Plata: A Peer-to-Peer Electronic Cash System Run by Phones

Pablo

November 2026

Abstract. Bitcoin showed that electronic cash can move from one party to another without going through a financial institution. But the machines that secure it have drifted out of reach of most of the people who might use it, and its signatures will not survive a large quantum computer. We propose a network whose validators are ordinary phones, and whose every signature is post-quantum from the first block. Phones serve whenever their owners allow, on Wi-Fi or mobile data. Each epoch, a small committee is drawn by secret tickets that only hashing protects. A block of payments is final as soon as two-thirds of the committee have signed it twice, as in Algorand, so there are no forks to wait out; while nobody pays, the network slows to a block every two minutes. The right to validate is not bought with machines or coins: a seat is given by three existing seats vouching for a person, its weight is earned slowly by service, and it is capped by the trust that reaches it from the rest of the network. No phone keeps the history; each keeps a recent window and a snapshot that the network itself certifies. The network is robust in its simplicity. Coins can be swapped for dollars on another chain without any bridge or custodian, by locks that only a hash opens. Two phones side by side can pay each other with no network at all, from an amount set aside in advance under a guarantee the chain holds. Programs and AI assistants pay as people do, within limits their owner sets. Phones reach each other only through Tor, inside a post-quantum key exchange of their own; every phone that serves is also a meeting point for others; phones come and go as they please, and on return catch up from the latest certified snapshot.

1. Introduction

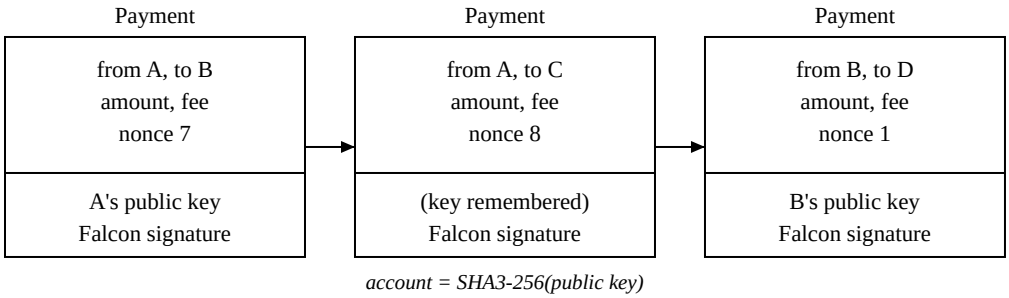
Commerce on the Internet still relies on financial institutions serving as trusted third parties to process electronic payments. Bitcoin removed that trust by replacing it with cryptographic proof and computing power [1]. It succeeded, but two of its assumptions have aged. The first is that anyone can contribute to securing the network. In practice, proof-of-work has concentrated in buildings full of special-purpose machines, and proof-of-stake, its usual replacement, gives the most power to whoever holds the most coins. The four billion people who carry a phone, and who would gain most from money that needs no bank, take no part in either.

The second assumption is that the digital signatures protecting coins will remain hard to forge. Bitcoin's elliptic curve signatures, and those of nearly every chain that followed it, can be broken by a sufficiently large quantum computer. Coins whose public keys have already been revealed could then be spent by anyone who builds one. Changing the signature scheme of a running network requires every holder to move their coins, and many never will.

What is needed is an electronic payment system in which the device most people already own is enough to take part in securing it, and in which no part of the protocol depends on a problem that quantum computers solve. In this paper, we propose such a system. It keeps what Bitcoin got right: no trusted third party, rules fixed in advance and the same for everyone, a fair launch with no coins before the first block, and a network that is secure as long as honest participants control most of it. It changes what a phone cannot do. A phone has no spare electricity to burn, cannot accept incoming connections, is asleep most of the day, and has little storage. Every design choice that follows comes from these limits.

2. Payments

We define an account as the SHA3-256 hash [7] of a Falcon-1024 public key [5]. A payment names the network, the sender, the receiver, the amount, a fee and a nonce, and is signed by the sender's key. The nonce increases by one with each payment, so a payment cannot be replayed. The signing bytes begin with a fixed label, so a payment signature can never be passed off as a vote or any other signed message. An account's first payment carries its sender's public key. From then on the network remembers the key, checked against the account it hashes to, and later payments travel without it: 1.3 kilobytes instead of 3.1, since the key is more than half of a payment.



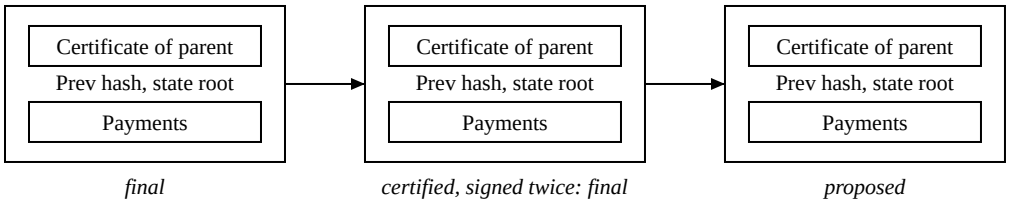
Unlike Bitcoin, we keep balances rather than unspent outputs. A phone cannot hold the set of all unspent outputs, but it can hold a proof of one balance. Balances and nonces are kept in a sparse Merkle tree [15], whose root every block commits to; a proof that an account holds a given balance, or does not exist, is under one kilobyte.

A payment can also be locked. A lock names a receiver, an amount, the SHA-256 hash of a secret and a deadline. Whoever shows the secret before the deadline pays the receiver; after the deadline, anyone may return the money to its sender. Deadlines are judged by the network's own clock, the middle of the times written in the last seven final blocks, so no single phone with a wrong clock can move them. Locks are what let Plata trade with other chains without trusting anyone (section 10).

The problem of course is that the receiver cannot verify that the sender did not spend the same balance twice, at the same nonce, in two places. As in Bitcoin, the only way to confirm the absence of such a payment is to be aware of all payments, and the network must agree on a single history of the order in which they were received. Bitcoin achieves this with proof-of-work. A phone cannot do work, so we need another way to decide who writes the history.

3. Blocks and Certificates

Time is divided into slots, and slots into epochs of 120. In each slot one member of the current committee proposes a block of payments. Each block contains the certificate of its parent: the signatures of more than two-thirds of the committee, by weight, on the parent's hash. Each certificate therefore reinforces the ones before it, as each proof-of-work does in Bitcoin, but with one difference that matters to a phone. A block becomes final in one of two ways. The first is the two-chain rule of the HotStuff family of protocols [8], as refined in Jolteon [9] and MonadBFT [10]: a block is final once it is certified and its child, in the very next slot, is certified too. The second is faster and applies to every block that carries payments. A member that holds a block's certificate signs a second time, a commit signature, and a block with commit signatures from more than two-thirds of the weight is final at once, without waiting for its child. These are Algorand's two steps of voting [2], on top of a pipelined chain. One rule makes the second signature safe: a member gives it at most once a slot, and never after giving up that slot or a later one. Without that rule our model checker finds two conflicting final blocks at once; with it, every reachable state at two slots, about 1.1 million with up to two attacker blocks, is safe, and random walks seven slots deep find nothing. Either way, once a block is final, no other block at its height can ever be final, so there is nothing to wait out and no reorganisation to fear.



Committee members vote on the order of payments, which keeps voting light, as in Monad's separation of ordering from execution [11]. The order inside a block is set by its leader's slot secret, committed long before (section 6), so the block can be applied the moment it is final. The leader knows that order, and could arrange for one payment to fail because another in the same block spent first; it could equally have left that payment out, which any leader can do, and a payment that fails is not recorded and can be sent again. With payments only and no contracts, the order of a block gives its leader nothing to gain.

A slot that gathers no certificate times out after fifteen seconds; more than two-thirds of the members' timeouts begin the next slot. A slot nobody may lead is given up after six seconds, even at the quiet pace below, and the network keeps full pace until the block before it is final. A new block is never made faster than every 2.5 seconds, a floor chosen because the data a phone downloads is proportional to the number of blocks it sees per hour. When phones reach each other through Tor, the network's own delays set the pace instead. Measured, a payment is final 1.9 seconds after it appears when four seats share one phone, and about 8 seconds across seven phones over Tor. On the running network, payments are final in about two to three seconds.

Members already hold the payments a leader proposes, since payments travel to every seat as soon as they are sent. A proposal therefore names its payments by their 32-byte fingerprints only. The

block's hash covers the same list either way, so it is the same block; a member missing a payment waits a second for it to arrive, and only then asks for the whole block. A block that names a payment already final is never fetched whole. Late votes (section 7) ride in later blocks the same way, named without their signatures, which members already hold; a member that lacks one asks the block's maker for it. Over links as slow as Tor's, these rules cut the data on each link between meeting points at full pace from about 110 to about 46 megabytes an hour, at the same pace of blocks. A block may carry up to 5,000 payments.

A network that makes blocks when nobody pays spends data and battery for nothing. The network may therefore run at a quiet pace: while no payment waits, a slot opens only after a wait the founding seats choose, thirty seconds, one minute or two minutes, and the running network uses two minutes; a payment opens the next slot at once, and so does a wallet that opens its payment screen, and full pace is kept for two minutes after it, since switching back and forth would cost more than it saves. The short wait for proposals then counts from the moment the payment arrived, not from the start of the quiet slot, so every member sees the same best block and the payment is final as quickly as at full pace. Measured on four seats over four hours with no payments, the quiet pace cut blocks from 1,400 to 132 an hour and data from 12.2 to 1.8 megabytes an hour per seat, while a payment every five minutes was still final in about two seconds; at two minutes, a seat with no payments to carry sends and receives about half a megabyte an hour. While nothing urgent waits, a seat that gives up a quiet slot repeats it ever more slowly, so a seat that sees work the others do not cannot flood the links. The pace changes nothing about validity or pay; it is recorded on the chain so that every phone keeps the same one.

4. Proof of Trust

To decide who signs the certificates without proof-of-work, we need a scarce resource that a phone has and that money cannot buy in quantity. Coins are not that resource: if voting power is proportional to coins, whoever holds the most coins decides, and the network ends where proof-of-work did. We use people instead. Using the coin requires none of what follows. Any phone can make a key, send and receive at once. Proof of Trust only decides who may hold a validator *seat* and share in its rewards.

A person receives a seat when three existing seats vouch for them. A seat invites a person with a one-time code, or the person tells the founding seats who they are in a sealed note; while the network is young, the founding seats give the three vouches. A seat may vouch at most five times in any rolling year, and only after it has served for 30 days; and one of the three vouchers of every new seat must be a founding seat or a seat at least a year old. That last rule matters: our simulations show that with a plain 90-day wait, fake seats vouching for fake seats grow until they hold almost all the weight, while with a senior voucher required, three hostile founders out of thirty are held to about a sixth of it after five years. A new seat starts at a tenth of the weight of a full seat and grows to a full seat over 90 days of service. Nobody holds more than one seat's weight, however much they own.

Vouches alone would let one bribed person vouch an unlimited cluster of fakes into the network over the years. Each epoch, therefore, every phone computes how trust flows from the founding seats

along vouches, in the manner of SybilRank [13], and a seat keeps its weight only as far as trust reaches it. Fake seats reachable only through the few real people who vouched for them receive little trust, and so little weight, however many there are. In our simulations, 500 fake seats behind one bribed seat weigh 2.7 seats in all, and 500 bribed members out of 10,000 are held to 5.1% of the weight after three years, against 30% without the cap. The computation uses integer arithmetic over finalised state only, so every phone arrives at identical weights.

These durations are measured in time, not in blocks. Each block carries its proposer's clock, and the network moves its own clock on by the median of those times, so a phone whose clock is wrong, or a minority of them, cannot shorten a wait; a block's time is held within three minutes of the network's clock, and a little more when blocks really are minutes apart, so that the clock keeps up at the quiet pace. A seat locks no deposit. A deposit that is returned the moment a seat stops protects nothing, and only holds back a newcomer's first earnings; the cost of cheating is the seat itself. A seat that votes for two different blocks in the same slot proves its own dishonesty with its two signatures, and so does a seat that shows each half of the network a different vote: every phone compares the votes carried inside each certificate with the votes it holds itself. The rules close a seat shown to have done either, and three people must then vouch for its owner again. For now, phones record such evidence and show it rather than submit it by themselves, since two honest founding seats were once closed by a report whose cause was not found; a founding seat may reopen a closed founding seat. Nobody is ever asked to judge anyone.

The trust starts with the founding seats, so their number and their independence from one another are the first defence of a young network, as the honesty of its earliest miners was for Bitcoin.

5. Network

The steps to run the network are as follows:

1. In the first half of each epoch, seats that are able to serve announce that they are available for the next two epochs, so that a phone briefly away while offers are open keeps its place.
2. In the second half, each available seat computes its committee ticket for the next epoch. The 100 best tickets per unit of weight form the committee; while 100 or fewer seats are available, all of them serve.
3. At the start of each slot, members whose slot ticket is low enough announce it; the holder of the best ticket sends its block of new payments.
4. Members accept the block only if all payments in it are valid and it extends the newest certified block they know.
5. Members express their acceptance by signing the block's hash and sending the vote to the whole committee; more than two-thirds of the weight forms a certificate.
6. Members that hold a certificate for a block of payments sign it a second time; more than two-thirds of these commit signatures make it final at once. Any other block is final once its child is certified.

Because leaders are secret, nobody knows where to send a vote, so votes go to every member, as in Algorand [2], and every member assembles the certificates itself. A member passes each new proposal on once. Members that saw only one of two different blocks from the same proposer may still certify it, but never two blocks in one slot. A proposer gains nothing by sending different blocks and loses its seat: the two blocks, each signed by it for the same slot, prove what it did, any member that sees both submits them, and the seat is closed as for a double vote. An epoch ends only at the first closing block on the finalised chain, and the next committee is computed from finalised blocks only. This is the reconfiguration rule of Diem [12]: it makes the end of an epoch a property of the chain rather than of what one phone happened to see. Our first rule was weaker, and a model checker found the flaw; the voting, commit and epoch rules are now model-checked, the two-chain rule exhaustively to three slots (46,304,267 reachable states, all safe) and the second signature as described in section 3.

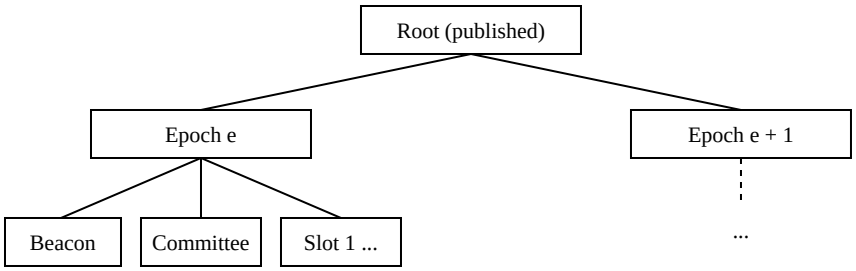
Phones connect to *meeting points*, which pass messages along. A meeting point can delay or drop a message, but never forge one, since every message is signed. Every phone that serves is itself a meeting point, reachable at an onion address it publishes on the chain, and each keeps two links: one to a meeting point of a founding seat, which can be reached at three onion addresses so that one failing cuts nobody off, and one to another phone, chosen among those that answer. A Tor circuit can die without a word, so each link carries a small sign of life every fifteen seconds, and a link silent for forty-five seconds is closed and opened again. A meeting point sends whole only the messages of the phones connected to it; for the rest it announces a short fingerprint, and another meeting point that lacks the message asks for it. Plain flooding cost five times the data. A phone that falls behind asks one seat for up to 128 blocks at a time, and a phone that has seen no final block for a while asks two committee members to bring it level. The final blocks it missed come slim: a payment whose key the network already holds travels without its signature, about a tenth of the bytes, in a batch signed by the seat that was asked. The phone neither votes nor answers wallets on such blocks until a fingerprint of the balances on the chain (section 9) covers them, and starts again from the network's state if one ever differs. If fewer than four seats are available, the chain pauses safely and resumes when they return.

A phone serves only when its owner allows it: on Wi-Fi by default, on mobile data if allowed, with its cost per hour shown first, and on battery down to 20% charge; it needs no charger. A phone that stops keeps its place on the committee until the epoch ends, but is not paid for the votes it does not give. More than two-thirds of the weight is enough to go on, so the network never waits for a phone that has left.

6. Randomness

Algorand selects its committees by verifiable random functions over elliptic curves [2], [3], which a quantum computer breaks, and no practical post-quantum equivalent exists. Falcon cannot stand in for it, because a signer could produce many valid signatures and keep the luckiest. We use commitments instead. When a seat joins, it publishes the root of a Merkle tree over secrets derived from a private key: for each epoch, one secret for the shared random number, one for its committee

ticket, and one for each slot. A tree covers 2^{17} epochs, about two and a half years, and is renewed before it runs out.



A secret is fixed long before it is used, so nobody can choose a convenient one; the only freedom left is to withhold it. A revealed secret, with its Merkle path, is 780 bytes and is checked by hashing alone. Only seats whose committee ticket falls under a published cut-off reveal it, and the committee is the 100 best tickets per unit of weight, a form of priority sampling [14] that favours a full seat over a half-earned one. The committee becomes known only a few minutes before it serves, and nobody knows who leads a slot until the leader proposes, so there is no schedule to attack.

The random number for epoch $e + 2$ is a chain of hashes: the previous number, then the committed secret of the leader of each block finalised in the first half of epoch e, in order. Our first rule took the first of those secrets alone, one contributor as in Algorand. A review found its flaw: the seat that led that one block knew the next number an epoch early, and could prepare tickets that win with it. Now nobody knows the number before honest leaders' secrets are out, half an epoch in, and a seat leads in an epoch only with tickets the chain already held when the epoch before began, so tickets made once the number is known come too late to lead with. A newcomer therefore votes and is paid for one epoch before it may lead. What is left to an attacker is the last secret of the half: to publish its block or stay silent and lose it, one choice between two numbers it cannot otherwise steer.

7. Incentive

By convention, every epoch the seats whose votes appear in the finalised blocks are paid, in proportion to how many they signed. A seat is paid for signing, not for leading or for being first. The first two-thirds of the votes make a block final, but a vote that arrives a few seconds later, as it will from a phone behind Tor, still proves that its seat holds the chain and checked the block. Such a vote may be carried by any block of the same epoch within the next ten slots and is paid in full, as Ethereum includes attestations after their slot [18]; each seat is paid at most once for each block. No block ever waits for a slow seat. A seat that is never online earns nothing. This adds an incentive for phones to support the network, and provides a way to distribute coins into circulation, since there is no central authority to issue them. There are no coins before the first block: every Plata is created by the network as it runs.

The pay of the seats is a budget that does not depend on traffic. While the network is small, each full seat adds 10,000 Plata a year to it, so the first participants are paid well but cannot end up owning most of the coins by being first. Once there are 10,000 seats, the budget is fixed at 100 million Plata

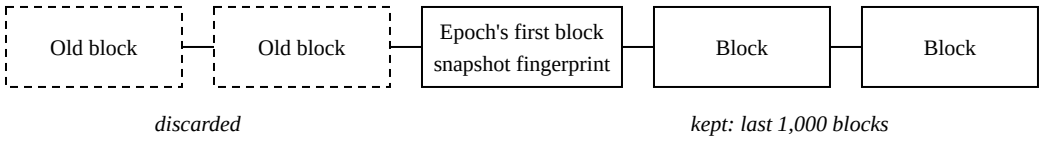
a year, forever. Each epoch pays for the time it lasted, measured from the clocks the blocks carry, so a faster or a quieter network creates the same coins in a year. Each epoch, fees fill the budget first and new coins make up the rest, so every fee paid is a coin that is not created. Fees beyond the budget are burned, except for a fifth that the seats keep. There is no maximum supply: supply grows while use is small and shrinks once fees exceed the budget, so it is pulled toward the level at which the year's fees equal the budget.

A tenth of every epoch's pot goes to a development fund, at an address fixed in the genesis block, where everyone can see it and nobody can change it. It is paid from the same pot, so users pay nothing extra for it. The founding seats hold powers that other seats do not, and we state them plainly: they may vouch without a yearly limit, so that a young network can grow; they may set the quiet pace of section 3 and how many payments a block carries, up to 5,000, which change how fast the network goes, never which payments are valid or who is paid; and a founding seat may reopen a founding seat that was closed (section 4). Software updates are signed by a separate release key. No party has any other role in the protocol: no reserved seats beyond the founding ones, no share of the coins.

The incentive may help encourage seats to stay honest. A seat that cheats loses its seat and the trust that took months to earn, and must persuade three people to vouch for it again. A group able to gather more than a third of the committee's weight would find it more profitable to keep earning by the rules than to stop a network whose coins it holds.

8. Reclaiming Disk Space

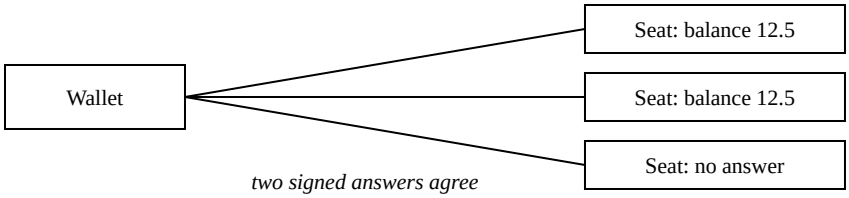
A phone has no room for the whole history, and it does not need it. Each node keeps the last 1,000 finalised blocks, its newest snapshot, and a 30-day index of blocks and payments that answers questions from wallets. At the end of each epoch every seat builds the same snapshot: balances, seats, the committees around the boundary, and the epoch's last block. The fingerprint of the snapshot must be written into the first block of the next epoch, which seats refuse otherwise, so the committee certifies it. Older blocks can then be discarded.



A phone that returns after days away, or a new one, asks every seat for the newest checkpoint and takes the one that two founding seats, or more than two-thirds of all seats, sign alike: the first answers to arrive do not decide. It then downloads the snapshot in chunks of 256 KB from any seat, each chunk checked against the list of hashes it carries and the whole checked against the certified fingerprint, and carries on from there. Catching up takes the size of the present, not the length of the past. The starting point can also come from a person: a phone that earns shows its checkpoint as a code, and a new phone scans it.

9. Wallets Without the Chain

It is possible to pay and be paid without running a seat. A wallet does not follow the chain. It asks for its balance with a short question, which every seat answers; each answer is signed by its seat, and the wallet accepts what two of them sign alike. While a wallet is open, seats also tell it at once when its balance changes, so it need not keep asking. It keeps a list of the seats it trusts, refreshed in the background from a checkpoint that two founding seats, or more than two-thirds of all seats, sign alike.

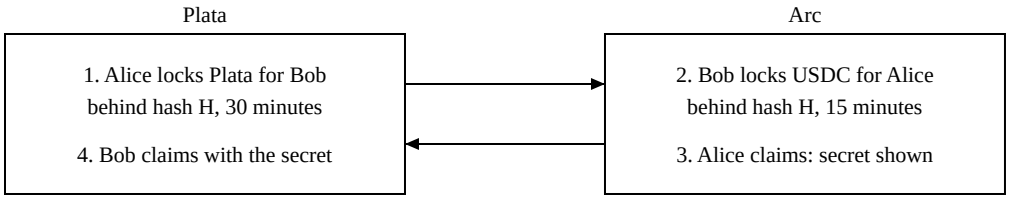


As such, the wallet is reliable as long as the seats that answer it are mostly honest, but more vulnerable if a group controls the seats it reaches. Every block commits to a root over all balances, and a proof of one balance against it is under a kilobyte. Two steps to use it are built. A seat signs the root once a block and answers with the path from one balance to it, which the wallet checks itself. And every block names the root of the balances two final blocks below it, which seats check before they vote, so the committee's certificate covers it: one answer from any seat, carrying that block and its certificate, is then enough, and the wallet needs only the committee's keys. Such evidence exists while the balances have not changed in the last two final blocks; right after a payment a wallet falls back to seats that agree. Wallets for programs (section 12) ask this way today. Compact certificates of the network's history in the manner of Algorand's state proofs [4] come after, and the same certificates are what a bridge to another chain would need.

Each seat signs at most twenty answers a second, eight of them for any one wallet, so questions cannot drain its battery; an answer by proof costs no new signature at all. Every answer goes only to the wallet that asked, so no wallet learns what others ask about.

10. Trading With Other Chains

A coin is more useful if it can be exchanged, but a bridge, a contract that holds coins on one chain and issues copies on another, is only as safe as its keepers, and bridges have been the largest thefts in the history of these systems. We use instead the atomic swap [20], which needs no keeper. Alice sells Plata for dollars that Bob holds on Arc, a chain where USDC is native. Alice chooses a secret and locks her Plata for Bob behind its hash for thirty minutes. Bob's phone checks the lock is final and exact, and locks his dollars on Arc for Alice behind the same hash, for fifteen minutes, ending well before hers. Alice claims the dollars, which shows the secret on Arc; Bob's phone reads it there and claims the Plata. If anyone stops, both locks expire and each side takes its money back. The one who knows the secret locks first and with the longer deadline, so the other always has time to use it.



both happen, or both are refunded

The contract on Arc is about a hundred and forty lines, with no owner, no pause and nothing to upgrade. Whoever submits a claim on Arc may keep a small tip from the locked dollars, so a seller with no dollars for Arc's fees is still paid. Offers to buy or sell are signed messages that meeting points pass along like any other, never written on the chain: each phone sees the same offers, takes the best with one tap, and its own phone runs every step, also with the screen off, and may cancel an offer at any time. An offer can also be kept open: the phone renews it each day and answers whoever takes it, in the background, until it is cancelled or sold out. A seller is paid at a new address each time, so two sales cannot be linked. Plata charges nothing for a swap beyond its usual fee; Arc's fees, paid in dollars, came to about a cent in our first swaps, which took about a minute. The same twenty-four words that hold a wallet's Plata also give it a dollar address on Arc, from which the owner can receive, swap, or send the dollars anywhere on Arc. The contract runs on Arc's main network, its code identical byte for byte to the one first tested on Arc's test network. A swap has no upper limit; each part of one is at least a dollar.

The other chain is not post-quantum. Arc's signatures are classical, its validators are approved, and the issuer of USDC can freeze an address; the amounts and the shared hash are public on both chains. The swap protects each side from the other, not from the other chain.

11. Paying Without the Internet

A payment needs the network, and networks fail: an outage, a place no signal reaches. Two phones side by side can still settle, if the payer prepared. While online, a wallet sets an amount aside on the chain: that amount to spend offline, and the same again locked as a guarantee. Offline, a payment is a signed slice of the amount set aside, from one mark to the next, so two payments of the same money overlap and anyone holding both can show it. The payer hands the payment over with a record of what was set aside, signed by two founding seats and at most 23 days old, and the receiver's phone checks all of it with no network: the signatures, the record, and that the slice lies beyond every slice it has been shown before.

The phones find each other by Bluetooth, with no pairing. The receiver shows a short code, the payer picks it among the phones nearby, and what they exchange is sealed under an ML-KEM-768 key made for that one payment. A payment with its record is about 8 kilobytes. Back online, either phone delivers it to the chain and the receiver is credited. What is set aside is released only 30 days after its owner stops the arrangement, and a record is good for 23, so a receiver always has at least seven days to deliver.

What stops a payer from handing the same slice to two people who never meet? Offline, nothing can; the chain can only make it cost. The first delivery of a slice is paid. A second that overlaps it, or

one beyond the amount, proves the cheat with the payer's own signatures: the whole guarantee is forfeited to the network's payouts, nothing set aside comes back, and the receiver is told plainly that the payment was not paid. We state the limit as plainly. A payer who cheats one person loses as much as was taken, but the person is not repaid, and a payer who cheats many at once gains more than the guarantee. Offline payments are for people who know each other, in amounts they could afford to lose; the app keeps what is set aside within a quarter of a wallet's balance, with as much locked beside it. The first payment between two phones with no network was made on 12 November 2026.

12. Programs and AI Assistants

A payment system that people use through an app can be used by software under the same rules; nothing in the protocol knows the difference. We provide a wallet for programs, with the app's own payment engine inside and the private connection built in, in three forms: a command, a library for Python and TypeScript, and a server that speaks the Model Context Protocol [21], through which an AI assistant uses tools. A program must never pay twice because it crashed or lost its connection. A payment is therefore signed and written to disk before it is sent, and only ever sent again exactly as it was: the same payment at the same place in its account's order, which the network accepts once. It is called final only when the account's history shows that very payment.

An assistant that can pay must not be the one that decides how much. Its owner starts the wallet with limits: the most for one payment, the most in 24 hours, and an amount beyond which the owner is asked. The program that holds the key enforces them, writes each payment down with the reason the assistant gave before making it, and never takes the assistant's word for the owner's. Where the assistant can ask its owner, it does. Where it cannot, it hands over a payment request that the owner pays from the app, and waits to see it arrive.

Software can also be paid. A service answers a request with the HTTP status "payment required" [22] and an invoice: an amount, and an address made for that one order from the seller's key. The buyer's program pays and asks again with its order's secret number; a paid order is served once. Neither side opens an account. On the running network, such a wallet, run on a phone, connected in 10 seconds, sent a payment in 1.0 second and saw it final 2.3 seconds later.

13. Post-Quantum Security

Every signature in the system is deterministic Falcon-1024, and there is no classical fallback: payments, votes, vouches, seat keys and the key that signs software updates. We use the deterministic variant that Algorand nodes use for their state proofs [6], vendored from Algorand's own code, which reproduces its 32 published test signatures byte for byte. A deterministic signer signs the same message the same way every time, so signing never depends on the quality of a phone's random number generator, and two votes from one seat for one slot can only be deliberate. Hashing is SHA3-256 throughout, and the tickets of section 6 are hash commitments, so nothing in consensus rests on elliptic curves. Addresses are hashes of public keys, and a public key appears on the chain only once, when its account first pays.

The cost is size. A Falcon-1024 signature is about 1.2 kilobytes, against 64 bytes for Bitcoin's, and post-quantum signatures cannot be cheaply combined. A first payment is about 3.1 kilobytes, 58% of it the sender's public key; once the network remembers the key, a payment is about 1.3 kilobytes. A certificate from a full committee carries up to 67 signatures, about 86 kilobytes, and votes are most of what a committee phone downloads: about 1.3 kilobytes per member per block. The second signature of section 3 adds from 4% to 24% more, depending on how often payments come; empty blocks receive none. Blocks travel without their certificates' signatures: deterministic Falcon gives each seat exactly one signature per vote, so a block names only which seats signed, and every member rebuilds the same certificate from the votes it already holds; late votes carried in later blocks travel without their signatures too. We accept this cost, rather than add a signature that will not last, and keep it bounded by the pace floor and the quiet pace of section 3.

Tor's own connections use classical key exchange, so every connection to a meeting point carries a second one inside it: an ML-KEM-768 key exchange [19], signed by the meeting point's Falcon seat key, after which every message is sealed with ChaCha20-Poly1305. The lock costs about 2% more data. A recording of today's traffic will therefore not yield its contents to a quantum computer later, and a phone knows it is speaking to the seat it meant to reach. A payment between two phones with no network (section 11) is sealed the same way, under an ML-KEM-768 key made for that payment alone. Two things remain, and we state them plainly. Tor's routing underneath is classical, so a quantum attacker with recordings could one day learn which phones connected to which meeting points, though never what they said. And the system rests on one signature algorithm; a second, unrelated key per account, based on hashing alone, would protect against a flaw found in Falcon itself, and is designed but not built.

14. Privacy

The traditional banking model achieves a level of privacy by limiting access to information to the parties involved and the trusted third party. Bitcoin kept privacy by keeping public keys anonymous [1]. We do the same, and go one step further for the network. Phones reach meeting points only at Tor onion addresses [16], and the chain accepts no other kind of address: no participant's Internet address is ever published, needed or shown. A meeting point does not learn where the phones that use it are, and a phone does not learn where its meeting point is. Requests that carry anything personal, such as a request to be vouched for, are sealed with ML-KEM for the one phone that will read them, and never written on the chain.

Amounts and accounts on the chain are public, as in Bitcoin. Shielded payments of the kind Zerocash introduced [17] hide them, but their proofs are heavy to produce on a phone and their soundness rests on assumptions that are not all post-quantum, so we do not use them. As an additional firewall, a new account should be used for each payment received, to keep them from being linked to a common owner. Wallets need no account with anyone; the only backup is twenty-four words.

15. Calculations

We consider the scenario of an attacker that controls a fraction q of the network's weight, after the trust cap of section 4, and wants a share of a committee large enough to harm it. With more than a third of a committee's weight, an attacker can halt finality for that epoch, and with an unlucky split of the honest members could try to finalise two histories; with a third or less, it can do neither. The attacker cannot choose the committee: tickets are fixed in advance and the random number is not known when they are committed. We therefore model each of the n committee seats as drawn independently, held by the attacker with probability q. The probability that the attacker holds more than a third of the seats is the tail of a binomial distribution:

$$P = \Sigma_{k > n/3}\, C(n, k)\, q^k\, (1 - q)^{n - k}$$

Converting to C code...

```
#include <math.h>
double AttackerHoldsAThird(double q, int n)
{
    double p = 1.0 - q;
    double term = pow(p, n);
    double sum = 0.0;
    int i;
    for (i = 0; i <= n; i++)
    {
        if (3 * i > n)
            sum += term;
        term *= (double)(n - i) / (i + 1) * q / p;
    }
    return sum;
}
```

Running some results for a committee of 100, with one committee drawn per epoch and, at the fastest pace, 105,120 epochs a year:

q=0.05	$P=1.3 \times 10^{-19}$	once in 7×10^{13} years
q=0.10	$P=7.0 \times 10^{-11}$	once in 136,000 years
q=0.15	$P=1.8 \times 10^{-6}$	once in 5.2 years
q=0.20	P=0.00074	about every 5 days
q=0.25	P=0.028	about every 3 hours
q=0.30	P=0.22	about every 23 minutes

Solving for the smallest committee that keeps the chance of such a committee under 0.1% a year:

q=0.05	n=36
q=0.10	n=75
q=0.15	n=147
q=0.20	n=318
q=0.25	n=897

A committee of 100 is therefore safe while an attacker holds under about a tenth of the network's weight, and the purpose of Proof of Trust is to keep it there: in our simulations, the trust cap held

500 bribed members out of 10,000 to 5.1% of the weight after three years. A larger committee would tolerate more, at the price of more votes for every phone to download. While the network is small, the committee is every available seat and the calculation does not apply: the network is then protected, as early Bitcoin was, by the honesty of its founding seats and by being worth little to attack.

We also measured how many payments the network can carry, with each key sent once and blocks that name their payments, at 5,000 payments a block. With four seats on one phone, every payment offered was final at 1,000 a second, still in about four seconds; the rules allow about 1,165. With seven phones over Tor, every payment was final at 400 a second, and the network carried up to 750. A block now costs each seat about 0.4 kilobytes per payment, and each payment's own journey about 1.3 kilobytes, so the limit is each phone's connection rather than the rules: about 55 payments a second through Tor at 100 kilobytes a second, and 140 at 250. On the running network, a test from one phone sent 24,000 payments in 34 seconds, about 705 a second, as fast as that phone could send them; the network made every one final, at about 800 a second, a median of 17.7 seconds after it was sent while the test ran. Appendix A gives the figures.

16. Conclusion

We have proposed a system for electronic payments that a phone can secure. We started with the usual framework of coins made from digital signatures, which provides strong control of ownership, and made every signature post-quantum. To agree on one history without proof-of-work or proof-of-stake, we gave the right to validate to people rather than to machines or money: a seat is vouched for by three others, its weight is earned by service and capped by the trust that reaches it. Committees and leaders are drawn by secret tickets protected by hashing alone, and a block of payments is final as soon as two-thirds of its committee have signed it twice. Seats are paid for every block they sign, even a few seconds late, so a phone far away earns as one nearby, and pay follows time, not blocks. Coins are exchanged with other chains by hash-locked swaps, with no bridge to trust. Two phones settle with no network, from money set aside under a guarantee the chain holds, and programs pay as people do, within limits their owner sets. Phones serve when they choose, keep only a recent window of the chain, catch up from snapshots the network certifies, and reach each other only through Tor, inside a post-quantum key exchange of their own. The network is robust in its unstructured simplicity. Phones work with little coordination, need not be identified to anyone but the three people who vouch for them, and can leave and rejoin the network at will, accepting the certified snapshot as proof of what happened while they were gone. Any needed rules and incentives are fixed in the genesis block and enforced by every phone alike.

Appendix A. The Numbers

The figures below come from three sources, named in each table: the running network as its Explorer showed it on 1 November 2026, the rules fixed in the genesis block and the code (version 1.56), and our whole-network simulations, which run the real code with delays like those of Tor. Where a figure is an estimate, it says so.

A.1 The network on 1 November 2026

What	Figure	Source
Running since	14 October 2026	network
Latest final block	110,102	network
Final block on 13 November 2026	168,399	network
Payout periods	942	network
Payments made final	170,363	network
Payments final in	2.9 s (median)	network
Partners (seats)	11, 6 of them on the committee	network
New block every	2 minutes while nobody pays (quiet pace)	network
Plata before the first block	0	genesis
Development fund	10% of every payout	genesis

A.2 For a partner

What	Figure	Source
Added to the yearly payouts by each full seat	10,000 Plata	rules
Earned by a full seat, on average	about 9,000 Plata a year, 24.7 a day	rules
Earned by a new seat (a tenth of full weight)	about 2.5 Plata a day	rules
Time from a tenth to full weight	90 days	rules
Payouts grow with the network until	10,000 full seats: 100 million Plata a year	rules
Payout period	120 blocks, paid for the time it lasted (at most 10 minutes at full pace, up to 5 hours at the quiet pace)	rules
Late votes still paid	within 10 slots, in full	rules
Deposit to hold a seat	none	rules
Joining	by a partner's invitation code, or a sealed note to the founding seats	app
Vouches needed for a seat	3, one from a founding seat or a seat a year old	rules
A seat may vouch after	30 days, 5 times a year (founding seats: no limit)	rules

A request to join expires after	7 days; 20 free requests per period	rules
Committee	at most 100 seats per period	rules
A place on the committee is offered for	the next two periods	rules
Two votes, or two blocks, for one slot	seat closed by the rules; for now recorded and shown (section 4)	rules
Data while serving	about 46 MB an hour per link at full pace; about 0.5 to 2 MB an hour at the quiet pace	simulation
Extra data for the post-quantum lock	about 2%	measured
Serving	on Wi-Fi by default; mobile data if allowed; on battery down to 20%, no charger needed	app

A.3 The network itself

What	Figure	Source
New block, full pace	every 2.5 s at the fastest	rules
New block, quiet pace (nobody paying)	every 30 s, 1 or 2 minutes, set by the founding seats (2 minutes on the network); full pace for 2 minutes after a payment	rules
Signatures for a final block	more than two-thirds of the committee's weight	rules
A slot gives up after	15 s; 6 s if nobody may lead it	rules
Payment final, four seats on one phone	1.9 s median, 4.7 s for 95%	simulation
Payment final, seven phones over Tor	about 8 s	simulation
Most payments a block	5,000	rules
Payments a second, four seats on one phone	1,000 offered, all final in 4.1 s; about 1,165 at most	simulation
Payments a second, seven phones over Tor	400 all final in 8.0 s; 750 at most	simulation
Payments a second on the running network, sent from one phone	24,000 sent at about 705 a second, all final, about 800 a second final; 17.7 s median under load	measured
Payments a second a phone's connection allows	about 55 at 100 KB/s, 140 at 250 KB/s	estimate
Block data per payment	about 0.4 KB (3.7 KB before keys were sent once and blocks named payments)	simulation
Size of a payment	3.1 KB the first time, 1.3 KB after	rules
Fees above the payouts	a fifth kept by the seats, the rest burned	rules
Blocks kept by each phone	the last 1,000, and the newest snapshot	rules
Snapshot pieces	256 KB each, checked one by one	rules
Catching up	up to 128 blocks or 1 MB per request	rules

Catching up, slim	final blocks without the keys and signatures the network already checked: about a tenth of the bytes, signed by the seat asked	rules
Fingerprint of the balances	named by every block, for two final blocks below it	rules
A link between meeting points	a sign of life every 15 s; closed and opened again after 45 s of silence	rules
Quiet pace of 30 s, four seats, four hours without payments	132 blocks an hour instead of 1,400; 1.8 MB an hour instead of 12.2	measured
Quiet pace of 2 minutes, no payments	about 30 blocks and 0.5 MB an hour per seat	simulation

A.4 For someone who pays

What	Figure	Source
Fee per payment	0.001 Plata	genesis
Payment final	about 2 to 3 seconds on the running network	network
Balance	answered by every partner; an open wallet is told at once when it changes	app
Signatures and keys	Falcon-1024 (deterministic), ML-KEM-768, SHA3-256	code
Internet address seen by anyone	never: Tor onion addresses only	rules
Backup	24 words	app
New app version found	within 10 minutes, fetched by itself on Wi-Fi	app
App size	about 16 MB	app
A swap of Plata for dollars on Arc	about 1 minute; no upper limit	measured
Swap cost to the seller	about 0.002 USDC, and a 0.01 USDC tip received	measured
Swap cost to the buyer	about 0.008 USDC in Arc fees, the 0.01 USDC tip, and 0.001 Plata	measured
Sending dollars from the app	0.005 USDC kept on each address for Arc's fee	app
Set aside to pay with no network	the amount, and the same again as a guarantee; in the app at most a quarter of the balance each	rules, app
A payment with no network	by Bluetooth, sealed with ML-KEM-768; about 8 KB with its record	measured
A record of what is set aside	signed by two founding seats; good for 23 days	rules
What is set aside comes back	30 days after its owner stops it	rules
The same money paid twice with no network	the first delivery is paid; the whole guarantee is forfeited	rules

A.5 For a program

What	Figure	Source
Ways in	a command, Python, TypeScript, and a server for AI assistants with 7 tools	code
A payment from a program on the running network	sent in 1.0 s, final 2.3 s later (run on a phone, 13 November 2026)	measured
A payment after a crash	sent again exactly as saved, never a second one	code
An assistant's limits	per payment, per 24 hours, and an amount above which its owner approves	code
Systems	Linux; Windows and Mac as previews	code

References

[1] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," 2008.

[2] Y. Gilad, R. Hemo, S. Micali, G. Vlachos, N. Zeldovich, "Algorand: Scaling Byzantine Agreements for Cryptocurrencies," *Proceedings of the 26th Symposium on Operating Systems Principles (SOSP)*, 2017.

[3] J. Chen, S. Micali, "Algorand: A secure and efficient distributed ledger," *Theoretical Computer Science*, vol. 777, 2019.

[4] S. Micali, L. Reyzin, G. Vlachos, R. Wahby, N. Zeldovich, "Compact Certificates of Collective Knowledge," *IEEE Symposium on Security and Privacy*, 2021.

[5] P.-A. Fouque, J. Hoffstein, P. Kirchner, V. Lyubashevsky, T. Pornin, T. Prest, T. Ricosset, G. Seiler, W. Whyte, Z. Zhang, "Falcon: Fast-Fourier Lattice-based Compact Signatures over NTRU," specification submitted to the NIST Post-Quantum Cryptography project, 2020.

[6] D. Lazar, C. Peikert, "Deterministic Falcon," specification and reference code, Algorand, 2021. https://github.com/algorand/falcon

[7] National Institute of Standards and Technology, "SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions," FIPS 202, 2015.

[8] M. Yin, D. Malkhi, M. K. Reiter, G. Golan Gueta, I. Abraham, "HotStuff: BFT Consensus with Linearity and Responsiveness," *Proceedings of the ACM Symposium on Principles of Distributed Computing (PODC)*, 2019.

[9] R. Gelashvili, L. Kokoris-Kogias, A. Sonnino, A. Spiegelman, Z. Xiang, "Jolteon and Ditto: Network-Adaptive Efficient Consensus with Asynchronous Fallback," *Financial Cryptography and Data Security*, 2022.

[10] M. M. Jalalzai, K. Babel, J. Komatovic, T. Klenze, S. Das, F. Elsheimy, M. Setrin, J. Bergschneider, B. Gilkalaye, "MonadBFT: Fast, Responsive, Fork-Resistant Streamlined Consensus," arXiv:2502.20692, 2025.

[11] Category Labs, "Asynchronous Execution," Monad documentation, 2025. https://docs.monad.xyz

[12] M. Baudet, A. Ching, A. Chursin, G. Danezis, F. Garillot, Z. Li, D. Malkhi, O. Naor, D. Perelman, A. Sonnino, "State Machine Replication in the Libra Blockchain," 2019.

[13] Q. Cao, M. Sirivianos, X. Yang, T. Pregueiro, "Aiding the Detection of Fake Accounts in Large Scale Social Online Services," *USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2012.

[14] N. Duffield, C. Lund, M. Thorup, "Priority sampling for estimation of arbitrary subset sums," *Journal of the ACM*, vol. 54, no. 6, 2007.

[15] R. C. Merkle, "Protocols for public key cryptosystems," *IEEE Symposium on Security and Privacy*, 1980.

[16] R. Dingledine, N. Mathewson, P. Syverson, "Tor: The Second-Generation Onion Router," *USENIX Security Symposium*, 2004.

[17] E. Ben-Sasson, A. Chiesa, C. Garman, M. Green, I. Miers, E. Tromer, M. Virza, "Zerocash: Decentralized Anonymous Payments from Bitcoin," *IEEE Symposium on Security and Privacy*, 2014.

[18] V. Buterin, D. Hernandez, T. Kamphefner, K. Pham, Z. Qiao, D. Ryan, J. Sin, Y. Wang, Y. X. Zhang, "Combining GHOST and Casper," arXiv:2003.03052, 2020.

[19] National Institute of Standards and Technology, "Module-Lattice-Based Key-Encapsulation Mechanism Standard," FIPS 203, 2024.

[20] M. Herlihy, "Atomic Cross-Chain Swaps," *Proceedings of the ACM Symposium on Principles of Distributed Computing (PODC)*, 2018.

[21] Anthropic, "Model Context Protocol," specification, 2024. https://modelcontextprotocol.io

[22] R. Fielding, M. Nottingham, J. Reschke, "HTTP Semantics," RFC 9110, section 15.5.3, 2022.