{E}[\underbrace{\delta^{N-1}(v+p_B N)}_{\text{discounted revenue}} - \underbrace{c\Big[\sum_{n=1}^{N-1}\delta^{n-1}+(1+\epsilon)\delta^{N-1}\Big]}_{\text{discounted cost}}] \quad (1)$$

The minimum profitable transaction can be expressed by the following inequality.

$$v > \delta^{-(N-1)}c\Big\{\Big[\sum\nolimits_{n=1}^{N-1}\delta^{n-1}+(1+\epsilon)\delta^{N-1}\Big]-\delta^{N-1}Np_B/c\Big\}$$

Since the time interval between blocks is short - approximately 13 s for Ethereum and 10 min for Bitcoin - the time discount factor will be very close to 1. Taking the limit as $\delta \to 1$ yields the expression for minimum profitable transaction size.

$$v > (c-p_B)N + c\xi \quad (2)$$

The first term reflects the attacker's profit from mining chain $\mathcal{A}$. The second term is the cost of the additional hashrate applied to the Nth post-fork block. When the attacker and incumbent hashrate cost is identical, $(c = p_B)$ is the market rate of profit and can be normalized to zero. In that case, the expression for minimum profitable transaction size is

$$v > c\xi \quad (3)$$

In Eqs. 2 and 3 ξ can be arbitrarily small. We conclude that the minimum transaction size required to render the ex-ante expected value of double-spend attack profitable is an increasing function of the spread between the cost of hashrate and the block reward $(c - p_B)$. When the attacker and incumbent hashrate cost is identical, the transaction size can be arbitrarily small.[7]

[7] Another dimension from which the double-spend attack can be analyzed is the expected profit from an attack using less than the honest miner hashrate. For example, an attack in which 30% of the hashrate applied to chain $\mathcal{IC}$ was applied to chain $\mathcal{A}$ would, in expectation, recoup 30% of its hashrate cost from block rewards and earn a third of the value of the transaction. In our example, the minimum value of the transaction required to make the attack profitable would be $v > 3\big(c\xi + (2/3)N\big)$. See Gervais et al. [3] for a detailed simulations in a Markov Decision process framework. Gervais et al. also evaluate the relationship of the stale block rate and the block reward to the expected profit of a double-spend attack. We do not evaluate those dimensions.

2.3 Retaliation

Moroz et al. [8] pointed out that a victim of a double-spend attack could retaliate by increasing the hashrate applied to the incumbent chain $\mathcal{IC}$. The possibility of retaliation means the blockchain may be less vulnerable to attack than is implied by the analyses of Budish or Gervais et al., neither of which address the possibility of retaliation. In the resulting war of attrition both attacker and victim are confronted with a similar marginal decision of whether to add another block[8]. In the model each numbered block (one for each chain) constitutes a round and the two chains add blocks at the same constant rate.[9] At the end of a round, each player decides whether to mine a block and enter the next round. Equation 4 depicts a variant of the Moroz et al. (2020) model. The expected payoff, for either player entering the next round $(N+1)$, assuming its opponent exits, is

$$\mathrm{E}[\quad \underbrace{v + p_B(N+1)}_{\text{revenue from becoming canonical}} \quad - \quad \underbrace{c(1+\gamma)^{N+1}}_{\text{cost of mining next block}} \quad] \tag{4}$$

There are equilibrium strategies that support a range of potential outcomes in this symmetric war of attrition. To achieve a determinate outcome would require a restriction on the strategy space or an equilibrium refinement.

2.4 A Deterrence Deficit

Nakamoto's claim that Bitcoin (and by extension, PoW blockchain protocols generally) provide incentives that minimize the security risk of a double-spend attack is not supported. Under free-entry and uniform hashrate costs a profit could be made from double-spending a transaction of any size. Moreover, when an attack is carried out, the possibility of counter-attack by the victim renders the outcome uncertain. The conclusion is that Nakamoto on its own neither deters nor promotes double-spend attacks. Any deterrence must come from outside the protocol. For example, a limitation on mining computers that elevates the attacker's hashrate cost above the honest miner's cost; or financing constraints that limit the capital an attacker can use to carry out a attack. We next begin our discussion of *ADESS*, which incorporates a double-spend attack deterrent inside the protocol.

3 The *ADESS* Protocol Modification

ADESS modifies Nakamoto by adding two protocols. The first addition is a criteria for a node to identify the attack chain and assign a penalty. This is the

[8] In the Moroz et al. [8] model, it is assumed that the victim and the attacker build the chains. The other miners step aside until the bifurcation is resolved.

[9] The Moroz et al. (2020) model assumes the puzzle difficulty remains constant, which implies $\gamma = 0$.

first subjective discontinuity of *ADESS*. The second addition is the application of the penalty to the attack chain. The attack chain's score changes from cumulative mining puzzle difficulty to weighted chain length, where the length of the penalized chain is discounted. When a node observes the discounted length of the penalized chain to exceed the un-discounted length of the incumbent canonical chain, the attack chain becomes canonical and its score switches back to cumulative mining puzzle difficulty. This is the absolute exponential scoring and the second subjective discontinuity of *ADESS*.

3.1 The Penalty Assignment

The objective is to assign the penalty to the chain that is built by the double-spend attacker. The proposed assignment rule is designed to optimize over two goals.

(i) Minimize the likelihood that the penalty is assigned to the chain to which the transaction is appended, and
(ii) Maximize the likelihood that the penalty is assigned to the attacker's chain.

The Victim's Incentive to Wait for Transaction Confirmation. When the transfer of tokens is first appended to a block on the canonical chain, the recipient of the tokens has an incentive to require that the chain remain canonical for one or more additional confirmation blocks before the exchange item is released to its counterparty (we refer to the number of confirmation blocks, inclusive of the block to which the transaction is appended, as "confirmation depth"). One motivation is to ensure that the receipt of tokens by a node is not later negated as a result of another chain becomes canonical. So-called "uncle-chains" are forks off of the canonical chain that are typically abandoned after a few blocks as consensus forms on a single chain. Uncle-chains arise with considerable frequency on some networks. For example, approximately 8.5% of valid blocks produced in the Ethereum network since inception are part of post-fork chains that were eventually abandoned.[10,11] On the other hand, the recipient will prefer to complete the transaction sooner rather than later.

This suggests a tradeoff between the recipient's desire for security - which increases with the passage of time - and the discounted value of the transaction - which decreases with the passage of time. Guo and Ren [4] elegantly formalized and proved the tradeoff as a relation between confirmation blocks k, block propagation delay upper bound Δ, mining rate λ and the fraction of honest hashrate ρ. Equation 5 is an upper bound on the probability that the recipient's receipt

[10] https://etherscan.io/uncles

[11] Another reason for increasing confirmation depth may be to protect against the possibility of a double-spend attack. However, as we demonstrated in Sect. 2.2, Nakamoto does not intrinsically deter double-spend attacks at any confirmation depth. Any security provided by increasing confirmation depth under Nakamto arises from the contingent circumstances in which the transaction takes place.

of tokens is later negated when a majority of hashrate is controlled by honest miners (subject to restrictions on $\lambda\Delta$).[12]

$$\left(2 + 2\sqrt{\frac{1}{p-1}}\right)4p(1-p)^k, \quad \text{where } p = \rho\epsilon^{\lambda\Delta} \tag{5}$$

When the attacker applies less than half the hashrate ρ, an increase in the number of confirmation blocks k, improves security by decreasing exponentially the probability that the receipt of tokens by a node is later reversed (i.e. by decreasing Eq. 5). We define the lower bound to confirmation depth $k = \alpha$ for a node as that which optimizes the tradeoff between security and timeliness when the node places a zero probability of being the target of a double-spend attack, i.e. $\rho = 1$. This forms a lower bound to the confirmation depth the node actually chooses. When a node is concerned that it might be the target of an attack, $\rho < 1$, it will need to increase its confirmation depth to maintain its security level. For the purposes of our analysis we will assume that α is the same for all nodes.[13]

The Attacker's Incentive to Build Its Chain in Secret. The attacker has an incentive to privately build its chain until after it has received the exchange item from its victim for two reasons. First, the broadcast of the attacker chain reveals that the attacker has emptied its wallet on the chain, which could alert the victim to the attack. Second, if the attacker broadcast a chain that had the most work, it would be canonical and the block to which the victim's receipt of tokens is appended would become part of an uncle chain, which would nullify the victim's receipt of tokens. Recall from Fig. 1 that the fork block for the attacker's chain $\mathcal{A}$ must be at a lower height than the block containing the transaction. Therefore, at the minimum, an attacker will not broadcast its chain for an interval of α blocks, counting from the fork-block.[14] Given the assumption that α is a lower bound to the confirmation blocks of all nodes, it follows that no double-spend attacker will broadcast its chain until after its victim has observed α confirmation blocks.

The Penalty Assignment Rule. These considerations suggest that (i) and (ii) can be optimized by exempting from the penalty the first chain that is observed to have an interval of α valid post-fork blocks and penalizing all other chains that start from the same fork block. This assignment rule reflects the equilibrium behavior of the attacker and the victim. At α blocks, the frequency of uncle blocks is low, which minimizes the probability that the transaction is

[12] Equation 5 is part of Theorem 1 of Gou and Ren (2022). The precondition in Theorem 1 requires that $\lambda\Delta < ln(\rho)ln(1/2)$.

[13] Alternatively, α can be viewed as the lower bound of node confirmation blocks.

[14] Guo and Ren [4] prove that any successful double-spend attack strategy can be carried out privately. It follows that an attack strategy that is optimal when the attacker chain is built in secret, is globally optimal.

on a chain that will subsequently be abandoned (goal (i)). The attacker will not broadcast its chain before α confirmation blocks have appeared, which ensures it will be penalized (goal (ii)).

Penalty Assignment Rule. When a node observes two chains with a common fork-block ancestor and at least one of the chains is of length α blocks, the chain that was first observed to achieve length α blocks is called "chain $\mathcal{IC}$". The other chain is called "chain $\mathcal{A}$". The penalty is assigned to chain $\mathcal{A}$. From the time of the penalty assignment to the time of the removal of the penalty, chain $\mathcal{IC}$ is canonical. □

3.2 The Penalty Function

When a penalty assignment has been made, *ADESS* diverges from the Nakamoto protocol. The canonical chain is chosen by comparing the number of post-fork blocks between chain $\mathcal{A}$ and chain $\mathcal{IC}$, rather than cumulative mining puzzle difficulty. In order to become canonical, the length of a chain $\mathcal{A}$ must exceed chain $\mathcal{IC}$ by some number of blocks. We refer to the manifold formed by the combinations of blocks - on chain $\mathcal{A}$ and chain $\mathcal{IC}$ - at which chain $\mathcal{A}$ becomes canonical, as the "canonical boundary". When a penalized chain $\mathcal{A}$ has crossed the canonical boundary, the penalty ceases to apply. The protocol for determining the canonical chain reverts to a comparison of cumulative mining puzzle difficulty.[15]

Penalty Function. The scoring formula has two elements.

(i) The scoring criteria for chain $\mathcal{IC}$ and chain $\mathcal{A}$ switches from cumulative mining puzzle difficulty to number of blocks. A per-block penalty weight of $\frac{1}{1+\xi}$ is applied to each block on chain $\mathcal{A}$, starting at its first post-fork block and continuing until the time that chain $\mathcal{A}$ is observed to have reached the canonical boundary. During the time interval that the penalty is applied to chain $\mathcal{A}$, the blocks on chain $\mathcal{IC}$ are unweighted. The score of an interval of N blocks on a chain $\mathcal{A}$ is $\frac{1}{1+\xi}N$ compared to a score of N for chain $\mathcal{IC}$.

(ii) After chain $\mathcal{A}$ has reached the canonical boundary, the protocol reverts to the Nakamoto criteria of comparing cumulative mining puzzle difficulty between the two chains, with a re-set of chain $\mathcal{A}$'s score. At the point of crossing, the score of chain $\mathcal{IC}$ is re-set to equal the cumulative mining puzzle difficulty of chain $\mathcal{IC}$, plus a small additional amount ϵ - which makes chain $\mathcal{A}$ canonical. □

Figure 2 displays the two elements of the penalty function. No penalty is assigned until α post-fork blocks on chain $\mathcal{IC}$ have been appended. When that

[15] The *ADESS* criteria for comparing chains after chain $\mathcal{A}$ has reached the canonical boundary is explored in Appendix.

occurs, the penalty is assigned to chain $\mathcal{A}$, starting retroactively from the fork-block. In order to become canonical, chain $\mathcal{A}$ must have $(1+\xi)N$ blocks at a time when chain $\mathcal{IC}$ is of length N.[16]

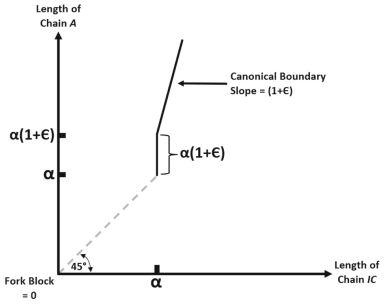

Fig. 2. The penalty function

3.3 The Exponentially Increasing Sunk Cost of a Double-Spend Attack

Consider the case where there is no latency, target time interval and initial hashrate are 1 and puzzle difficulty fully adjusts after each block. In that case, the expected hashrate required to append the next block in the target time interval is equal to the expected hashrate applied to the prior block. Suppose the attacker applies a constant growth rate of γ.[17]

The interaction of a growth rate above target with the mining puzzle difficulty adjustment imposes an ex-ante expected exponentially increasing cost to carry out a double-spend attack. At the first post-fork block the attacker applies $(1+\gamma)$ units of hashrate. At the second block, after the mining puzzle difficulty has increased to require $(1+\gamma)$ units of hashrate to achieve the target, the attacker must apply $(1+\gamma)^2$ units of hashrate to maintain its growth rate. At each post-fork block n the attacker must apply $(1+\gamma)^n$ units of hashrate. The increase in the attacker's per block cost above the block reward p_B is a sunk cost that it cannot recover from block rewards when chain $\mathcal{A}$ crosses the canonical boundary. Figure 3 displays the sunk cost that the exponentially increasing penalty imposes

[16] It is worth pointing out that *ADESS* conforms to the Axioms of Leshno and Strack [5] since it does not alter the underlying Nakamoto entry and reward structure for miners.

[17] The target blockchain growth rate T can be represented as a Bernoulli Process $\mathbb{E}[T] = D/h$, where D denotes mining puzzle difficulty; the probability of guessing the puzzle solution is $1/D$ and hashrate is h. In our example $T = 1$ and initial $h = 1$. The Bernoulli Process yields $D = 1$ at the first block. Full adjustment implies that an increase of hashrate to $1+\gamma$ causes difficulty to increase to $D = 1+\gamma$ and so forth.

on the attacker. It is the ex-ante expectation of exponentially increasing sunk cost that provides the deterrent to launching a double-spend attack.

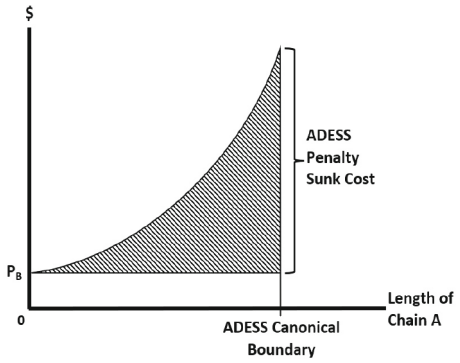

Fig. 3. The ex-ante expected exponentially increasing sunk cost of a double-spend attack

3.4 The Intrinsic Deterrent to Double-Spend Attacks

The Penalty Assignment Rule and the Penalty Function ensure that, to succeed, a double-spend attacker must use more hashrate under *ADESS* compared to Nakamoto. This is a direct result of the penalty and stands independent of any strategic considerations or models. The key implementation issue is the choice α. We do not, at present, know the lower bound of confirmation blocks. A goal of future research is to attempt to estimate α. Until then, it may be reasonable to use an existing convention, such as Nakamoto's six-block confirmation rule for Bitcoin. It is also possible that, once chosen, α will become a Schelling focal point whereby recipient nodes wait at least α confirmation blocks before sending their exchange item.[18] Proposition 1 states the relative increase in hashrate required to carry out a double-spend attack under *ADESS*.

Proposition 1 (The Increased Cost of Attack Under *ADESS***).** *The ex-ante expected hashrate required to carry out a double-spend attack under ADESS, where the penalty is bounded away from zero, i.e.* $\xi > e$ *for some* $e > 0$*, is weakly greater than the ex-ante expected hashrate required to carry out a double-spend attack under Nakamoto.*

Proof. Consider any post-fork block N on chain $\mathcal{IC}$ at which chain $\mathcal{A}$ becomes canonical under *ADESS*. The expected hashrate is $N(1+\xi)$ if $N < \alpha$ or there is no mining puzzle adjustment at any block. A mining puzzle adjustment at any block increases the expected hashrate for succeeding blocks by a exponential

[18] Schelling [10].

factor. Under Nakamoto, the attack strategy from Sect. 2.2 to become canonical at block N on chain $\mathcal{IC}$ is $N + e$. Since $\xi > e$, it follows that the expected hashrate under $ADESS$ always exceeds the expected hashrate under Nakamoto. The proposition follows from the fact that hashrate costs money. □

4 A Model of *ADESS*

In this Section, we formalize a model of the $ADESS$ protocol and evaluate its resilience to double-spend attacks under a set of baseline conditions. The model applies to an attacker who is deciding whether to launch a double-spend attack on a single node to which is has sent tokens in a single transaction. It evaluates the attacker's 'go', 'no go' decision to launch an attack, based on the ex-ante expected profit from an optimal attack strategy. It does not address any change in the attacker's plan after the attack is launched.[19] We make two assumptions that simplify the analysis and enable us to focus on how the $ADESS$ protocol affects the decision to launch a double-spend attack. The first assumption limits the strategic space in the game played by miners. The second assumption imposes a deterministic chain growth rate.

4.1 Mining Game Structure

There is a single attacker who attempts to double-spend a single counterparty. The attacker mines in secret until it has achieved two milestones; chain $\mathcal{A}$ has crossed the canonical boundary and the attacker has received the exchange item from its victim, whereupon the attacker broadcasts chain $\mathcal{A}$ to the network. There are multiple honest miners of chain $\mathcal{IC}$. These miners do not engage in double-spend attacks. They only become aware of a double-spend attack when the attacker broadcasts its chain. If chain A meets the criteria to be canonical, the honest miners switch from mining chain $\mathcal{IC}$ to mining chain $\mathcal{A}$. Honest miners may (or may not) have rational expectations about the possibility of attack and may require a premium profit to compensate for the risk of losing their escrowed post-fork block rewards on chain $\mathcal{IC}$ if an attack occurs and chain $\mathcal{A}$ becomes canonical. The model is agnostic as to the competitiveness of the mining market. The attacker is the only strategic player in the game we analyze. We characterize the equilibrium by evaluating the attacker's decision to attack.

Assumption 1 (Strategic Space). *The attacker makes a strategic choice of whether and when to launch a double-spend attack. Honest miners work on the canonical chain.* □

4.2 Certainty Equivalence for Blockchain Growth Rate

Miners make a succession of independent guesses of puzzle solutions over time. The time distribution of new blocks is a Bernoulli Process parameterized by

[19] The model derives an upper bound to double-spend vulnerability, since an attack, once launched, may be discontinued before it reaches completion.

the hashrate and puzzle difficulty, with the mean reverting to the target growth rate by periodic adjustments to puzzle difficulty. This random process can create opportunity for strategic behavior at any point in time, for example when a miner solves a puzzle faster than the target, as analysed by Eyal and Sirer [2]. However, the decision to launch an attack, which *ADESS* is designed to influence, is based on, inter alia, the ex-ante expected growth rate of the blockchain, which is a deterministic function of hashrate and puzzle difficulty. In order to focus on the decision to launch a double-spend attack, our model uses the certainty equivalence of the ex-ante expected growth rate.

Assumption 2 (Certainty Equivalence of Mining Puzzle Solutions). *Blockchain growth is a deterministic function of hashrate and mining puzzle difficulty.* □

4.3 The Baseline Case

Initially we impose four restrictions on the model. The first two are relaxed in Sect. 6 and the latter two are relaxed in the Appendix.

(i) After each new block, the mining puzzle difficulty fully adjusts - to set the growth rate at 1 block per unit of time at the hashrate applied to the previous block.
(ii) All nodes observe blocks in the same temporal order.
(iii) There is one attacker chain $\mathcal{A}$ and one victim.
(iv) The attacker chooses a constant growth rate γ for chain $\mathcal{A}$.

In order to reduce the number of variables in the analysis we impose several additional restrictions throughout which do not affect the implications of the model. The growth rate target is 1 block per unit of time; hashrate on chain $\mathcal{IC}$ is 1 per unit of time; the block reward, p_B, is constant; the attacker faces the same cost of 1 unit of hashrate, c - in dollars - as the honest miners; the profit from honest mining is normalized to zero (i.e. $P_B = c$)[20]; the crypto to dollar exchange rate at 1 and the time discount rate at $\delta \in (0, 1]$. Finally, we assume the attacker has a consistent estimate of the victim's confirmation depth, which may exceed α as defined in Sect. 3 based on, inter alia, the transaction value or the identity of, or its relationship with, the sender. In a slight abuse of notation, we use α to denote the confirmation depth in the model of this section.[21]

[20] The normalization reflects that the attacker's profit from mining is equal to the market return on investment. This may, or may not, imply a restriction on mining market structure. For example, if a mining oligopoly resulted from active miners having a lower cost of hashrate compared to other miners, the attacker would not earn an extra-normal return from its mining. On the other hand, a low cost miner does not need to create a fork (i.e. launch an attack) in order to earn an extra-normal profit. It can earn the profit so by mining chain $\mathcal{IC}$.

[21] Confirmation depth α is an equilibrium outcome of a game between the counterparties that we do not model.

The Attacker's Strategic Plan. We decompose the attacker's strategic plan into three parts. The first decision is the choice of which block to fork and when to begin building chain $\mathcal{A}$. The second decision is the block on $\mathcal{IC}$ at which chain $\mathcal{A}$ reaches the canonical boundary. The third decision is when to broadcast chain $\mathcal{A}$. We evaluate each decision.

As to the first decision, the attacker can start to build chain $\mathcal{A}$ by forking a block on the canonical chain at any time. The optimal choice of starting time is the solution of a dynamic optimization problem. A complicating factor in the analysis is that the wait time for the transaction to appear on a block is stochastic. There are two levels of uncertainty. One is the probability of a transaction being chosen by a miner, which can be affected inter-alia by the tip fee offered by the attacker relative to other tip fees in the transaction pool. The other is the probability of a miner being first to solve the mining puzzle. We denote the number of blocks on chain $\mathcal{IC}$ between the fork-block and the block onto which the transaction is appended by the random variable σ.

We do not solve the full dynamic programming problem, however, we are able to show that the attacker will fork the head of the canonical chain. Suppose the attacker forks a block that is τ blocks earlier than the head of chain $\mathcal{IC}$. In order for chain $\mathcal{A}$ to cross the canonical boundary when chain $\mathcal{IC}$ has N post-fork blocks, it needs to append $N(1+\xi)$ blocks in $N-\tau$ units of time. Denote γ as the growth rate of the attacker chain required to reach the canonical boundary at that time. The optimal τ is derived by solving Eq. 6 to minimize the hashrate required to cross the canonical boundary.

$$\underbrace{N(1+\xi)}_{\text{chain } \mathcal{A} \text{ blocks}} = N \underbrace{(1+\gamma)}_{\text{growth rate to reach canonical}} - \tau \tag{6}$$

Equation 6 implies that setting $\tau = 0$ minimizes cost (since the growth rate is a function of hashrate which has a unit cost c), which means the attacker will fork an end-block. This solution, in turn, implies that the attacker will choose the growth rate $(1+\xi)$, which matches the penalty.[22] We do not evaluate the decision of which head-block of the canonical chain the attacker will commence building chain $\mathcal{A}$. Rather, we analyse the attacker's decisions contingent on having forked a blockchain head.

As to the second decision, Eq. 7 is the attacker's problem for choosing the length of the attack, which is the block N on chain $\mathcal{IC}$ at which it reaches the canonical boundary.[23] The objective is to minimize cost.

$$\arg\min_{N} \text{ cost of attack } = c \sum_{n=0}^{\lceil N(1+\xi)-1 \rceil} \delta^{n/(1+\xi)} \Big(\underbrace{1+\xi}_{\text{Growth rate}} \Big)^n \tag{7}$$

[22] The problem can be stated as $\arg\min_{\tau} \gamma = N(1+\xi) + \tau + 1$.

[23] The exponent of the discount rate is divided by the growth rate to adjust for the intervals of time between blocks on chain $\mathcal{A}$ prior to reaching the canonical boundary, which is less than 1. The bracketed expression $\lceil N(1+\xi) - 1 \rceil$ indicates that the number of blocks on chain $\mathcal{IC}$ is rounded up from the value inside the brackets.

Subject to $N \geq \sigma + \alpha$, which is the minimum interval over which the penalty is applied. It is immediate that cost is increasing in N. We conclude that chain $\mathcal{A}$ will reach the canonical boundary at exactly $N = \alpha + \sigma$ post-fork blocks on chain $\mathcal{IC}$.

The attacker's third decision is the time to broadcast chain $\mathcal{A}$. The question is whether the attacker will continue to mine chain $\mathcal{A}$ in secret past the time it reaches the canonical boundary. Let B denote the blocks that the attacker secretly mines after chain $\mathcal{A}$ has reached the canonical boundary. The attacker will have to apply at least 1 unit of hashrate on chain $\mathcal{A}$ on each of the B blocks in order to remain canonical when it is broadcast. This follows from the Penalty Function, which states that once it has reached the canonical boundary, chain $\mathcal{A}$'s score is adjusted to the cumulative mining puzzle difficulty on chain $\mathcal{IC}$ plus a small ϵ. Chain $\mathcal{A}$ needs to keep pace with chain $\mathcal{IC}$ to ensure it will be canonical when it is broadcast. Equation 8 is the problem the attacker solves to determine when to broadcast its chain.[24]

$$\arg\max_{B} \pi(B;\xi,N,v,p_B) = \underbrace{\delta^{(\alpha+\sigma)+B-1}\{v + p_B(\lceil N(1+\xi)\rceil + B)\}}_{\text{discounted revenue}} - \underbrace{c\Big[\sum_{n=0}^{\lceil(\alpha+\sigma)(1+\xi)-1\rceil}\{\delta^{n/(1+\xi)}(1+\xi)^n\} + \sum_{b=0}^{B-1}\delta^{(\alpha+\sigma)+b}\Big]}_{\text{discounted cost}} \quad (8)$$

Revenue is discounted by the time elapsed from the commencement of the attack to the broadcast of the attack, reflecting that the attacker is not able to spend the block rewards from chain $\mathcal{A}$ until it is broadcast. Costs are discounted from the time they are incurred. The discounted profit from secretly mining one block after the time at which the canonical boundary is crossed (equivalently, past $\alpha+\lambda$ blocks on chain $\mathcal{A}$)is

$$M(B) = \underbrace{(\delta^{(\alpha+\sigma)} - \delta^{(\alpha+\sigma-1)})\{v + p_B(\lceil N(1+\xi)\rceil + 1)\}}_{\text{discount of mining rewards on ancestor blocks}} + \underbrace{\delta^{(\alpha+\sigma)}p_B}_{\text{disc rev from added block}} - \underbrace{\delta^{(\alpha+\sigma)}c}_{\text{disc cost of added block}} \quad (9)$$

Rearranging the terms of Eq. 9 into Eq. 10 shows that the marginal profit from secretly mining past the canonical boundary is negative. It follows that the attacker will broadcast chain $\mathcal{A}$ at $\alpha + \sigma$ post-fork blocks on chain $\mathcal{IC}$.

[24] In an abuse of notation, λ in the attacker's ex-ante decision problem denotes the mean of the distribution of the random variable σ.

$$M(B) = \underbrace{(\delta^{(\alpha+\sigma)} - \delta^{(\alpha+\sigma-1)})}_{<0}\underbrace{\{v + p_B(\lceil N(1+\xi)\rceil + 1)\}}_{>0} + \underbrace{\delta^{(\alpha+\sigma)}(p_B - c)}_{=0} \leq 0 \tag{10}$$

We conclude that the attacker's optimal strategic plan is to fork a head-block and grow chain $\mathcal{A}$ at the rate of $(1+\xi)$ (the first decision); to reach the canonical boundary when chain $\mathcal{IC}$ has $\alpha + \sigma$ post-fork blocks (the second decision) and thereupon to broadcast chain $\mathcal{A}$ (the third decision). If the expected profit from this plan is positive, the attack will be launched.[25]

5 Key Properties of the *ADESS* Protocol

In this section we state the main result of our model, which is that the penalty parameter ξ can be set to render unprofitable a double-spend attack on a transaction of any size. Equation 8 is the attacker's ex-ante expected profit. Noting that N, p_B and c are fixed and $B = 0$, profitability is determined by the interaction of the value of the transaction v and the penalty parameter ξ. For a given v, an increase in ξ raises discounted revenue and cost. Equation 11 is the marginal effect of an increase in ξ, by $d\xi$, on the attacker's profit, where N is the interval of blocks on chain $\mathcal{IC}$ between the fork block and the broadcast of chain $\mathcal{A}$ and the marginal increase in ξ causes the number blocks on chain $\mathcal{A}$ required to reach the canonical boundary to increase by one block (i.e. $\lceil N(1+(\xi+d\xi))\rceil = \lceil N(1+\xi)\rceil + 1$).

$$M(\xi) = \underbrace{\delta^N p_B}_{\text{additional block reward}} - c\underbrace{\sum_{n=0}^{\lceil N(1+\xi)-1\rceil} d/d\xi(\delta^{n/(1+\xi)}(1+\xi)^n)}_{\text{cost increase on current blocks}} - \underbrace{\delta^N c}_{\text{cost of additional block}} \tag{11}$$

The derivative for the discounted cost of an attack at nth block on chain $\mathcal{A}$ is

$$d/d\xi(\delta^{n/(1+\xi)}(1+\xi)^n) = n(\xi+1)^{n-2}\delta^{n/(\xi+1)}[(\xi+1) - log(\delta)]$$

This expression is positive since the time discount rate $\delta \in (0,1)$, which implies that $log(\delta) < 0$. Under the assumption that $p_B = c$, it is the case that $M(\xi) < 0$ for any $\xi > 0$, i.e. that the ex-ante expected profit from an attack is a declining

[25] This is not a complete characterization of the attacker's decision problem since it does not pin down the block that the attacker will fork. The conclusions are conditional on λ. However, this imprecision turns out not to matter for the results we are interested in.

function of the penalty parameter ξ.[26] The key property of *ADESS* is that, for every transaction value v there is a penalty parameter ξ above which attack is unprofitable. We state this property in the following theorem.

Theorem 1 (*ADESS* Bound on Double-Spend Profitability). *For any transaction value v, there is a value of the penalty parameter $\underline{\xi}$ above which a double-spend attack is unprofitable.*

Proof. Equation 11 shows that ex-ante expected profit from a double-spend attack is a decreasing function of the penalty parameter ξ. The derivative of $M(\xi)$ with respect to ξ is

$$\frac{d}{d\xi}M(\xi) \quad = \quad -c \sum_{n=0}^{\lceil N(1+\xi)-1 \rceil} d^2/d\xi^2(\delta^{n/(1+\xi)}(1 + \xi)^n) \quad < \quad 0 \quad (12)$$

The second derivative for the discounted cost of an attack at nth block on chain $\mathcal{A}$ is

$$d^2/d\xi^2(\delta^{n/(1+\xi)}(1+\xi)^n) = \\ n(\xi+1)^{n-4}\delta^{n/(\xi+1)}[n-(n-1)(\xi+1)]2log(\delta) + (n-1)(\xi+1)^2$$

This expression is positive when the term $n-(n-1)(\xi+1) < 0$, which holds for $\xi > 1/(n-1)$. Equation 12 applies whether or not $d\xi$ induces an additional block to be added to chain $\mathcal{A}$ before reaching the canonical boundary. $M(\xi) > 0$ and $\frac{d}{d\xi}M(\xi) > 0$, imply that, for any transaction value v, there is a value of ξ above which the expected profit from a double-spend attack is a strictly concave decreasing function of the penalty parameter, with a discontinuous drop in profit where an additional block is added. It follows that there is a value of ξ for which $\pi(\xi) < 0$.[27] This proves the Theorem. Figure 4 displays the relationship between ξ and the attacker's profit.

□

[26] If $d\xi$ does not cause the number blocks on chain $\mathcal{A}$ required to reach the canonical boundary to increase, the leftmost and rightmost terms of Eq. 11 drop out and $M(\xi)$ remains negative.

[27] The proof is as follows. By strict concavity, for $\xi' > \xi$" $\frac{\pi(\xi')-\pi(0)}{\xi'} \leq \frac{\pi(\xi")-\pi(o)}{\xi"} \implies \pi(\xi') \leq (\xi'(\pi(\xi") - \pi(0)) + \pi(0)$. Since $\pi(\cdot)$ is decreasing in ξ, profit is negative for any $\xi > \xi'$.

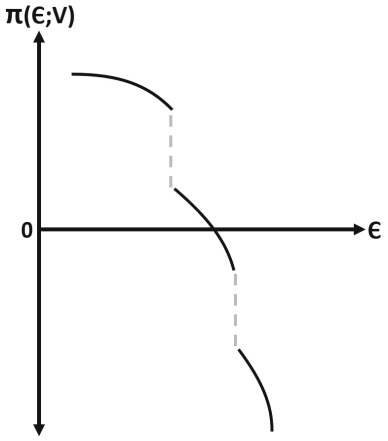

Fig. 4. The attacker's profit function

ADESS has the analogous property that every penalty setting renders unprofitable some interval of transaction values.

Corollary 1. *For any penalty parameter $\xi > 0$ there corresponds an interval of transaction values $[0, v)$ for which a double-spend attack is unprofitable.*

Proof. Set $v = 0$ in Eq. 8. Under our assumption that $p_B \leq c$, it is immediate that $\pi(\xi, v = 0) < 0$ for any $\xi > 0$. It follows that $\pi(\xi, v) < 0$ for any $v \in [0, -\pi(\xi, v = 0)$. This proves the Corollary. □

6 Relaxing the Baseline Model Constraints

In this section we relax two of the restrictive assumptions in the baseline model of Sect. 4.3 and evaluate how the key properties of the *ADESS* protocol are affected in each instance.

6.1 Incomplete or Infrequent Adjustment of Mining Puzzle Difficulty

There are two cases to consider here. The first case occurs when the puzzle difficulty does not fully adjust after each block (but it does adjust symmetrically). Consider the case where the hashrate applied to the first block is $(1 + \gamma)$. Under Assumption 2 the puzzle is solved in $1/(1 + \gamma)$ units of time. Full adjustment implies the puzzle difficulty is increased to require a hashrate of $(1 + \gamma)$ to solve the puzzle for block #2. With a partial adjustment factor $\beta \in (0, 1]$, the required hashrate is $(1 + \beta\gamma)$, which a lesser increase than full adjustment.[28] The hashrate growth of a sequence of blocks is the sum

[28] In the context of footnote 15, partial adjustment implies that an increase of hashrate to $1 + \gamma$ causes difficulty to increase to $D = 1 + \beta\gamma$, and so forth.

$(1+\gamma)+(1+\beta\gamma)(1+\beta\gamma)+...+(1+\gamma)(1+\beta\gamma)^{n-1}+....$ β slows the exponential growth of the attacker's cost. The only effect this partial adjustment has on the $ADESS$ protocol is to reduce the effective penalty by the mapping $\xi \implies \beta\xi$. The effective penalty can still be set at any value.

The second case occurs when the mining puzzle adjustment is made periodically, after an interval of blocks has been appended to a chain (called an "epoch"). For example, in Bitcoin the puzzle difficulty is adjusted every 2016 blocks. Inside of an epoch, the cost of achieving the penalty growth rate grows linearly rather than exponentially. The increased cost of growing chain $\mathcal{A}$ at the rate of $(1+\xi)$ is an additional $c\xi$ per block on chain $\mathcal{IC}$. If the block reward equals cost, this cost is recoverable from block rewards. The conclusion is that, for a given penalty parameter ξ, the effect of the penalty on attack cost increases with the frequency of mining puzzle adjustments.

6.2 Network Latency

Consider a synchronous network with a block propagation delay upper bound of $\Delta > 0$ units of time and no other channel of communication between nodes. In this setting it is possible that some nodes receive the broadcast of post-fork block $\alpha+\sigma$ on chain $\mathcal{IC}$ before it receives the broadcast of post-fork block $\lceil N(1+\xi)\rceil$ on chain $\mathcal{A}$, even when $\mathcal{A}$ was the first to initiate a broadcast. These nodes will not recognize chain $\mathcal{A}$ as canonical. Under $ADESS$, a split in opinion at the canonical boundary will persist if the following two conditions hold: (i) a portion of honest miners do not observe that chain $\mathcal{A}$ has reached the canonical boundary and continue to mine chain $\mathcal{IC}$ and the other portion mine chain $\mathcal{A}$ and (ii) the hashrate applied to mining chain $\mathcal{IC}$ is weakly below the hashrate applied to chain $\mathcal{A}$. In this case, there is more hashrate applied to chain $\mathcal{A}$ than chain $\mathcal{IC}$, but it is a constant amount. As a result, miners of chain $\mathcal{A}$ will observe that their chain remains canonical since it has more post-crossing cumulative mining puzzle difficulty than chain $\mathcal{IC}$, and miners of chain $\mathcal{IC}$ will observe their chain remains canonical since cumulative mining puzzle difficulty on chain $\mathcal{A}$ is not growing exponentially while it remains penalized.

The attacker can prevent a breakdown of consensus at the canonical boundary by broadcasting $(N+\Delta)(1+\xi)$ blocks on chain $\mathcal{A}$ at the time N post-fork blocks on chain $\mathcal{IC}$ are broadcast. This requires chain $\mathcal{A}$ to grow at the accelerated rate of $1+\{\xi+\frac{\Delta}{N}(1+\xi)\}$. A node that receives chain $\mathcal{A}$'s broadcast Δ units of time in the future while receiving the broadcast of chain $\mathcal{IC}$ without delay will observe that chain $\mathcal{A}$ reached the canonical boundary at the time chain $\mathcal{IC}$ has $N+\Delta$ post-fork blocks. Theorem 1 and Corollary 1 continue to hold when the growth rate ξ replaced by the accelerated growth rate.[29]

[29] Note that this result holds when latency slows the growth rate of chain $\mathcal{IC}$ due to conflicts arising from uncle chains.

7 Blockchain Security of *ADESS* vs. Nakamoto

In this section we compare $ADESS$ to Nakamoto with respect to two matters not previously covered; a malicious attack and an uninformed node. Our conclusion is that, in these two matters, $ADESS$ does not introduce any serious security vulnerabilities that are not present in Nakamoto.

7.1 Malicious Attack

We define a malicious attack as an attempt to break the consensus around one canonical chain. In Sect. 6.2 we showed that a permanent split in the consensus opinion under $ADESS$ could arise when chain $\mathcal{A}$ reaches the canonical boundary if the crossing is not seen by some honest miners. In the event of a split where $\leq 50\%$ of honest hashrate was applied to chain $\mathcal{IC}$, a malicious attacker could withdraw its hashrate and the split would be permanent. To achieve this result the attacker must incur (a) the exponentially increasing cost of reaching the canonical boundary and (b) the cost of eclipsing some miner nodes around the canonical boundary to ensure that the requisite number do not see chain $\mathcal{A}$ reach the canonical boundary. Once that has been achieved, honest miners will work on both chains under the assumptions of our model. More generally, consensus cannot be re-established under the $ADESS$ protocol.

A permanent split in the consensus opinion under Nakamoto can be achieved if the attacker either (i) indefinitely applies hashrate to ensure that chain $\mathcal{IC}$ and the attacker's chain $\mathcal{A}$ grow at the same rate or (ii) the attacker permanently eclipses miners representing $< 50\%$ of hashrate so as to prevent them from seeing chain $\mathcal{IC}$ while broadcasting chain $\mathcal{A}$ to them. In that case, the eclipsed miners will mine chain $\mathcal{A}$ and the un-eclipsed miners will apply their hashrate to chain $\mathcal{IC}$, which they observe to have the most cumulative mining puzzle difficulty.

This suggests a tradeoff between $ADESS$ and Nakamoto. Under $ADESS$ a malicious attacker must overcome the penalty to carry out the attack, which implies a larger initial investment to launch the attack compared to Nakamoto, but no long-term expenditures. On the other hand, Nakamoto requires a perpetual application of computing power to maintain the consensus split. Figure 5 is a graphical display of the relative costs of a malicious attack under each protocol. If costs are discounted at the rate of $\delta \in (0, 1)$ per unit of time, the early costs incurred under $ADESS$ will be weighted more heavily than the further out costs incurred under Nakamoto. In that case, it is possible that the present value of the $ADESS$ attack cost could be higher than the present value of the Nakamoto attack cost, even when the actual $ADESS$ cost is below the Nakamoto cost. There is no apriori way to rank the difference in the present value cost of attack between the two protocols.

7.2 A Node Is Not Connected When a Fork Occurs

A node that is not connected to the network when the fork occurs does not observe the temporal order of the broadcast of the initial blocks of the post-fork

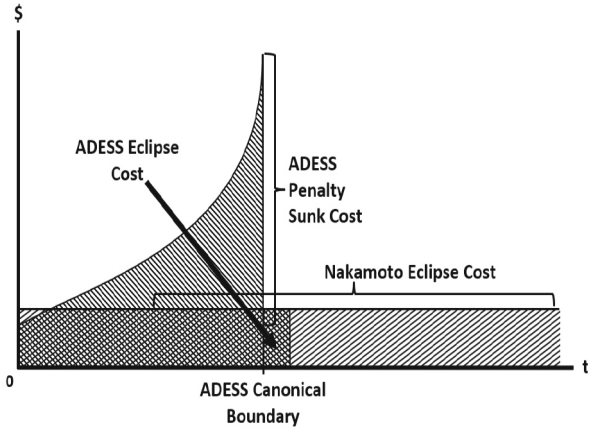

Fig. 5. *ADESS* v Nakamoto - cost of malicious attack

chain segments. Under *ADESS*, such a node cannot make a determination of whether, or to which chain, a penalty should be assessed and therefore cannot determine which chain is canonical. By contrast, Nakamoto does not require a node to observe the temporal sequence of new blocks. A node that newly enters the network can determine the canonical chain by comparing cumulative mining puzzle difficulty. This is a weakness in the security of *ADESS* compared to Nakamoto. Nevertheless, it is unclear what practical effect this weakness represents. *ADESS* provides an incentive for miners and frequent transactors, such as exchanges, to maintain several nodes - which are inexpensive - in order to ensure it is always connected to the network. Less frequent participants in the network can infer the canonical chain by observing which chain is being actively mined.

8 Conclusion

We reviewed the literature on the vulnerability of PoW blockchains to double-spend attacks that is relevant to our analysis. That literature shows that such attacks can be profitable at any transaction size and that the possibility of retaliation by the victim is not necessarily an effective deterrent. We then proposed a modification to the standard PoW protocol, called *ADESS*, that increases ex-ante expected cost of carrying out a double-spend attack by assigning a penalty to the scoring of the attacker's fork chain. There are two parts to *ADESS*. The first part is a criteria for identifying the attacker's chain and assigning the penalty to it. We argued that identification is made possible by the fact that a rational counterparty will convey its item of value only after the block to which the transfer transaction is appended and several descendant confirmation blocks have been added to the canonical chain. A rational attacker will not broadcast its chain until after it has received the item of value from its counterparty,

which requires it to wait until after the confirmation blocks have appeared before broadcasting its chain. The resultant delay in broadcasting the attacker's chain suggests the criteria of assigning the penalty to the chain that does not broadcast its blocks until after the other chain has added several post-fork blocks.

The second part of *ADESS* is the penalty function. Once the penalty has been assigned, the criteria for comparing chains shifts from cumulative mining puzzle difficulty to the number of blocks. The penalty discounts the value of block on the penalized chain. The penalized chain must therefore grow at a faster rate to overcome the penalty. The interaction of faster growth with mining puzzle difficulty adjustments that require ever higher hahsrate to maintain an elevated blockchain growth rate, result in a exponential increase in hashrate - and cost - for the attacker's chain to overcome the penalty. We showed that the expected cost of carrying out a double-spend attack under *ADESS* is weakly higher than Nakamoto.

We constructed a model which enabled us to prove that, for a transaction of any size, there is an *ADESS* penalty that renders a double-spend attack unprofitable. We then demonstrated that the main results continue to hold in the presence of incomplete adjustment of mining puzzle difficulty and network latency. Finally, we argued that the requirement that nodes observe the temporal sequence of blocks in *ADESS* does not practically create a serious impediment to the participation of nodes in the network and that there is no apriori way to rank the vulnerability to a malicious attack between *ADESS* and Nakamoto.

Appendix

Generalizing *ADESS* to Multiple Chains and Multiple Forks

ADESS can be extended to a general blockchain network in which there are multiple forks, each with two or more descendant chains where penalty assignments are made at each fork. To accommodate this, we generalize the *ADESS* protocol so that a penalized chain cannot become canonical until it has overcome all penalties assigned to it. Proposition 1 and its Corollary prove that there is always exactly one canonical chain.

Tree Graph Representation of a Blockchain Network. A blockchain network can be represented as a directed tree graph. A chain is a unique directed path running from a fork-block to a chain head.[30] At any time t there are N fork-blocks in the blockchain network, each one denoted $f_n,\ n \in \{1, ..., n, ..., N\}$ and M heads, each one denoted $B_m,\ m \in \{1, ..., m, ..., M\}$. We also denote B_m as the chain connecting a fork-block to a chain head. Figure 6 displays the blockchain network as a directed tree graph at t periods after the Genesis block was broadcast.

[30] Unique paths, running from the root node to end nodes, are a feature of directed tree graphs.

Penalty Nomenclature. We simplify the presentation by assuming that an attacker forks the parent of the block to which the transaction is appended, so that $\lambda = 0$.[31] The baseline chain for fork-block f_n is the first chain to broadcast α post-fork blocks, which is denoted $\alpha : f_n$. For example in Fig. 6, $\alpha : f_2$ indicates that chain B_4 is the baseline chain relative to f_2, since it was the first chain to broadcast a chain segment with α blocks, starting at f_2. A more complicated example involves chain B_1. $\alpha : f_1$ indicates that B_1 was the first chain to broadcast a chain segment with α blocks, starting at f_1 and $\alpha : f_3$ indicates that chain B_2 was the first chain to broadcast a chain segment with α blocks, starting at f_3. In this case B_1 is the baseline chain for fork f_1 and is not the baseline chain for fork f_3. At each fork f_n a penalty may (or may not) be assigned to one or more chains B_m. When a assignment is made, the penalized chain is compared to the baseline chain $B_{m'}$.[32]

The penalty is "active" at fork-block f_n so long as the penalized chain has not overcome the penalty. The tuple $< B_m : \widehat{f_n} : B_{m'} : t >$ denotes that B_m is actively penalized relative to $B_{m'}$ at f_n at time t. The penalty is "inactive" after it has been overcome by the penalized chain. An inactive penalty is denoted $< B_m : f_n : B_{m'}, t >$. In the latter case, B_m has become the baseline chain relative to fork f_n. In general, $\widehat{f_n}$ indicates a currently active penalty and f_n indicates a past penalty that has been overcome. The set of active and inactive penalties assigned to B_m at time t is denoted by a list such as $\{< B_m : \widehat{f_n} : B_{m'} : t >, < B_m : f_{n+i} : B_{m'+j} : t >, ...\}$.

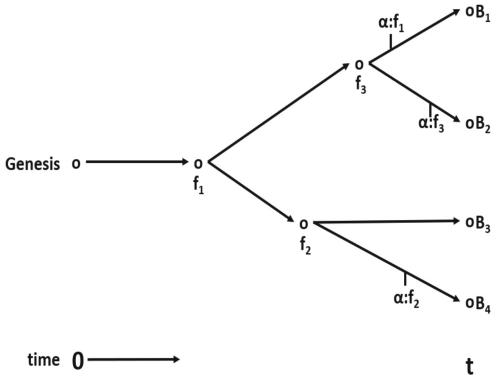

Fig. 6. A blockchain network

[31] From Sect. 4.3, λ is the number of blocks in chain IC (in the generalized context the baseline chain $B_{m'}$) between the fork-block and the block onto which the transaction is appended. $\lambda > 0$ at fork n would be denoted $\alpha + \lambda(f_n) : f_n$, which would not affect Proposition 1.

[32] A baseline chain at one fork-block can be a penalized chain at another fork-block.

Generalized Penalty Assignment Rule and Penalty Function. We restate the penalty assignment rule and penalty function for the general case.

Generalized Penalty Assignment Rule

(i) Apply the *ADESS* Penalty Assignment Rule to each fork-block f_n (i.e. a chain that is not first to broadcast α post-fork blocks is assigned a penalty relative to the chain that is first to broadcast α post-fork blocks) with the exception that
(ii) If the chain that is first to broadcast α blocks after a fork-block f_n is subject to an active penalty at the time of the broadcast, then no penalty assignment is made at fork-block f_n. □

Generalized Penalty Function

(i) Apply the *ADESS* Penalty Function to each penalized chain at fork-block f_n.
(ii) At the time a chain B_m has overcome every penalty assigned to it, it has no active penalties and the protocol for B_m reverts to the Nakamoto criteria of comparing cumulative mining puzzle difficulty with other chains that do not face active penalties. When it has overcome its last penalty, the score of B_m is re-set to equal the cumulative mining puzzle difficulty of the baseline chain for the last penalty, plus a small additional amount ϵ. For example, if $< B_m : f_n : B_{m'} : t >$ is the last penalty to become inactive, B_m is assigned an adjusted cumulative mining puzzle difficulty = ($B_{m'}$ cumulative puzzle difficulty at t) + ϵ. If B_m has more than one active penalties and all are overcome at the same time, the baseline chain with the highest score is used for the re-set. □

The Canonical Chain. At any time the set of chains can be partitioned into two groups. One group are chains that have been assigned at least one penalty that has not been overcome. These chains are not eligible to be canonical. Among chains in the other group, those ranked by (possibly adjusted) cumulative mining puzzle difficulty, one will be canonical - provided there is at least one chain in this group. Proposition 2 establishes that there is at least one chain that is eligible to be canonical.

Proposition 2. *Under the generalized ADESS Protocol, there is at least one chain to which no penalty has ever been applied at any time.*

Proof. We prove the proposition by construction. Start at the Genesis block and proceed to the first fork-block f_1. There will be at least one post-fork chain that is not penalized. Choose one of the non-penalized chains and proceed to the next fork-block. There will be at least one post-fork chain that is not penalized. Choose one of the non-penalized chains and proceed to the next fork-block, and so forth until a head B_m is reached. The chain B_m is not penalized at any fork. □

The example in the proof is displayed in Fig. 6 when comparing chains B_1 and B_2. If B_1 is the un-penalized chain at f_1, then either B_1 or B_2 must be un-penalized.

Corollary 2. *There is exactly one canonical chain under generalized ADESS.*

Proof. Proposition 2 states that there is at least one chain to which no penalty has ever been applied at any time. Such a chain is eligible to be canonical. Suppose there is more than one chain without active penalties at a point in time. These chains are compared on the basis of cumulative mining puzzle difficulty. Under Nakamoto the chain with the most work is canonical. □

Finally, *ADESS* applies to a circumstance where there is only one fork-block with actively mined descendant chains and none of those chains have forks. *ADESS* is the application of generalized *ADESS* in the case where there is one fork-block, two fork chains and neither chain has an active penalty from a prior fork. In that case, chain $\mathcal{A}$ is penalized relative to chain $\mathcal{IC}$.

Relaxing the Restriction on Growth Rate of Chain $\mathcal{A}$

The model limits the attacker to choosing a constant growth rate γ for chain $\mathcal{A}$. We now relax that restriction and allow the attacker to choose the growth rate of each block n on chain $\mathcal{A}$ as the function $\gamma(n, \xi)$. Equation 11 becomes

$$M(\xi) = \underbrace{p_B}_{\text{additional block reward}} - \underbrace{c \sum_{n=0}^{\lceil N(1+\xi)-1 \rceil} d/d\xi(\delta^{n/(1+\xi)}(1+\gamma(\xi,n))^n)}_{\text{cost increase on current blocks}} - \underbrace{c}_{\text{cost of additional block}} \quad (10')$$

We do not evaluate all possible functional forms of $\gamma(n, \xi)$. We show that Theorem 1 and Corollary 1 continue to hold if the growth rate is a affine function of ξ. Let $\gamma(n, \xi) = \rho + \xi f(n)$ for some scalar $\rho > 0$ and function $f(n) > 0$. The derivative for the discounted cost of an attack at nth block on chain $\mathcal{A}$ is

$$\begin{aligned} d/d\xi(\delta^{n/(1+\gamma(\xi,n))}(1+\gamma(\xi,n))^n) = \\ nf(n)(\xi f(n)+\rho)^{n-1}\delta^{n/(\xi f(n)+\rho+1)} \\ - nf(n)log(\delta)[(\xi f(n)+\rho)^n+1]\delta^{n/(\xi f(n)+\rho+1)}[\xi f(n)+\rho+1]^{-2} \end{aligned}$$

The expression is positive, noting that $log(\delta) < 0$, since $\delta \in (0, 1)$. Therefore Equation 10' is negative. Noting that n, δ and ξ are strictly positive, the expression is bounded away from zero, which implies that there is no upper bound to the expression. It follows that there is a value $\underline{\xi}$ for which $\pi(\underline{\xi}) < 0$, which proves Theorem 1 and Corollary 1.

References

1. Budish, E.: The Economic Limits of Bitcoin and the Blockchain. National Bureau of Economic Research Working Paper 24717, June 2018 (2018). https://www.nber.org/papers/w24717
2. Eyal, I., Sirer, E.G.: Majority is not enough: bitcoin mining is vulnerable. In: Christin, N., Safavi-Naini, R. (eds.) FC 2014. LNCS, vol. 8437, pp. 436–454. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-45472-5_28
3. Gervais, A., Glykantzis, V., Karame, G.O., Ritzdorf, H., Wurst, K., Capkun, S.: On the security and performance of proof of work blockchains. In: CCS 2016: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, New York, NY, United States, pp. 3–16. Association for Computing Machinery (2016)
4. Guo, D., Ren, L.: Bitcoin's Latency - Security Analysis Made Simple arxiv2203.06357v3 (2022). https://arxiv.org/abs/2203.06357
5. Leshno, J., Strack, P.: Bitcoin: an axiomatic approach and an impossibility theorem. Am. Econ. Rev. Insights **2**(3), 269–286 (2020). American Economic Association
6. Lovejoy, J.: Reorgs on Bitcoin Gold: Counterattacks in the wild (2020). Medium https://medium.com/mit-media-lab-digital-currency-initiative/reorgs-on-bitcoin-gold-counterattacks-in-the-wild-da7e2b797c21
7. MIT Media Lab Digital Currency Initiative reorg tracker (2020). 51% attacks - reorg tracker. https://dci.mit.edu/51-attacks
8. Moroz, D., Aronoff, D., Narula, N., Parkes, D.: Double Spend Counterattacks (2020). http://arxiv.org/abs/2002.10736[cs.CR] https://doi.org/10.48550/arXiv.2002.10736
9. Nakamoto, S.: Bitcoin; A Peer - to - Peer Electronic Cash System (2008). https://bitcoin.org/bitcoin.pdf.Original Bitcoin code https://satoshi.nakamotoinstitute.org/code/ (2008) Original Bitcoin code https://satoshi.nakamotoinstitute.org/code/
10. Schelling, T.C.: The Strategy of Conflict. Harvard University Press, Cambridge (1960)
11. Singer, A.: Fight fire with fire: MIT scholar suggests ETC counters 51% attacks. Cointelegraph Set. 15 (2020). https://cointelegraph.com/news/fight-fire-with-fire-mit-scholar-suggests-etc-counters-51-attacks
12. Wood, G.: Ethereum: A secure Decentralised Generalised Transaction Ledger - Berlin Version d77a387, 26 April 2022 (2021). https://ethereum.github.io/yellowpaper/paper.pdf

Author Queries

Chapter 11

Query Refs.	Details Required	Author's response
AQ1	This is to inform you that corresponding author has been identified as per the information available in the Copyright form.	
AQ2	As per Springer style, both city and country names must be present in the affiliations. Accordingly, we have inserted the city and country names "New York, USA" in the affiliation 2. Please check and confirm if the inserted city and country names are correct. If not, please provide us with the correct city and country names.	