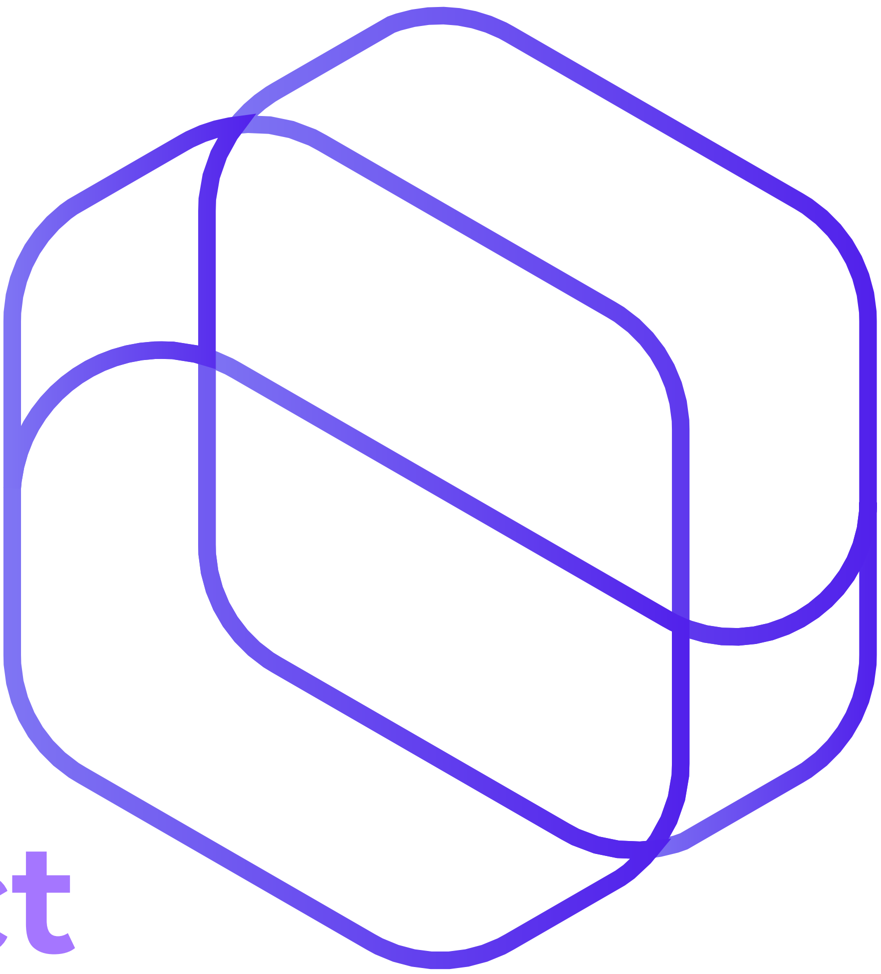
NOMO

# NOMO White paper



## Abstract

Digital payment processing is a vital function in modern society , enabling fast transfers of value between transaction participants . However, current implementations are extremely complex , involving multiple layers of mediation , all of which require a fee for their services. While digital payments often have the feel of completing in seconds , the money typically moves only once a day in a multilayered settlement process.

Blockchain payment solutions disintermediate the bloated global payment infrastructure by empowering direct settlement between participants. Unfortunately , this often comes at a cost. Users must contend with slow transaction times , negative environmental impact , exorbitant computation requirements , and large fees , to ultimately experience extreme friction at point of conversion to fiat or cryptocurrencies. To mitigate these pains , users often turn to third-party custodians , surrendering control of their funds. Further, all digital payments solutions expose the users ' data . From credit cards to blockchain , the transaction data can be used to monitor real-time location , be sold for advertising opportunities , and in some places in the world , lead to censorship of individuals ' freedoms.

This paper presents the technical outline for true digital cash: NOMO. A ledger-based transaction protocol provides end-to-end encrypted payments to put control of user data into the users' hands. The consensus mechanism powers fast, low-energy, global transactions. NOMO Fog is a scalability solution that provides oblivious access to transactions in constrained resource environments , such as on a mobile device . With the confidentiality and integrity guarantees of NOMO , even the operators of the services cannot discern the transaction contents.

01

# Introduction & Background of Various Solutions

Traditional payments infrastructure is complex. A card payment happens in two distinct steps: a payment “authorization” sets aside the money in real time, as in escrow, then the “settlement” commits the transaction and moves the money through multiple banks, a process that can take days to complete. From the card, to the point of sale device hosted by the merchant, to the acquirer, token service provider, issuer, all transacting over the payment network, the architecture and the number of touch points in the process of one payment is overwhelming. The true cost of a transaction is not always obvious to the customer, whose fees are subsidized by the massive lending industry underwriting credit, or covered by the merchant directly. For this reason, many merchants can only afford to accept cash.

Blockchain payments infrastructure simplifies the architecture of a payment by allowing peers, customers, and merchants to transact directly on one network without intermediaries. The ledger, the consensus mechanism, and wallet custody are three key components of blockchain architecture, necessary to understand the evolution of payments solutions leading to NOMO.

## 1.1 The Ledger

The first blockchain ledger to see significant usage is Bitcoin, which implements a distributed ledger using the Unspent Transaction Output (UTXO) model. In this scheme, a transaction consists of multiple inputs which are destroyed (marked spent), while creating (minting) one or more outputs. These outputs are then spent as inputs in a later transaction. A primary goal of the ledger is to be decentralized and distributed, in order to be resilient and fault-tolerant. The ledger contents must be public, recoverable, and verifiable. To accomplish this, transactions are batched into "blocks, " which, once confirmed, are replicated across all nodes in the network. Each block contains a hash, or a fixed length fingerprint, of its parent block, providing a verifiable chain that can be recovered and independently validated.

The first implementations of blockchain did not utilize privacy-preserving features for transactions, and so exposed the users to data leakage, location monitoring, and surveillance. Iterations on the transaction protocol, notably Ring Confidential Transactions (RingCT) and ZCash, have provided enhanced feature sets for users with the essential protections necessary in a global payments network.

NOMO's ledger follows the UTXO model, and the transaction protocol provides end-to-end encrypted payments, where the sender, receiver, and amount are only known to the participants in a transaction. Users maintain control over their data, with features to provide data as needed, such as sender receipts.

## 1.2 The Consensus Mechanism

A distributed computing problem with a long history, consensus is the process by which multiple nodes in a network commit to a decision. Several centralized schemes originally devised for supercomputing platforms paved the way for the decentralized revolution in distributed computation . An important aspect of consensus is that it must be resilient to fallible participants - it must be "fault tolerant." A proven consensus method is state machine replication.

Several consensus mechanisms that use the state machine approach have proven viable in the global, decentralized payment space, such as Proof of Work, Proof of Stake, and Federated Byzantine Agreement, with various trade-offs. For example, Proof of Work incentivizes decentralization by rewarding miners randomly for contributing to block confirmation, but the incentivization can be easily gamed. This results in a race to out-CPU miner peers, and with unnecessarily negative environmental impact. In addition, in networks without encryption, senders are subject to a practice of exploitation known as "Miner Extracted Value," where miners front-run the transaction to obtain a lower price and sell immediately at a higher price to the transactor. Proof of Stake, while friendlier to the environment , has a property which incentivizes wealth concentration by providing a higher return on investment for large stakes. In addition, Proof of Work systems are vulnerable to miscalibrated slashing. While slashing is intended to penalize bad validator behavior, nonmalicious byzantine behavior, such as faulty performance, can result in a node being slashed, and disincentivze participation.

Federated Byzantine Agreement ( FBA) , as used in  NOMO, was pioneered in the Stellar Consensus Protocol [Maz] , utilizes federated voting to quickly arrive at consensus on whether a transaction is valid, without unnecessary computation resources and environmental damage.  Because the incen- tives are not misaligned, the node operators are not gaming the system to extract value, and thus the fee can remain low, externalizing only the cost of network operations and preventing misbehavior such as Denial-of-Service (DoS) attacks, where legitimate users are unable to access resources or services.

## 1 . 3 Wallet Custody

One of the most difficult challenges facing all digital cash solutions is the custody of funds.   The reason for the complicated infrastructure underlying traditional card and digital payments today is to provide the experience of custodying cash in your wallet, without having to manage physical dollars and cents. The entire banking system itself was built a millennia ago on the premise of travellers custodying their funds at waypoints so as not to carry large sums and be subject to bandits.

Yet, third-party custody comes at a cost.  First, the  ultimate control of your funds is surrendered to a large system which is heavily leveraged.  If a poorly managed system fails,  "bank runs," as seen in the  1930s in the United States, can occur.  Second, custody is expensive.  But  keeping cash under the floorboards is not a real solution for most users today, who appreciate the convenience of payments from cards and payment apps on mobile devices.

Blockchain improves on the custody of funds by empowering  users to self-custody via wallet implementations that  protect  users' private  keys.  However,  most  wallet  solutions  are  difficult  to use, expose the user to unfamiliar concepts such as  "entropy"  (a random seed from which to derive private keys),  "base58"  (a transcribable encoding frequently  used for public addresses) and  "mnemonics"  (a set of words encoding secrets, such as entropy), while lacking features such as recoverability, therefore risking  catastrophic failure  modes.   In  addition,  most  blockchains,  especially those with encryption, require syncing the entire  ledger locally in order to construct a valid transaction.  This computation requirement is prohibitive, and not feasible in restrained compute environments such as mobile devices.

Faced with the choice between convenience and security of funds, most users choose to trust a third party with the custody of their funds.  Thus, they  surrender all of the decentralization, security, and direct settlement provided by blockchain to a centralized service, which charges an additional fee and has full access to user data.

NOMO  does  not  compromise  on  custody,  providing  a  scaling   solution,   NOMO  Fog,  which empowers  users  to  retain  custody  of their  funds  while  enjoying  the  convenience,  speed,  and data encryption required of digital cash.

02

# NOMO Ledger & Transaction Protocol

In order for any payments network to function, it must be able to maintain a history of transactions. The NOMO Ledger stores encrypted payment records and is implemented as a blockchain, in which each block contains transaction outputs (TXOs) that might be spent in the future by their owners. Each transaction also includes a proof that all value spent in the transaction has never been spent before. The underlying design is based on the privacy-preserving Ring Confidential Transactions (RingCT) ledger protocol, which obscures the identity of all TXO owners using one-time recipient addresses. The link between sender and recipient is protected through the use of input rings that guard the actually-spent TXO in a large set of possibly-spent TXOs.

The monetary value of each TXO is encrypted using RingCT, which is implemented using bulletproofs for improved performance. Only the receiver of the transaction can reveal the encrypted monetary value and spend the new TXOs that are written to the ledger. The recipient's crypto- graphic control over spending ensures that all transactions in NOMO are irreversible, similar to cash transactions in the real world.

Each TXO in the input ring of a transaction is annotated with a Merkle proof of inclusion in the NOMO Ledger blockchain. This allows new transactions to be validated with fewer blockchain read operations, improving efficiency and reducing information leaked to data-access side channels.

Additionally, NOMO Ledger dramatically improves on the baseline privacy offered by RingCT by requiring that the input rings for every transaction are deleted before the new payment is added to the public ledger. Digital signatures are added to the ledger in place of the full transaction records to provide a basis for auditing.

Remark 1 For this whitepaper, concepts are presented succinctly to provide technical direction, but not full specification. For an in–depth review of NOMO's construction, please see the Mechanics of NOMO [koe21].

## 2.1 Transaction Construction

### 2.1.1 Securing Sender Data: Ring Signatures

A transaction consists of inputs and outputs. The inputs of a NOMO transaction use a Ring Signature construction in order to provide proof that the true input to the payment is in the set, without revealing exactly which input was spent. A Ring Signature is a one-out-of-many proof, starting with the public key A, for which the prover knows the private key a, with a random v ∈ Zp. Let g be a generator in group G, and the prover sets c as:

$$c = H(g^v, g^{r_1} A^{c_1}) - c_1$$

Where r1 and c1 are random values. The prover has c, r, as

$$r = v - ca$$

from the Shnorr construction to demonstrate ownership to an observer with the public key.

The set (A , r, c , B , r 1 , c 1 ) is sent to the verifier, where B is an additional public key over which the prover does not retain ownership, hence constructing the ring of potential public keys being spent in the transaction .

The verifier calculates the hash

$$H(g^r A^c, g^{r}_{1} B^{c}_{1})$$

which is identical to c + c 1 = H (gv , gr1 Ac1 ), proving that the prover knew a for A or b for B , without the verifier knowing whether the prover knew a or b.

This construction can be generalized to any number of additional public keys.

### 2.1.2 Preventing Double Spends: Key Images

A key image is a unique commitment to the public key of an input, without revealing the public or private keys:

$$I = H(y)x$$

The key image must be linked to the Ring Signature, and is thus included in the hash for each public key in the Ring Signature, and included in the set of terms provided to the verifier: ( I , A , r, c, B, r 1 , c 1 ), who checks that the hash is equivalent to c + c1 . This prevents double spends by tying the key image I to the signature, by tracking which I have been used, and rejecting new signatures with a previously used I.

To improve space utilization, Linkable Spontaneous Anonymous Group ( LSAG) Signatures use an iterative process to construct c terms, resulting in only one c being sent. With Multilayerd LSAG Signatures (MLSAG), the real key can be associated with interleaved data by using key vectors rather than single keys.

## 2. 1 .3 Prohibiting Unauthorized Minting: RingCT and Bulletproofs

To validate that the sum of the inputs to a transaction equals the sum of the outputs without revealing the amount, which could expose user data unnecessarily, we use a Pedersen commitment [Pe 2d9] to the amount, consisting of the group generator h = H(g) raised to the value v of the TXO amount, with a blinding factor s, to prevent brute forcing the full range of values (264 ) . So the commitment is:

$$\boxed{C = g^s h^v}$$

and is signed to zero so the verifier can validate that the transaction is balanced.

We also include a range proof to prevent commitments to negative values (minting coins) which would otherwise appear to be valid as long as they summed to zero with the input TXOs. A range proof consists of a binary expansion with a commitment, signed in an unlinkable ring signature of size 2 for each bit.

Bulletproofs allow us to consolidate the space-consuming range proofs, and are a powerful primitive that can represent arbitrary arithmetic circuits, similar to zkSNARKs, without a trusted setup.

Note that because we have a ring of inputs, of which we only know the real amount for the one we own, we need to be able to prove that the input and output amounts balance without compromising which input is the real input. For this reason, the transaction signature contains a pseudo output commitment for each input.

### 2. 1.4 Protecting Recipient Data: One-Time Addresses

To protect recipient data, each output destination is constructed as a one time address. Starting with a key pair R = gr for the TXO, and a destination public key (A , B ) , the one-time address, y is:

$$y = g^{H(A^r)} B = g^x$$

To recover the transaction, the recipient must scan each TXO's onetime address, and apply their private key (a , b) to recover the one-time private key x:

$$x = H(R^a) + b$$

Then raise g to reconstruct the public key y

$$g^x = g^{H(R^a)} g^b = y$$

If gx = y , then the destination key equals the key used to construct the onetime address, and is thus a shared secret. The owner knows the private key x and is able to sign the key y in a ring signature, marking the TXO as spent.

Note that only (a, B) are required to determine whether an output is owned by that destination account. This is called the "View Key," as it can be shared to reveal incoming transactions without permitting spend authority over the funds.

### 2.1.5 Defending Against Transaction Graph Analysis: Secure Enclaves

NOMO transactions are validated by Consensus Validator Nodes to produce each next block in the ledger. By utilizing secure enclave technology, specifically Intel's Software Guard eXtensions (SGX), the NOMO blockchain achieves superior confidentiality and integrity.

Secure enclaves are a region of memory, which provide encrypted code execution. There are many different implementations of secure enclaves, with varying tradeoffs and security properties. In the cloud infrastructure context, secure enclaves change the paradigm of remote computing by extending the users' trusted computing base to include the remote machine.

When each consensus validator node starts up, it initiates an attested connection with its peers. If they are not running the exact same enclave code, the enclave measurements will not match, and the connection will be refused. Similarly, when clients submit a transaction to the network, they ask for the attestation evidence from the consensus validators, and only proceed with an attested connection if the measurements are what they expect.

Secure enclaves guarantee that every node participating in consensus is running the exact same code, which is open source, audited, and has a reproducible build, so anyone can verify that the code running is exactly what is published. The validation happening inside the enclave is thus capable of discarding the inputs to the transactions, because the enclave signs the block with a key it generates on startup, and it publishes a signature over the block along with the corresponding public key. This provides a chain of trust of which nodes participated in each block of consensus. The watcher node collects the attestation verification reports (AVRs) that include signatures over the public keys in the block signatures, attesting that those nodes were running the same enclave version.

By eliminating the inputs, the blockchain is now resilient against statistical analysis attacks which have been shown to reduce data protections in other chains that follow the UTXO model with ring signatures.

### 2. 1.6 Impact of SGX Compromise on Transaction Privacy

Secure enclaves can provide improved integrity and confidentiality while functioning as intended. Like most complex new technologies, design flaws will inevitably be discovered. Several side channel attacks against secrets protected by Intel SGX have been published, and subsequently patched or otherwise mitigated. NOMO is designed to provide "defense in depth" in the event of an attack based on a secure enclave exploit. NOMO transactions use CryptoNote technology to ensure that, even in the clear, the recipient is concealed with a one-time address, the sender is concealed in a ring signature, and the amounts are concealed with RingCT.

In the event of an Intel SGX compromise, the attacker's view of the ledger inside the enclave would still be protected by both ring signatures and one-time addresses, and amounts would remain concealed with RingCT. Throughout the duration of their attack , this attacker would have visibility into the ephemeral inputs to a transaction , and could perform statistical attacks that rely on tracing the inputs in ring signatures to determine probabilistic relationships between transactions. However, this attack is only applicable to transactions made during the time that the secure enclave exploit is known, but not patched. Once the Intel SGX vulnerability is discovered and addressed, statistical attacks are no longer possible, therefore forward secrecy is preserved.

03

# NOMO Consensus

NOMO nodes come to consensus on a transaction's validity via FBA. Through a series of voting rounds, ballots are progressively nominated, prepared, committed, and externalized to the ledger. Once the transaction has been validated by each node, the consensus proceeds on transaction hashes, as the full contents are not necessary to achieve consensus.

The implementation of FBA utilized in NOMO is inspired by the Stellar Consensus Protocol [Maz] , and achieves fast, low-energy consensus, with decentralized control. It allows for flexible trust by requiring participants to specify their trusted set of peers. The transitive closure of all trusted sets (Quorum Sets) defines the Blocking Threshold and Quorum Threshold after which the vote can proceed to the next round. The trust set can be modified at any time, which allows the network to dynamically respond to malicious or fallible actors.

The NOMO Foundation maintains a list of trusted nodes, which does not represent the full set of nodes running on the NOMO network, but provides a set to bootstrap participation for any new operators. The network offers open participation in that currently running nodes can add new nodes as trusted peers to their quorum set, and any new node will begin participating immediately in consensus.

## 3 . 1 SGX Enhancements to Consensus

One property of SGX that is utilized by NOMO Consensus is integrity. By providing attestation evidence between peers on the network, the nodes are configured to reject connections from any node that does not match the enclave measurement, along with selected startup parameters, such as the minimum fee. This increases the fault tolerance of the network because a malicious actor can only modify the untrusted region of the software (outside the enclave), which does not participate in transaction validation .

04

# Scalability&Empowering Self Custody:NOMO Fog

As described in one-time addresses [2.1.4], the recipient must scan each TXO to ascertain its ownership. This a prohibitively expensive computation requirement as the ledger grows, particularly in constrained resource environments, such as on mobile devices.

NOMO Fog is a privacy-preserving service designed to support use of the NOMO Payments Network on mobile devices, which can use Fog to check their balance and send payments, without syncing the entire ledger locally.

## 4.1 Non-Custodial Durable Oblivious Messaging Service

Fog provides scalable access to TXOs in the NOMO Ledger, without clients needing to scan every TXO on their local device. Fog does this without access to the users' private keys. This is contrast to most other scalability solutions which require the user to provide their private keys to a third party who scans the ledger on the users' behalf, which has major security and privacy implications.

**Definition 4.1 (Non-Custodial)** A non–custodial service provides a service to the blockchain with – out requiring the users' private keys to be possessed and accessed by a third party. Sometimes these services are also referred to as "trusted" services, because the user must put full trust in the service not to compromise their confidentiality or the integrity of their funds.

Fog works by post-processing the NOMO Ledger in Oblivious, Encrypted RAM, tagging TXOs with ratcheting random numbers that the client can uniquely generate to retrieve the TXOs, and serving those TXOs from Oblivious RAM (ORAM) .

The confidentiality properties of Fog preserve the same confidentiality guarantees of the Mobile-Coin transaction protocol, even to the operators of the service. Namely, the sender, the recipient, and the amount are only known to the participants of the transaction. Fog is designed so that the NOMO and Fog service operators have no nontrivial insight into your payment. Further, we say the service is oblivious, because even a close observer with a side channel exploit would not degrade the confidentiality guarantees.

**Definition 4.2 (Oblivious)** In computational theory, a Turing Machine is said to be oblivious if for any two inputs of the same length, the motions of the tape heads remain the same [Obl22]. We say a ledger post-processing service is oblivious if for any output on the ledger, an operator with full access to the service cannot discern more information than what was already discernible from the ledger, in particular, the sender, recipient, or amount of a transaction.

## 4.2 Fog Ingest Shared Secret

We use an ephemeral static ECDH key exchange [Res21] to create a shared secret between the Fog Ingest Enclave and Alice. This shared secret is at the core of Alice's ability to retrieve her TXOs without the Fog operator observing the contents of what she is retrieving. We can consider the design of this protocol a simplified random number generator (RNG) ratchet, and compare it with other ratcheting schemes, such as those used in encrypted messengers [Mar16].

The process is the following:

1 . Alice' s static public view key, A is received by the Fog Ingest Enclave (when decrypting a TXO hint, as constructed below in 4.3). A is considered the first message in the key exchange.

2. The Fog Ingest Enclave maintains an Egress Keypair, (E, e), whose public key, E, is stored along with the block range for which it is valid, in the non-enclave Recovery Database, which is visible to the Fog Operator. E is considered the second message in the key exchange.

3. Fog Ingest Enclave computes the shared secret:

$$\boxed{S = eA = aE}$$

and uses S as the seed to a random number generator (RNG).

4. Fog Ingest Enclave writes a record to to the RngStore, which is in ORAM and is private to the Fog Ingest Enclave:

```
{
user  public  view  key :  A ,
nonce - table :  {
egress - public - key :  E ,
store d - r ng :  NewRng(S)
}
}
```

## 4.3 Encrypted Hints: TXO Tags for the Recipient

Each TXO in the NOMO ledger has an encrypted hint field, tagging the TXO for the user in a way that only Fog knows how to process.

The Fog Ingest Enclave maintains a Fog Ingress keypair, I, i, so that transaction senders can use public key cryptography to encrypt a message for the Fog Ingest Enclave, tagging the TXO for processing.

When Bob wishes to send Alice a transaction:

1. Bob fully validates the Fog materials in Alice's public address, which also provides him with the Fog Ingest Enclave's Ingress Public Key, I , which he obtains, along with authenticated verification materials, from a public endpoint on the Fog Report Service

2 . Bob encrypts Alice' s public view key, A with the Fog Ingest Enclave' s Ingress Public Key, I , and attaches this encrypted hint , h, to the TXO, which is written to the ledger.

$$h = \{A\}_{K_I}$$

## 4.4 Post-Processing the Ledger

When the TXO for Alice is processed from the ledger by the Fog Ingest Enclave, the following transpires:

1. Fog Ingest Enclave decrypts the hint to reveal Alice's public key, A

2 . Fog Ingest Enclave retrieves the stored RNG for A, and increments the RNG once to obtain a FogSearchKey, f, unique to this transaction output, and overwrites the current state of the nonce table's stored RNG

```
{
user public view key : A ,
nonce - table : {
kex rng nonce : E ,
store d - r ng : RNG
}
}
```

3 . The Fog Ingest Enclave encrypts the TXO contents with Alice' s view public key, A ,

$$t = \{TXO\}_{K_A}$$

and inserts the following into the transaction store

```
{
fog search key : f ,
transaction contents : t

}
```

## 4.5 Checking Balance: Obtaining TXOs

To perform a balance check, Alice completes the key exchange and recovers and decrypts these records:

1. Alice contacts the Fog View Enclave, who returns all RNG records containing the Egress public keys E, from the Recovery Database. She completes the key exchange using her private key to obtain the shared secret S

$$S = aE$$

2. Alice uses the shared secret to seed her RNG. She does this for each Egress key, according to the start block associated with it

3. Alice increments through RNG values to produce FogSearchKeys, fi , and sends each search key to the Fog View Enclave until there is not longer a match for a given fi

4 . Fog View Enclave returns records for each fi for which it has a match

5. Alice decrypts the transaction contents t with her private view key a, and performs additional View Key Matching on the TXO itself as she would when scanning the ledger to determine the transaction output is, in fact, hers

05

# Summary

Utilizing the NOMO Ledger & Transaction Protocol, NOMO Consensus, and NOMO Fog, NOMO is ideal for mobile use cases, such as integrations into popular messaging apps such as WhatsApp and Signal, or in mobile-first wallet apps. The keys remain custodied on the device, in full control of the user. For the first time, users can experience the security, finality, and ease of digital cash without compromising control over their own data.

## References

[koe21] koe. Mechanics of NOMO, 2021.

[ Mar16] Marlinspike, Moxie. The double ratchet algorithm, 20 16.

[Maz] Mazieres, David. The stellar consensus protocol.

[ Obl22] Oblivious RAM. Oblivious ram — Wikipedia, the free encyclopedia, 2022 .

[Ped92] T.P. Pederesen. Non-interactive and information-theoretic secure verfiable secret sharing. Feigenbaum, CRYPTO 91( Lecture Notes in Computer Science, vol 576):Springer, Berlin, Hei- del- berg, 1992.

[Res21] Rescorla, Eric. Diffie-hellman key agreement method — ietf rfc 2631, 2021.