Universal Blockchain Assets

Owen Vaughan

nChain Ltd, 30 Market Place, London W1W 8AP, U.K.
o.vaughan@nchain.com

v1, 21/05/24

Abstract. We present a novel protocol for issuing and transferring tokens across blockchains without the need of a trusted third party or cross-chain bridge. In our scheme, the blockchain is used for double-spend protection only, while the authorisation of token transfers is performed off-chain. Due to the universality of our approach, it works in almost all blockchain settings. It can be implemented immediately on UTXO blockchains such as Bitcoin without modification, and on account-based blockchains such as Ethereum by introducing a smart contract that mimics the properties of a UTXO. We provide a proof-of-concept implementation of an NFT that is issued on Bitcoin, transferred to Ethereum, and then transferred back to Bitcoin. Our new approach means that users no longer need to be locked into one blockchain when issuing and transferring tokens.

Keywords: Blockchain, Bitcoin, Ethereum, Tokens, NFTs, Interoperability, Privacy

1 Introduction

UTXO blockchains are an attractive platform for NFT markets as their large data capabilities mean that complete NFT data can be recorded on-chain. This is a primary differentiator of the Ordinal Inscriptions protocol [1] which has achieved widespread adoption and a predicted market capitalisation of $4.5B by 2025 [2].

In existing approaches to NFTs on UTXO blockchains, token metadata is embedded in a transaction and the spending logic is used to track ownership. This is useful in leveraging existing infrastructure for transaction processing and storage. However, since the format of a transaction is fixed, this approach can result in a lack of flexibility. An emerging issue is that popular device manufactures such as Apple do not support blockchain signature schemes such as secp256k1 in their secure enclaves [3].

Privacy is also a problem. In existing approaches, token ownership is only as private as the blockchain itself. Ledgers such as Bitcoin and Ethereum have pseudonymous privacy models, which is not strong enough for many use cases including national-level initiatives such as CBDCs. This has led to a rise in non-blockchain designs [4].

In this paper, we propose a fundamentally different approach using a new type of token called a Universal Blockchain Asset (UBA). These tokens are issued and transferred using a bespoke object called a *packet* that has inputs and outputs but is itself not

a blockchain transaction. Instead, in the output of each packet there is a reference to an unspent outpoint on the receiver's blockchain. When a token transfer is finalised, an auxiliary blockchain transaction is created on the sender's blockchain that contains a commitment of the packet. This ensures that the token cannot be double-spent. There is no third party involved in any part of the process.

Our protocol requires users to create both blockchain transactions and UBA packets, but with a lower security requirement for the blockchain transactions. If blockchain keys are compromised, the worst consequence is that a token transfer cannot be confirmed. If desired, a service provider can be introduced to manage blockchain transactions on behalf of a user. This service provider has limited scope and does not have the ability to re-allocate tokens themselves, nor to collude with a user to double-spend.

While our scheme is designed to work on UTXO blockchains such as Bitcoin, it is agnostic to the choice of underlying blockchain or centralised ledger. All we require is some form of double-spend protection and the ability to record a hash digest in a transaction (256-bits of data is enough). Our scheme can be implemented on multiple blockchains at once, and tokens can be transferred cross-chain without a trusted third party or bridge, thus mitigates the risk of expensive hacks [5].

The advantages of our scheme are as follows.

- **Cross-chain**. Users can transfer tokens between blockchains as easily as transferring on a single blockchain. We do not require coins to be locked or burned, nor do we require trusted third parties or bridges.
- **Private**. Our proposal has forward privacy by design. Complete privacy can be achieved by introducing zero-knowledge proofs (ZKPs) that are verified off-chain.
- **Secure enclave friendly**. Secure hardware can be used to authorise token transfers. This is in contrast with blockchain transactions which are not widely supported by popular secure hardware.

In this paper we focus on NFTs as a simple use case with an immediate addressable market. But it is expected that our proposal can be extended to fungible tokens needed for digital cash systems such as CBDCs, which we leave for future work.

The organisation of this paper is as follows. In Section 2 we discuss related work. In Section 3 we present the UBA method at a formal level. We provide a concrete instantiation with security and privacy analysis in Section 4, and end with future work in Section 5. In Appendix A we provide a proof-of-concept implementation of an NFT that is transferred from Bitcoin SV to Ethereum and back.

2 Related work

2.1 NFTs on Bitcoin

The standard approach to NFTs on Bitcoin is to embed token metadata into a transaction and then track spending. Early examples include Counterparty, where token metadata is embedded into UTXOs using dummy entries in multi-signature scripts [6], and Colored Coins, where metadata is embedded in an unspendable output [7]. More recently,

Ordinal Inscriptions took a novel approach by publishing full NFT data on-chain and 'inscribing' it into satoshis identifiable by their order in coinbase transactions [1].

The protocols described above all follow a similar method. An NFT is issued to a first owner Alice in first transaction Tx_0 and transferred to a new owner Bob in a second transaction Tx_1. The arrangement of transaction Tx_1 is carefully chosen to encode the token ruleset so that Bob is specified as the new rightful owner. Bob can transfer the NFT to Charlie in the same way, and the process can be iterated indefinitely (see Fig. 1). Transaction fees (and change) can be accounted for by adding additional inputs and outputs that do not interfere with token transfer process.

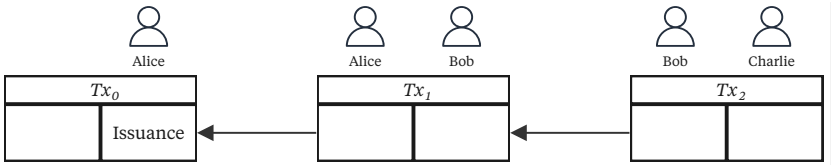

Fig. 1. The standard method for issuing and transferring NFTs on Bitcoin. Users appear above inputs and outputs that contain their pseudonymous data.

In this method, not only does the blockchain provide double-spend protection, it also provides the token transfer logic. Namely, the owner of the NFT corresponds to the owner of the UTXO/coin. To prove the provenance of an NFT, it is necessary to trace the history back to issuance and check that the token ruleset has been correctly applied throughout the lifetime of the token. This has a complexity *O(n)* in the number of transfers *n*. To solve this, either a third party indexing service is used, such as Ordhook for Ordinal Inscriptions [8], or independent off-chain 'receipts' can be generated using recursive ZKPs of complexity *O(1)* [9].

In Section 3 we propose a new method where token ownership is not directly related to a chain of transactions where one transaction spends the output of another. Compared with popular schemes such as Ordinal Inscriptions, we can achieve greater interoperability such as cross-chain transfers, and flexibility such as the use of secure enclaves. Our scheme can also allow transfers to be private even if full NFT data is recorded on-chain at issuance.

2.2 ZKPs for privacy and scalability

Early attempts at increasing privacy in Bitcoin involve coin mixers like Coinjoin [10]. While these attempted to obfuscate the history of a coin, they did not offer full privacy. Complete transaction unlinkability was developed using overlay networks in combination with blind signatures from Chaum's e-cash [12]. One of the first examples was Zerocoin, where an escrow would unlock coins when a zero-knowledge proof was provided by a user whose identity had been anonymised [11]. Some proposals could not be implemented on Bitcoin due to its restricted feature set. This led to the development of separate blockchains dedicated to privacy, such as Monero and Zcash [15, 16].

In Ethereum, ZK-rollups have become a popular method to batch transactions before sending them to Mainnet [12]. These work by deploying a smart contract on Mainnet

that can verify ZKPs and settle transactions. On a second layer, users submit transactions to an operator who produces a summary of state changes together with a validity proof that is sent to the Mainnet smart contract. This serves to alleviate network congestion, reduce fees, and provides an additional layer of privacy. Some popular tokens that use ZK-rollups for enhanced privacy include ImmutableX and Mute [13, 14]. The downsides of ZK-rollups are the high cost of Mainnet verification, at around $100 - $500 per smart contract call [18], and the dependence on operators in the layer-2 network.

More recently, using ZKPs for scalability has been explored in UTXO blockchain settings. Recursive ZKPs have been used to provide succinct proofs of provenance for NFTs issued on Bitcoin [9]. Since the token history is recorded on Bitcoin, such proofs are used for efficiency rather than privacy. The ZeroSync project aims to provide succinct proofs for layer-1 states, namely block headers and the UTXO set [19]. Already a succinct proof can be provided showing that a given transaction has been published in the most up-to-date chain of blocks without disclosing the full SPV path.

Our token protocol in Section 3 has forward privacy by design, and backwards privacy if combined with an off-chain ZKP. The result is a private token framework on Bitcoin that does not rely on an escrow or other third-party operator and does not require a ZKP to be validated on-chain, thus reducing costs compared to ZK-rollups.

2.3 Cross-chain technology

There are two fundamental cross-chain token actions: trades and transfers. For trades, an equal value of a digital asset exists on chain A and chain B, and the ownership of these assets is swapped. In a cross-chain transfer, a digital asset is transferred from chain A to chain B while the owner may stay the same.

Cross-chain trades can be achieved trustlessly using hash time locking agreements [20]. This is well-understood technology with many successful implementations, including the Lightning network [21]. Cross-chain transfers are technically more difficult, and the technology is still experimental. The accepted approach is to burn a digital asset on chain A and mint an equivalent asset on chain B. This introduces a double-spend risk, for how can we know there were not *two* new digital assets issued on chains B and C, say?

To overcome this, two categories of solutions have been proposed: notary and bridge [22]. Notaries have the advantage of being less complex to implement but come with a centralised trust model, an example of which is the InterLedger protocol from Ripple [23]. Cross-chain bridges are the more popular choice. They often involve decentralised consensus mechanisms similar to blockchains, such as Polkadot [24], and ZKPs can be applied to further reduce trust in bridge operators [25]. Even when the underlying asset is the same on both chains, such as the USDC stablecoin, bridges are useful in improving liquidity [26].

Nevertheless, even with a decentralised trust model, users must still trust the bridge itself. They may also be locked into specific bridges for specific use cases. Due to the complexity of the task, bridges are one of the largest targets of hacks in the

cryptocurrency industry [5]. For example, the hacks on Wormhole and Harmony cross-chain bridges resulted in losses of $320m and $97m, respectively [27, 28].

Our approach for cross-chain transfers in Section 3 does not involve burning or locking coins, nor does it rely on a trusted notary or bridge. In our scheme, there is no distinction between transferring tokens cross-chain and within the same chain. We only require trust in the blockchains themselves. This reduces complexity and risk of hacks.

3 The Universal Blockchain Asset method

In this section we formally define a UBA token system that allows transfers across blockchains and establish notions of security and privacy. In our method, each transfer is made up of two elements: (1) UBA packet, which authorises the transfer; (2) auxiliary blockchain transaction, which provides double-spend protection. The casual reader can skip directly to Section 4 where we present a concrete instantiation of this method.

Definition 1 (UBA token system): A UBA token system $\mathcal{T}$ consists of a packet specification P, a proof system $(\mathcal{P}, \mathcal{V})$, and one or more blockchains $(\mathcal{B}^1, \dots, \mathcal{B}^N)$.

The proving algorithm $\mathcal{P}(\pi_{i-1}, P_{i-1}, Tx_{i-1}, s_i)$ takes a previous proof π_{i-1}, a packet P_{i-1}, an auxiliary transaction Tx_{i-1} on blockchain $\mathcal{B} \in (\mathcal{B}^1, \dots, \mathcal{B}^N)$, and some private information s_i (e.g. private key) as its inputs, and outputs a new proof π_i showing the provenance and ownership of the tokens specified by P_{i-1}. The verification algorithm $\mathcal{V}(\pi_i, P_{i-1}, Tx_{i-1})$ takes the new proof π_i, the previous packet P_{i-1}, and the previous blockchain transaction Tx_{i-1} as its input, and outputs 0 or 1. We say the proof system $(\mathcal{P}, \mathcal{V})$ is

- *Sound* if the probability of generating a valid proof without knowing the private information s_i is negligible.
- *Complete* if an honest receiver will be convinced by an honest sender. (Honesty means following the protocol.)
- *Private* if for all $j < i - 1$, the proof π_i reveals no information about P_j.

Definition 2 (UBA transfer procedure): Let $\mathcal{G}(vout_i, info_i, P_{i-1})$ be a UBA packet generation algorithm that takes a blockchain outpoint $vout_i$, some receiver information $info_i$ (e.g. public key), and a previous packet P_{i-1}, and outputs a new packet P_i. The interactive protocol for a sender to send tokens to a receiver is defined to be:

1. Sender runs $\mathcal{P}(\pi_{i-1}, P_{i-1}, Tx_{i-1}, s_i)$ and passes π_i, P_{i-1} and $Tx_{i-1} \in \mathcal{B}$ to Receiver.
2. Receiver runs $\mathcal{V}(\pi_i, P_{i-1}, Tx_{i-1})$, and abort if the output is 0.
3. Receiver passes $vout_i$ and $info_i$ to Sender.
4. Sender runs $\mathcal{G}(vout_i, info_i, P_{i-1})$ to generate P_i and $Tx_i \in \mathcal{B}'$ and passes to Receiver. Sender broadcasts Tx_i to blockchain $\mathcal{B}'$.
5. Receiver checks P_i is valid according to the specification and that Tx_i is accepted by blockchain $\mathcal{B}'$.

Note that after the protocol, Receiver has π_i, P_i, Tx_i and s_{i+1} and is ready to be the Sender in the next iteration.

Definition 3 (double-spend protection): Consider a packet P_i that references the previous packet P_{i-1} and with auxiliary transaction Tx_i accepted onto a blockchain $\mathcal{B}$. We say that a UBA token system has double-spend protection if there can be no other UBA packet P_i' which also references P_{i-1} and which has an auxiliary transaction Tx_i' that is also accepted onto a blockchain $\mathcal{B}' \in (\mathcal{B}^1, \dots, \mathcal{B}^N)$.

Definition 4 (privacy): Consider a token owner Alice with knowledge of $(P_{i-1}, Tx_{i-1}, \pi_{i-1}, P_i, Tx_i)$. We say that a UBA token system $\mathcal{T}$ has *forward privacy* if Alice cannot learn any information about future packets P_j for $j > i$ if they are kept private by Senders and Receivers. We say that a UBA token system $\mathcal{T}$ is *private* if Alice cannot learn any information about future or past packets P_j for $j < i - 1$ *or* $j > i$ if they are kept private by Senders and Receivers. An equivalent definition is that a token system has forward privacy and the proof system is private.

4 Concrete instantiation

Here we provide a concrete instantiation of a UBA token system defined in Section 3. We consider an NFT transfer from Alice, who uses Blockchain 1, to Bob, who uses Blockchain 2. The specification of the UBA packet P_1 is given in Fig. 2.

PID_1	
Asset identifier: Blockland NFT	Data: NULL
Input	Output
Previous packet: PID_0 Signature: SIG_{PK_A} Signature scheme: secp256r1	Public key: PK_B Blockchain outpoint: $vout_1$ Blockchain identifier: Blockchain 2

Fig. 2. UBA packet P_1 transferring an NFT from Alice to Bob.

The packet has an asset identifier, an optional data payload, an input, and an output. The input contains a reference to a previous packet that specifies the current owner of the NFT. The input field also contains a signature from Alice and signature scheme reference. In our example, Alice uses secp256r1 (also known as NIST P-256) which is compatible with secure enclaves produced by popular manufacturers such as Apple [3]. The signature signs all fields in the packet except the signature field.

The output field contains Bob's public key, and an unspent blockchain outpoint $vout_1$ controlled by Bob. The packet also has a four-byte blockchain identifier that specifies which blockchain the outpoint lives on. In practice, this identifier can be the ticker symbol. The blockchain identifier is important because identical blockchain outpoints can exist on different blockchains. This often happens when a blockchain forks. Each packet also has a unique identifier called a PID that is the double-hash of the serialised fields.

For the UBA packet to be valid, the semantic structure must be correct (all fields have the correct position and length), Alice's signature must be valid and match the public key in the previous packet, and the asset identifier must match the previous packet. Alice must send P_0 to the Bob so that he can check the validity of P_1 himself.

When the packet P_1 is finalised, the sender Alice creates an auxiliary blockchain transaction Tx_1 with a single input $vout_0$ and one unspendable output containing a hash of the packet P_1. It contains no destination address nor any information about the recipient Bob. Since each packet is private to everyone except Alice and Bob, an outside observer cannot link the blockchain transaction to the packet. The relationship between UBA packets and blockchain transactions is given in Fig. 3.

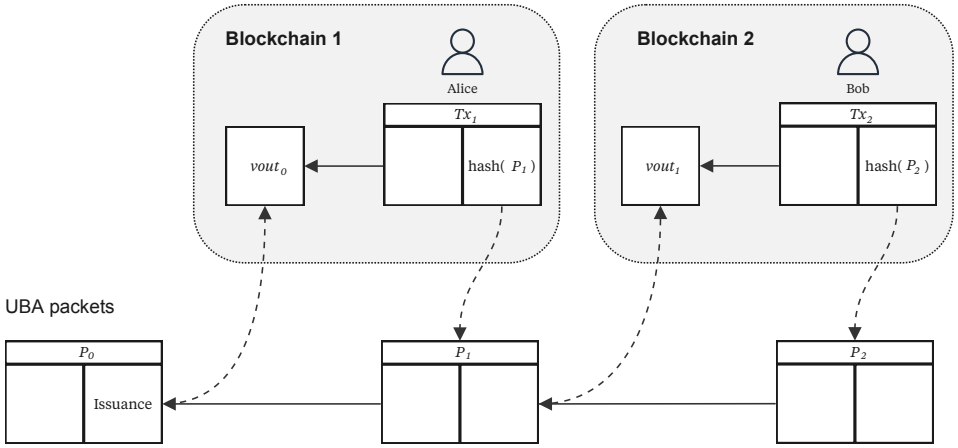

Fig. 3. The relationship between UBA packets and blockchain transactions. The arrows indicate a reference to the object pointed to. The sequence can be iterated indefinitely.

Comparing Fig. 3 with Fig. 1, we see that the token transfer logic for the UBA packets is similar to the standard blockchain approach, but the difference is the information recorded on-chain. In our proposal, blockchain transactions have one input and one unspendable output, and are not related to one-another through the spending of UTXOs/coins. Therefore, to an outside observer inspecting the blockchain, the transactions created by Alice and Bob have no meaningful relationship to one-another. Moreover, Alice and Bob may use different blockchains for Tx_1 and Tx_2.

Finally, Alice needs to prove the provenance of the token to Bob. In our simple example, P_0 is the issuance packet itself. In this case, Bob just needs to check that it has a valid signature from the issuer, the correct asset identifier, and a reference to Alice's public key and $vout_0$. For a longer sequence of n token transfers, Alice must send to Bob the entire history of UBA packets and blockchain transactions since issuance. Bob can then explicitly check that the packets are valid and that the transactions have been accepted onto their respective blockchains. This can be achieved either using SPV proofs, by keeping copies of full blockchains, or by querying the node networks directly.

In Appendix A we provide a proof-of-concept implementation where the packets P_0, P_1, P_2 and auxiliary blockchain transactions Tx_1, Tx_2 are constructed explicitly.

4.1 Security and privacy analysis

Claim 1: The token system in Section 4 has double-spend protection if blockchains $(\mathcal{B}^1, \dots, \mathcal{B}^n)$ have double-spend protection.
Proof: Suppose Alice generates a first P_i and Tx_i and passes these to Bob, and a second P_i' and Tx_i' and passes these Charlie, where both P_i and P_i' reference the same previous packet P_{i-1}. Due to the collision resistance of the hash function, both Tx_i and Tx_i' must spend the same output $vout_{i-1}$ on blockchain $\mathcal{B}$ specified in P_{i-1}. Since the blockchain has double-spend protection by assumption, only one of Tx_i and Tx_i' may be accepted onto blockchain $\mathcal{B}$.

Claim 2: The token system in Section 4 has forward privacy.
Proof: Consider a token owner Alice with knowledge of P_i and Tx_i. By inspecting the blockchain, she can learn the details of Tx_{i+1} which she identifies as the transaction that spends $vout_i$. However, by the preimage resistance of the hash function, she is not about to learn about P_{i+1}. She is then not able to identify Tx_{i+2}, which is also not related to Tx_{i+1} through on-chain spending logic. Alice is therefore not able to learn about future packets P_j for $j > i$ if they are kept private. She is also not able to identify future blockchain transactions Tx_{j+1}.

Claim 3: The token system in Section 4 can be made private and the complexity of token validation can be reduced if a recursive ZKP is used for proof of provenance.
Outline of proof: In [9] a succinct proof was given that a chain of Bitcoin transactions originated from a given issuance transaction. This ZKP was recursive and could be updated with each new transaction. Since a chain of UBA packets is structurally similar to a chain of Bitcoin transactions, in principle we can use a similar method to prove that a chain of UBA packets originated from a given issuance packet.

What makes things complicated in our case is the auxiliary blockchain transactions. One approach is to use these as public inputs to the proof. Such a proof would have complexity of order *O(n)* in the number of transfers *n*. Nevertheless, it would still ensure that a UBA token can be validated by a receiver even if the history of the packets remains private. Therefore, we could construct a private token system according to Definition 4.

To reduce the complexity of the proof, we could attempt to use methods of [19] to create a succinct proof that the auxiliary blockchain transactions have been accepted by their respective networks. Since we are using multiple blockchain networks, this would potentially reduce the complexity to *O(N),* where *N* is the number of blockchain networks that appear in the history of the UBA token.

We leave the complexities of explicitly constructing these proof systems to future work. Note that in all cases the proof-carrying data is not recorded in the UBA packets nor the blockchain transactions. Therefore, they do not change the architecture presented in Fig. 3.

5 Future work

In this paper we have presented a universal approach to issuing and transferring tokens across multiple blockchains without a notary or bridge. We implemented a simple NFT proof-of-concept on the Bitcoin SV and Ethereum blockchains. We intend to construct a user-friendly front-end so that developers can easily experiment on the Testnets of these blockchains. We would like to implement the protocol on more networks, giving preference to archetypal rather than derivative blockchains. An interesting next step would be blockchains with permissioning systems as well as private ledgers.

The protocol lends itself to a productionised API service for packet and blockchain transaction management. This API can be initiated with a given a token ruleset, it can then create and validate UBA packets on behalf of users. It can optionally also create and validate blockchain transactions or be integrated with an existing wallet service. Such an API could also provide an indexing service for UBA packets, similar to Ordhook for Ordinal Inscriptions. This would allow a blockchain agnostic NFT marketplace to be established.

On the theoretical side, it would be useful to extend the instantiation of the UBA method to include fungible tokens. In this case, the UBA packets need to have multiple inputs and outputs. One is now faced with a choice: either introduce a different blockchain outpoint for each packet output, or to use a single blockchain outpoint for all packet outputs. It would be interesting to weight up the pros and cons of both approaches, at the same time paying attention to other implementation details that are specific to fungible tokens.

In Section 4 we outlined a proof system that would allow a UBA token system to be private and tokens to be validated succinctly. To achieve this, work needs to be done to create a recursive ZKP that can validate UBA packets back to issuance. Given that we have a lot of flexibility in the design of UBA packets, one could investigate whether there are different design choices that allow proofs to be constructed more efficiently, for example by using different hash functions. For the auxiliary blockchain transactions, succinct blockchain inclusion proofs would help reduce the overall complexity of proof generation for UBA packets. Such proofs would be blockchain-specific and be of independent utility to their respective chains.

Acknowledgements

This work is built on top of many great ideas around linked blockchain data commitments that were generated at nChain between 2020 – 2024. Here we essentially turn the Chain-of-Commitments protocol into a cross-chain token protocol by introducing blockchain outpoints at the level of data packets and adding signatures for authenticity. The author thanks Dr. Wei Zhang and Dr. Michaella Pettit for their valuable feedback on the manuscript, and to Josie Wilden, John Murphy, and Arthur Gordon for implementing the proof-of-concept.

References

1. Ordinal theory handbook, https://docs.ordinals.com, last accessed 21/05/24
2. Thorn A. et al: Bitcoin Inscriptions & Ordinals A New $5bn Market. Galaxy Research White Paper (2023)
3. Protecting keys with the Secure Enclave, Apple, https://developer.apple.com/documentation/security/certificate_key_and_trust_services/keys/protecting_keys_with_the_secure_enclave, last accessed 21/05/24
4. Goodell, G., Toliver, D.R., Nakib, H.D.: A Scalable Architecture for Electronic Payments. In: Matsuo, S., *et al.* Financial Cryptography and Data Security. FC 2022 International Workshops. Lecture Notes in Computer Science, vol 13412. doi.org/10.1007/978-3-031-32415-4_38, arxiv.org/abs/2110.13840
5. Lee S-S., Murashkin A., Derka M., Gorzny J.: SoK: Not Quite Water Under the Bridge: Review of Cross-Chain Bridge Hacks. arxiv.org/abs/2210.16209
6. Counterparty, https://counterparty.io/, last accessed 21/05/24
7. Assia Y., Buterin V., Hakim L., Rosenfeld M., Lev R.: Colored Coins whitepaper. https://www.etoro.com/wp-content/uploads/2022/03/Colored-Coins-white-paper-Digital-Assets.pdf
8. Introducing Ordhook: a Reliable Index for Ordinals, https://www.hiro.so/blog/introducing-ordhook-a-reliable-index-for-ordinals, last accessed 21/05/24
9. Kiraz M. S., Larraia E., Vaughan O.: NFT Trades in Bitcoin with Off-chain Receipts. Applied Cryptography and Network Security Workshops: ACNS 2023 Satellite Workshops, AIBlock, Kyoto, Japan, June 19–22, 2023, Proceedings Jun 2023, Pages 100–117 https://doi.org/10.1007/978-3-031-41181-6_6, eprint.iacr.org/2023/697
10. Maxwell G.: CoinJoin: Bitcoin privacy for the real world. Post on bitcointalk. org. https://bitcointalk.org/index.php?topic=279249, last accessed 21/05/24
11. Miers, I., Garman, C., Green, M., Rubin, A.D.: Zerocoin: Anonymous distributed e-cash from Bitcoin. In: *Proceedings of IEEE SP*, 2013, 397–411. http://eprint.iacr.org/2014/349
12. Chaum, D.: Blind Signatures for Untraceable Payments. In: Advances in Cryptology: Proceedings of Crypto 82(3), pp. 199–203, 1983. https://chaum.com/wp-content/uploads/2022/01/Chaum-blind-signatures.pdf
13. Monero, https://www.getmonero.org/, last accessed 21/05/24
14. ZCash, https://z.cash/, last accessed 21/05/24
15. ZK-rollups. https://ethereum.org/en/developers/docs/scaling/zk-rollups/, last accessed 21/05/24
16. ImmutableX, https://www.immutable.com/, last accessed 21/05/24
17. Mute, https://mute.io/, last accessed 21/05/24
18. L. Hioki: A pre-consensus mechanism to secure instant finality and long interval in zkRollup. https://ethresear.ch/t/a-pre-consensus-mechanism-to-secure-instant-finality-and-long-interval-in-zkrollup/8749, last accessed 21/05/24
19. ZeroSync, https://zerosync.org/, last accessed 21/05/24

20. M. Herlihy: Atomic Cross-Chain Swaps. PODC '18: Proceedings of the 2018 ACM Symposium on Principles of Distributed ComputingJuly 2018. Pages 245–254. https://doi.org/10.1145/3212734.3212736
21. Poon, J., Dryja, T.: The bitcoin lightning network: scalable off-chain instant payments (2016).
22. Li, L., Wu, J., Cui, W.: A review of blockchain cross-chain technology. In: IET Blockchain, Volume3, Issue3, September 2023, Pages 149-158. doi.org/10.1049/blc2.12032.
23. Hope-Bailie, A., Interledger, T.S.: Creating a standard for payments. In: Proceedings of the 25th International Conference Companion on World Wide Web, pp. 281–282. ACM, New York (2016) https://doi.org/10.1145/2872518.2889307
24. Wood, G.: Polkadot: vision for a heterogeneous multi-chain framework. *White Pap.* **21**, 2327–4662 (2016), https://assets.polkadot.network/Polkadot-whitepaper.pdf
25. Xie T., et al.: zkBridge: Trustless Cross-chain Bridges Made Practical. In: CCS '22: Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications SecurityNovember 2022Pages 3003–3017 https://doi.org/10.1145/3548606.3560652
26. Cross-Chain Transfer Protocol, https://www.circle.com/en/cross-chain-transfer-protocol, last accessed 21/05/24
27. Wormhole Bridge Exploit Incident Analysis, Certik, https://www.certik.com/resources/blog/1kDYgyBcisoD2EqiBpHE5l-wormhole-bridge-exploit-incident-analysis, last accessed 21/05/24
28. Harmony Incident Analysis, Certik, https://www.certik.com/resources/blog/2QRuMEEZAWHx0f16kz43uC-harmony-incident-analysis, last accessed 21/05/24

A Proof-of-concept: BSV, ETH, BSV

Here we provide a proof-of-concept implementation of the UBA token system presented in Section 4. We issue an NFT to Alice on Bitcoin SV (BSV), transfer it to Bob on the Ethereum, and then to Charlie back on BSV. The NFT is an image of the space shuttle stored on IPFS at location

`QmSgvgwxZGaBLqkGyWemEDqikCqU52XxsYLKtdy3vGZ8uq` .

Below, we provide the data of the UBA packets P_0, P_1, P_2 and links to the auxiliary blockchain transactions Tx_1, Tx_2 that are available online on the BSV and Ethereum Testnets, respectively. We encourage the reader to explicitly verify that the packets and transactions adhere to the protocol outlined in Section 4. (Recall that the struture of the packets is given by Fig. 2, and the auxiliary blockchain transactions by Fig. 3.)

A.1 Issue to Alice on BSV

The packet P_0 records that an NFT is issued Alice on the BSV blockchain. The data is given as follows.

```
"P_0":
{
    "asset_id": "Space Shuttle launch",
    "data": "QmSgvgwxZGaBLqkGyWemEDqikCqU52XxsYLKtdy3vGZ8uq",
    "previous_packet": null,
    "signature":
"d748d8d02c9babed9aaf0c408629267ec06a7ee02ed706d150fc9a5dfa1e0f8f02c1682b0
909ff2c9d3d26a26102a3522ef22eee6ed987dde229973c39fa8359",
    "signature_scheme": "NIST256p",
    "public_key":
"025900eec3232e7322efcc326d0da932e845344641e190dcb10da6bba7c89fc176",
    "blockchain_outpoint":
"f520e23dd06e8921dc58f64236a63df455ef1e7cc8fc72c8111f7a1a7e82b784:1",
    "blockchain_id": "BSV"
}

"PID_0": "a8bcb5c0a2f707d14592f392c69bdedd25d1702ee50216775436bd82e19570f5"
```


Let us break down the fields above. The first entry is the asset ID which should be constant for all packets. The second entry contains an optional data payload that we have chosen to be the IPFS location of the space shuttle image. The following three fields represent the packet's input. First comes a reference to a previous packet, which is null in this case. Next is a signature, and, since this is the issuance packet, this is from the issuer. (In this toy model are re-using Alice's public key here. We can think of this as Alice issuing an NFT to herself.) The signature is over the serialised packet fields excluding the signature and signature scheme. Specifically, the signature signs the data

```
sighash_preimage = { asset_id || data || previous_packet || public_key
              || blockchain_outpoint || blockchain_id } .
```

The signature scheme is specified to be NIST P-256 and can be verified using open-source libraries such as OpenSSL and python-ecdsa.

The final three fields represent the output of the packet. The public key belongs to Alice as the recipient of the NFT. The blockchain outpoint is controlled by Alice, and the blockchain ID specifies that this outpoint is on BSV.

The last entry in the box above is the packet ID. For this simple example, we re-use the sighash so we have `PID_0 = sighash_ 0`. In practice, a double hash of the serialised packet fields is recommended.

A.2 Transfer to Bob on Ethereum

Since Ethereum operates an account-based model as opposed to a UTXO model, one cannot directly specify an unspent outpoint on Ethereum. However, it is straight-forward to implement a smart contract that can mimic a UTXO. This smart contract is universal and can be called by anyone. Essentially, the smart contract has two main functions: `createUTXO` and `spendUTXO`. When the `createUTXO` function is called, an 'outpoint' object is created, and a unique ID is returned. This 'outpoint' has a status of 'unspent'. The `spendUTXO` function accepts an 'outpoint' ID and data payload as input. When the `spendUTXO` function is called, the 'outpoint' status is changed to a fixed final state 'spent' and the data payload is assigned to the 'outpoint'. This is similar a Bitcoin

transaction with one input and one unspendable output with data payload, as required in Fig. 3. Our implementation of the smart contract is available here

`0xf120D32bb10A2aE2971f9Aa026aBE8F0dA9709fb` .

Our smart contract also contains the functions `isUTXOspent` and `getCpid`. The corresponding solidity code is available upon request.

Using the above contract, Bob is now able to create an 'outpoint' $vout_1$ on Ethereum. Alice uses this to construct packet P_1 transferring ownership of the NFT to Bob.


```
"P_1":
{
    "asset_id": "Space Shuttle launch",
    "data": "QmSgvgwxZGaBLqkGyWemEDqikCqU52XxsYLKtdy3vGZ8uq",
    "previous_packet":
"a8bcb5c0a2f707d14592f392c69bdedd25d1702ee50216775436bd82e19570f5",
    "signature":
"f305de9e68ab9d8b156104cc8be9b5249fb5f056fb5abb08f3314a827dfd6cbb721c3f25a
a63a981d46110e4e4e4e506b98c971d6f4973dce87109e74ffebe40",
    "signature_scheme": "NIST256p",
    "public_key":
"021b955dff7a28f722b25304f3daba78c83718ec3ab38486c523c748b9f27f6ce0",
    "blockchain_outpoint":
"0x18a33e2702059102eb10097333d850314a42efb84da6b6ec676748a904251749",
    "blockchain_id": "ETH"
}

"PID_1": "f12d4ba9592137828718ffd50ee3ec5ab43e012e3e707c006c40dbeb4d1e3f89"
```


To confirm the transfer, a corresponding blockchain transaction Tx_1 is created by Alice and broadcast to the BSV network

`8e0c17a1d9b352c78c1fdb3ff18fa25bd1d01cd81f428e12bec52fb703753621` .

This transaction has one input $vout_0$ and one unspendable output containing PID_1. Bob is now the rightful owner of the NFT and it cannot be double-spent by Alice.

A.3 Transfer to Charlie on BSV

Bob now sends the NFT to Charlie who operates on BSV. To do this, Bob creates packet P_2 with the following data.


```
"P_2":
{
    "asset_id": "Space Shuttle launch",
    "data": "QmSgvgwxZGaBLqkGyWemEDqikCqU52XxsYLKtdy3vGZ8uq",
    "previous_packet":
"f12d4ba9592137828718ffd50ee3ec5ab43e012e3e707c006c40dbeb4d1e3f89",
    "signature":
"0b4db3336d4070f4af2f8f56608c824b9d45064880026214788c4860b55f2d74f508b352c7
fb3a995559e7eae43b7e6dd3bbfdc54961d148cb64cbb1cc685158",
    "signature_scheme": "NIST256p",
    "public_key":
"02a1002a5f24d40ac2f87fa4562cca0202007c5eefe601d8c0d7b0e636813f92b9",
    "blockchain_outpoint":
"2a1c1dea9efa62c88a0b4e553d1e32f631b86ad78a1760fe49f7eabb13607a3b:1",
    "blockchain_id": "BSV"
}

"PID_2": "2cf8337a75f2ad32fac567a537460a56fcad33d7553551ab46e5bc77fd34c9c9"
```

To confirm the transfer, a corresponding auxiliary blockchain transaction Tx_2 is created by Bob and broadcast to the Ethereum network (note PID_2 in the data field)
`0xf082c469b5dcbb83e1aa362a570ac825c941adb32d03dcc8df148d6cd52eae36` .
This sets the 'outpoint' status to 'spent' and attaches an immutable record of PID_2. Charlie is now the rightful owner of the NFT and it cannot be double-spent by Bob.