Cryptography for Blockchain Applications

Noemi Glaeser

Abstract

Blockchains were introduced in 2008 by Satoshi Nakamoto as a way to implement a trusted but decentralized append-only ledger. This simple functionality has given rise to a plethora of decentralized applications utilizing the blockchain as a public bulleting board. In recent years, it has become clear that this basic functionality is not enough to prevent widespread attacks on both the privacy and security of blockchain users, as evidenced by the blockchain analytics industry and the billions of dollars stolen via cryptocurrency exploits to date. This work explores the role cryptography has to play in the blockchain ecosystem to both enhance user privacy and secure user funds.

Contents

1 Introduction

Bitcoin [Nak08] was the first digital currency to successfully implement a fully trustless and decentralized payment system. todo: ... Ethereum [But14] introduced programmability via *smart contracts* todo: ...

1.1 Model and Preliminaries

Definition 1 (NP Relation)**.** *An NP relation $\mathcal{R}$ is todo: ...*

The language corresponding to $\mathcal{R}$ is defined as $\mathcal{L}_{\mathcal{R}} := \{Y : \exists y \text{ such that } (Y, y) \in \mathcal{R}\}$.

1.2 Definitions

2 Privacy-Enhancing Building Blocks

Although cryptocurrencies are often treated as fully anonymous digital currencies, numerous works have shown how to link transactions or even fully deanonymize users in many popular cryptocurrencies [BKP14, BT19, KKM14, MSH+18, KYMM18]. Numerous mitigations have been suggested, including cryptocurrency mixers (described below) and privacy-first cryptocurrencies like Zcash [zca] and Monero [mon].

2.1 Zero-Knowledge Proofs and their trust assumptions

Non-interactive zero-knowledge proofs (NIZKs) are ubiquitous building blocks in many blockchains. Zcash [zca], as indicated by the letter "Z" in its name, relies heavily on a type of NIZK called zkSNARK (zero-knowledge succinct argument of knowledge) to achieve private payments: zkSNARKs are used to prove a party has sufficient funds to make a payment without revealing anything more about those funds [BCG+14]. On Ethereum, (zk-)rollups enhance scalability by leveraging the succinctness of (zk)SNARKs, though they may or may not offer the zero-knowledge property.

To achieve such a high level of succinctness, SNARKs rely on a trusted setup to generate a *common reference string (CRS)*. In keeping with the primary innovation of the blockchain, which is the elimination of a trusted third party (TTP), practitioners use various approaches to minimize the trust in the CRS generation. Zcash uses a multi-party computation ceremony [zca16] with many independent participants to distribute the trust among several parties. Another trust-minimizing approach consists of using SNARKs with universal and updatable CRS [GKM+18, MBKM19, CHM+20, GWC19]. A universal CRS can be reused across applications, avoiding a new complicated setup ceremony for every use. Updatable CRSs allow any participant in a system to contribute randomness to the CRS at any point, including once the CRS is in production use, to enable a "one-out-of-many" trust scenario in which the user must only

trust themselves to contribute (and then delete) good randomness to the CRS in order for the whole system to be secure.

An orthogonal concern is maintaining the security of SNARKs when they are composed with other protocols in the complex blockchain ecosystem. Formally, this is modeled by universally composable security via the UC framework [Can01]. Unfortunately, most SNARKs in deployment today are not provably UC-secure. Although compilers to transform any SNARK or NIZK into a UC variant exist [KZM+15, GKO+23], these are not compatible with the aforementioned trust-minimizing properties like updatability. A generic compiler which adds UC-security while maintaining updatability would help ensure confidence in both the trusted setup and the operational security of deployed NIZKs.

2.1.1 Circuit-Succinct Universally Composable NIZKs with Updatable CRS [AGRS24]

In this section, we summarize the contributions and constructions of [AGRS24]. todo: ...

	UC		succinctness-preserving		
	SE	BBE	in $\|C\|$	in $\|w\|$	upd. CRS
C∅C∅ [KZM+15]	✓	✓	✓	✗	✗
DS [DS19]	✓	✗	✓	✓	✗
Lamassu [ARS20]	✓	✗	✓	✓	✓
This work [AGRS24]	✓	✓	✓	✗	✓
Concurr. work [GKO+23]	✓	✓	✓	✓	✗

Table 1: Comparison with concurrent and previous work.

3 Privacy-Enhancing Applications

3.1 Cryptocurrency Mixers

Noemi: How to position this? Our focus will be on *off-chain* mixers, whose focus is on enabling *scalability* and *interoperability* rather than privacy... todo: Talk about how these also solve scalability and interoperability?

todo: citations [RMK14, SNBB19, TLK+18, BNM+14], Bitcoin (CoinJoin, CoinShuffle), Bolt, Blindcoin, Mixcoin Cryptocurrency mixers add a measure of k-anonymity to cryptocurrency tokens by employing a central party, or *mixer*, to shuffle the tokens among users of the service. Users deposit their coins into the service, and later retrieve them again (using a different address, otherwise anonymity is trivially broken). Any particular token (retrieved from the mixer) cannot be tied to a particular source (user who deposited money from the mixer):

each of the k users is equally likely to be the source of a given token. For security, a mixer must offer *atomicity*, i.e., a user pays $c + \epsilon$ coins if and only if the "recipient" (normally the same user, but under a new address) is paid c coins (ϵ is a parameter which represents the mixer and transaction fees). todo: exit scams

Mixers come in two flavors: on- and off-chain mixers. On-chain mixers todo: [] are simply accounts into which users can deposit coins and later retrieve them (or allow another party to retrieve them) by redeeming some token, with atomicty enforced via an on-chain script.Noemi: check this

Off-chain mixers normally require more complicated protocols to enforce the atomicity requirement without the scripting functionality offered by the underlying blockchain. One line of work, initiated by TumbleBit [HAB+17], uses a protocol paradigm which we refer to as a *synchronization puzzle*. A synchronization puzzle is a three-party protocol between a sender (Alice), a mixer (Hub), and a recipient (Bob) todo: insert figure. Synchronization puzzle protocols consist of four steps: (1) Hub and Bob execute *puzzle promise* phase with respect to some message m_{HB}, which outputs a puzzle τ containing a hidden signature s on m_{HB}. (2) Bob sends τ to Alice via a private channel, who (3) uses it to execute the *puzzle solver* phase with Hub with respect to another message m_{AH}. At the end of this phase, Alice obtains the signature s on m_{HB} and Hub learns s' on m_{AH}. To conclude, (4) Alice sends s to Bob. A synchronization puzzle protocol should satisfy the following properties:

- **Blindness**: In the puzzle solver phase, Hub *blindly* helps solve τ, i.e., the phase should not reveal anything about τ to Hub. (This keeps Alice and Bob unlinkable from the point of view of Hub.)
- **Unlockability**: If the puzzle solver phase completes successfully, s must be a valid secret for τ. (This ensures that the Hub cannot learn s' without revealing s.)
- **Unforgeability**: Bob cannot output a valid signature s on m_{HB} before the puzzle solver phase completes. (This ensures Bob cannot learn s without Hub learning s'.)

To use a synchronization puzzle to realize an atomic payment, the parties set $m_{AH} : (A \xrightarrow{c+\epsilon} H)$ and $m_{HB} : (H \xrightarrow{c} B)$, where $(U_i \xrightarrow{c} U_j)$ denotes a payment of c coins from user U_i to U_j. Then s' and s are set to the signatures authorizing m_{AH} and m_{HB}, respectively. Thus, at the completion of the protocol Alice will have sent c coins to Bob (and paid a fee of ϵ to Hub).

Tumblebit realizes a synchronization puzzle via todo: ???.

Before we describe the approach to instantiating synchronization puzzles taken by follow-up work [TMM21], we introduce the notion of *adaptor signatures*.

Definition 2 (adaptor signature [AEE+21])**.** *An adaptor signature scheme* $\Pi_{\mathsf{ADP}} := (\mathsf{KGen}, \mathsf{PreSig}, \mathsf{PreVrfy}, \mathsf{Adapt}, \mathsf{Vrfy}, \mathsf{Ext})$ *is defined with respect to a digital signature scheme* Π_{ADP} *and an NP relation* $\mathcal{R}$*:*

$\mathsf{KGen}(1^\lambda) \to (\mathsf{vk}, \mathsf{sk})$**:** *The key generation algorithm is the same as in the underlying digital signature scheme, i.e.,* $\Pi_{\mathsf{ADP}}.\mathsf{KGen}$.

$\mathsf{PreSig}(\mathsf{sk}, m, Y) \to \tilde{\sigma}$**:** *The pre-signing algorithm takes as input a signing key* sk*, message* m*, and instance* Y *of the relation* $\mathcal{R}$ *and returns a pre-signature* $\tilde{\sigma}$.

$\mathsf{PreVrfy}(\mathsf{vk}, m, Y, \tilde{\sigma}) \to \{0,1\}$**:** *The pre-verification algorithm checks that a pre-signature is well-formed.*

$\mathsf{Adapt}(\tilde{\sigma}, y) \to \sigma$**:** *Given a witness* y *for the instance* Y*, this algorithm adapts the pre-signature* $\tilde{\sigma}$ *into a valid signature* σ.

$\mathsf{Vrfy}(\mathsf{vk}, m, \sigma) \to \{0,1\}$**:** *The verification algorithm is the same as in the underlying digital signature scheme, i.e.,* $\Pi_{\mathsf{ADP}}.\mathsf{Vrfy}$.

$\mathsf{Ext}(\tilde{\sigma}, \sigma, Y) \to y$**:** *Given a pre-signature* $\tilde{sigma}$ *and a signature* σ *generated with respect to some instance* Y*, the extract algorithm outputs the corresponding witness* y *such that* $(Y, y) \in \mathcal{R}$.

An adaptor signature scheme should satisfy the following properties: *pre-signature correctness*, which guarantees that for all instances $Y \in \mathcal{L}_Rel$ and honestly generated $\tilde{\sigma}, \sigma$, the pre-signature and signature pass (pre-)verification and the extracted witness $y' \leftarrow \mathsf{Ext}(\tilde{\sigma}, \sigma, Y)$ should satisfy $(Y, y') \in \mathcal{R}$; *unforgeability*, which is a straightforward extension of the standard existential unforgeability notion (EUF-CMA) for digital signatures; *pre-signature adaptability*, which states that for any $Y \in \mathcal{L}_\mathcal{R}$ and corresponding pre-signature $\tilde{\sigma}$, pre-verification implies that $\tilde{\sigma}$ can be adapted to a verifying signature σ; and *witness extractability*, which says that it is difficult for an adversary to adapt an honestly-generated pre-signature $\tilde{\sigma}$ into a signature σ which verifies but where $y' \leftarrow \mathsf{Ext}(\tilde{\sigma}, \sigma, Y)$ such that $(Y, y') \notin \mathcal{L}_\mathcal{R}$. We refer the reader to [GMM+22] for formal definitions.

Anonymous Atomic Locks (A^2L) [TMM21] uses a rerandomizable CPA-secure encryption scheme Π_E and an *adaptor signature* scheme Π_ADP to realize a synchronization puzzle which is compatible with a wider range of cryptocurrencies. todo: ...

3.1.1 Blind Conditional Signatures

In this section, we summarize the contributions and constructions of [GMM+22]. In this paper, we analyzed the A^2L protocol [TMM21] and found that, in contrast to its claims, it is not secure. Although A^2L was proven secure in the universal composability (UC) [Can01] framework, we show that a gap in their formal model allows two constructions which are completely insecure despite meeting their definitions: one admits a key recovery attack and the other allows a colluding sender and recipient to steal coins from the mixer. To close this gap, we introduce a new primitive called blind conditional signtaures (BCS) which

captures the core coin mixing functionality. We give game-based security definitions for BCS and show how to modify A^2L to obtain a new protocol, A^2L^+, which meets these definitions. We also give a UC-secure construction of BCS, dubbed A^2L^{UC}, which requires much more complex machinery.

Counterexamples to A^2L. We show that there exist cryptographic primitives which satisfy the prerequisites of A^2L's main theorem, but allow *(a)* a *key recovery attack*, in which a malicious user is able to learn the long-term secret of the hub or *(b)* a *one-more signature attack*, in which a sender and recipient can collude to obtain n tokens from the hub while only sending $n-1$ tokens. Both attacks run in polynomial time and succeed with overwhelming probability. These instantiations of A^2L are specifically crafted allow an attack and do not imply that all instantiations of A^2L are broken; however, we cannot prove the security of A^2L either. This gap is discussed further below.

Below we give informal descriptions of both attacks below. We refer the reader to [GMM+22] for detailed descriptions and analysis. Both attacks rely on the fact that the A^2L protocol offers a malicious Alice what amounts to a decryption oracle. This oracle, which we refer to as $\mathcal{O}^{A^2L}$, takes as input a verification key vk, message m, group element h, ciphertext c, and pre-signature $\tilde{\sigma}$. It computes todo: ...

Key recovery attack.

todo: write intuitive explanation of the attacks, refer to full paper for details

Definitions. todo: Write game-based defs, refer reader to A2L and our paper for UC ideal functionality

Constructions. todo: Write both constructions?

3.2 On-chain private voting

3.2.1 Cicada: A framework for private non-interactive on-chain auctions and voting [GSZB23]

In this section, we summarize the contributions and constructions of [GSZB23]. todo: ...

4 Proposed Work

4.1 Registration-Based Encryption as a Web3 service

Noemi: Unclear if this can be included

4.2 Threshold cryptocurrency wallets in the hot-cold paradigm

5 Timeline

todo:

References

[AEE+21] Lukas Aumayr, Oguzhan Ersoy, Andreas Erwig, Sebastian Faust, Kristina Hostáková, Matteo Maffei, Pedro Moreno-Sanchez, and Siavash Riahi. Generalized channels from limited blockchain scripts and adaptor signatures. In *Advances in Cryptology–ASIACRYPT 2021: 27th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 6–10, 2021, Proceedings, Part II 27*, pages 635–664. Springer, 2021.

[AGRS24] Behzad Abdolmaleki, Noemi Glaeser, Sebastian Ramacher, and Daniel Slamanig. Circuit-succinct universally-composable NIZKs with updatable CRS. In *37th IEEE Computer Security Foundations Symposium*, 2024.

[ARS20] Behzad Abdolmaleki, Sebastian Ramacher, and Daniel Slamanig. Lift-and-shift: Obtaining simulation extractable subversion and updatable SNARKs generically. In Jay Ligatti, Xinming Ou, Jonathan Katz, and Giovanni Vigna, editors, *ACM CCS 2020*, pages 1987–2005. ACM Press, November 2020.

[BCG+14] Eli Ben-Sasson, Alessandro Chiesa, Christina Garman, Matthew Green, Ian Miers, Eran Tromer, and Madars Virza. Zerocash: Decentralized anonymous payments from bitcoin. In *2014 IEEE Symposium on Security and Privacy*, pages 459–474. IEEE Computer Society Press, May 2014.

[BKP14] Alex Biryukov, Dmitry Khovratovich, and Ivan Pustogarov. Deanonymisation of clients in bitcoin P2P network. In Gail-Joon Ahn, Moti Yung, and Ninghui Li, editors, *ACM CCS 2014*, pages 15–29. ACM Press, November 2014.

[BNM+14] Joseph Bonneau, Arvind Narayanan, Andrew Miller, Jeremy Clark, Joshua A. Kroll, and Edward W. Felten. Mixcoin: Anonymity for bitcoin with accountable mixes. In Nicolas Christin and Reihaneh Safavi-Naini, editors, *FC 2014*, volume 8437 of *LNCS*, pages 486–504. Springer, Heidelberg, March 2014.

[BT19] Alex Biryukov and Sergei Tikhomirov. Deanonymization and linkability of cryptocurrency transactions based on network analysis. In *2019 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 172–184, 2019.

[But14] Vitalik Buterin. A next-generation smart contract and decentralized application platform. *White paper*, 2014.

[Can01] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *42nd FOCS*, pages 136–145. IEEE Computer Society Press, October 2001.

[CHM+20] Alessandro Chiesa, Yuncong Hu, Mary Maller, Pratyush Mishra, Psi Vesely, and Nicholas P. Ward. Marlin: Preprocessing zkSNARKs with universal and updatable SRS. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part I*, volume 12105 of *LNCS*, pages 738–768. Springer, Heidelberg, May 2020.

[DS19] David Derler and Daniel Slamanig. Key-homomorphic signatures: definitions and applications to multiparty signatures and non-interactive zero-knowledge. *Designs, Codes and Cryptography*, 87:1373–1413, 2019.

[GKM+18] Jens Groth, Markulf Kohlweiss, Mary Maller, Sarah Meiklejohn, and Ian Miers. Updatable and universal common reference strings with applications to zk-SNARKs. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part III*, volume 10993 of *LNCS*, pages 698–728. Springer, Heidelberg, August 2018.

[GKO+23] Chaya Ganesh, Yashvanth Kondi, Claudio Orlandi, Mahak Pancholi, Akira Takahashi, and Daniel Tschudi. Witness-succinct universally-composable SNARKs. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part II*, volume 14005 of *LNCS*, pages 315–346. Springer, Heidelberg, April 2023.

[GMM+22] Noemi Glaeser, Matteo Maffei, Giulio Malavolta, Pedro Moreno-Sanchez, Erkan Tairi, and Sri Aravinda Krishnan Thyagarajan. Foundations of coin mixing services. In Heng Yin, Angelos Stavrou, Cas Cremers, and Elaine Shi, editors, *ACM CCS 2022*, pages 1259–1273. ACM Press, November 2022.

[GSZB23] Noemi Glaeser, István András Seres, Michael Zhu, and Joseph Bonneau. Cicada: A framework for private non-interactive on-chain auctions and voting. Cryptology ePrint Archive, Paper 2023/1473, 2023. https://eprint.iacr.org/2023/1473.

[GWC19] Ariel Gabizon, Zachary J. Williamson, and Oana Ciobotaru. PLONK: Permutations over lagrange-bases for oecumenical noninteractive arguments of knowledge. Cryptology ePrint Archive, Report 2019/953, 2019. https://eprint.iacr.org/2019/953.

[HAB+17] Ethan Heilman, Leen Alshenibr, Foteini Baldimtsi, Alessandra Scafuro, and Sharon Goldberg. TumbleBit: An untrusted bitcoin-compatible anonymous payment hub. In *NDSS 2017*. The Internet Society, February / March 2017.

[KKM14] Philip Koshy, Diana Koshy, and Patrick McDaniel. An analysis of anonymity in bitcoin using P2P network traffic. In Nicolas Christin and Reihaneh Safavi-Naini, editors, *FC 2014*, volume 8437 of *LNCS*, pages 469–485. Springer, Heidelberg, March 2014.

[KYMM18] George Kappos, Haaroon Yousaf, Mary Maller, and Sarah Meiklejohn. An empirical analysis of anonymity in zcash. In William Enck and Adrienne Porter Felt, editors, *USENIX Security 2018*, pages 463–477. USENIX Association, August 2018.

[KZM+15] Ahmed Kosba, Zhichao Zhao, Andrew Miller, Yi Qian, Hubert Chan, Charalampos Papamanthou, Rafael Pass, abhi shelat, and Elaine Shi. C∅c∅: A framework for building composable zero-knowledge proofs. Cryptology ePrint Archive, Report 2015/1093, 2015. https://eprint.iacr.org/2015/1093.

[MBKM19] Mary Maller, Sean Bowe, Markulf Kohlweiss, and Sarah Meiklejohn. Sonic: Zero-knowledge SNARKs from linear-size universal and updatable structured reference strings. In Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz, editors, *ACM CCS 2019*, pages 2111–2128. ACM Press, November 2019.

[mon] Monero. https://getmonero.org.

[MSH+18] Malte Möser, Kyle Soska, Ethan Heilman, Kevin Lee, Henry Heffan, Shashvat Srivastava, Kyle Hogan, Jason Hennessey, Andrew Miller, Arvind Narayanan, and Nicolas Christin. An empirical analysis of traceability in the monero blockchain. *PoPETs*, 2018(3):143–163, July 2018.

[Nak08] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. *Decentralized business review*, 2008.

[RMK14] Tim Ruffing, Pedro Moreno-Sanchez, and Aniket Kate. CoinShuffle: Practical decentralized coin mixing for bitcoin. In Miroslaw Kutylowski and Jaideep Vaidya, editors, *ESORICS 2014, Part II*, volume 8713 of *LNCS*, pages 345–364. Springer, Heidelberg, September 2014.

[SNBB19] István András Seres, Dániel A. Nagy, Chris Buckland, and Péter Burcsi. MixEth: efficient, trustless coin mixing service for Ethereum. Cryptology ePrint Archive, Report 2019/341, 2019. https://eprint.iacr.org/2019/341.

[TLK+18] Muoi Tran, Loi Luu, Min Suk Kang, Iddo Bentov, and Prateek Saxena. Obscuro: A bitcoin mixer using trusted execution environments. In *Proceedings of the 34th Annual Computer Security Applications Conference*, pages 692–701, 2018.

[TMM21] Erkan Tairi, Pedro Moreno-Sanchez, and Matteo Maffei. A^{2}L: Anonymous atomic locks for scalability in payment channel hubs. In *2021 IEEE Symposium on Security and Privacy*, pages 1834–1851. IEEE Computer Society Press, May 2021.

[zca] Zcash. https://z.cash.

[zca16] The design of the ceremony, 10 2016. https://electriccoin.co/blog/the-design-of-the-ceremony/.