Shift: Building Blockchains Without Validators

Sureshot Labs

Abstract

What if it were possible to build a decentralized blockchain network similar to Bitcoin and Ethereum, but without any miners or validators to process transactions? What if a sender and a recipient could independently settle transactions among themselves without compromising security? If such a protocol could be realized, it would enable a truly peer-to-peer digital financial system fundamentally altering our understanding and perception of blockchains. Practically, this system could offer unparalleled scalability, eliminate transaction fees altogether, enable instant settlement, and obviate the necessity for economic subsidies, for instance token inflation. This paper presents such a system called **Shift**, a novel protocol that re-imagines the foundational structure of blockchain networks and the financial ecosystems built upon them.

1 Introduction

The main issue in designing a decentralized digital currency is the so-called *double-spend problem*, *i.e.*, how to prevent a user from spending the same money more than once. In a centralized digital currency model, the issuer or a trusted third party (TTP), such as a bank, maintains an up-to-date book of each user's balance. Every money transfer from a sender to a recipient is routed through the TTP, who then updates the sender's and recipient's balance accordingly, thereby preventing double-spends.

The first viable solution to the double-spend problem in a decentralized setting was proposed in the Bitcoin whitepaper [21]. The paper introduced the idea of a *peer-to-peer* (P2P) digital cash that replaces the central minting and accounting authority with a distributed network of nodes that detect and reject double-spend transactions. This is achieved by replicating the bookkeeping process of a centralized entity on nodes across the entire network. In Bitcoin, this involves keeping track of which monies are yet to be spent, *aka*, Unspent Transaction Outputs (UTXOs).

However, making the bookkeeping decentralized is not sufficient as malicious nodes in the network may maintain a copy that is different from that of the honest nodes. To address this, Bitcoin relies on a distributed consensus protocol referred to as the *Nakamoto consensus* to agree on the state of UTXOs.

Ethereum [30], Solana [31] and other next-generation protocols extended the idea of Bitcoin from processing of simple payments to execution of stateful Turing-complete programs called *smart contracts*. These protocols operate in a similar manner, where instead of a central party executing smart contracts, the entire distributed network of nodes called validators is tasked to execute state transition functions and agree on the post-execution program state using a Byzantine Fault Tolerance (BFT) consensus protocol.

1.1 Problem Statement

While Bitcoin and all the ensuing smart contract networks such as Ethereum have been oft-referred to as P2P networks, they are in fact far removed from being a truly P2P system as every transaction to either transfer value or to execute code needs to be shared with a large number of decentralized network of nodes. On the other hand, a true P2P system like physical cash does not rely on another centralized or decentralized system to process the transfer of funds. A cash transfer involves only the sender and the recipient.

Although the transactional model in Bitcoin, Ethereum or Solana is effective in achieving decentralization and trustlessness, it introduces several key limitations:

1. **Scalability:** Each node must maintain a complete copy of the blockchain, limiting the network's ability to handle high transaction volumes. Currently, Bitcoin can process around 7 transactions per second, far below the throughput required for mainstream financial use. Ethereum can process about 100 transactions per second. Scaling solutions like rollups can offer better scalability, but their capacity is limited by the underlying settlement layer.

2. **Long Finality Time:** With an average block time of 10 minutes, Bitcoin transactions can take up to an hour to reach finality, making it unsuitable for real-time payment. Ethereum and other scaling solutions have better finality, but they still require 12s-12mins for finality.

3. **High Transaction Costs:** As transactions need to be validated by a large decentralized network, the cost for end users is much higher compared to the cost they are used to paying in traditional banking systems.

4. **Reliance on Economic Subsidies:** All these systems heavily rely on incentives to miners and validators through token inflation and subsidies. Such incentives are often not sustainable and will eventually lead to high transaction costs when those subsidies are either not meaningful or not available.

5. **Privacy Concerns:** Every transaction is broadcast to and validated by the network, compromising user privacy as transaction details are publicly accessible.

A truly P2P transactional network, similar to physical cash would address all these limitations. For instance, physical cash transfers are by design parallelizable in the sense that any number of two users could be transacting with each other in parallel without creating any processing bottlenecks. Additionally, physical cash transfers are instantaneously settled, incur zero cost and are by design privacy-preserving. However, the main downside of physical cash is that it is not digital and therefore can not be used over a network like the Internet which strictly restricts its usage to face-to-face transactions. Moreover, by the virtue of being physical, cash cannot be used as programmable money.

In light of these issues, this paper raises the following two key questions:

Question 1: *Is it possible to build a truly P2P decentralized protocol, where a recipient directly verifies if the sender's transaction is not a double-spend?*

Such a system, if possible to design, dramatically changes the way blockchains like Bitcoin, Ethereum and Solana have been perceived and designed so far which all assume a network of nodes to validate and process transactions. This new system would resemble a blockchain devoid of a network of validators. And importantly, the system will not require any consensus protocol to agree on a post-transaction state.

> **Question 2:** *Can such a P2P system support programmability thereby enabling more feature-rich applications like P2P trading, P2P lending etc.?*

A true P2P digital payment system and by extension remittances in itself is novel, but, it would be ideal if the system could support programmable money allowing for more feature-rich applications such as trading or lending/borrowing of assets.

2 Intermezzo: (Im)Possibility of Building Such a System

Without a centralized entity or a decentralized network validating user transactions, it may appear that it is not possible to build such a system. The main issue is how should a sender convince a recipient that a certain transfer is not a double-spend transaction without attestation from a TTP or validation from an external network.

2.1 Strawman Solution I

A curious reader may wonder why can not such a system be built using zero-knowledge proofs given that one can construct a *non-interactive zero-knowledge* (NIZK) proof for any NP statement [9]. In order to analyze this possibility, we have to assume client-side proof generation, *i.e.*, the sender of a transaction can generate a ZK proof that she is not attempting a double spend. The sender should then be able to convince a recipient that she indeed has a certain balance and create a ZK proof that the recipient can verify locally without the need to rely on any network or TTP.

However, the sender can always send the same ZK proof to any other recipient and unless the two recipients communicate with each other, it will be impossible to detect double-spends.

2.2 Strawman Solution II

Another potential solution could be to make use of a *one-time signature* (OTS) scheme, first proposed by Lamport in [17]. Similar to a standard signature scheme, OTS allows a signer to use a key pair to sign a single (arbitrary) message. But, if a key pair is used to sign a second, different message, no security guarantees are given. A variant of OTS is *extractable one-time signature* (EOTS) scheme. An EOTS scheme has the property that if two different messages are signed using the same secret key, then one can use those two signatures to recover the secret key. Such a scheme built using a variant of Schnorr's signature scheme [24] is used in Babylon [28] to dissuade nodes from equivocating in a BFT protocol.

When applied to our setting, a sender would sign a transaction using an EOTS scheme, and send the transaction to the recipient. If the sender signs another transaction to spend the same money, anyone with access to the two signed transactions can extract the sender's signing key, therefore disincentivizing the sender to do so.

The EOTS-based solution has better properties than the one using ZK proofs, however, relying on incentives in a fully P2P system is not ideal. As the sender may send those two transactions privately to two different non-communicating recipients which would make collating the two transactions to recover the secret key nearly impossible.

2.3 Key Takeaway

For either of the strawman solutions to work, the P2P network should form a complete graph, *i.e.*, the P2P network has a topology where every peer is connected to every other peer and that every transaction received by a peer gets shared with every other peer reliably. This is clearly impractical.

The analysis of the solutions however reveals the need of a crucial primitive: *one-time proofs* (a cryptographic proof that can only be verified once) or more generally, a primitive that can restrict how many times a certain signing key can be used without the need for the recipients to communicate.

Albeit in a different context, this notion of one-time proofs was first formally studied by Goldwasser *et al.* in [14]. One-time proofs are cryptographic proofs that can only be verified once and then become useless and unconvincing. It was proposed as an application of *one-time programs* (OTP), which are programs that can be executed on a *single* input, whose value can be specified at run time. An OTP is like a black box function that may be evaluated once and then "self destructs".

An obvious takeaway would be to use such a primitive for our construction. However, prior works [7, 13] suggest that it is impossible to build such a system purely using cryptography. In light of the impossibility results, the key insight in [14] is that it is possible to build a universal OTP compiler for any underlying program, if we assume access to a secure hardware. The hardware required is a secure memory with no compute capabilities eliminating side channel attacks and other esoteric attacks that rely on extracting information from the compute being run in the hardware.

3 Contributions

Building upon the work of Goldwasser *et al.* [14] and [11, 32], we propose a novel protocol called **Shift** to affirmatively answer Questions 1 and 2. Given the impossibility results on building such a system purely using cryptography, Shift leverages a minimal and secure hardware. The hardware is minimal in the sense that its supported feature set particularly in terms of its compute capabilities is much restricted compared to that provided by off-the-shelf *trusted execution environment* (TEE) chips such as Intel SGX [1] which are known to be prone to numerous side channel and other attacks [23].

Despite the restricted compute capabilities of the hardware used in our design, Shift can support a wide variety of applications while keeping the surface area for hardware-related attacks to an absolute minimum to a point where most if not all can be completely eliminated.

Our Solution. The hardware used in Shift can in fact be seen as an extension of existing hardware wallets such as Ledger[1] [2] with two additional capabilities:

1. **It "encumbers" the private key immediately after a transaction has been signed. Key encumbrance [18] is done in such a way that an encumbered key cannot be used to sign any other transaction.**

   By encumbering the signing key, the system ensures that funds controlled by the key can only be spent once, thereby solving the double-spend problem.

[1] Ledger is a hardware that securely manages signing keys and signs messages sent to it without the signing key ever leaving the device.

2. **It supports *remote attestation* [5] — a security service that allows a remote verifier to reason about the identity and integrity of a device.**

   Remote attestation ensures that the party receiving the funds provides an address that is generated from within a device with the key encumbrance property. This ensures that the double-spend is prevented at the recipient-level and it cascades to the next recipient.

By limiting the hardware capabilities, we show that even a simple Ledger-like hardware if equipped with only two additional capabilities can be used to build an entirely new financial system. We use this hardware in Shift to design the first truly P2P protocol that completely eliminates the need for a decentralized network for transaction validation.

Transactional Flow. For ease of explanation, we assume a Bitcoin-style UTXO model. It is worth highlighting that relying on a UTXO model does not restrict our protocol in any material way, as prior works have shown functional equivalence of UTXO and account-based models [3].

In this setting, the transactional flow in Shift is as follows: The sender checks whether the recipient's address is indeed an address generated from within the secure device. This is done via remote attestation. The sender's device then signs a transaction to transfer funds to a recipient's address. Immediately after signing the transaction, the sender's device encumbers the private key in a way that it cannot be used to sign any other transaction, thereby spending the entirety of the funds controlled by the private key. The signed transaction is then sent to the recipient's device which in turn verifies if the signature on the transaction is valid and that it came from an untampered device. This is again done via remote attestation. In the happy path, if all the verification checks have passed, then the transfer is considered successful.

Benefits. Shift is a simple and elegant protocol to enable instant, direct P2P transfer of assets and more, with zero fees, and without the economic subsidies typical of blockchain networks. It presents a new way to design financial networks that is drastically different from that of existing blockchains which rely on validators and full nodes to secure transactions and therefore are required to process every single transaction. Shift by design eliminates many pitfalls of existing blocckhain designs.

As transactions in Shift do not need to be sent to any external network, there is no need to run any BFT-style consensus protocol to agree on which monies are yet to be spent. Furthermore, by eliminating the need for any participant to maintain any global state, each sender and recipient pair is in effect tasked to independently secure their transactions. In other words, the system operates autonomously.

Another implication of Shift's design is that it eliminates the need of different tertiary blockchain services such as indexers, RPC endpoints, block explorers, etc. Note that a traditional blockchain is nearly unuseable without any of these services. Shift renders these services fully redundant which makes bootstrapping a fully-functioning ecosystem around Shift much easier, cost-effective and low-effort compared to that of traditional blockchains.

Applications. Beyond the obvious usecase of P2P payments and remittances, a simple modification to Shift and the hardware can allow for P2P trading of assets and P2P lending protocols by leveraging hashed-time lock contracts [15].

We further believe that the underlying hardware used by SHIFT is of independent interest for different applications. One of these is a new kind of hardware wallet that could be considered "hot" as, unlike Ledger, it can connect to the Internet. This new device combines the security of a cold, hardware-based wallet (such as Ledger) with the enhanced user experience of hot wallets, such as browser-based wallets like MetaMask[2].

Moreover, unlike existing hardware such as Ledger that are designed for transaction signing initiated by a human, the hardware used by SHIFT can be used by humans, machines as well as software particularly, AI agents. The autonomous nature of SHIFT is in fact ideal to build a rather inexpensive mechanism for agent-to-agent transactions. The underlying hardware can also be adapted in different industry settings such as robotics and IoT devices, where the hardware is embedded in the device, allowing one device to seamlessly transact with another in an inexpensive manner.

SHIFT and the underlying hardware can also serve non-financial usecases such as building a digital voting device, allowing a user or a machine (read robots or agents) to vote only once using the device. The key encumbrance property will guarantee that the device is only allowed to cast its vote once. Another useful application (previously explored in the literature) is to build access tokens, where the device can prove to a verifier that it holds some secret for a certain time period.

Summary. To summarize, this paper makes the following contributions:

1. Presents the design of the first truly P2P system called SHIFT to prevent double-spends using a minimal extension of a Ledger-like hardware. The system is designed to avoid pitfalls of existing blockchains and decentralized networks.

2. Demonstrate several approaches to building such a hardware by identifying its minimally required feature set. We consider a modular design that allows us to customize the different stacks resulting in a flexible architecture to adapt the system based on the specific usecase that may require a certain underlying chip, a specific key encumbrance policy or a preferred remote attestation mechanism.

3. Discuss several applications of SHIFT such as: A) a hardware-wallet that is designed to be used as a hot wallet, opening up a new design space for wallets that take the best of hardware-based cold wallets such as Ledger and software-based hot wallets such as MetaMask, B) a new payment system that is truly P2P, C) extensions to support P2P trading and P2P lending/borrowing via hashed time-lock contracts and more.

4 System Setting

In this section, we outline the system and network settings under which SHIFT operates, the underlying transaction model, and the required feature set from the secure hardware.

P2P Network. We assume a network of participants, these could be humans, machines or software each owning a secure hardware with certain properties (as presented later in the section). We assume that any two transacting entities have an insecure and semi-reliable communication channel

[2] https://metamask.io/

between them, whereby, the channel can be public for anyone to see, but any messages posted in the channel outbox can only be delayed by up to a certain fixed upper time bound Δ. For simplicity, we also assume that the same communication channel is available to all the participants. Finally, at any given time, any pairs of these participants can be transacting with each other via the communication channel.

Transaction Model. Without loss of generality, we assume a UTXO-model for transactions. Each UTXO represents coins that are yet to be spent and is associated with a certain public key, such that it can only be spent by providing a signature using the corresponding private key. Each transaction in this model consists of certain number of input and output UTXOs. The input UTXOs represent monies being spent, while output UTXOs represent who receives the monies. Unlike account-based models, all coins in an input UTXO need to be spent in the transaction and therefore the transaction may have an output UTXO to send any change back to the transaction sender.

Secure Hardware. We assume a minimal extension of existing hardware wallets like Ledger. The hardware device is assumed to be equipped with the following capabilities:

1. **Secure Memory:** Each device contains a secure memory, which securely stores private keys for the users. It can generate as many keys as possible for the user.

2. **Secure Signing:** The hardware can sign user transactions using a specified private key without exposing the private key to the user or any other external client. The private key should never leave the device.

3. **Key Encumbrance:** Once a transaction is authorized, the hardware encumbers the key in such a way to prevent the user from reusing the key to sign multiple transactions.

4. **Attestation Reports:** The hardware generates a remote attestation report for each transaction, confirming the identity and integrity of the hardware.

5. **Inter-Device Verification:** Devices can verify remote attestation reports from other devices, enabling users to authenticate each other's transactions directly.

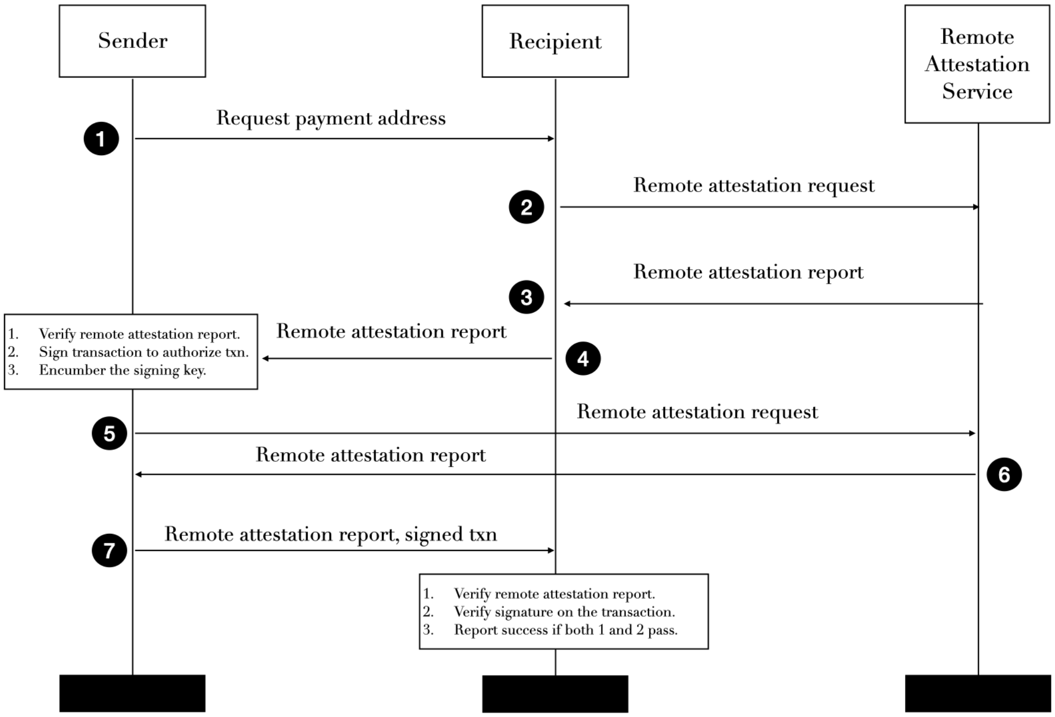
Properties 1, 2 are already offered by hardware wallets such as Ledger. Properties 3-5, are additional features that we add to a hardware like Ledger.

5 Overview of Shift

Assuming such a hardware can be built, the system operates as follows.

Transaction Life Cycle.

1. **Address Verification:** A sender requests the recipient an address and a proof (in the form of an attestation report) that the address was generated from within the device.


Sender
Recipient
Remote Attestation Service
1
Request payment address
2
Remote attestation request
3
Remote attestation report
1. Verify remote attestation report.
2. Sign transaction to authorize txn.
3. Encumber the signing key.
Remote attestation report
4
5
Remote attestation request
Remote attestation report
6
7
Remote attestation report, signed txn
1. Verify remote attestation report.
2. Verify signature on the transaction.
3. Report success if both 1 and 2 pass.

2. **Transaction Initialization and Signing:** The sender's device then verifies the attestation report and if the proof is valid, then the sender initiates a transaction to transfer funds and requests the hardware to sign it with a corresponding private key. The transaction format could be either UTXO or account-based. In the UTXO model, the transaction takes an existing UTXO as input and creates a new UTXO that sends the monies to the recipient and optionally another UTXO to receive any change back to a fresh address created by the hardware.

3. **Key Encumbrance:** Once the transaction has been signed, the device will irreversibly encumber the signing key in way that the same key cannot be used to sign any other transaction. As keys are only-used once, the address to receive any change must be a fresh unused address.

4. **Attestation Generation:** The hardware then generates an attestation report that serves as proof of hardware's identity and integrity guarantees, including the proof of key encumbrance.

5. **P2P Verification:** The recipient's hardware then verifies the attestation report, confirming the transaction's authenticity and accepting it as final.

6 Threat Model

Unlike blockchain networks like Bitcoin, Ethereum, and Solana, which prevent double spending through state machine replication, Shift relies on secure hardware for security. Therefore, we refer to [25] for a brief discussion of the threat model, as it serves as a helpful reference in the context of secure hardware (see Figure 1).

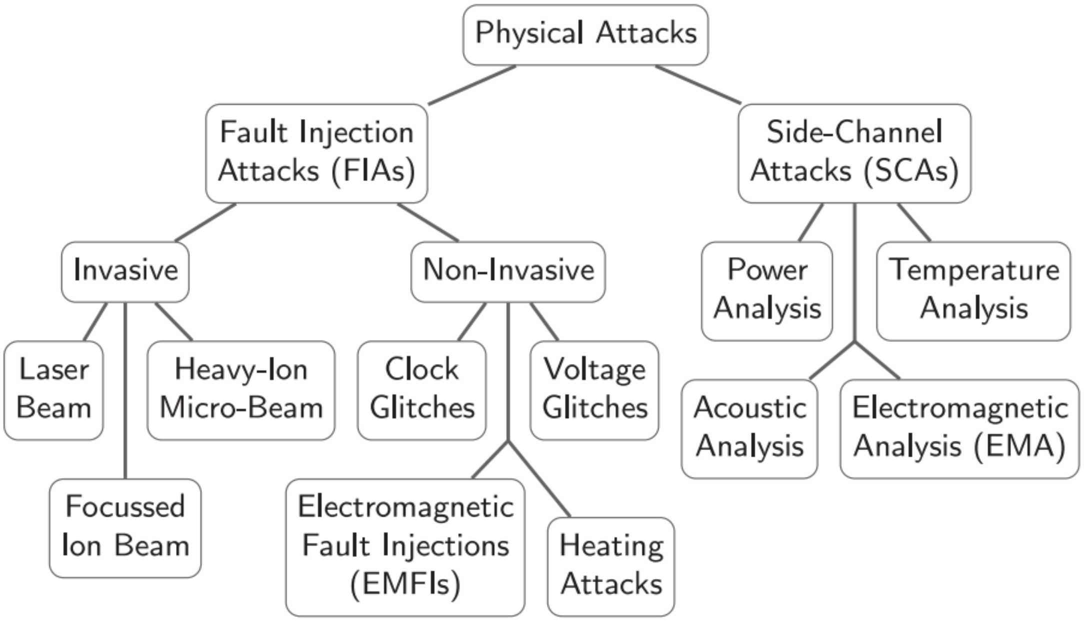


Figure 1: Figure from [25] — taxonomic scope for physical attacks.

Moreover, unlike many real-world systems (such as those relying on Intel SGX), which assume secure servers operated by a cloud service provider with restricted physical access, Shift's P2P design requires each user to possess the underlying hardware. Consequently, our threat model assumes that adversaries have full access to both the hardware and any associated software components. This enables them to launch attacks that fall into the following broad categories:

(1) Software attacks (including software-based micro-architectural attacks),

(2) Physical side-channel and fault injection attacks [25].

(3) Other hardware attacks (e.g., NAND mirroring [26], hardware trojans [29]).

Attestation Key Extraction Adversary. The most serious attack that an adversary may mount is extracting the *attestation key* — hardware *root-of-trust* embedded in the hardware by the chip manufacturer in certain types of hardware. The attestation key is used in Shift to prove the integrity of the hardware and the code being run on it to any remote party. Key extraction therefore allows the adversary to impersonate the hardware and affect a double spend.

To mitigate this risk, we plan to transition from chips embedded in mobile devices to chips that have stronger security guarantees against physical attacks. Such chips use physical unclonable function (PUF) [12] to generate and store the attestation key, making it substantially much more difficult to perform chip attacks to extract the key while at rest. In order to guard against side channel and fault injection attacks that would be used to extract the key meanwhile it is being used, more secure chips would use hardware masking [10] and circuit redundancy [6,20]. Given this approach, we must also adopt an evolving threat model that incrementally accounts for stronger adversaries, adapting to hardware advancements to further harden security against such attacks.

The implementation will proceed in phases (more details in Section 8). In the first phase, we will rely on currently available hardware. Consequently, our threat model will assume that chip manufacturers are honest and that physical attackers have limited capabilities to perform side-channel attacks, fault-injection attacks, or other hardware-based attacks to extract the attestation key.

In the second phase, while we will likely still need to trust chip manufacturers, we plan to leverage emerging hardware technologies—such as PUF-based chips—that offer stronger security guarantees against physical attacks. In the third phase, we will no longer assume the manufacturer is honest. Instead, they will be considered a potential adversary and incorporated into the threat model. To that effect, chip verification techniques such as in [22] could be used.

Our ultimate goal is to develop an implementation that is resilient against all forms of attacks, including all types of physical attacks. The challenges of such an implementation are not trivial and assumming that a perfectly secure device could be constructed we still have the challenge of remotely verifying the device, which is achieved via remote attestation with current trusted hardware. One core challenge with remote attestation is that it requires trusting a chip verifier to establish a trusted chip registry.Our long term vision is to move towards a device-independent protocol [19].

7 Attestation Key Extraction: Detection, Mitigation and Recovery

In this section, we consider the absolute worst-case scenario, where the security of the underlying hardware is only relative to the attacker's capabilities and may not withstand physical attacks — particularly those aimed at extracting the hardware root of trust (*i.e.*, the attestation key). As discussed in Section 6, such an attack would compromise the security of Shift if it relied solely on hardware security.

Below, we propose a hardened version of Shift that incorporates a public bulletin board. This hardened design enables the detection of attacks — including, but not limited to, attestation key extraction, that could result in double spending.

It is worth noting that detecting a key extraction attack is inherently challenging. However, once such an attack is identified, recovery is relatively straightforward. This is because the compromised attestation key can be removed from the chain of trust, effectively preventing the attacker from executing any double-spend attempts. This also presents a disincentive to mount key recovery attacks.

7.1 Hardened Protocol

In the hardened protocol, we assume the existence of a public bulletin board, such as a public blockchain, where transaction data and UTXOs (representing the system state) can be periodically posted. This blockchain however does not require a runtime or the ability to execute smart contracts. Hence, a data availability layer such as Celestia [4] that enables persistent storage of each user's last checkpoint would also be a suitable alternative. For simplicity, however, we assume the bulletin board is a public blockchain like Ethereum or Solana.

The hardened protocol introduces the following two modifications, while the rest of the Shift protocol remains unchanged:

1. **Checkpointing by the Sender:** In the hardened version, the sender must post the transaction data (signed by the attestation key) to the blockchain, along with the updated set of UTXOs (*i.e.*, any newly created UTXOs from the transaction). This checkpointing mechanism ensures that, even if the sender were to extract the attestation key in the future, an audit trail exists to verify the provenance of any funds being spent. Additionally, the transaction data sent to the recipient now includes an on-chain reference to the creation of the input UTXO.

2. **Additional Verification by the Recipient:** In the hardened version, the recipient is required to verify that the UTXO being spent was indeed created in a previous transaction, as referenced on the bulletin board. Only if this verification check passes can the recipient accept and settle the transfer.

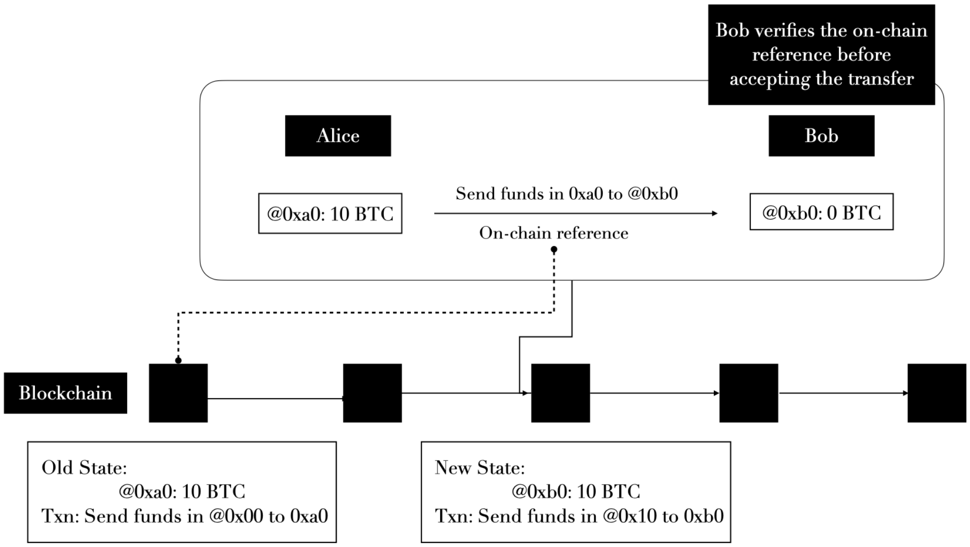

Bob verifies the on-chain reference before accepting the transfer
Alice
Bob
@0xa0: 10 BTC
Send funds in 0xa0 to @0xb0
On-chain reference
@0xb0: 0 BTC
Blockchain
Old State:
@0xa0: 10 BTC
Txn: Send funds in @0x00 to 0xa0
New State:
@0xb0: 10 BTC
Txn: Send funds in @0x10 to 0xb0

Figure 2: Hardened protocol to detect attestation key extraction attacks. Transaction data and the modified state is posted on a public bulletin board such as a blockchain.

Illustrative flow: As shown in Figure 2, consider a user, Alice, who has 10 BTC in an address labeled `0xa0`. She received these funds as part of a transaction from an address `0x00`, and this transaction was recorded on the bulletin board as required by the protocol. The data published on-chain is signed by the sender's hardware.

Now, when Alice wants to send funds from `0xa0`, she requests an address from Bob (`0xb0`) and creates a transaction signed by her hardware to move the funds. Along with this transaction, Alice also sends the on-chain reference to the transaction where she originally received the funds in `0xa0`. Upon receiving the transaction, Bob verifies all necessary signatures and additionally cross-checks the on-chain pointer by directly querying the bulletin board.

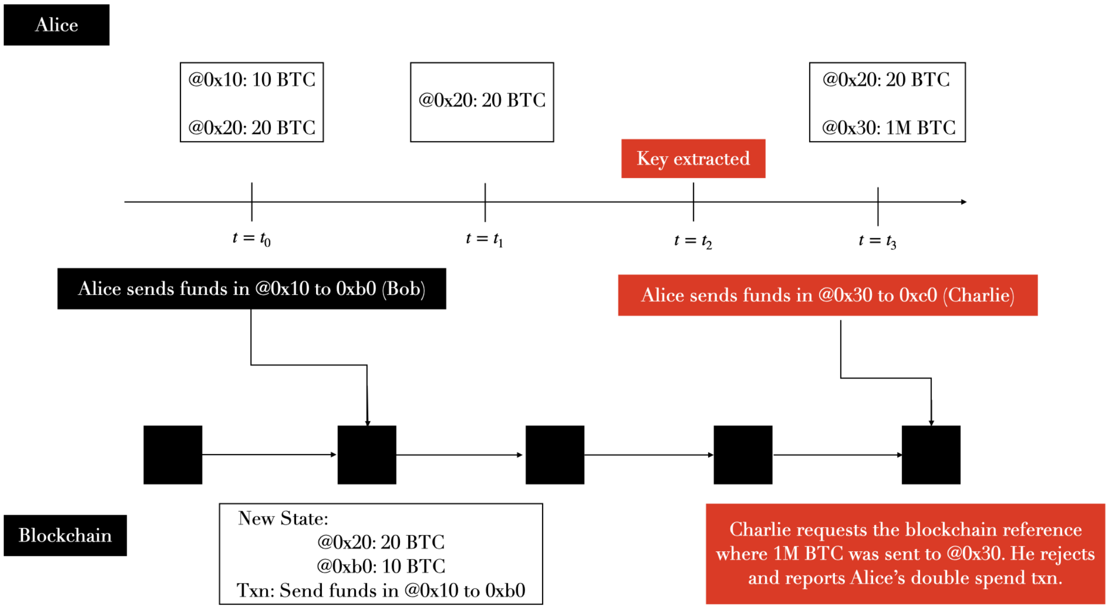

Alice
@0x10: 10 BTC
@0x20: 20 BTC
@0x20: 20 BTC
@0x20: 20 BTC
@0x30: 1M BTC
Key extracted
t = t0
t = t1
t = t2
t = t3
Alice sends funds in @0x10 to 0xb0 (Bob)
Alice sends funds in @0x30 to 0xc0 (Charlie)
Blockchain
New State:
@0x20: 20 BTC
@0xb0: 10 BTC
Txn: Send funds in @0x10 to 0xb0
Charlie requests the blockchain reference where 1M BTC was sent to @0x30. He rejects and reports Alice's double spend txn.

Figure 3: Detection of attestation key extraction attack in the hardened protocol.

7.2 Security Analysis

To analyze the security of the hardened protocol (refer to Figure 3), let us assume a malicious user, Alice, who is honest until a certain time $t = t_2$, when she successfully extracts the attestation key from her hardware. We assume that at $t = t_0$ and at $t = t_1$, Alice behaves honestly as she has not yet been able to recover the attestation key. Therefore, the transfer of funds to Bob at $t = t_0$ happens as per the protocol and therefore at $t = t_1$, the only UTXO Alice owns is the one labelled `0x20`.

At $t = t_2$, Alice successfully extracts the attestation key. At this point, she can sign arbitrary messages using the attestation key. Alice attempts to perform a double spend, however, since each address in the protocol is only used once, any re-use of an address can be easily detected via the bulletin board. As a result, Alice's attestation key will be removed from the chain of trust, preventing the double spent from being validated.

Alice is therefore obliged to use a fresh address to attempt to mint money out of thin air. She creates a transaction to spend 1M BTC (created out of thin air) from an address 0x30 to Charlie (0x30). Charlie, unaware that Alice has extracted the attestation key, perceives any signed message from Alice to be valid, as he still trusts the authenticity of Alice's attestation proofs. However, However, because Alice is now using a new address and her previous keys have been removed from the chain of trust, the transfer is ultimately subject to verification through the bulletin board.

Alice now creates a transaction to spend 1M BTC (created out of thin air) and uses the extracted attestation key to generate an attestation proof, claiming that the transaction was signed using untampered hardware. However, as part of the transaction, Alice is required to provide an on-chain reference to where she had received the 1M BTC from. Since the 1M BTC were created out of thin air, she cannot provide a legitimate past reference. Any false reference can be cross-checked by Bob by querying the bulletin board. This allows Bob to detect the attack and present the evidence to a remote attestation authority, which can then remove the rogue hardware from the chain of trust, effectively invalidating Alice's double spend attempt.

7.3 Performance Analysis

The hardened protocol eliminates the incentive to extract the attestation key from the hardware. However, it introduces certain overhead in terms of latency, throughput and has an impact on transactional privacy. We discuss these three points categorically below.

Throughput. As each transaction data in the hardened version needs to be posted on the bulletin board, the throughput of SHIFT will depends on the data bandwidth of the underlying bulletin board. However, as there is no need for any compute, the protocol can leverage any high data availability network without sacrificing considerably on the throughput of the system as users can still transact in parallel.

Latency. Similar to the previous argument, the hardened protocol will now have the finality of the bulletin board. This may not be ideal if we are using a public blockchain as the bulletin board. In order to address the finality concerns, users may optimistically settle transactions similar to optimistic rollups which will make transactions almost instantaneous.

Privacy. As every transaction needs to be posted on the bulletin board, the privacy properties of the original design are no longer guaranteed. A potential solution to this could be to leverage zero-knowledge proofs where the sender proves to the recipient that the coin being spent is one of those on the bulletin board without revealing the exact reference. Such a protocol was first proposed in the context of ZeroCash [8] — a privacy preserving version of Bitcoin.

8 Implementation Roadmap

In this section we provide a high-level directions to implement the first prototypes of SHIFT. We explore multiple possible iterations, in which the system's security is improved incrementally. Since SHIFT relying on a secure hardware, one of the core challenges of the implementation is to ensure that the security of the hardware holds against attacks, more particularly physical attacks. Hence, the security and maturity of our system will likely be limited by that of the available hardware.

Keeping this in mind, we propose four main phases to our implementation, that will be centered around the currently and *projected* available hardware. We propose the following timeline:

Year 1-3: Leverage novel chips that will have hardened the physical security with respect to physical attacks. These chips would likely be closed source and thus difficult to verify, and would require us to still put heavy trust into designers and manufacturers, but nevertheless, would offer stronger security guarantees such as PUF-based key generation and storage, and hardware masking to protect against side-channel attacks for instance. To help reduce the trust in manufacturers, we could try to leverage verification techniques such as IRIS [16]. In the very optimistic case, remote attestation could be anchored into the chip's unique PUF key. Trust in the PUF key, rather than being backed by the chip manufacturer would be backed by a set of verifiers using a verification technolgoy such as IRIS [16]. One of the main drawbacks of this phase is that we would still be relying on difficult to verify closed-source hardware. Another drawback is that even in the best case in which we can verify chips for hardware trojans, these verification techniques are likely to be limited and the verification process will introduce a perhaps challenging process that will require some kind of simple consensus protocol. In other words, we may have to trust a committee of chip verifiers.

Year 3-5: Leverage novel open source chips that will facilitate their verification in order to lower the need to trust manufacturers. These chips would also offer increased physical security, with respect to both side-channel and fault injection attacks. With open source chips we should be able to perform extensive verifications to ensure that chips have been implemented as per their design such as in [22]. The verification process will still likely require setting up a committee of verifiers to protect against malicious verifiers. The verification process would act a trust establishment phase in which the PUF public key of a chip would be recorded as belonging to a vetted open source chip design. Remote attestation would then be anchored in this "trusted" PUF public key.

Year 5+: The optimistic case, the previous phase is likely to need improvements and security hardening. One hope for the coming years, is that chip verification techniques would become non-destructive which would open the door to multiple and repeated verifications by different parties. We should also by then have a better understanding of the security limits of classical hardware with respect to PUFs, and hardware masking for instance. Moreover, it is reasonable that quantum-based hardware approaches will have evolved considerably, especially considering already existing developments and ideas such as in [27].

It needs to be very clear that in order for our system to be reliable, the hardware we use must be verifiable and crypto-physically secure. By "crypto-physically" secure we mean that the security of the hardware is guaranteed by cryptography and physics. By "verifiable" we mean that a remote verifier must be capable to verify that it is communicating with hardware that is crypto-physically secure. As far as we understand, this level of secure hardware is not available today, but there's significant research and development towards realizing much more secure hardware, and it could be argued that it's a good timing to paddle in order to catch the wave. Meanwhile the R&D that will hopefully yield the required hardware, is on-going, prototypes can be built in a modular way, that will allow swapping the less secure hardware for more secure hardware.

9 Conclusion

We have a lot of work to do!

References

[1] Intel software guard extensions (intel sgx), https://www.intel.com/content/www/us/en/products/docs/accelerator-engines/software-guard-extensions.html

[2] Ledger: Hardware wallet & crypto wallet - security for crypto, https://www.ledger.com/

[3] Adler, J.: Accounts, strict access lists, and utxos (2020), https://forum.celestia.org/t/accounts-strict-access-lists-and-utxos/37

[4] Al-Bassam, M.: Lazyledger: A distributed data availability ledger with client-side smart contracts (2019), https://arxiv.org/abs/1905.09274

[5] Banks, A.S., Kisiel, M., Korsholm, P.: Remote attestation: A literature review (2021), https://arxiv.org/abs/2105.02466

[6] Bar-El, H., Choukri, H., Naccache, D., Tunstall, M., Whelan, C.: The sorcerer's apprentice guide to fault attacks. Proceedings of the IEEE **94**(2), 370–382 (2006). https://doi.org/10.1109/JPROC.2005.862424

[7] Barak, B., Goldreich, O., Impagliazzo, R., Rudich, S., Sahai, A., Vadhan, S., Yang, K.: On the (im)possibility of obfuscating programs. J. ACM **59**(2) (May 2012), https://doi.org/10.1145/2160158.2160159

[8] Ben Sasson, E., Chiesa, A., Garman, C., Green, M., Miers, I., Tromer, E., Virza, M.: Zerocash: Decentralized anonymous payments from bitcoin. In: 2014 IEEE Symposium on Security and Privacy. pp. 459–474 (2014). https://doi.org/10.1109/SP.2014.36

[9] Blum, M., Feldman, P., Micali, S.: Non-interactive zero-knowledge and its applications. p. 103–112. STOC '88, Association for Computing Machinery, New York, NY, USA (1988). https://doi.org/10.1145/62212.62222, https://doi.org/10.1145/62212.62222

[10] Cassiers, G., Masure, L., Momin, C., Moos, T., Standaert, F.X.: Prime-field masking in hardware and its soundness against low-noise sca. IACR Transactions on Cryptographic Hardware and Embedded Systems **2023**(2), 482–518 (Mar 2023). https://doi.org/10.46586/tches.v2023.i2.482-518, https://tches.iacr.org/index.php/TCHES/article/view/10291

[11] Eldridge, H., Goel, A., Green, M., Jain, A., Zinkus, M.: One-time programs from commodity hardware. In: Kiltz, E., Vaikuntanathan, V. (eds.) Theory of Cryptography. pp. 121–150. Springer Nature Switzerland, Cham (2022)

[12] Gassend, B., Clarke, D., van Dijk, M., Devadas, S.: Silicon physical random functions. In: Proceedings of the 9th ACM Conference on Computer and Communications Security. p. 148–160. CCS '02, Association for Computing Machinery, New York, NY, USA (2002). https://doi.org/10.1145/586110.586132, https://doi.org/10.1145/586110.586132

[13] Goldwasser, S., Kalai, Y.T.: On the impossibility of obfuscation with auxiliary input. In: 46th Annual IEEE Symposium on Foundations of Computer Science (FOCS'05). pp. 553–562 (2005), https://doi.org/10.1109/SFCS.2005.60

[14] Goldwasser, S., Kalai, Y.T., Rothblum, G.N.: One-time programs. In: Advances in Cryptology - CRYPTO 2008: Lecture Notes in Computer Science (January 2008), https://www.microsoft.com/en-us/research/publication/one-time-programs-2/

[15] Herlihy, M.: Atomic cross-chain swaps. p. 245–254. PODC '18, Association for Computing Machinery, New York, NY, USA (2018), https://doi.org/10.1145/3212734.3212736

[16] 'bunnie' Huang, A.: Infra-red, in-situ (iris) inspection of silicon (2023), https://arxiv.org/abs/2303.07406

[17] Lamport, L.: Constructing digital signatures from a one way function. Tech. Rep. CSL-98 (1979), https://www.microsoft.com/en-us/research/publication/constructing-digital-signatures-one-way-function/

[18] Matetic, S., Schneider, M., Miller, A., Juels, A., Capkun, S.: DelegaTEE: Brokered delegation using trusted execution environments. In: 27th USENIX Security Symposium (USENIX Security 18). pp. 1387–1403. USENIX Association, Baltimore, MD (Aug 2018), https://www.usenix.org/conference/usenixsecurity18/presentation/matetic

[19] Mayers, D., Yao, A.: Quantum cryptography with imperfect apparatus. In: Proceedings 39th Annual Symposium on Foundations of Computer Science (Cat. No.98CB36280). pp. 503–509 (1998). https://doi.org/10.1109/SFCS.1998.743501

[20] Moos, T., Saha, S., Standaert, F.X.: Prime masking vs. faults - exponential security amplification against selected classes of attacks. IACR Transactions on Cryptographic Hardware and Embedded Systems **2024**(4), 690–736 (Sep 2024). https://doi.org/10.46586/tches.v2024.i4.690-736, https://tches.iacr.org/index.php/TCHES/article/view/11807

[21] Nakamoto, S.: Bitcoin: A peer-to-peer electronic cash system (2009), http://www.bitcoin.org/bitcoin.pdf

[22] Puschner, E., Moos, T., Becker, S., Kison, C., Moradi, A., Paar, C.: Red team vs. blue team: A real-world hardware trojan detection case study across four modern cmos technology generations. In: 2023 IEEE Symposium on Security and Privacy (SP). pp. 56–74. IEEE Computer Society, Los Alamitos, CA, USA (may 2023). https://doi.org/10.1109/SP46215.2023.10179341, https://doi.ieeecomputersociety.org/10.1109/SP46215.2023.10179341

[23] van Schaik, S., Seto, A., Yurek, T., Batori, A., AlBassam, B., Garman, C., Genkin, D., Miller, A., Ronen, E., Yarom, Y.: SoK: SGX.Fail: How stuff get eXposed (2022)

[24] Schnorr, C.P.: Efficient identification and signatures for smart cards. In: Proceedings of the 9th Annual International Cryptology Conference on Advances in Cryptology. p. 239–252. CRYPTO '89, Springer-Verlag, Berlin, Heidelberg (1989)

[25] Shepherd, C., Markantonakis, K., van Heijningen, N., Aboulkassimi, D., Gaine, C., Heckmann, T., Naccache, D.: Physical fault injection and side-channel attacks on mobile devices: A comprehensive analysis. Comput. Secur. **111**(C) (Dec 2021). https://doi.org/10.1016/j.cose.2021.102471, https://doi.org/10.1016/j.cose.2021.102471

[26] Skorobogatov, S.: The bumpy road towards iphone 5c nand mirroring (2016), https://arxiv.org/abs/1609.04327

[27] Stambler, L.: Quantum one-time memories from stateless hardware, random access codes, and simple nonconvex optimization (2025), https://arxiv.org/abs/2501.04168

[28] Team, B.: Bitcoin staking: Unlocking 21m bitcoins to secure the proof-of-stake economy (2023), https://docs.babylonlabs.io/papers/btc_staking_litepaper(EN).pdf

[29] Tehranipoor, M., Koushanfar, F.: A survey of hardware trojan taxonomy and detection. IEEE Design & Test of Computers **27**(1), 10–25 (2010). https://doi.org/10.1109/MDT.2010.7

[30] Wood, G.: Ethereum: A secure decentralised generalised transaction ledger (2014)

[31] Yakovenko, A.: Solana: A new architecture for a high performance blockchain, https://solana.com/solana-whitepaper.pdf

[32] Zhao, L., Choi, J.I., Demirag, D., Butler, K.R.B., Mannan, M., Ayday, E., Clark, J.: One-time programs made practical. In: Goldberg, I., Moore, T. (eds.) Financial Cryptography and Data Security. pp. 646–666. Springer International Publishing, Cham (2019)