A Hitchhiker's Guide to Privacy-Preserving Cryptocurrencies: A Survey on Anonymity, Confidentiality, and Auditability

MATTEO NARDELLI*, FRANCESCO DE SCLAVIS*, and MICHELA IEZZI*, Bank of Italy, Italy

Cryptocurrencies and central bank digital currencies (CBDCs) are reshaping the monetary landscape, offering transparency and efficiency while raising critical concerns about user privacy and regulatory compliance. This survey provides a comprehensive and technically grounded overview of privacy-preserving digital currencies, covering both cryptocurrencies and CBDCs. We propose a taxonomy of privacy goals—including anonymity, confidentiality, unlinkability, and auditability—and map them to underlying cryptographic primitives, protocol mechanisms, and system architectures. Unlike previous surveys, our work adopts a design-oriented perspective, linking high-level privacy objectives to concrete implementations. We also trace the evolution of privacy-preserving currencies through three generations, highlighting shifts from basic anonymity guarantees toward more nuanced privacy-accountability trade-offs. Finally, we identify open challenges at the intersection of cryptography, distributed systems, and policy definition, which motivate further investigation into the primitives and design of digital currencies that balance real-world privacy and auditability needs.

CCS Concepts: • **Security and privacy** → **Privacy-preserving protocols**; *Pseudonymity, anonymity and untraceability*; *Domain-specific security and privacy architectures*; • **Computer systems organization** → Distributed architectures; • **General and reference** → **Surveys and overviews**.

Additional Key Words and Phrases: Privacy-Enhancing Technologies, Cryptocurrencies, CBDC

1 Introduction

The rise of cryptocurrencies and the rapid development of central bank digital currencies (CBDCs) have fueled renewed interest in digital currencies. This momentum stems from the advances in decentralized technologies (i.e., blockchains and smart contracts), the introduction of decentralized finance, and the increasing concerns over data privacy and financial surveillance. While cryptocurrencies have demonstrated that permissionless, cryptographically secure value transfer is feasible, central banks are interested in modernizing monetary infrastructures with digital-native capabilities. However, these innovations raise fundamental questions about user privacy, institutional control, and regulatory compliance (e.g., [9, 69, 78]). Public blockchains typically offer pseudo-anonymity, using addresses decoupled from real identities. However, the transparency of public blockchains enables transactions tracing, address clustering, and ultimately users de-anonymization, thus exposing financial behavior and undermining confidentiality (e.g., [78, 81, 101, 103]). As a result, *privacy-preserving digital currencies* have emerged as a critical research field, attracting attention from cryptographers, system designers, economists, and policymakers.

Researchers have developed a variety of privacy-enhancing technologies (PETs), including mixing protocols (e.g., [98, 122, 150]) and confidential transactions (e.g., [13, 32, 74]), and privacy-native cryptocurrencies (e.g., [83, 107, 136, 144]). These solutions implement various notions of privacy, such as anonymity, confidentiality, and unlinkability. However, privacy alone is not sufficient: compliance with regulatory frameworks—such as Anti-Money Laundering (AML) and Combating the Financing of Terrorism (CFT)—requires mechanisms for auditability and selective disclosure. Therefore, recent work has explored cryptographic protocols such as zero-knowledge proofs (ZKPs), threshold encryption, and homomorphic commitments to reconcile these conflicting goals.

*All views are those of the authors and do not necessarily reflect the position of Bank of Italy.

Authors' Contact Information: Matteo Nardelli, matteo.nardelli@bancaditalia.it; Francesco De Sclavis, francesco.desclavis@bancaditalia.it; Michela Iezzi, michela.iezzi@bancaditalia.it, Bank of Italy, Rome, Italy.

This survey proposes a unifying, design-oriented perspective on privacy-preserving digital currencies, spanning both decentralized cryptocurrencies and CBDCs. Unlike previous surveys that focus on broad overviews (e.g., [9, 68, 78]), specific technologies (e.g., [1, 69, 135]), or do not consider auditability (e.g., [61, 78, 111]), our work aims to bridge these dimensions. We introduce a systematic taxonomy of privacy and auditability properties and map them to concrete implementations across a broad range of systems. Structured as a hitchhiker's guide, our survey defines key design goals and illustrates how they have been implemented in the literature through cryptographic primitives, protocols, and system architectures. Our contributions are threefold. First, we introduce a taxonomy that classifies privacy-preserving digital currencies along multiple dimensions, including system architecture, trust model, value representation, privacy guarantees, and auditability mechanisms. Second, we systematize how privacy properties are formalized and realized in digital currency systems, mapping high-level goals to technical mechanisms. We trace the evolution of privacy-preserving digital currencies across three generations. Third, we identify research challenges stemming from the design of privacy-preserving digital currencies, which lie at the intersection of cryptography, distributed systems, and privacy engineering. Our survey aims to organize this diverse and technically rich landscape, supporting both foundational research and practical protocol design in the rapidly evolving domain of digital currencies.

This paper is organized as follows. Section 2 reviews related surveys and outlines how our work complements and differs from them. Section 3 introduces the key cryptographic primitives used in digital currencies and cryptocurrencies. Section 4 presents a detailed analysis of privacy-preserving currency designs along key architectural and cryptographic dimensions. Section 5 outlines open challenges and future research directions, and Section 6 concludes the paper.

2 Related Surveys

A large body of literature has explored privacy-preserving digital currencies, motivated by both the technological challenges and the societal implications of financial surveillance. This includes surveys on PETs, systematization of cryptocurrencies, and privacy investigations for CBDCs.

Surveys on Privacy in Cryptocurrencies. Among the earliest works, Bonneau et al. [29] reviewed extensions of Bitcoin and introduced altcoins (i.e., alternatives to Bitcoin), including early privacy-native protocols. Genkin et al. [68] offered a broad and accessible overview of techniques for anonymity in decentralized cryptocurrencies, while emphasizing the absence of a unified definition of privacy, which hinders fair comparisons between systems. More recent surveys have attempted to formalize anonymity notions. For instance, Amarasinghe et al. [3] developed a taxonomy of anonymity features, and evaluated Bitcoin-derived solutions against it. Our work draws from and extends this analysis by further considering the trade-offs between anonymity, confidentiality, and *auditability* across a broader range of systems, including CBDCs. Feng et al. [61] offered a layered classification of privacy-preserving mechanisms from network-level obfuscation to cryptographic primitives. While technically detailed, their coverage of cryptocurrency-specific protocols and privacy-accountability trade-offs is limited. Similarly, a systematic literature review by Herskind et al. [78] catalogs techniques such as stealth addresses, confidential transactions, and network anonymity. However, it does not analyze core system features such as value representation models or system architectures. Peng et al. [111] identify PETs and illustrate how various protocols integrate them, but do not investigate protocol evolution or design trade-offs.

Other related surveys take a narrower or orthogonal perspective. Almashaqbeh and Solomon [1] focus primarily on privacy in blockchain computation, i.e., smart contracts, but do not consider concerns of accountability in payments. Zhang [148] and Wang et al. [141] provide useful overviews

of privacy in deployed cryptocurrencies, but lack systematic analysis of technical contributions. Alsalami and Zhang [2] present a classification of PETs but do not address accountability aspects.

Privacy in CBDCs. A distinct body of work addresses privacy in CBDCs, driven by the need to balance individual privacy with institutional oversight. Auer et al. [8, 9] distinguish between soft privacy, based on access controls and trust in intermediaries, and hard privacy, based on cryptographic techniques. While they offer a high-level classification and conceptual framework, their discussion does not consider low-level protocol mechanisms or specific cryptographic solutions. Darbha and Arora (Bank of Canada)[1] and Pocher and Veneris [113] explore PETs from a compliance and governance perspective. These works offer blueprints but lack detailed cryptographic analysis. Our work bridges this gap by providing a comparative review of cryptographic primitives, protocols, and system architectures that spans both CBDCs and cryptocurrencies.

Specialized Surveys. Several surveys focus on specific technologies or techniques. Examples include Bitcoin [69], smart contracts [21, 88], DeFi [15], mixers [7], accountability mechanisms [43], and ZKPs [1, 90, 135]. These contributions do not consider the full design space of privacy-preserving digital currencies as we do, nor do they explore how different privacy features interact—particularly in settings requiring auditability. Koerhuis et al. [84] present a forensic analysis of Monero and Verge, and highlight how implementation flaws threaten privacy; a concern we also emphasize.

Contribution of This Survey. Compared to previous works, our survey offers a comprehensive framework for analyzing privacy-preserving digital currencies. We propose a unified taxonomy that integrates both decentralized cryptocurrencies and CBDCs. Specifically, we present a systematic analysis of existing solutions to identify key value representation models, privacy features, and auditability guarantees. We focus on how such features are realized through cryptographic protocols and specific design choices, resulting in a classification of contributions that connects design goals to technical implementations.

3 Background

We first introduce key cryptographic primitives, including commitments, encryption schemes, and ZKPs. Then, we briefly review key cryptographic protocols used in privacy-preserving currencies, such as anonymity-enhanced signatures, multi-party computation, and one-time public keys.

3.1 Commitments and Homomorphic Commitments

A commitment scheme is a cryptographic primitive that lets someone commit to a value in a way that he cannot change it later. The scheme consists of two phases: (i) *commit phase*, where a commitment is made to a value v, but the value is kept secret from third parties; (ii) *opening phase*, where the committed value v is revealed. A commitment is *hiding* if an adversary cannot distinguish which of two values corresponds to the commit; it is *binding* if the committer cannot generate the same commit from two different values. These properties can be either *computational*, if the adversary is computationally bounded, or *perfect*, if not. A commitment scheme cannot be both perfectly hiding and perfectly binding, due to fundamental cryptographic limitations [70].

Homomorphic commitments, such as Pedersen [110] or ElGamal [57], preserve the addition or multiplication of corresponding values. For example, let g and h be elements of a prime q group $\mathbb{G}_q$ such that $\log_g h$ is not feasible to compute, let r_i be a random value over $\mathbb{Z}_q$ and let $v_i \in \mathbb{Z}_q$ be the committer's value. Perdersen commitments are defined as $c_{v_i} = g^{v_i} h^{r_i}$. The homomorphic properties of Pedersen commitments guarantee that $g^{v_1} h^{r_1} \cdot g^{v_2} h^{r_2} = g^{v_1+v_2} h^{r_1+r_2}$ namely, the sum of the commitments of v_1 and v_2 is equal to the commitment of the sum of the two values under the sum

[1] https://www.bankofcanada.ca/2020/06/staff-analytical-note-2020-9/

of the two randomness r_1 and r_2, allowing multiple operations on the committed messages. Pedersen commitments are perfectly hiding, computationally binding, and additively homomorphic; ElGamal commitments are computationally hiding, perfectly binding, and multiplicatively homomorphic.

3.2 Encryption Schemes

3.2.1 Identity-based Encryption. Identity-based encryption (IBE) schemes [132] are public-key encryption schemes in which the public key is a string that uniquely identifies a user (e.g., an email). The corresponding private keys for decryption are derived from a master private key, by a trusted central authority. IBE can be used to simplify key management and recover, and access control within an organization. Anonymous variants make the public key indistinguishable from the ciphertext, effectively hiding the identity of the recipient of the message.

3.2.2 Threshold Encryption. Threshold encryption allows data to be encrypted in such a way that it can only be decrypted when a predefined quorum of participants agrees to reveal it. Threshold encryption can help maintain privacy while removing single point of failures. A popular implementation uses the ElGamal encryption scheme [57]. A *threshold-issuance* variant of Boneh-Franklin IBE is used in UTT [136], to distribute the master secret key among multiple authorities.

3.2.3 Homomorphic Encryption. Homomorphic encryption schemes [119] allow to compute additions, multiplications or both, without revealing the plaintext. Operations can be performed directly on ciphertexts such that the result corresponds to the encryption of the operation applied to the plaintexts, i.e., $\mathrm{Enc}(m_1) \star \mathrm{Enc}(m_2) = \mathrm{Enc}(m_1 \star m_2)$, where $\mathrm{Enc}(\cdot)$ represent an encryption function and $\star$ denotes either addition or multiplication. This allows to delegate computation to a third party while maintaining confidentiality. In payment systems (e.g., [32, 97]), it enables confidential transactions when used as a building block for ZKPs, like to homomorphic commitments.

3.3 Zero-knowledge Proofs

A ZKP is a protocol enabling one party (the prover) to convince another party (the verifier) that a statement is true, without disclosing any information beyond the validity of the statement itself. A dishonest prover can convince the verifier of a false statement, only with negligible probability (*soundness*). If the prover is computationally bounded (*computational soundness*), the proof is known as *argument* [25]. ZKPs can be either *interactive*, requiring the parties to exchange messages, or non-interactive (NIZK), when the verifier is convinced through a single message. ZKPs can also be *proofs of knowledge* when the prover can convince the verifier that he knows a certain value, e.g., a secret key (or *arguments of knowledge*, if the prover is computationally bounded). If the ZK proof is short and fast to verify, it is *succinct*. If it requires no trusted setup, it is *transparent*.

A conceptual framework for ZK protocols is the Σ-protocol (or Sigma-protocol), which is a type of 3-move interactive proof system: the prover commits a value, the verifier provides a challenge, and the prover provides a proof based on the commit and the challenge, so that they cannot choose a "malicious" proof. Using the Fiat-Shamir transform, Σ-protocols can be made non-interactive [62]. Building on this, various NIZK protocols have been designed, including zk-SNARK [25], Bulletproofs [33], and—to some extent—zk-STARK [19]. Zk-SNARKs are succinct non-interactive arguments of knowledge. Many zk-SNARK constructions have been proposed in past years (e.g.,[25, 27, 48, 73]), with different approaches also for setup. Among them, the most popular approach is Groth16 [73], an instance optimized for performance but with some trade-offs like requiring a trusted setup. Bulletproof is optimized for range proof and does not require trusted setup. zk-STARK is a recent approach offering efficient proving times and no need for trusted setup. STARKs are succinct transparent arguments of knowledge. However, in the current state, zk-STARK

generates large proof size (a few hundred kilobytes), so they are not as succinct as SNARKs. A recent survey on ZKPs can be found, e.g., in [50, 55]. We usually refer to NIZK as ZK.

3.4 Anonymity-enhanced Signatures

3.4.1 Blind Signatures. Blind signatures [44] are a form of digital signature in which the signer produces a valid signature on a message without learning its content: before signing, the message is *blinded*, i.e., transformed through a random element that makes it unrecognizable. Once the contents of the signed message are unblinded, they can be publicly verified as usual.

3.4.2 Randomizable Signatures. Randomized signatures provide a way to randomize a previously generated signature, transforming it into a new valid signature, but making it impossible to link the two instances together. This enables a level of anonymity by unlinking user identities from their transaction history, as each transaction appears distinct even if performed by the same user. The original idea was proposed by Camenisch and Lysyanskaya [37, 38] but suffered from a linear size in the number of messages to be signed. Later, Pointcheval-Sanders proposed more efficient signature and verification algorithms [115]. Existing works (e.g., [13, 129, 136, 143]) adopt either Pointcheval-Sanders [115] or Coconut [134], a threshold scheme derived from it. The Pointcheval-Sanders scheme has the ability to produce signatures of a message directly from its commitment. This effectively makes it a blind signature, thanks to the hiding property of commitments.

3.4.3 Group Signatures. Group signature schemes [47] enable any member of a designated group to generate a valid signature without revealing who signed (ensuring anonymity). A verifier uses a group public key to confirm that a valid signature was produced by some group member. This property also implies *unlinkability*: it is not possible to determine whether two different signatures were produced by the same user [112]. However, group signatures can be made *linkable* to prevent double spending in transactions (e.g., [147]). Group signatures also support *user traceability*: a designated group manager holds a group secret key that enables him to revoke a signer's anonymity and reveal his identity when necessary.

3.4.4 Ring Signatures. Ring signatures [120] are similar to group signatures, as they hide the signer's identity behind a group. Differently from group signatures, user anonymity is permanent, because there is no revoke mechanism. Indeed, the signer simply joins their public key with a set of other public keys during the signing phase. A verifier needs the whole list of public keys to verify the validity of the signature, and cannot discern which one of the n participants corresponding to the public keys is the signer, but can only guess with probability $1/n$. There are some variant ring signatures, where anonymity is somewhat limited and the signer can be traced under special circumstances, e.g., after double spending. These are called *traceable* ring signatures [65, 66]. Ring signatures can also be *linkable* [92], if they allow anyone to determine if two signatures were produced by the same signer (e.g., [89, 139]).

3.4.5 Threshold Signatures. Threshold signatures are a type of signature that can be jointly produced by a committee of signers. A t-of-n signature scheme requires at least t signers out of a total of n participants to produce a valid signature. Works that use threshold signatures [4, 13, 83, 129] use threshold blind signatures, based on Coconut [134].

3.5 Secure Multi-party Computation

Secure Multi-Party Computation (MPC) allows multiple parties to jointly compute a result over their private inputs without any party having to reveal its own input to the others. The key idea is that the parties collaborate to perform some computation and, at the end, they all learn the correct output of the function, but nothing else. This technique is used, e.g., in CoinParty [150].

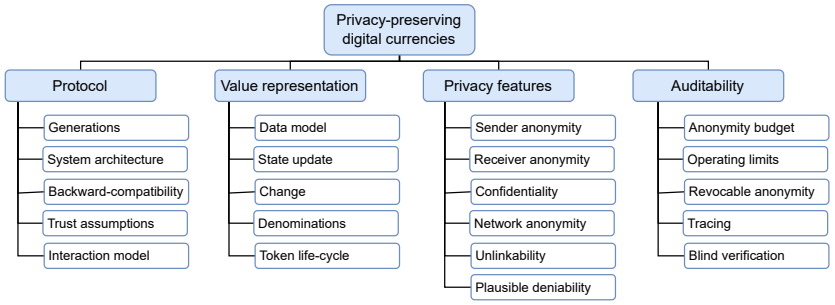

Fig. 1. Dimension of analysis along which we review privacy-preserving digital currencies.

3.6 One-time Public Keys

3.6.1 Stealth Addresses. A stealth address hides the recipient's identity by making each payment unlinkable. It is a one-time public key generated by the transaction sender with public key provided by the transaction receiver. Recipient does not update his public key, but the sender can derive fresh one-time keys using stealth addresses. Let G be a generator of an elliptic curve group $\mathbb{G}$ with scalar field $\mathbb{Z}_q$. Each receiver R has a public view key $V = vG$ and a public spend key $S = sG$, where $v, s \in \mathbb{Z}_q$ are secret scalars. A sender chooses a random scalar $r \in \mathbb{Z}_q$ and computes the shared secret $H = \mathcal{H}_p(rV) \in \mathbb{Z}_q$, with $\mathcal{H}_p(\cdot)$ a hash-to-scalar function, and the one-time public key $P = HG + S$. Then, the sender includes P and $R = rG$ in the transaction. The recipient scans the blockchain and for each P and R computes $P' = \mathcal{H}_p(vR)G + S$. If $P' = P$ then the output belongs to the recipient.

3.6.2 Updatable Public Keys. Updatable public keys allow to update the recipient's public key in a decentralized manner, producing dynamic and evolving public keys. Quisquis [60] introduces this technique to update both sender and recipient public keys. Formally, given a prime-order group $\mathbb{G}_p$ with p the prime order of the group, and G its generator, each user has a secret key s_i and a public key $P_i = (G^r, G^{r \cdot s_i}) = (U, V)$, for some random value $r \in \mathbb{Z}_q$. To perform a transaction, the sender updates the public key by choosing a random value $z \in \mathbb{Z}_q$ and computing $P_i' = (U^z, V^z)$. The updated key is indistinguishable from a fresh key, thanks to the Decisional Diffie-Hellman assumption. The recipient verifies whether a given updated public key P_i' corresponds to his secret key s_i. This works because updated keys still verify correctly with the original secret key.

4 Design Dimensions of Privacy-Preserving Digital Currencies

The research community has explored various solutions to ensure anonymity, confidentiality, and auditability in digital currencies. We examine the most relevant contributions from the literature, analyzing how they address key design challenges. To provide a comprehensive overview of the diverse approaches studied so far, we identify key dimensions of analysis—protocol, value representation, privacy features, and auditability—and highlight the most significant design choices. These interconnected dimensions embed privacy features into the system's design, linking them to the ledger structure, value representation, fund exchange, and auditability. Figure 1 outlines the dimensions that structure this section. Table 2 summarizes key contributions, whereas Table 1 provides a description of the labels used in our analysis.

4.1 Protocol

Over time, different generations of approaches have emerged, each addressing specific features and evolving needs. These developments have been driven by key factors, including the introduction

Table 1. Description of the labels used in Table 2.

Dimension	**Legend**			
Protocol				
Type	CD	CBDC	D	Digital currency
Blockchain Compatibility	B	Bitcoin	E	Ethereum
Value Representation				
Data Model	A	Account-based	CT	Chaumian token
	T2B	Transaction per bank table	UT	Unspent token
Other				
Auditability	AB	Anonymity budget	L	Operating limits
	RA	Revocable anonymity	TR	Coin tracing
	ZD	Verification without disclosure		
Cryptographic Primitives	BS	Blind signature	C	Commitment
	DS	Digital signature	GS	Group signatures
	HC	Homomorphic commitment	IBE	Identity-based encryption
	MS	Multi signature	MT	Merkle tree
	O-RAM	Oblivious RAM	PE	Public key encryption
	RS	Ring signature	SA	Stealth address
	UP	Updatable public keys	ZK	Zero-knowledge proof
	X^t	Threshold-based version of X	X^r	Randomizable version of X
	X^*	Custom variant of X		
Generics	✓	Yes	×	No
	◐	Partial	⇔	Alternative
	○	Abstracted		

Table 2. Classification of existing approaches for privacy-preserving digital currencies (see Table 1 for a legend).

	Protocol							*Value Representation*					*Privacy*								*Other*	
Paper	**Year**	**Type**	**Decentralized**	**Blockchain Compatibility**	**No-Trust Assumptions**	**Non-Interactive**	**Implemented**	**Data Model**	**Change**	**Different Denominations**	**Transferable Token**	**Immediate Claiming**	**Sender Anonymity**	**Receiver Anonymity**	**Transaction Value Confidentiality**	**Network Anonymity**	**Unlinkability**	**Plausible Deniability**	**Auditability**	**Cryptographic Primitives**
Chaum [44]	1983	D	×	×	✓	×	×	CT	×	×	×	✓	✓	×	×	×	×	✓	×	BS
Hayes [75]	1990	D	×	×	×	×	×	CT	×	×	✓	×	◐	×	×	×	×	✓	×	BS*
Sander and Ta-Shma [127]	1999	D	×	×	×	×	×	CT	×	×	×	×	◐	×	×	×	×	✓	RA	MT
Camenisch et al. [36]	2006	D	×	×	×	×	×	CT	×	×	×	×	✓	×	✓	×	×	✓	AB	HC, PE*, ZK
ZeroCoin [102]	2013	D	✓	◐	×	✓	×	UT	✓	×	×	✓	✓	×	×	×	✓	×	×	HC, ZK
CryptoNote [139]	2013	D	✓	⇔	✓	✓	×	UT	✓	✓	×	✓	✓	✓	×	×	✓	×	×	RS, SA
EZC [5]	2014	D	✓	◐	✓	✓	×	UT	✓	✓	×	✓	✓	◐	✓	×	✓	×	×	HC, ZK
ZeroCash [20]	2014	D	✓	◐	×	✓	×	UT	✓	✓	×	✓	✓	✓	✓	×	✓	×	×	C, PE, ZK
Monero [107]	2014	D	✓	⇔	✓	✓	✓[2]	UT	✓	✓	×	✓	✓	✓	✓	✓	✓	✓	×	PE, RS, SA
MimbleWimble [114]	2016	D	✓	⇔	◐	×	×	UT	✓	✓	×	✓	✓	✓	✓	×	◐	✓	×	HC, MS, ZK
Solidus [40]	2017	CD	×	○	×	×	✓	A	✓	✓	×	✓	✓	✓	✓	○	✓	✓	RA	PE, O-RAM, ZK
GNU Taler [53]	2017	D	×	×	×	×	×	CT	✓	✓	×	✓	✓	×	×	×	✓	✓	×	BS
Garman et al. [67]	2017	D	✓	◐	×	✓	×	UT	✓	✓	×	✓	✓	✓	✓	×	✓	×	L,RA,TR	DS, PE, ZK
zkLedger [105]	2018	CD	✓	⇔	✓	✓	✓	T2B	✓	✓	×	✓	✓	✓	✓	×	✓	✓	ZD	HC, PE
Quisquis [60]	2019	D	✓	⇔	✓	✓	×	A	✓	✓	×	✓	✓	✓	✓	×	✓	✓	×	HC, UP, ZK
Traceable Monero [89]	2019	D	✓	×	✓	✓	✓	UT	✓	✓	×	✓	✓	✓	✓	✓	✓	✓	RA	PE, RS, SA
PRCash [143]	2019	D	◐	⇔	×	×	✓	UT	✓	✓	×	✓	✓	✓	✓	×	◐	✓	AB	DS^r, HC, ZK
Zhang et al. [147]	2019	CD	×	⇔	×	✓	✓	A	✓	✓	×	✓	◐	◐	×	×	◐	✓	RA	GS
Androulaki et al. [4]	2020	D	✓	⇔	×	×	×	UT	✓	✓	×	✓	◐	◐	◐	×	◐	✓	RA	BS^t, HC, PE, ZK
Zether [32]	2020	D	×	✓ (E)	✓	✓	✓	A	✓	✓	×	✓	✓	✓	✓	×	✓	×	×	HE, ZK
PGC [49]	2020	D	✓	⇔	×	✓	✓[3]	A	✓	✓	×	✓	×	×	◐	×	×	✓	RA	DS, PE, ZK
DCAP [91]	2020	D	✓	⇔	×	×	◐[4]	A	×	✓	×	✓	◐	◐	×	×	◐	×	RA	ZK
Gross et al. [72]	2021	CD	✓	×	×	×	×	A	✓	✓	×	✓	✓	✓	✓	×	✓	×	L	C, ZK
DSC [97]	2021	D	✓	⇔	✓	✓	×	A	✓	✓	×	✓	×	×	✓	×	×	×	×	HE, ZK
Zef [13]	2022	D	✓	✓ (E)	✓	×	✓[5]	A	✓	✓	×	✓	✓	✓	✓	○	✓	×	×	BS^r, C, DS^t, ZK, SA

(continued on next page)

[2] https://github.com/monero-project/monero
[3] https://github.com/yuchen1024/Kunlun/tree/master/adcp
[4] https://github.com/colyn91/-Smart-Contract-on-DCAP
[5] https://github.com/novifinancial/fastpay/tree/extensions

Table 2. Classification of existing approaches for privacy-preserving digital currencies (see Table 1 for a legend).

	Protocol							*Value Representation*						*Privacy*								*Other*	
Paper	**Year**	**Type**	**Decentralized**	**Blockchain Compatibility**	**No-Trust Assumptions**	**Non-Interactive**	**Implemented**	**Data Model**	**Change**	**Different Denominations**	**Transferable Token**	**Immediate Claiming**	**Sender Anonymity**	**Receiver Anonymity**	**Transaction Value Confidentiality**	**Network Anonymity**	**Unlinkability**	**Plausible Deniability**	**Auditability**	**Cryptographic Primitives**
BlockMaze [74]	2022	D	✓	⇔	✓	×	✓[6]	A	✓	✓	×	✓	✓	✓	✓	×	✓	×	×	C, ZK
PEReDi [83]	2022	CD	✓	×	×	×	×	A	✓	✓	×	✓	✓	✓	✓	×	✓	✓	RA	C, ZK, BS^t, PE^t
AT-CBDC [94]	2022	CD	✓	⇔	×	×	✓	A	✓	✓	×	✓	✓	✓	×	×	×	✓	RA	BS, ZK
UTT [136]	2022	CD	✓	×	×[7]	✓	✓[8]	CT	✓	✓	×	✓	✓	✓	✓	○	✓	✓	AB	$DS^{t,r}$, IBE^t, ZK
Platypus [144]	2022	CD	×	×	✓	×	✓	A	✓	✓	×	✓	✓	✓	✓	○	✓	✓	L	C, DS, ZK
ERCE [145]	2022	D	✓	⇔	×	✓	◐	UT	✓	✓	×	✓	◐	◐	✓	×	◐	×	L,RA	C, MT, ZK
Bank of Australia [118]	2023	CD	×	✓ (E)	×	✓	×	A	✓	✓	✓	✓	×	×	✓	×	×	✓	×	PE
BIS Tourbillon [23]	2023	CD	×	×	✓	×	×	CT/UT	×	✓[9]	×	✓	✓	×	×	×	◐	✓	×	BS
KAIME [52]	2023	CD	✓	×	✓	✓	✓[10]	A	✓	✓	×	✓	◐	◐	✓	×	×	×	RA	HC,PE^t, ZK
Goodell et al. [71]	2023	CD	✓	⇔	✓	×	×	CT	◐	×	✓	×	✓	×	×	×	×	✓	×	BS, PE
PayOff [18]	2024	CD	×	×	×	✓	✓	A	✓	✓	✓	×	✓	✓	✓	×	◐	✓	RA	C, ZK
PCRAP [56]	2024	D	✓	⇔	×	×	×	UT	✓	✓	×	✓	✓	✓	✓	×	✓	✓	×	HC, MT, RS, SA
Friolo et al. [64]	2024	D	✓	×	×	×	×	CT	×	×	×	✓	✓	✓	×	×	✓	✓	×	PE, ZK
AQQUA [109]	2024	D	◐	⇔	×	✓	×	A	✓	✓	×	✓	✓	✓	✓	×	✓	✓	L	HC, UPK, ZK
Yu et al. [146]	2024	D	×	◐	×	✓	×	UT	×	×	×	✓	◐	×	×	×	◐	×	RA	HC, PE
PARScoin [129]	2025	D	✓	✓ (E)	✓	✓	×	A	✓	✓	×	×	✓	✓	✓	×	×	✓	×	$BS^{t,r}$, HC, PE

[6] https://github.com/Agzs/BlockMaze
[7] Threshold encryption relaxes trust requirements from a single entity to a quorum of entities.
[8] https://github.com/definitelyNotFBI/utt
[9] Denominations in power of 2
[10] https://github.com/midmotor/kaime_cbdc_proof_test

of blockchain technology, advancements in cryptographic primitives, and the growing interest in designing CBDCs. The *protocol* dimension provides a broad perspective on the diverse range of existing solutions.

4.1.1 Generations of Privacy-preserving Digital Currencies. The term *digital currency* encompasses a range of different forms of money that is managed, stored, and exchanged on digital computer systems over the Internet[11]. Cryptocurrency and CBDC represent two specific forms of digital currency. A cryptocurrency is defined, stored, and exchanged over *peer-to-peer* systems, such as the blockchains. A CBDC is digital currency issued by a Central Bank [17], where the term is often used to indicate that the currency generally does not have a classical physical form of a fiat currency. We identify three main generations of solutions for privacy-preserving digital currencies.

First Generation. The *first generation* focuses on the design of digital currencies leveraging digital signatures and a centralized authority (i.e., the issuing bank) [44]. The fundamental problem relates to guaranteeing anonymity of sender (i.e., the payer) although he must interact with the central bank to obtain valid tokens (representing coins) to perform subsequent payments. Chaum [44] proposes one of the first attempts to construct a digital currency, commonly referred to as *ecash*. Later, several works improved ecash with different features, including support for different coin denominations (e.g., [53]), partial spending (e.g., [53]), token transferability (e.g., [11, 14, 42, 75, 125]), offline spending (e.g., [45]), and auditability (e.g., [36, 53]).

Second Generation. The *second generation* of works emerged with the introduction of Bitcoin [104]. Bitcoin and, later, Ethereum [34] propose public and permissionless blockchains, meaning that the ledger contains transactions visible to everyone and anyone can participate in the process of updating the public ledger by following a specific (validation and consensus) protocol. Being public, these blockchains use addresses (acting as pseudonyms) to exchange money in a pseudo-anonymous manner, as addresses are not directly tied to user identities. Public blockchains cannot easily and fully satisfy two privacy-related features: untraceability and unlinkability [117]. *Untraceability* ensures that, for any given incoming transaction, all potential senders are equally likely. *Unlinkability* means that, for any two outgoing transactions, there is no way to prove they were sent to the same recipient. While generating a new address for each transaction can enhance these properties, various studies indicate that achieving true anonymity and unlinkability on public blockchains remains challenging. An analysis of the transaction graph can reveal clusters of pseudonyms or patterns that link with high probability two or more addresses (e.g., [78, 81, 101, 103]). The second generation of solutions mainly focuses on improving unlinkability, anonymity, and confidentiality, often considering decentralized architectures (e.g., [6, 13, 60, 64, 77, 100, 102, 129, 139]).

We classify this second generation of approaches into three main categories: privacy-native protocols, mixing protocols, and off-ledger solutions. Changing the protocol of existing blockchains is hard, as it involves forks. Therefore, *privacy-native protocols* define (mostly) novel currencies with improved confidentiality, anonymity, and unlinkability features (e.g., [20, 107, 136, 139]). Conversely, mixing protocols propose supplementary services for permissionless blockchains aimed to collect a group of transactions, scramble senders and receivers, and publish them as a single group transaction (e.g., [24, 30, 98, 116, 122, 138, 142, 150]). Mixing protocols break the link between sender and receiver, thus reducing the possibility of linking together addresses of the blockchain (i.e., they enhance unlinkability of transactions). Off-ledger solutions are a specific approach related to blockchains (which serve as Layer 1) and the ability to build decentralized peer-to-peer Layer 2 systems on top of them. Layer 2 solutions process transactions off the main

[11] https://en.wikipedia.org/wiki/Digital_currency

chain, reducing the need to record every transaction on the shared ledger (e.g., [13, 116, 130, 142]). Since we focus on on-ledger privacy, we omit a detailed analysis of Layer 2 privacy solutions.

Third Generation. While protecting the privacy and anonymity of payers and payees remains essential, the *third generation* of privacy-preserving digital currencies introduces mechanisms that balance these guarantees with auditability and regulatory compliance (e.g., [4, 18, 40, 49, 67, 83, 89, 105, 109, 143–147]). These approaches are particularly motivated by concerns around AML and CFT. Auditability refers to the system's ability to track, verify, or selectively disclose information about transactions, without fully compromising user privacy. This capability enables the enforcement of specific policies that apply to users, coins, or transactions behaviors over time. To achieve this, modern privacy-preserving systems integrate mechanisms such as revocable anonymity, trapdoor-based decryption, and ZKPs for correctness and compliance.

4.1.2 System Architecture. We examine the system architectures underlying privacy-preserving digital currencies and mixing protocols, which enhance transaction unlinkability.

Centralized Systems. The first generation of privacy preserving currencies typically rely on *centralized architectures* involving up to three or four trusted authorities, each responsible for distinct functions. They include: (i) a central bank, which issues digital coins and maintains monetary control; (ii) a tracing authority, responsible for auditing and tracing under specific conditions; (iii) a registration authority, in charge of user on-boarding and identity verification; and, in some cases, (iv) a technical support entity, which collaborates with the tracing authority to enable conditional de-anonymization. Most early proposals assume a single central bank (e.g., [18, 40, 44, 53, 147]). For privacy, special cryptographic primitives are used, such as blind signatures [44], which allow the bank to issue valid coins without learning user identities. To support auditability and regulatory compliance, systems often include a separate tracing authority that can decrypt transactions or enforce specific policies (e.g., check or bypass an anonymity budget). This entity operates independently from the central bank, thereby preserving some separation of duties (e.g., [127, 144]). Yu et al. [146] introduce a technical support authority that must cooperate with the tracing authority to extract sender identities. In the context of CBDCs, system architectures usually include a registration authority, responsible for enrolling user registration and managing identity records (e.g., [147])—an essential requirement considering regulatory frameworks (e.g., AML and CFT).

Decentralized Systems. Blockchains introduced a fundamental shift in the design of digital currencies by leveraging decentralization and cryptographic protocols. Decentralization serves two main purposes. First, it enhances resilience against malicious actors or censorship by distributing trust across multiple entities. This is often supported by cryptographic techniques such as threshold encryption, secret sharing, and multi-signature schemes. Second, it enables intermediary-free value exchange, using commitments and digital signatures to ensure integrity, authenticity, and non-repudiation. Many privacy-native solutions adopt decentralized architectures in line with these principles (e.g., [4, 97, 105, 107, 114, 139, 145]). Moreover, the programmability of blockchains has enabled privacy-preserving overlays (e.g., [20, 102]) and smart contracts[12] implementing private tokens "pegged" to the coin of the primary chain (e.g., [13, 32, 129]).

Architecture of Mixing Protocols. Different architectural styles have been explored for mixing protocols, including both centralized (e.g., [30, 76, 93, 116, 138]) and decentralized solutions (e.g., [24, 108, 122, 123, 142, 150]). In centralized solutions, a third party (often called *mixer* or *tumbler*) provides the mixing service for a fee. While such designs are simple and support large

[12] Although smart contracts are logically centralized, they are executed on each node; therefore, we consider smart contracts as a decentralized solution.

anonymity sets, they introduce risks such as availability, theft, and potential privacy breaches. Accordingly, different trust models have been proposed to mitigate these concerns (see Section 4.1.4). Decentralized mixers remove the need for a trusted party by using cryptographic protocols or peer-to-peer coordination, enhancing security and censorship resistance, but they often incur higher complexity, require user coordination, and typically support smaller anonymity sets. Notable examples include Xim [24], CoinShuffle++ [124], and ValueShuffle [122]. Both centralized and decentralized approaches face challenges such as Sybil attacks and regulatory scrutiny. In a Sybil attack, an adversary inject fake participants into the mix to degrade anonymity.

4.1.3 Protocol Integration and Compatibility with Existing Blockchains. Although based on blockchain principles, most second-generation privacy-preserving cryptocurrencies are not fully compatible with baseline systems. We analyze how they interact with existing digital currencies, whether through compatibility, non-compatible extensions, or entirely different approaches.

Compatible Solutions. Mixing protocols propose a supplementary services for existing blockchains, in particular for Bitcoin, and are specifically designed to be fully compatible (e.g., [116, 122, 124, 138]). Another category of compatible solutions leverages blockchain programmability to introduce privacy-enhanced currencies through overlays (e.g., [13, 32]). For instance, Zether [32] employs a smart contract on Ethereum to store and manage private accounts, whereas Zef [13] introduces opaque coins in the form of off-chain certificates exchanged between Ethereum user accounts.

Non-compatible Extensions. Most privacy-enhancing currency proposals result in non-compatible extensions of existing blockchain protocols. For example, Zerocoin [102] is an overlay over Bitcoin that allows user to convert bitcoins in zerocoins. A zerocoin can be proved (using ZK techniques) to originate from a valid and unspent bitcoin, without revealing the link to the original bitcoin. Zerocoin does not provide full anonymity: it obscures coin traceability, but it does not hide the number of transactions, recipient addresses, and balances. Building on Zerocoin, Zerocash [20] enhances privacy by concealing both sender and recipient addresses, supporting arbitrary transaction values, and enabling change handling. Both Zerocoin and Zerocash require new transaction types and payment semantics, and break compatibility with Bitcoin. Garman et al. [67] extends Zerocash with no backward compatibility, by introducing policy enforcement, selective user tracing, and tracing of tainted coins—features that further diverge from Bitcoin's existing design. Yu et al. [146] propose an overlay on Bitcoin for anonymous yet regulated transactions using a coin backed by bitcoins. Their approach relies on three centralized entities that cooperate to enable batch linkability and payer tracing while preserving privacy. These changes are not backward-compatible with Bitcoin.

Novel Approaches. Most approaches present greenfield solutions for privacy-native cryptocurrencies (e.g., [4, 49, 60, 105, 107, 114, 139, 143, 147]), often extending open-source solutions (e.g., [89, 109]). Both centralized (e.g., [40, 147]) and decentralized architectures (e.g., [4, 56, 71, 91, 97, 105]) for privacy-native solutions have been proposed, with a sensible preference for the latter. Early approaches like CryptoNote [139], MimbleWimble [114], and Monero [107] were among the first to address Bitcoin's privacy limitations by still proposing fully decentralized alternatives. CryptoNote [139] enhances sender and receiver anonymity through one-time ring signatures and NIZK proofs, inspiring cryptocurrencies like ByteCoin, DigitalNote, and Aeon [3]. MimbleWimble [114] improves privacy and scalability by aggregating confidential transactions instead of storing individual transaction data. Cryptocurrencies building on MimbleWimble include Grin[13]

[13] https://grin.mw/

and Beam[14]. Monero [107] strengthens privacy with stealth addresses, ring signatures, and confidential transactions, while also integrating network anonymity protocols.

Other approaches propose partially decentralized solutions (e.g., [143]), combine concepts from centralized mixers (e.g., [60]), or consider hierarchical bank-intermediated settings (e.g., [40, 105]). Wüst et al. [143] propose PRCash to strike a trade-off between performance (fast payments), user privacy, and regulatory oversight. Basically, PRCash includes two centralized entities, which respectively issue money and oversee compliance, and a permissioned blockchain, which records MimbleWimble-like transactions. Quisquis [60] builds on the concept of mixing to *redistribute wealth*: each transaction updates a bunch of other accounts on chain, thus preventing addresses from appearing multiple times on the blockchain, reducing the risk of de-anonymization and linkability. Quisquis ensures integrity through ZKPs while maintaining privacy using commitments and updatable public keys. Cecchetti et al. [40] propose Solidus, a hierarchical, bank-intermediated systems where a small number of banks manage transactions of on-chain assets on behalf of a large number of users. This setup ensures transaction-graph confidentiality: sender and receiver of a transaction cannot be publicly identified[15], even by pseudonyms.

4.1.4 Trust Assumptions. We now focus on the degree of trust required, such as reliance on a trusted third party, to facilitate secure transactions. Most solutions for digital and cryptocurrencies do not usually rely on trusted parties for confidentiality (e.g., [13, 60, 74, 89, 97, 98, 105, 107, 108, 129, 136, 139, 142]), even when they propose a centralized system (e.g., [44, 46, 144]).

The solutions that introduce trust assumptions do it for different purposes. The approaches that use zk-SNARK as ZKP require a trusted setup to initialize and correctly share some cryptographic parameters (e.g., [20, 67, 102, 145]). A few works require a trusted central bank to issue coins that can be later exchanged anonymously (e.g., [64, 71]). Elfadul et al. [56] assume the presence of a regulatory supervision that grants permissions to users to perform transaction; this centralized authority could asymmetrically revoke such authorization. Contributions that explore special forms of auditability related to revocable anonymity (e.g. [91, 143, 145, 146]), trapdoor (e.g., [4, 49, 56, 147]), or user tracing (e.g., [67]) require trust in the tracing authority that uncovers transaction details only when necessary. PEReDi [83] reduces trust requirements by relying on quorum of centralized entities that must agree to perform specific operations, such as decrypting transaction information.

As regards centralized mixing protocols, since the third party represents a single point of failure, different protocols focused on different threat models, where the mixer may be trusted and accountable (e.g., [30, 138]) or untrusted (e.g., [76]). For instance, Mixcoin offers a signed warranty that will enable his customers to unambiguously prove if the mix has misbehaved [30]. Conversely, Tumblebit [76] prevents coin thief using an on-chain escrow and ZKPs. The rise of off-chain solutions has let to the exploration of alternative mixer definitions. For instance, BlindHub [116] implements mixing within payment channel hubs, where the hub acts as the tumbler. It employs blind signature-based protocols to hide users' addresses from the hub, ensuring privacy.

4.1.5 Interaction Models. Privacy-preserving digital currencies differ in the extent to which transactions require direct or indirect interaction between the payer and the payee. *Interactive payment protocols* require active cooperation between payer and payee during the transaction process. This is typical in many account-based models where the recipient must be involved in constructing or approving the updated account state, often through ZKPs or digital signatures. For instance, PEReDi [83] and Platypus [144] enforce synchronized interaction: both parties participate in updating the ledger with proofs and commitments, ensuring mutual accountability and preventing

[14] https://www.beam.mw/

[15] While banks can identify their respective clients, other entities only learn the identities of the banks.

attacks such as dusting or replay. Similarly, Chaumian token-based systems often require the payee to claim tokens immediately after receipt to prevent double spending (e.g., [53]). In contrast, *non-interactive payment protocols* allow the payer to unilaterally construct a transaction and broadcast it to the network. Many token-based systems, especially those based on unspent tokens, adopt this strategy (e.g., [20, 107, 114, 139]). This model improves user convenience and enables asynchronous payments, but may require periodic ledger scanning or local computation by the receiver[16]. Some account-based protocols implement delayed claiming to approximate this non-interactive behavior. Sarencheh et al. [129] implement a non-interactive variant of PEReDi, which works in two stages. In the transfer stage, the sender interacts with a set of issuers to update his account and provide information in ZK. The claim stage credits the receiver account, exploiting information enclosed in ZKP by the sender. While interactive designs enable rich policy enforcement and conditional privacy, non-interactive models better suit high-latency or asynchronous networks.

4.2 Value Representation

The *value representation* dimension examines the approaches used to represent the currency, the user balance, and the exchange of value. For the purpose of this survey, we investigate these aspects with a specific focus on their relation with privacy-related features.

4.2.1 Ledger Data Models. Digital currencies and cryptocurrencies follow three main approaches for representing and tracking ownership of digital assets: accounts, tokens, and tabular ledger.

Account-based. Account-based ledgers associate a balance with each account address, which is debited or credited by transactions. This simplifies state management and enables simple definitions of policies based on cumulative balances. Nonetheless, it requires safeguards against replay and double-spending attacks, often implemented by storing a sequence number (or *nonce*) in each account and updating it with every transaction. For example, Ethereum [34] associates to each account[17] a nonce (indicating the number of sent transactions), a balance, and two hashes referencing the account's storage contents and associated code. Since the balance of an account is explicitly represented, obtaining anonymity and unlinkability is challenging, especially in systems where users cannot freely generate multiple addresses (e.g., [74]). Privacy-preserving account-based solutions replace plain-text balances with commitments, achieving *confidentiality* of balances and transactions (e.g., [13, 32, 60, 74, 129]). Liu et al. [94] propose an account-based CBDC with sender and receiver anonymity but with no unlinkability. The system follows a two-tier model where the central bank can identify users via encrypted identifiers, while commercial banks cannot. To guarantee anonymity while ensuring *unlinkability*, most of the approaches change users' account address at each transaction and use an authenticator to prove correctness. The authenticator can be implemented, e.g., using digital signatures by trusted or centralized authorities (e.g., [83]) or ZKPs (e.g., [32, 60, 109, 144]). Platypus [144] rely on ZKPs; conversely, in PEReDi [83], account states must be (blindly) signed by a quorum of maintainers, and since each transaction reveals prior account snapshots to construct new authenticators, randomized signatures are used to prevent linkability. Quisquis [60] and AQQUA [109] use updatable public keys to update accounts in a decentralized manner. The account-based model lends itself well to *auditability*, as it allows simple definition of policies on holding limits (e.g., [72, 109, 144]) or spending and receiving limits (e.g., [109]). For example, the account state by Gross et al. [72] includes an accumulator of all spending in the current epoch, enabling policies to rely on this information. AQQUA [109], instead, includes

[16] Off-chain communication can alleviate the burden of scanning the ledger, but current protocols lack this feature.
[17] Ethereum also defines standards to represent fungible tokens (ERC-20) and non-fungible tokens (ERC-721). ERC-20 can be used to design privacy-preserving currencies (e.g., [32]).

commitments to three different counters (i.e., balance, total amounts sent and received). Conversely, Platypus [144] leaves the specification of the additional information to be application dependent.

Token-based. Token-based ledgers do not maintain explicit balances per address, but instead track individual tokens. The user's balance can be derived from the sum of all owned tokens. This model enables parallel validation of user transactions and stronger privacy guarantees (as balances are not explicitly materialized) but introduces management complexity, particularly when users distribute their coins across multiple addresses for added privacy. There are two main types of token structures: Chaumian tokens and unspent tokens. Although both represent digital value, they differ in structure, spending mechanisms, privacy guarantees, and centralization.

Chaumian tokens [44] use blind signatures and were first deployed in centralized, bank-issued electronic cash schemes. When issuing a token, the bank blindly signs a random value provided by the payer, allowing the payer to later unblind it for spending. Blind signatures prevent the bank from linking withdrawals to subsequent transactions, thus ensuring *sender anonymity*. These tokens rely on a trusted authority to issue and verify their validity; to this end, the authority must hold an ever increasing list of spent tokens that might introduce scalability challenges. Some implementations introduce expiration to limit circulation (e.g., [53, 95]). The Chaumian model inspired decentralized privacy-preserving cryptocurrencies (e.g., Monero [107], Zerocoin [102], and Zerocash [20]), which eliminate the need for a central issuer by leveraging a combination of different cryptographic techniques. For example, Zerocash supports transparent (like Bitcoin) and shielded transactions, with the latter hiding sender, recipient, and amount. Shielded payments encrypt these values with one-time keys and use zk-SNARK proofs to validate inputs-outputs equality, concealing all on-chain details.

Unspent tokens, introduced in Bitcoin [104], represent transaction outputs that can be used as inputs for future transactions, which in turn create new unspent tokens. As such, they are inherently traceable unless PETs are used [10]. Unspent tokens are issued in a decentralized manner, usually as result of the mining process. Their core attributes include an identifier, value (or its commitment), and ownership information. Their recording in a public, transparent ledger prevents double-spending but complicates privacy. Privacy-focused cryptocurrencies employ stealth addresses, ring signatures, confidential transactions, and ZKPs to *obscure* transaction details and *unlink* payments (e.g., [20, 102, 107, 114])—these properties are discussed in more detail later. Tokens can also include auxiliary data to support the evaluation of specific *policies* (e.g., [145, 146]). For example, Xue et al. [145] include information to prove in ZK that the total amount of the cryptocurrency transferred and the transaction frequency in a time period is limited. For a detailed discussion of auditability in token-based systems, see Section 4.4.

Tabular Ledger. Narula et al. [105] propose zkLedger, a third approach tailored to a specific use case: a bank-intermediated CBDC. It uses a tabular ledger where rows represent transactions and columns represent banks. Each cell contains a Pedersen commitment to hide values, while enabling cryptographic verification. Transactions include ZKPs to prove balance, asset ownership, and consistency with audit records. This structure supports rich privacy-preserving *auditing*: the system includes an auditor who can issue a rich set of auditing queries—using primitives like sums, moving averages, variance, standard deviation, ratios—and receive answers that are provably consistent with the ledger. As downside, this structure does not scale well with many banks and transactions, making it suitable for small, privacy-focused networks.

4.2.2 Ledger State Update. Updating the ledger is another operation that may reveal sensitive information to other parties. We review the approaches for account-based and token-based ledgers.

Account-based. A generic state update transaction for account-based ledger includes the sender and receiver addresses, the exchanged amount, and the sender digital signature. For privacy-enhanced account-based, a naïve state update would reveal the updated accounts. Therefore, privacy-preserving state updates in account-based ledger usually adopt three strategies (e.g., [32, 60, 83, 109, 144]): to obtain confidentiality, they update a commitment of the new account state; to obtain unlinkability, they always update the public key associated with the (recipient) account; and to obtain anonymity, they update a group of accounts at the same time.

Token-based. Token-based ledgers store the list of valid (unspent) tokens or burnt (spent) tokens. A spending operation must invalidate an existing unspent token, to prevent double-spending. With Chaumian tokens (e.g., [44, 53, 95]), payment systems include a withdrawal operation, which creates a new valid token that the payer can spend later in time, and a spend operation. The state of the system is represented by the list of spent tokens, which is updated at spending time; unspent tokens are usually not explicitly stored (if not by payers). Privacy-preserving systems based on Chaumian tokens do not usually guarantee transaction confidentiality (e.g., [44, 53, 64, 71, 75, 127]), therefore they still maintain the list of spent tokens. Unlike other approaches, Sander and Ta-Shma [127] represent valid tokens in a Merkle tree. The bank does not blind sign tokens; instead, valid tokens come with a hash chain proving membership in the Merkle tree. The Merkle root is publicly available to ensure transparency and auditability. Withdrawal coins can be invalidated if needed, addressing certain attack scenarios (e.g., blackmail). Friolo et al. [64] employ a ZKP-based protocol to achieve *unlinkability*. Transactions are represented as commitments, and when spent, their opening is revealed along with a ZKP demonstrating that it is linked to a valid, unspent token. In contrast, UTT [136] ensures *transaction confidentiality*, and it recurs to *nullifiers* to track spent coins. Nullifiers enable the bank to detect double-spending without knowing transaction specifics. In [136], a nullifier is a pseudo-random value computed from the coin's serial number (and a secret). The system checks their validity through ZKPs and, if valid, stores them in a list for future checks.

With unspent tokens, the payment operation simultaneously invalidates previous tokens and generate new ones. The state of the system is represented by the list of unspent tokens[18]. Unspent tokens cannot be easily marked as spent without compromising *sender's anonymity*. Most of the approaches, including CryptoNote [139], Zerocoin [102], Zerocash [20], and their extensions (e.g., [5, 67, 145]), let the payer spend a token by using a serial number and a ZKP that indicates that the number is a valid opening to some commitment stored in an unspent token. Monero [107] (and its extension Traceable Monero [89]) uses one-time signatures for unspent tokens, which can be used only once to spend the token. At spending time, the ledger registers a salted hash of the public key. MimbleWimble (and PRCash that reuses its transaction format [143]) eliminates the idea of performing transactions between addresses; instead, it uses a binding and hiding commitment scheme [110], together with Bulletproof range proofs [33]. By using commitments as outputs, coins act as having their own private key, the knowledge of which, along with the knowledge of their value, would enable spending the funds. This guarantees *anonymity* and *unlinkability*.

4.2.3 Handling Change. Account-based ledgers use transactions to transfer exact value between payer and payee, therefore these kind of systems do not need to support change.

Instead, token-based ledgers propose different approaches for handling change. Systems relying on Chaumian tokens do not easily support change or partial spending (e.g., [53]). Tomescu et al. [136] allow multiple inputs and outputs for a single transaction, but requires the payer to prove *value preservation* in ZK. The proof consists in two parts: the first proves that the sum of inputs equals

[18]Since blockchains store transactions in an append-only log, they trace the derivation history of each unspent token. This helps implement a decentralize yet scalable system.

the sum of outputs; the second is a range proof on the output value (needed because ZKPs works with modulo arithmetic and the equality proof is not enough to guarantee value preservation). System based on unspent tokens inherit the change implementation strategy by Bitcoin, where the change is another unspent token owned by a (new) address managed by the payer (e.g., [4, 56, 102]) or a commitment controlled by the payer (e.g., [107, 143]).

4.2.4 Support for Different Denominations. Chaum [44] considered a simple system where tokens have implicit value. A few extensions consider the possibility to emit tokens with different denominations (e.g., [53, 136]). For example, Dold [53] in GNU Taler use multiple denomination key pairs for blind-signing coins of different monetary values. Conversely, Tomescu et al. [136] explicitly represent the token value by concealing it within a commitment. As regards unspent token-based ledgers, Zerocoin does not support arbitrary denominations [102]. This design simplifies the ZKP used for anonymity but introduces usability limitations, as users must split or combine coins when making payments. Zerocash [20] supports arbitrary denominations by using homomorphic commitments, allowing users to transact any amount while preserving privacy. The remaining approaches allow transacting tokens of different denominations, as each unspent token internally holds its value (or a commitment to it). Among them, CryptoNote [139] does not hide transaction amounts, making it vulnerable to statistical analysis—e.g., identifying change as the smaller output. Using multiple change addresses can obscure amounts but leads to issues like proliferation of *dust transactions* [107]. Hiding values with commitments prevents such analysis.

4.2.5 Token Life-cycle.

Double Spending Protection. Two main approaches to double spending protection emerge: deposit-time checks and publicly verifiable ledgers. Centralized systems usually relying on Chaumian tokens do not provide a transparent ledger or the primitives for proactive verification of double spending for the payee. Instead, they check token validity at deposit time, i.e., when the recipient interact with the bank to deposit the token (e.g., [44, 53, 71, 136]). The public visibility of the ledger in blockchains (e.g., [104]) allows payee to autonomously check whether an received token has been already spent. This validity check is also possible in privacy preserving systems, which usually require the payer to provide a ZKP of token validity (e.g., [89, 102, 107, 114, 145, 146]).

Transferable Tokens. Only a few works (e.g., [71, 75]) explore mechanisms for transferring Chaumian tokens, i.e., enabling safe ownership changes. Hayes [75] proposes a system where each coin owner signs a chain of transactions to ensure traceability. For example, if user A receives a token from the bank, he can pay another user B by issuing a token for B and attaching the token A received from the bank. When the token chain becomes too long, the owner can exchange it for a fresh token via the bank. However, this approach is vulnerable to double spending, which is mitigated by imposing limited token lifetimes. Similarly, Goodell et al. [71] define tokens that carry a *proof of provenance*: it verifies validity without requiring external interaction. This proof is a list of token commitments including the initial commitment and signature by the issuer.

Immediate Claiming. With Chaumian tokens, the payer-payee interaction takes place off-the-ledger; therefore, the payee should be online to deposit the token, after a validity check by the token issuer. This kind of interaction requires the payee to immediately claim the token to prevent double spending attacks [44]. One-time signatures can be used to reveal misbehaving senders' identity, enabling revocation and blacklisting [75]. Similarly, in [36], spending a token multiple times reveals the sender's identity. Conversely, Sander and Ta-Shma [127] avoids relying on immediate claiming of tokens by using Merkle hash trees to publicly authenticate coin validity through inclusion proofs, rather than secret signatures. Coins remain valid as long as users maintain updated hash chains to

public tree roots, which are periodically broadcast. This enables offline, asynchronous payments where coins can be spent without real-time interaction with the bank.

4.3 Privacy Features

We delve into the details of the technical solutions used to obtain the different privacy goals. Privacy in digital currencies is often defined through different sub-properties focusing on anonymity of participants, value confidentiality, plausible deniability, and unlinkability (e.g., [1, 3, 22, 68]).

4.3.1 Sender Anonymity. *Sender anonymity* relates to the protection of the payer's identity, i.e., the transaction sender. Some work implement *asymmetric anonymity*, where only one party remains anonymous (e.g., [5, 23, 44, 53, 71, 102, 127, 146, 147]). The main techniques for achieving sender anonymity are: blind signatures (e.g., [44, 53, 71, 94]) and their variants—one-time (e.g., [75]), threshold (e.g., [4, 83, 129]), and randomizable (e.g., [136, 143])—, ring signatures (e.g., [56, 107, 139]), group signatures (e.g., [147]), and commitments with ZKPs (e.g., [20, 102]).

Chaum [44] recurs to *blind signature* to unlink the sender request for signing a token and its later spending. To obtain revocable anonymity and reveal the sender on double spending, Hayes [75] introduces a *one-time blind signature* schemes. However, since each transaction requires a new signing key, the storage and computation management is cumbersome; moreover, one-time signatures often require large signature size and expensive verification, making them impractical for high-throughput systems. UTT [136] realizes blind signatures using *randomizable signatures* by Pointcheval and Sanders [115], which allow signatures to be altered using fresh randomness without compromising their validity. All these schemes rely on a central issuer, who can represent a single point of failure. This can be mitigated, e.g., by using *threshold blind signatures*, where a quorum of authorities must cooperate to create a valid blind signature (e.g., [4, 83, 128]).

Other works reuse the idea of mixing protocols through the concept of *ring signatures*, which conceals the sender's public key—hence, his identity—in a group of other users' public keys (e.g., [56, 89, 107, 139]); this group is referred to as *anonymity set*. For example, CryptoNote [139] uses one-time (linkable) ring signatures for sender anonymity. For each transaction, the sender generates a one-time key and derives a *key image*, then selects an anonymity set and produces a ring signature including the key image and a verification challenge. The key image is stored on a public ledger managed by a centralized entity to detect double-spending. Verifying a signature requires the public keys of all users in the ring; thus, larger anonymity sets increase privacy but also transaction size. Different privacy-preserving cryptocurrency systems use ring signatures for obtaining sender's anonymity, including Monero[19] [107], Traceable Monero [89], and PCRAP [56]. However, studies show that Monero's anonymity can be compromised through transaction analysis (e.g., [81, 86, 103]).

Considering the requirement of revoking sender's anonymity if needed, Zhang et al. [147] use *group signatures* where a trusted centralized entity sets up the anonymity set and can, optionally, uncover the senders' identity for inspection. Differently from ring signatures where the anonymity set is dynamic and can change for each transaction, group signatures have a less flexible management.

To be valid, unspent tokens require proving their ownership. To implement sender's anonymity, existing approaches rely on two key tricks. First, the unspent token includes a commitment to its serial number; the serial number is revealed only to mark the token as spent (see Section 4.2.5). Second, the spending operation includes a ZKP that demonstrates knowledge of the opening to a commitment of an unspent coin. Sander and Ta-Shma [127] use multiple Merkle trees to track valid Chaumian tokens. To prove validity, the sender provides a NIZK proof showing that he knows a valid coin in the tree and that the coin's serial number and a hash chain correctly lead to one of the

[19] Möser et al. [103] showed weaknesses in Monero's key sampling: 66.09% of transactions do not include any other public key, sampling is not really random, and real inputs can be inferred with 80% accuracy.

live Merkle tree roots. Zerocoin [102], and its extensions (e.g., [20, 67]), adopts ZKPs as well. In these systems, the payer publishes a transaction that includes the recipient's address, an empty origin address, a ZKP, and the commitment opening. The ZKP prevents linking the sender to any specific commitment; the commitment itself hides the opening using a random value that is never revealed, preventing any association between the two.

Privacy-preserving account-based models mostly rely on ZKPs to obtain sender's anonymity (e.g., [18, 72, 83, 144]). Specifically, a commitment to the account state is used to trace the balance (along with other information, as detailed later); account state updates include ZKPs to prove correctness. For example, Gross et al. [72] use the ideas of Zerocash by committing on the accounts state and by invalidating them on update. The authors suggest using Merkle trees to prove that a transaction proposal refers to an existing commitment in the ledger without pointing to (and thus revealing) it. Platypus [144] uses a model where payer and payee exchange with the central bank commitments to their updated accounts, the transaction amount, a serial number, and two ZKPs proving the update's validity and its link to a prior certified state. The payer sends his part to the payee, who completes it and interacts with the central bank. If the serial and the ZKPs are valid, the bank updates a publicly available log. Conversely, PEReDi [83] achieves sender anonymity through anonymous channels, an encrypted ledger, and a threshold signature scheme. Specifically, the sender and receiver each submit partial (specular) transactions that update only their respective account states. These transactions are encrypted using threshold encryption. Sender and receiver must both actively engage in updating their accounts. Solidus [40] uses banks as proxies; therefore, transactions are not explicitly exchanged between users but among their respective banks.

4.3.2 Receiver Anonymity. The key techniques for obtaining receiver's anonymity are one-time destination key or stealth address (e.g., [56, 107, 139]), encrypted address (e.g., [20, 136]), commitments as output (e.g., [114, 143]), and updatable public key (e.g., [60, 109]). Obtaining receiver anonymity with Chaumian tokens is challenging, as the payee must interact with the bank to deposit the received tokens. Hayes [75] suggests using *one-time aliases* to obtain privacy, even though the idea is not detailed for practical usages. A similar idea can be found in CryptoNote [139], where the sender generates a *one-time destination key* by combining a random number with the receiver's public key—essentially performing a Diffie-Hellman exchange to derive a shared secret. Because this process requires two elliptic curve keys, a CryptoNote address is twice the size of a standard elliptic curve public key. The transaction is then sent to the one-time public key and includes a commitment to the random value. The receiver scans all the incoming transaction using his private key, reconstructs the one-time destination public key from the commitment, and checks if it matches; if it does, he recognizes the payment as intended for him. The sender and receiver never interact directly. The most obvious disadvantage of this approach is that every receiver has to read every transaction to identify those belonging to him. This approach has been later integrated in other privacy-preserving cryptocurrencies, including Monero [107], which calls the one-time receiver address as *stealth address*, Traceable Monero [89], and PCRAP [56] (see Section 3.6.1).

In Zerocash [20] (unspent token-based ledger), the sender includes an *encrypted version of the recipient's address* in the transaction (using asymmetric encryption). Therefore, periodically, each user has to scan the ledger, decrypt transactions using his private key, and identify those intended for him. Similarly, UTT [136] (Chaumian token-based system) ensures receiver's anonymity using *homomorphic commitments*, randomizable signatures, and anonymous (Boneh-Franklin) IBE scheme [28]. A coin is a commitment to a tuple containing the owner's identifier, a serial number issued by the central bank, and its value. To be valid, each coin comes with a (randomizable) signature by the central bank. To make a payment, the sender creates a transaction for the payee, which includes the receiver's identity commitment, the output value commitment, the encrypted

coin details (using the recipient's key), and ZKPs for correctness. The transaction is sent to the bank for validation. If valid, the bank blindly issues new coins for the recipient and records them on a public ledger—without learning identities or amounts, as they remain hidden via homomorphic commitments[20]. The recipient periodically scans the ledger for transactions encrypted for him, decrypts payments with his private key, verifies the amounts, and retrieves the coins' details. Then, he can randomize the coins before adding them to his wallet.

The unspent token-based ledger MimbleWimble [114] (and PRCash [143]) performs transactions between *commitments* rather than addresses; this guarantees receiver's anonymity and unlinkability.

Account-based approaches typically achieve receiver anonymity using the same technique as for sender anonymity (e.g., [13, 18, 32, 40, 60, 72, 74, 83, 91, 109, 129, 144]). This is done by *updating account state commitments* and employing ZKPs to verify correctness without revealing the receiver's identity. We mention Quisquis [60], where an account is a pair of public key and a homomorphic commitment to its balance. A transaction updates both the public key and the commitment using *updatable public key* (see Section 3.6.2). Senders select public keys—including the recipient's and decoys—to form the transaction input, updating only the sender's and recipient's balances while leaving others unchanged. Since each address appears at most twice on the ledger, de-anonymization is harder. However, this approach requires users to periodically scan the ledger to update their keys, thus introducing some transaction serialization.

4.3.3 Transaction Value Confidentiality. Two main approaches emerge from the literature to conceal transaction details: commitments (e.g., [4, 5, 18, 20, 56, 60, 67, 72, 74, 105, 107, 114, 136, 143, 144]) and public key encryption (e.g., [32, 83, 97, 129]). All the approaches using commitments and most of those adopting public key encryption (e.g., [32, 97]) use ZKPs to prove correctness.

Commitments. Camenisch et al. [36] use commitments with verifiable encryption to hide the commitment's opening: the opening is a random value generated and encrypted during withdrawal, and decrypted and stored at deposit time to prevent double spending. In 2013, Maxwell proposed confidential transactions[21], which use *Pedersen commitments* to hide transaction amounts while ensuring correctness (also using range proofs). Monero adopted and extended this technique by combining ring signatures for sender's anonymity, obtaining RingCT [107]. Zerocash [20] uses ZKPs to achieve full transaction privacy: a cryptographic commitment is made to a new coin, including its value, owner, and serial number; then, a ZKP ensures transaction validity without exposing sensitive data. Commitment-based schemes are the most widely adopted to obtain confidential transaction value, especially because their homomorphic versions allow to combine commitments without leaking sensitive data. To prevent users from creating negative or infinite amounts, range proofs prove that a committed value is greater than zero and within a valid range.

Public Key Encryption. In PEReDi [83], senders and receivers update their accounts by submitting *encrypted transaction information* to several maintainers who operate a distributed ledger. Transactions are encrypted using threshold ElGamal encryption [57], which requires a quorum of maintainer to cooperate to revoke confidentiality and decrypt transaction details. Authorities can trace both transactions and identities. PARScoin [129] extends this approach to introduce an offline payment protocol that does not require interaction between sender and receiver. Androulaki et al. [4] extend Zerocoin to implement auditability: each user is assigned to an auditor at registration time, who can decrypt and inspect the user's transactions when required. Using encryption instead of commitments allows to revoke anonymity and implement auditability and tracing. In

[20]The homomorphic properties of the commitment scheme allows to directly combine the value commitment, identity commitment, and serial number commitment in a single coin commitment.

[21]https://elementsproject.org/features/confidential-transactions/investigation

Solidus [40], Publicly Verifiable Oblivious RAMs (PVORMs) enable banks to update the accounts of their clients without revealing exactly which accounts are being updated. PVORM provides a ZKP demonstrating that the updates are correct with respect to the transaction triggering them.

A few solutions implement privacy-preserving currencies using smart contracts on public blockchains. To ensure confidentiality, transactions are encrypted with the sender's public key and supplemented with a NIZK proof for public verifiability (e.g., [32, 97]). For example, Zether [32] uses smart contracts to define a privacy-preserving tokens that can be exchanged between public keys associated with a confidential balance. They use ElGamal encryption, which has homomorphic properties, for hiding each account's balance. Zether introduces the Σ-Bullets (an enhancement of the Bulletproofs range ZKP) to efficiently prove statements over the encrypted transfer value and the new sender balance. However, Zether currently allows only one anonymous transfer per epoch (to prevent double spending), and execution costs (namely, gas costs on Ethereum) are very high [1]. Similarly, DSC [97] optimizes privacy within Ethereum-style smart contract environments, but does not support anonymity: it uses ZKPs and homomorphic encryption to hide balances and transferred amounts. User balances are stored on the ledger in encrypted form. We briefly mention few other works that focus on privacy in smart contracts. For example, Hawk [126] uses MPC to split a contract into private and public parts, allowing confidential state. Enigma [151] uses secure enclaves to run contract code on encrypted inputs. These solutions are beyond currency transfers, but they show that account privacy often requires splitting roles or off-chain computation. Recent research projects aim for private smart contracts via ZKP frameworks, often assuming a model with private channels or truster parties (e.g., [31]).

4.3.4 Network Anonymity. Network anonymity concerns the protection of sender and receiver IP addresses, which could otherwise reveal real-world identities. Most of the existing approaches do not investigate (or even mention) the network anonymity feature. Few works mention it but consider this issue as out of scope of their investigation (e.g., [13, 40, 136, 144]). Monero [107] and Traceable Monero [89] explicitly support network anonymity by using TOR (The Onion Router[22]) and I2P (Invisible Internet Project[23]) to hide the real IP address [133]. While *TOR* routes traffic through multiple relays to obscure the user's IP address, *I2P* is a network optimized for anonymous peer-to-peer communication, using fully decentralized routing and encryption to enable different anonymous services. Moreover, Monero uses *Dandelion++* [59], an improvement of Dandelion [26], which prevents malicious attackers from linking transactions with their source IPs. Dandelion++ uses randomization mechanisms for message forwarding, which happens in two phases: stem (anonymization) and fluff (broadcast). In the stem phase, a transaction is first relayed privately across a small, randomly chosen subset of nodes; each node probabilistically decides whether to continue forwarding the transaction along the stem or to switch to the second phase. In the fluff phase, the node broadcasts to the entire network using traditional flood propagation protocol.

4.3.5 Linkability and Unlinkability. Unlinkability ensures that transactions cannot be correlated or traced back to the same user. For instance, it should be impossible to conclude that two transactions were delivered to or sent from the same user (other definitions can be found in [3, 44, 148]). Unlinkability can refer to the sender, receiver, or both. A few works break this property into two facets, internal and external. For instance, CoinJoin [98] does not preserve internal unlinkability as each participant in the group that mixes transactions can link entities within the group.

[22] https://www.torproject.org/
[23] https://geti2p.net/en/

Obtaining unlinkability in public ledgers is challenging as linking between addresses in cryptocurrency systems stems from different patterns (e.g., [35, 41, 58, 80, 96, 121, 140, 149]). A non exhaustive list of patterns and heuristics to link addresses is the following.

- *Common input ownership heuristic*: If multiple addresses appear as inputs in a single transaction, they are likely controlled by the same entity;
- *Change address*: In a transaction with two outputs, one is the payment, the other one may be the change returned to the sender;
- *Address reuse*: Using the same address for multiple transactions make it easy to link payments to the same user[24];
- *Temporal analysis*: If transactions occur in rapid succession or at predictable intervals, they may be linked to the same entity;
- *Clustering analysis*: It deals with identifying wallet behavior such as round numbers, specific fee patterns, or common spending habits;
- *Dust attack analysis*: Small amounts of cryptocurrency are sent to many addresses to track their movements (similar to a phishing attack); if dust is later spent with other transactions, it links addresses together.

Existing countermeasures primarily aim to obscure input-output relations within transactions through mixing protocols and address reuse prevention. Mixing protocols disrupt common ownership heuristics and help mitigate change address attacks. Address-changing techniques generate new sender or receiver addresses for each transaction to prevent linkability. Additionally, ensuring network anonymity helps further obscure user identities. To counter dust attacks, the simplest defense is to ignore dust transactions, preventing adversaries from establishing links between addresses. We review mixing protocols and approaches adopted in privacy-native currencies.

Mixing Protocols. Different mixing protocols have been proposed, and we summarize them in Appendix A, Table 6. The naïve approach for mixing transactions consists in a centralized service that collects transactions, and publishes on the ledger a single large transaction with multiple inputs and outputs. This hides the information regarding which inputs flows towards which output, thus limiting (if not preventing) linkability between sender and receiver. The Mixcoin centralized protocol considers an accountable mixer, which offers a signed warranty that will enable his customers to unambiguously prove if the mix has misbehaved [30]. Blindcoin [138] extends this protocol by considering blind signatures for the warrants; this hides the link between input and output addresses of a user even from the mixer, achieving full unlinkability; however, the mixer can still thief coins. Tumblebit [76] prevents coin thief using an on-chain escrow and ZKPs. In BlindHub [116], mixing is performed within payment channel hubs, where the hub acts as tumbler and uses blind signatures to hide the customers' addresses from itself. Obscuro [137] is a mixing protocol that utilizes Trusted Execution Environments (TEEs) and enables participants to verify the mixer's integrity via remote attestation. It is specifically designed to defend against two key threats: the participation rejection attack, where malicious users try to manipulating mixing by withdrawing their deposits before the TEE reads them, and the blockchain forking attack, where adversaries present stale blocks to mislead the mixer. Numerous papers also investigate decentralized protocols. CoinJoin [98] is a technique that allows multiple users to combine their transactions into a single joint transaction; the author does not propose a fully fledged protocol, but only describes the mixing idea. CoinShuffle [123] allow to combine multiple users' inputs and outputs in a single transaction but may compromise anonymity if transaction values differ or the anonymity set is small. Later, Ruffing et al. [124] proposed CoinShuffle++, which uses a

[24]This works because address are pseudonymous but not private.

more efficient mechanism for anonymity, i.e., *Dining Cryptographers Networks* (DC-nets) instead of mix-nets: while a mix-net requires sequential processing, hence a number of communication rounds linear in the number of participants, DC-nets enable to process mixing in parallel, hence it requires only a constant number of communication rounds. Xim [24] is a multi-round protocol for anonymously finding mix partners based on advertisements placed on the blockchain, with the property that no outside party can identify participants that pair up. SecureCoin [79] creates an aggregated temporary deposit address and requires the collaboration of involved peers to generate novel destination addresses and collectively redistribute coins. Leveraging *Confidential Transactions* (which nonetheless is available only in a soft fork of Bitcoin), ValueShuffle [122] provides full privacy besides parties anonymity. Confidentiality enables mixing funds of different values without compromising unlinkability. CoinParty [150] is a decentralized mixing protocol that distributes the mixing process across a group of semi-trusted parties. They use an MPC protocol to jointly perform sensitive operations (i.e., encryption, decryption, and address shuffling), ensuring that no single party can link input and output addresses. Threshold signatures is used to enable claiming funds upon agreement of a majority of mixing nodes. It also includes mechanisms to detect and exclude misbehaving participants, increasing the overall protocol reliability.

Decentralized tumblers have also been designed using *smart contracts* on Ethereum. Möbius [99] uses ring signatures and stealth addresses to obfuscate the senders and recipients associations (as Monero does [107]). First, senders furnish both their funds and stealth keys to the smart contract; then, each recipient creates a ring signature to withdraw their funds from the contract and transfer them to an ephemeral address. The size of the anonymity set is limited to the size of the ring, and the gas cost of the withdrawing transaction increases linearly with the size of the ring. MixEth [130] aims to minimize gas costs by utilizing Neff's verifiable shuffle [106] and off-chain messages. Once the sender deposits coins, the MixEth contract enters rounds of shuffling and challenging, during which anyone can challenge the correctness of the preceding shuffle. When recipients believe that sufficient shuffling has been performed, they can withdraw their funds from the MixEth contract. We also mention AMR [87] which disrupts the traceability connection between coins deposited and withdrawn by the same user. Participants deposit a predetermined quantity of coins into a smart contract, which establishes a Merkle tree structure over the deposits. To withdraw coins, a zk-SNARK prove coins inclusion in the Merkle tree. AMR leverages lending platforms to provide users with the opportunity to accrue interest on their deposited assets.

Although improving unlinkability, mixing protocols are still vulnerable to transaction graph analysis to some extent. For example, if the mixing process does not involve adequate randomization or if there are patterns in the amounts or timing of transactions, likely connections can be identified, even after several rounds of mixing. Also, if a participant does not properly anonymize the change, it can be traced back to the sender. To maximize privacy, participants need to combine mixing protocols with other privacy-preserving techniques and follow best practices (e.g., distributing coins across multiple addresses) to further obscure their transaction history and reduce the risk of being traced through transaction graph analysis. Further details on mixing solutions for Bitcoin and Ethereum, e.g., in [7] and on transaction graph analysis, e.g., in [41, 63, 121].

Integrated Solutions. Privacy-preserving cryptocurrencies leverage a variety of strategies to improve unlinkability, focusing primarily on address-changing, transaction mixing, and use of ZKPs. Each of these techniques adds a layer of privacy, with trade-offs in terms of anonymity, computational complexity, and vulnerability to analysis. One-time destination addresses, *stealth addresses* (e.g., [107, 139]), or updatable public keys (e.g., [60, 109]) help obtain recipient unlinkability. While this ensures the recipient's privacy, it does not address the issue of transaction amounts, which are still visible on the ledger. Moreover, existing protocols leveraging updatable public keys

require the receiver to scan the ledger for incoming payments, which adds complexity for the user. *Ring signatures* (e.g., [107, 139]) build upon the idea of mixing, which redistributes funds in a manner that obscures the origins (and destinations) of the coins. With ring signatures, a sender creates a ring with multiple decoy inputs, making it impossible to determine which input belongs to the spender. However, the privacy protection is limited by the number of decoys in the ring, and sophisticated heuristics can sometimes reduce the anonymity set, thus compromising privacy. For the most robust privacy, *ZKPs* offer a powerful solution (e.g., [20, 114]). They allow transactions to prove their validity without revealing any details about the sender, receiver, or transaction amount. With this approach, transactions are recorded as cryptographic proofs (instead of sender-receiver addresses), ensuring that outputs cannot be linked to inputs. Among ZK-based approaches, we mention the *commitments and nullifiers* strategy (e.g., [20, 136]): a coin contains a commitment to its serial number, and spending the coin involves publishing a nullifier, i.e., a pseudo-random value derived from the coin serial number. This ensures unlinkability to the original coin, while marking it as spent. However, designing a system with commitments and nullifiers is more complex than ring signatures or stealth addresses, and is also more computational demanding (see Section 4.5).

4.3.6 Plausible Deniability. With *plausible deniability*, we intend whether a user can deny intentional participation in a transaction, preventing others and adversaries from proving their involvement (e.g., [1, 150]). Existing work is roughly evenly divided between those that support this property (e.g., [71, 83, 105, 107, 109, 114, 136, 144]) and those that do not (e.g., [13, 32, 67, 72, 74, 98, 139, 146]). Basically, protocols "on-demand", such as mixing protocols, subsume an explicit user intention in using the additional service to obtain a privacy feature (such as unlinkability). Likewise, overlay systems implementing an anonymous cryptocurrency, such as Zerocoin [102], Zerocash [20], EZC [5], Zether [32], require explicit intention to convert native currencies to a privacy-preserving one. Conversely, systems that force anonymity or unlinkability as default, readily provide the property of plausible deniability. Among these systems, we mention Monero [107], MimbleWimble [114], UTT [136], PEReDi [83], or Platypus [144], that do not provide the functionality of combining anonymous and public transactions.

4.4 Auditability

Auditability refers to the ability to examine, track, and verify transactions or balances while preserving a certain degree of privacy. It is a key feature of the third generation of privacy-preserving cryptocurrencies and, in particular, CBDCs. The concept of auditability is closely related to that of *policy*, which refers to the rules enforced by the protocol or by users to govern behaviors (e.g., to restrict who can transact and under what conditions). While policies involve *ex-ante* enforcement (i.e., preventing undesired actions before they occur), auditability supports *post-facto* verification by auditors and regulators. We point out that *accountability* is also important, as it implies linking actions to individuals, e.g., through identity-binding mechanisms such as those provided by a registration authority to comply with AML/CFT regulations.

Full privacy often conflicts with compliance: if transactions are completely opaque, regulators have no oversight [12]. Some privacy designs therefore incorporate policy-controlled privacy. We identify the following approaches that enable or support auditability: anonymity budget, operating limits, revocable anonymity and user tracing, coin tracing, and verification without disclosure.

4.4.1 Anonymity Budget. An *anonymity budget* limits the participant's ability to perform anonymous payments. Each transaction consumes a portion of this budget, and once the participant runs out of it, he loses the ability to make further anonymous payments (e.g., [36, 54, 136, 143]). Camenisch et al. [36] introduce the *bounded-anonymity business model*, where there is a publicly-known limit to the number of coins a user may anonymously transfer to a specific merchant.

PRCash [143] restricts the total amount of anonymous payments a user can *receive* within a time window (or epoch). The anonymity budget is a special non-spendable output. By adjusting the budget size and duration, authorities can regulate the flow of anonymous funds. Once the budget is exhausted, or if the user opts for non-anonymous transactions, the payee creates an output that includes his identity encrypted with the regulator's public key. UTT [136] employs the anonymity budget to balance privacy with accountability for *spending* operations. Beyond the allocated budget, transactions may either lose privacy or require auditing. An auditor periodically refreshes anonymity budgets and can approve transactions that exceed the budget, e.g., by requesting additional transaction details before signing. In all these systems, the anonymity budget is implemented as a special token that must be included in any anonymous but accountable transaction.

4.4.2 Operating Limits. Operating limits indicate constraints beyond which transactions cannot be carried out. Limits are usually imposed on the single transaction value, overall user's balance, amount of sent or received funds, and number of performed transactions within a time window. Compliance with limits is generally proven in zero-knowledge. Garman et al. [67] extend Zerocash by introducing several building blocks to support auditability, including regulatory closure (i.e., restricting transactions to assets of the same type), spending limits, selective user inquiries, and coin tracing. A central idea is the enforcement of a *per-transaction spending limit*, whereby a transaction becomes invalid if it exceeds the defined limit, unless it is signed by an authorized party. Another proposal involves maintaining a *per-user counter* incremented with each outgoing transaction. The system can then reject transactions if the user's counter is above a predefined threshold. Building on these ideas, Gross et al. [72] propose a two-layered account-based system for implementing a CBDC. To enforce the spending policy, each account maintains an *epoch turnover*, representing the total amount spent during the current epoch. Compliance with spending limits is proven using ZKPs as part of each account update transaction. Similarly, AQQUA [109] uses *multiple counters* per account, which are publicly stored as three commitments: one to the account balance, and two to the cumulative amounts sent and received. AQQUA also introduces a registration authority, which links users' real-world identities to their (possibly, multiple) public keys, and an audit authority, which verifies compliance with defined policies. During an audit, the authority selects a user by his public key and requests the opening of two historical account state snapshots (for every user's accounts). These openings allow the authority to directly inspect whether the user complies with specific policies. Five types of policies exist: spending limits, receiving limits, transaction value limits, non-participation (i.e., inactivity over a time period), and transaction opening (i.e., selective disclosure of the value sent or received in a specific transaction). After the audit, the user randomizes his accounts (and provides a ZKP of valid update), thereby restoring his privacy. Xue et al. [145] use per-user counters, as well. They design an unspent token-based ledger, which stores counters in a Merkle tree managed by regulators, which are also in charge of user on-boarding. Users can define policies using ZKPs[25] on counters, which are updated for each transaction through user-regulator interaction. Counters enable policies on total amount and *frequency* of fund transfers within a time period. When suspicious transactions are detected, the participants' identities can be recovered by regulators. Similarly, Chen et al. [49] consider three policies—on transaction limits, rates, and value—and recur to ZKPs to attest compliance.

4.4.3 Revocable Anonymity and User Tracing. User tracing mechanisms enable authorities to discover the user's identity or to identify all transactions associated with a specific user when needed. This concept is often related to revocable or conditional anonymity. *Revocable anonymity* refers to the privacy model where users remain anonymous by default, but their identity can be

[25] Specifically, zk-SNARKs are used for arithmetic statements and Sigma Protocols for algebraic statements.

revealed under certain conditions or rules. Conversely, *user tracing* refers to technical mechanisms that allow authorized entities to identify users or link their transactions when required.

Some contributions use special *digital signatures* that reveal the sender's identity if he double spends (e.g., [18, 75, 147]). Other approaches *encrypt* user-related information—such as identifiers, keys, or transaction metadata—within the transaction, in special *tags*, making them accessible only to tracing authorities with decryption capabilities (e.g., [40, 67, 83, 89, 91, 94, 147]). This is often referred to as the *trapdoor* approach, where decryption is enabled through a private key or other privileged information. Many systems also rely on registration authorities that link real-world identities to cryptographic keys, making traced identities resolvable (e.g., [94, 147]). For example, Cecchetti et al. [40] and Liu et al. [94] consider a setting with a central bank. While Cecchetti et al. [40] manage in oblivious RAM the transactions across users' commercial banks, Liu et al. [94] require that each transaction includes the user's identity encrypted with the central bank public key. In a more general setting, Zhang et al. [147] introduce a supervision authority that can extract the payer's identity from any transaction using its private key (and the registration authority). Garman et al. [67] propose a tracing mechanism where each user receives a *unique encryption key*: if the user is being traced, the key is a randomized version of the authority's key; otherwise, it is derived from a null key. Users cannot tell which type of key they received. To enhance transparency, periodically the system spends tokens that publicly demonstrate how many users were traced without revealing their identities. Androulaki et al. [4] assign each user to an auditor at registration time; then, the system encrypts the transaction information under the public keys of the *sender's and the receivers' auditors*. Xue et al. [145] encode the payer's identity in each transaction using the regulator's key and the transaction's epoch identifier. The epoch identifier enables temporal scoping, which limits the impact of key compromising and can enable granular tracing.

A few works assign *long-term keys* to users, from which they derive one-time keys for individual transactions. User tracing mechanisms aim to recover the long-term keys that uniquely identify a user. For example, Lin et al. [91] propose a system in which users perform transactions using *anonymous keys* derived from their long-term public addresses. To enable tracing, a designated manager can use his private key and the anonymous transaction key to recover the user's long-term public address (and his identity). Traceable Monero [89] introduces a method for generating one-time key pairs for *payees*, where each one-time address is associated with a tag. This tag can be decrypted by the tracing authority to recover the payee's long-term public key. The tag is designed to support both *backward and forward tracing*: it can identify both the accounts that sent funds to the current account and the subsequent accounts that receive funds originating from it.

The ability of tracing authorities to decrypt transactions places a high trust burden on a *centralized entity*. To mitigate this, Yu et al. [146] require collaboration between *two distinct authorities* to detect correlations among a batch of transactions—specifically, those originating from the same user—and to trace user identities when necessary. To enable this functionality, the authors introduce the concept of a *mark*[26], which embeds the sender's public key encrypted with random values. These values can be reconstructed only through the joint effort of both authorities. Conversely, PEReDi [83, 128] proposes distributing the decryption power across multiple parties, requiring them to cooperate and reach a *quorum* in order to access private information. This very idea is then also used by Sarencheh et al. [129].

4.4.4 Coin Tracing. Garman et al. [67] propose an approach to trace certain coins, avoiding an extensive transaction auditing. In particular, the scheme enables tracing by marking specific coins

[26]Although the concepts of tag and mark are closely related, we follow the terminology of [146], using mark to denote data that enables a two-party computation protocol in a privacy-preserving manner.

as traceable, i.e., *tainted coins*[27]. When such a coin is spent, all resulting output coins also become traceable. To implement this, each coin contains a fresh *encryption key*, and tracing data for the outputs is encrypted under the input coin's key and under the authority's public key to prevent the sender from performing unauthorized tracing. To limit cascade tracing and collusion risk (i.e., when a user and the authority jointly trace non-tainted coins), exchanges are allowed to verifiably remove tracing using a public dummy key. Keller et al. [82] propose a method to de-anonymize users in CoinJoin and ring signature schemes. They model a scenario where a law enforcement agency collaborates with certain users—referred to as witnesses—who can confirm that their inputs were not used in a given mixing transaction. As the number of such witnesses increases, the anonymity set of the remaining users is reduced, making de-anonymization more feasible.

4.4.5 Verification without Disclosure. Auditability mechanisms that enable verification without disclosure allow a party to confirm compliance with a policy without learning any private data. These mechanisms rely on cryptographic proofs (usually NIZK) to show that hidden data satisfies specific constraints (e.g., [32, 40, 97, 105, 144]). Crucially, the verifier learns nothing beyond the validity of the claimed statement. Besides the approaches we already mentioned in *Operating limits*, here we review those supporting policies beyond basic balance or transfer checks. For example, zkLedger [105] uses Pedersen commitments enabling aggregate computations. zkLedger supports complex auditing via a map-reduce model: banks generate commitments over filtered and transformed data (e.g., checking for non-zero values or squaring for variance) along with NIZKs proofs of correctness, enabling auditors to compute metrics like means and variances. Platypus [144] defines account states with opaque *auxiliary data* to support custom policy checks. Solidus [40] uses PVORM and ZKPs to publicly verify a bank's transaction log compliance. Zether [32] implements confidential transactions in Ethereum, using NIZKs to verify properties like non-negative balances. DSC [97] enforces compliance of encrypted transactions with range and authorization proofs.

4.5 Summary and Takeaways

The analysis presented in this section highlights the diverse and evolving landscape of privacy goals in digital currencies. While early systems primarily targeted anonymity and unlinkability through address obfuscation and mixing techniques, more recent designs emphasize stronger notions of confidentiality and auditability. These are supported by advanced cryptographic tools such as ZKPs, homomorphic commitments, and threshold encryption.

Each privacy goal—anonymity, confidentiality, unlinkability, and auditability—requires distinct mechanisms and introduces specific trade-offs. For instance, unlinkability may conflict with auditability, and encrypted transactions that ensure confidentiality can complicate regulatory oversight. No single protocol fully satisfies all privacy and compliance objectives; rather, systems tend to prioritize different goals based on their threat models, trust assumptions, and intended deployment contexts. Table 3 provides a high-level summary of how these goals are addressed in practice. For each privacy property, we list relevant cryptographic primitives, system mechanisms, and example of representative protocols. This mapping serves as a reference point for researchers and practitioners navigating the trade-offs inherent in privacy-preserving digital currency design.

In general, achieving comprehensive privacy requires combining multiple techniques, each with specific trade-offs in terms of confidentiality, unlinkability, and computational demand. For example, blind signatures are lightweight and provide cash-like anonymity but lack unlinkability without centralized coordination. Ring signatures offer anonymity that scales with ring size, but increasing the anonymity set directly impacts computational cost and latency. Encryption, threshold

[27] A tainted coin is a traceable digital asset, which could be linked to illicit or suspicious activity, often flagged for enhanced scrutiny or restricted use.

Table 3. Summary of key technologies for achieving privacy goals and corresponding example protocols

Privacy Goal	Key Cryptographic Tools	System Mechanisms	Example Protocols
Sender Anonymity	Blind, ring, group, and randomizable signatures, commitments and ZKPs	One-time addresses, anonymity set, anonymous addresses	eCash, UTT, Zerocoin
Receiver Anonymity	One-time public keys, encrypted addresses, commitments	One-time addresses	MimbleWimble, Quisquis, Zerocash
Confidentiality	Commitments, encryption, ZKPs	Confidential values and balances	Zether, PEReDi, Monero
Network Anonymity	TOR, I2P, Dandelion++	Obfuscate IP addresses and routing paths	Monero
Unlinkability	Mixing protocols, ring signatures, one-time addresses, commitments	Fresh key generation, decoy input, output set hiding	Platypus, BlockMaze, CoinShuffle++
Auditability	Authenticated tokens, threshold encryption, anonymous keys, trapdoor, ZKPs	Anonymity budget, operating limits, tracing hooks, revocable anonymity	PEReDi, Solidus, zkLedger

encryption, and commitments facilitate selective auditability but do not provide sender anonymity on their own. ZKPs offer robust privacy guarantees, but they are computationally intensive, may require substantial RAM (e.g., [1, 90]), and require careful engineering for practical deployment. One of the well-known ZKP-based systems is zk-SNARK, which makes transactions compact for verification but needs a trusted setup that, if mishandled, could compromise the security of the system. Importantly, as shown in Section 4.4, implementing auditability often requires careful design of ledger data structures and interaction protocols to avoid unintended information leaks while still enabling policy verification. This multidimensional design space underscores the need for a holistic, design-oriented approach: effective privacy-preserving digital currencies must integrate layered technical solutions that balance user privacy with institutional accountability.

5 Open Challenges and Research Directions

Despite significant advances in privacy-preserving digital currencies, numerous open challenges remain. These challenges stem from the unique intersection of cryptographic capabilities, monetary policy requirements, and societal expectations of privacy and auditability. We highlight a few directions that merit further investigation.

5.1 Formal Definitions of Privacy and Auditability

Existing systems use ad-hoc definitions for privacy and auditability. As pointed out in different previous works (e.g., [1, 68]), there exists several theoretical gaps related to the definition of unified privacy models and auditability. Most protocols informally claim anonymity, unlinkability, or confidentiality; however, formal and composable definitions that encompass transaction graph obfuscation, timing metadata, and value-hiding under active adversaries are lacking. We also point out that protocols combining multiple primitives often lack formal, compositional security proofs in frameworks like UC [39] or IITM [85]. Auditability mechanisms (e.g., view keys, trapdoor disclosures) are rarely formalized. There is a lack of frameworks to reason about selective disclosure, accountability guarantees, or conditional anonymity under adversarial inspectors.

5.2 Scalability and Efficiency

Many cryptographic protocols and, more broadly, PETs remain computationally demanding. This includes fully homomorphic encryption, ZKPs, and MPC. However, real-world cryptocurrencies and CBDCs must support high transaction throughput with low latency and high reliability. To

bridge this gap, future research must focus on developing optimized, lightweight, and parallelizable privacy-preserving protocols that can operate efficiently in high-throughput environments. Promising directions include hardware acceleration (e.g., via TEEs) and the exploration of protocol variants that balance privacy guarantees with resource efficiency. ZKPs play a central role in many of the systems analyzed, yet several fundamental challenges remain. General-purpose ZKP schemes (e.g., zk-SNARKs, Bulletproofs) face issues related to trusted setup, proof aggregation, and post-quantum security. For instance, zk-SNARKs typically require a trusted setup, introducing centralization risks and limiting composability. While transparent SNARKs (e.g., Plonky2[28] and Spartan [131]) and STARKs mitigate these concerns, they often incur higher computational costs. Achieving efficient, setup-free ZKPs without compromising performance remains an open research problem. Additionally, support for efficiently generating proofs that can verify other proofs—known as recursive proof composition—is still limited, even though this capability is crucial for scalable applications like off-chain rollups. Developing ZKPs that are not only succinct and recursive, but also universally verifiable (i.e., verifiable by anyone without special access), remains a technically challenging and active area of research. Finally, no current ZKP scheme offers both strong post-quantum guarantees[29] and performance comparable to elliptic curve-based constructions, highlighting the need for further advances in quantum-resistant proof systems.

5.3 Balancing Privacy with Regulatory Compliance

Total anonymity is generally incompatible with regulatory oversight, making auditable privacy and selective disclosure critical design goals. An open challenge is how to enable systems where honest users maintain strong privacy guarantees, while still allowing authorities to investigate illicit activities under well-defined conditions. Several preliminary approaches explore this trade-off, yet significant work remains to formalize disclosure policies, define the types of information subject to auditing, and clarify the roles and interactions of participating entities. Relevant research directions include ZKPs with selective opening, trapdoor, witness encryption, and anonymity budgeting—each enabling varying levels of conditional transparency. For instance, while fully underexplored, witness encryption holds promise for enabling advanced privacy-preserving policies, such as verifiable time locks or conditional disclosures without trusted third parties (see Appendix A.3.5). Additionally, emerging requirements such as time-bound anonymity, jurisdiction-aware confidentiality, and programmable disclosure call for more flexible and context-aware privacy controls. Designing PETs that can support such adaptive and policy-compliant behaviors, while remaining robust against misuse, poses a complex challenge involving technical, legal, and usability considerations.

5.4 Adaptive Privacy and Role Rotation

Privacy-preserving systems fundamentally rely on the concept of anonymity sets. However, managing these sets in a secure, scalable, and adaptive manner remains an open challenge, especially in decentralized or evolving environments. Many existing systems, such as those based on ring signatures, operate with fixed-size anonymity sets or employ heuristic-based decoy selection. These approaches struggle to maintain robustness as usage patterns shift, and they are vulnerable to poisoning attacks, where adversaries inject tainted or known outputs into the anonymity set to degrade privacy. Developing cryptographically sound methods for dynamic anonymity set construction that grows or shrinks with system usage, while resisting active attacks, remains an unsolved problem. Beyond the anonymity set, adaptive privacy must also account for the dynamic nature of

[28] https://github.com/0xPolygonZero/plonky2

[29] ZKPs based on bilinear pairings (e.g., Plonk, Groth) or Pedersen vector commitments (i.e., Bulletproof) assume at least the discrete logarithm problem, hence are subject to quantum threats. Other SNARKs (e.g., Ligero, Orion) and STARKs using Merkle tree and other protocols to build a polynomial commitment are post quantum resistant.

system participants, particularly in regulatory or issuance roles. In systems like CBDCs or regulated cryptocurrencies, authorities responsible for auditing, compliance verification, or even token issuance may change over time due to policy shifts, jurisdictional realignments, or organizational restructuring. Supporting the evolution of designated entities, such as auditors or currency issuers, without compromising user anonymity or transaction confidentiality is a significant technical challenge. Current systems typically assume static trust anchors or fixed multi-signature sets. However, introducing cryptographic mechanisms to dynamically reconfigure these roles (e.g., via threshold cryptography, distributed key generation, or policy-based encryption) requires careful balance between adaptability and privacy. Specifically, open issues relate to securely update the set of auditing entities, policies evolving over time, or support role rotation or revocation while preserving unlinkability, correctness, and forward secrecy.

5.5 Network-Level Anonymity and Cross-layer Privacy Leakage

Network-layer anonymity in cryptocurrencies is typically outsourced to external systems (e.g., Tor, mix networks). However, native integration of network-level privacy mechanisms into cryptocurrency protocols remains largely unresolved. This delegation creates a critical cross-layer vulnerability, where even perfectly private on-chain transactions can be deanonymized through timing analysis, transaction ordering, volume inference, or network topology correlation. Joint modeling of privacy across the ledger layer and network layer is underexplored. Existing protocols often assume independence between transaction content and communication patterns, which adversaries can exploit. For instance, gossip-based mempool propagation leaks significant metadata about the origin and timing of transactions, enabling traffic correlation attacks. Emerging research directions include the design of privacy-preserving mempools that incorporate indistinguishable broadcasting, cover traffic, oblivious relaying, or dandelion-style [59] diffusion to mask transaction origins. These approaches aim to reduce the information available to global or local adversaries observing network traffic (e.g., [16, 51]). However, these solutions often introduce latency, bandwidth overhead, or synchronization issues, leading to difficult trade-offs between efficiency, liveness, and anonymity guarantees. Addressing network anonymity in cryptocurrency systems requires cross-layer designs that align messaging protocols with the cryptographic guarantees of the ledger. This is a highly active area of research, particularly in adversarial environments where well-funded surveillance actors may target both layers simultaneously.

6 Conclusion

Designing digital currencies that balance user privacy with institutional auditability remains one of the most intricate and impactful challenges in modern cryptographic and systems engineering. Over the past decade, cryptocurrencies and CBDC prototypes have pushed innovation across cryptographic protocols, PETs, and system architectures. Yet, despite substantial progress, the design space remains fragmented, and no universally accepted blueprint has emerged.

This survey offers a structured and technically grounded overview of the evolving landscape of privacy-preserving digital currencies. We systematize the literature across decentralized cryptocurrencies and CBDCs, introducing a comprehensive taxonomy that connects privacy and auditability goals with underlying cryptographic primitives, protocol designs, and system architectures. In doing so, we bridge conceptual objectives—such as anonymity, confidentiality, and auditability—with concrete technical mechanisms and design models. Our analysis traces the evolution of privacy-preserving digital currencies across three generations, reflecting an increasing interest in addressing privacy-accountability trade-offs. We underscore that privacy is not a monolithic property but a multifaceted design goal shaped by system requirements and regulatory constraints. Finally, we outline several open research challenges that call for interdisciplinary collaboration across

cryptography, distributed systems, economics, and policy. Key directions include the development of scalable and composable privacy primitives, architectures that balance auditability with confidentiality, and formal models that capture realistic adversarial and compliance scenarios. As digital currencies transition from academic prototypes to operational systems, ensuring robust privacy alongside regulatory compliance will require continued technical innovation and interdisciplinary coordination. By providing a unified taxonomy and analysis framework, this survey aims to support future research and guide the design of privacy-preserving digital currency systems.

Acknowledgments

We thank Marco Benedetti, Giuseppe Galano, Sara Giammusso, Antonio Muci, and Fabiana Rossi for their insightful discussions in the early stages of this work and for their valuable feedback throughout its development.

References

[1] Ghada Almashaqbeh and Ravital Solomon. 2022. SoK: Privacy-Preserving Computing in the Blockchain Era. In *Proc. of IEEE EuroS&P'22*. IEEE, New York City, US, 124–139.

[2] Nasser Alsalami and Bingsheng Zhang. 2019. SoK: A Systematic Study of Anonymity in Cryptocurrencies. In *Proc. of IEEE DSC'19*. 1–9.

[3] Niluka Amarasinghe, Xavier Boyen, and Matthew McKague. 2019. A Survey of Anonymity of Cryptocurrencies. In *Proc. of ACSW '19*. ACM, Article 2, 10 pages.

[4] Elli Androulaki, Jan Camenisch, Angelo De Caro, Maria Dubovitskaya, Kaoutar Elkhiyaoui, and Björn Tackmann. 2020. Privacy-preserving auditable token payments in a permissioned blockchain system. In *Proc. of AFT'20*. ACM, 255–267.

[5] Elli Androulaki and Ghassan O. Karame. 2014. Hiding Transaction Amounts and Balances in Bitcoin. In *TRUST*. Springer, 161–178.

[6] Elli Androulaki, Ghassan O. Karame, Marc Roeschlin, Tobias Scherer, and Srdjan Capkun. 2013. Evaluating User Privacy in Bitcoin. In *Proc. of FC'13*. Springer, 34–51.

[7] Alireza Arbabi, Ardeshir Shojaeinasab, Behnam Bahrak, and Homayoun Najjaran. 2023. Mixing Solutions in Bitcoin and Ethereum Ecosystems: A Review and Tutorial. arXiv:2310.04899

[8] Raphael Auer, Rainer Böhme, Jeremy Clark, and Didem Demirag. 2022. Mapping the Privacy Landscape for Central Bank Digital Currencies. *Queue* 20, 4 (2022), 16–38.

[9] Raphael Auer, Rainer Böhme, Jeremy Clark, and Didem Demirag. 2025. Privacy-enhancing technologies for digital payments: mapping the landscape. *BIS: Working Papers* (2025), 23.

[10] Foteini Baldimtsi, Kostas Kryptos Chalkias, Varun Madathil, and Arnab Roy. 2024. SoK: Privacy-Preserving Transactions in Blockchains. IACR ePrint 2024/1959.

[11] Foteini Baldimtsi, Melissa Chase, Georg Fuchsbauer, and Markulf Kohlweiss. 2015. Anonymous Transferable E-Cash. In *Proc. of PKC'15*. Springer, 101–124.

[12] David Ballaschk and Jan Paulick. 2021. The public, the private and the secret: Thoughts on privacy in central bank digital currencies. *J. Paym. Strategy Syst.* 15, 3 (2021), 277–286.

[13] Mathieu Baudet, Alberto Sonnino, Mahimna Kelkar, and George Danezis. 2023. Zef: Low-latency, Scalable, Private Payments. In *Proc. of WPES'23*. ACM, 1–16.

[14] Balthazar Bauer, Georg Fuchsbauer, and Chen Qian. 2021. Transferable E-Cash: A Cleaner Model and the First Practical Instantiation. In *Proc. of PKC'21*. Springer, 559–590.

[15] Carsten Baum, James Hsin yu Chiang, Bernardo David, and Tore Kasper Frederiksen. 2023. SoK: Privacy-Enhancing Technologies in Finance. IACR ePrint 2023/122.

[16] Joseph Bebel and Dev Ojha. 2022. Ferveo: Threshold Decryption for Mempool Privacy in BFT networks. IACR ePrint 2022/898.

[17] Morten L Bech and Rodney Garratt. 2017. Central bank cryptocurrencies. *BIS Quarterly Review September* (2017).

[18] Carolin Beer, Sheila Zingg, Kari Kostiainen, Karl Wüst, Vedran Capkun, and Srdjan Capkun. 2024. PayOff: A Regulated Central Bank Digital Currency with Private Offline Payments. arXiv:2408.06956

[19] Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev. 2019. Scalable Zero Knowledge with No Trusted Setup. In *CRYPTO'19*. Springer, 701–732.

[20] Eli Ben Sasson, Alessandro Chiesa, Christina Garman, Matthew Green, Ian Miers, Eran Tromer, and Madars Virza. 2014. Zerocash: Decentralized Anonymous Payments from Bitcoin. In *Proc. of IEEE S&P'14*. 459–474.

[21] Daniel Benarroch, Bryan Gillespie, Ying Tong Lai, and Andrew Miller. 2024. SoK: Programmable Privacy in Distributed Systems. IACR ePrint 2024/982.

[22] Jorge Bernal Bernabe, Jose Luis Canovas, Jose L. Hernandez-Ramos, Rafael Torres Moreno, and Antonio Skarmeta. 2019. Privacy-Preserving Solutions for Blockchain: Review and Challenges. *IEEE Access* 7 (2019), 164908–164940.

[23] BIS Innovation Hub. 2023. *Project Tourbillon: Exploring privacy, security and scalability for CBDCs.* Technical Report. BIS.

[24] George Bissias, A. Pinar Ozisik, Brian N. Levine, and Marc Liberatore. 2014. Sybil-Resistant Mixing for Bitcoin. In *Proc. of WPES'14.* ACM, 149–158.

[25] Nir Bitansky, Ran Canetti, Alessandro Chiesa, and Eran Tromer. 2012. From extractable collision resistance to succinct non-interactive arguments of knowledge, and back again. In *Proc. of ITCS'12.* ACM, 326–349.

[26] Shaileshh Bojja Venkatakrishnan, Giulia Fanti, and Pramod Viswanath. 2017. Dandelion: Redesigning the Bitcoin Network for Anonymity. *Proc. ACM Meas. Anal. Comput. Syst.* 1, 1, Article 22 (2017), 34 pages.

[27] Dan Boneh, Justin Drake, Ben Fisch, and Ariel Gabizon. 2020. Halo Infinite: Recursive zk-SNARKs from any Additive Polynomial Commitment Scheme. IACR ePrint 2020/1536.

[28] Dan Boneh and Matthew Franklin. 2003. Identity-Based Encryption from the Weil Pairing. *SIAM J. Comput.* 32, 3 (2003), 586–615.

[29] Joseph Bonneau, Andrew Miller, Jeremy Clark, Arvind Narayanan, Joshua A. Kroll, and Edward W. Felten. 2015. SoK: Research Perspectives and Challenges for Bitcoin and Cryptocurrencies. In *Proc. of IEEE S&P'15.* 104–121.

[30] Joseph Bonneau, Arvind Narayanan, Andrew Miller, Jeremy Clark, Joshua A. Kroll, and Edward W. Felten. 2014. Mixcoin: Anonymity for Bitcoin with Accountable Mixes. In *Proc. of FC'14.* Springer, 486–504.

[31] Sean Bowe, Alessandro Chiesa, Matthew Green, Ian Miers, Pratyush Mishra, and Howard Wu. 2020. ZEXE: Enabling Decentralized Private Computation. In *Proc. of IEEE S&P'20.* 947–964.

[32] Benedikt Bünz, Shashank Agrawal, Mahdi Zamani, and Dan Boneh. 2020. Zether: Towards Privacy in a Smart Contract World. In *Proc. of FC'20.* Springer, 423–443.

[33] Benedikt Bünz, Jonathan Bootle, Dan Boneh, Andrew Poelstra, Pieter Wuille, and Greg Maxwell. 2018. Bulletproofs: Short Proofs for Confidential Transactions and More. In *Proc. of IEEE S&P'18.* 315–334.

[34] Vitalik Buterin et al. 2013. Ethereum white paper. https://ethereum.org/it/whitepaper/.

[35] Ferenc Béres, István A. Seres, András A. Benczúr, and Mikerah Quintyne-Collins. 2021. Blockchain is Watching You: Profiling and Deanonymizing Ethereum Users. In *Proc. of IEEE DAPPS'21.* 69–78.

[36] Jan Camenisch, Susan Hohenberger, and Anna Lysyanskaya. 2006. Balancing Accountability and Privacy Using E-Cash (Ext. Abstr.). In *Proc. of SCN'06.* Springer, 141–155.

[37] Jan Camenisch and Anna Lysyanskaya. 2003. A Signature Scheme with Efficient Protocols. In *Secur. Commun. Netw.* Springer, 268–289.

[38] Jan Camenisch and Anna Lysyanskaya. 2004. Signature Schemes and Anonymous Credentials from Bilinear Maps. In *Proc. of CRYPTO'04.* Springer, 56–72.

[39] Ran Canetti. 2000. Universally Composable Security: A New Paradigm for Cryptographic Protocols. IACR ePrint 2000/067.

[40] Ethan Cecchetti, Fan Zhang, Yan Ji, Ahmed Kosba, Ari Juels, and Elaine Shi. 2017. Solidus: Confidential Distributed Ledger Transactions via PVORM. In *Proc. of ACM SIGSAC CCS'17.* ACM, 701–717.

[41] Wren Chan and Aspen Olmsted. 2017. Ethereum transaction graph analysis. In *Proc. of ICITST'17.* 498–500.

[42] Melissa Chase, Markulf Kohlweiss, Anna Lysyanskaya, and Sarah Meiklejohn. 2014. Malleable Signatures: New Definitions and Delegatable Anonymous Credentials. In *Proc. of IEEE CSF'14.* 199–213.

[43] Panagiotis Chatzigiannis, Foteini Baldimtsi, and Konstantinos Chalkias. 2021. SoK: Auditability and Accountability in Distributed Payment Systems. In *Proc. of ACNS'21.* Springer, 311–337.

[44] David Chaum. 1983. Blind Signatures for Untraceable Payments. In *CRYPTO.* Springer, 199–203.

[45] David Chaum, Amos Fiat, and Moni Naor. 1990. Untraceable Electronic Cash. In *Proc. of CRYPTO'88.* Springer, 319–327.

[46] David Chaum, Christian Grothoff, and Thomas Moser. 2021. How to Issue a Central Bank Digital Currency. arXiv:2103.00254

[47] David Chaum and Eugène van Heyst. 1991. Group Signatures. In *Proc. of EUROCRYPT'91.* Springer, 257–265.

[48] Thomas Chen, Hui Lu, Teeramet Kunpittaya, and Alan Luo. 2023. A Review of zk-SNARKs. arXiv:2202.06877

[49] Yu Chen, Xuecheng Ma, Cong Tang, and Man Ho Au. 2020. PGC: Decentralized Confidential Payment System with Auditability. In *Proc. of ESORICS'20.* Springer, 591–610.

[50] Miranda Christ, Foteini Baldimtsi, Konstantinos Kryptos Chalkias, Deepak Maram, Arnab Roy, and Joy Wang. 2024. SoK: Zero-Knowledge Range Proofs. Crypt. IACR 2024/430.

[51] Claudia Diaz, Harry Halpin, and Aggelos Kiayias. 2021. The nym network.

[52] Ali Dogan and Kemal Bicakci. 2023. KAIME : Central Bank Digital Currency with Realistic and Modular Privacy. IACR ePrint 2023/713.

[53] Florian Dold. 2019. *The GNU Taler system: practical and provably secure electronic payments.* Ph. D. Dissertation. Université de Rennes.

[54] ECB. 2019. Exploring anonymity in central bank digital currencies. *Focus, Dec. (4)* 1, 11 (2019).

[55] Mohammed El-Hajj and Bjorn Oude Roelink. 2024. Evaluating the Efficiency of zk-SNARK, zk-STARK, and Bulletproof in Real-World Scenarios: A Benchmark Study. *Information (Switzerland)* 15, 8 (2024), 463.

[56] Issameldeen Elfadul, Lijun Wu, Rashad Elhabob, and Ahmed Elkhalil. 2024. A privacy and compliance in regulated anonymous payment system based on blockchain. *J. Ambient Intell. Humaniz. Comput.* 15, 8 (2024), 3141–3157.

[57] T. Elgamal. 1985. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Trans. Inf. Theory.* 31, 4 (1985), 469–472.

[58] Dmitry Ermilov, Maxim Panov, and Yury Yanovich. 2017. Automatic Bitcoin Address Clustering. In *Proc. of IEEE ICMLA'17.* 461–466.

[59] Giulia Fanti, Shaileshh Bojja Venkatakrishnan, Surya Bakshi, Bradley Denby, Shruti Bhargava, Andrew Miller, and Pramod Viswanath. 2018. Dandelion++: Lightweight Cryptocurrency Networking with Formal Anonymity Guarantees. *Proc. ACM Meas. Anal. Comput. Syst.* 2, 2, Article 29 (2018), 35 pages.

[60] Prastudy Fauzi, Sarah Meiklejohn, Rebekah Mercer, and Claudio Orlandi. 2019. Quisquis: A New Design for Anonymous Cryptocurrencies. In *Proc. of ASIACRYPT'19.* Springer, 649–678.

[61] Qi Feng, Debiao He, Sherali Zeadally, Muhammad Khurram Khan, and Neeraj Kumar. 2019. A survey on privacy protection in blockchain system. *J. Netw. Comput. App.* 126 (2019), 45–58.

[62] Amos Fiat and Adi Shamir. 1987. How To Prove Yourself: Practical Solutions to Identification and Signature Problems. In *Proc. of CRYPTO'86.* Springer, 186–194.

[63] Michael Fleder, Michael S. Kester, and Sudeep Pillai. 2015. Bitcoin Transaction Graph Analysis. arXiv 1502.01657.

[64] Daniele Friolo, Geoffrey Goodell, Dann Toliver, and Hazem Danny Nakib. 2024. Private Electronic Payments with Self-Custody and Zero-Knowledge Verified Reissuance. arXiv:2409.01958

[65] Eiichiro Fujisaki. 2011. Sub-linear Size Traceable Ring Signatures without Random Oracles. In *Proc. of CT-RSA'11.* Springer, 393–415.

[66] Eiichiro Fujisaki and Koutarou Suzuki. 2007. Traceable Ring Signature. In *Proc. of PKC'07.* Springer, 181–200.

[67] Christina Garman, Matthew Green, and Ian Miers. 2017. Accountable Privacy for Decentralized Anonymous Payments. In *Proc. of FC'17.* Springer, 81–98.

[68] Daniel Genkin, Dimitrios Papadopoulos, and Charalampos Papamanthou. 2018. Privacy in decentralized cryptocurrencies. *Commun. ACM* 61, 6 (may 2018), 78–88.

[69] Simin Ghesmati, Walid Fdhila, and Edgar Weippl. 2022. SoK: How private is Bitcoin? Classification and Evaluation of Bitcoin Privacy Techniques. In *Proc. of ARES'22.* ACM, Article 5, 14 pages.

[70] Oded Goldreich. 2004. *Foundations of Cryptography, Volume 2.* Cambridge university press Cambridge.

[71] Geoff Goodell, D. R. Toliver, and Hazem Danny Nakib. 2023. A Scalable Architecture for Electronic Payments. In *Proc. of FC'22 Workshops.* Springer, 645–678.

[72] Jonas Gross, Johannes Sedlmeir, Matthias Babel, Alexander Bechtel, and Benjamin Schellinger. 2021. Designing a central bank digital currency with support for cash-like privacy. *Available at SSRN 3891121* (2021), 44 pages.

[73] Jens Groth. 2016. On the Size of Pairing-Based Non-interactive Arguments. In *Proc. of EUROCRYPT'16.* Springer, 305–326.

[74] Zhangshuang Guan, Zhiguo Wan, Yang Yang, Yan Zhou, and Butian Huang. 2022. BlockMaze: An Efficient Privacy-Preserving Account-Model Blockchain Based on zk-SNARKs. *IEEE Trans. Dependable Secure Comput.*. 19, 3 (2022), 1446–1463.

[75] Barry Hayes. 1990. Anonymous one-time signatures and flexible untraceable electronic cash. In *AUSCRYPT '90.* Springer, 294–305.

[76] Ethan Heilman et al. 2016. TumbleBit: An Untrusted Bitcoin-Compatible Anonymous Payment Hub. Crypt. IACR 2016/575.

[77] Jordi Herrera-Joancomartí. 2015. Research and Challenges on Bitcoin Anonymity. In *Proc. of DPM'15.* Springer, 3–16.

[78] Lasse Herskind, Panagiota Katsikouli, and Nicola Dragoni. 2020. Privacy and Cryptocurrencies-A Systematic Literature Review. *IEEE Access* 8 (2020), 54044–54059.

[79] Maged Hamada Ibrahim. 2017. Securecoin: a robust secure and efficient protocol for anonymous bitcoin ecosystem. *Int. J. Netw. Secur.* 19, 2 (2017), 295–312.

[80] Harry Kalodner, Malte Möser, Kevin Lee, Steven Goldfeder, Martin Plattner, Alishah Chator, and Arvind Narayanan. 2020. BlockSci: Design and applications of a blockchain analysis platform. In *Proc. of Security'20.* USENIX, 2721–2738.

[81] George Kappos, Haaroon Yousaf, Mary Maller, and Sarah Meiklejohn. 2018. An Empirical Analysis of Anonymity in Zcash. In *Proc. of Security'18.* USENIX, 463–477.

[82] Patrik Keller, Martin Florian, and Rainer Böhme. 2021. Collaborative Deanonymization. In *Proc. of FC'21 Workshops*. Springer, 39–46.

[83] Aggelos Kiayias, Markulf Kohlweiss, and Amirreza Sarencheh. 2022. PEReDi: Privacy-Enhanced, Regulated and Distributed Central Bank Digital Currencies. In *Proc. of ACM SIGSAC CCS'22*. ACM, 1739–1752.

[84] Wiebe Koerhuis, Tahar Kechadi, and Nhien-An Le-Khac. 2020. Forensic analysis of privacy-oriented cryptocurrencies. *Forensic Sci. Int.: Digit. Investig.* 33 (2020), 200891.

[85] Ralf Kuesters, Max Tuengerthal, and Daniel Rausch. 2013. The IITM Model: a Simple and Expressive Model for Universal Composability. IACR ePrint 2013/025.

[86] Amrit Kumar, Clément Fischer, Shruti Tople, and Prateek Saxena. 2017. A Traceability Analysis of Monero's Blockchain. In *Proc. of ESORICS'17*. Springer, 153–173.

[87] Duc V. Le and Arthur Gervais. 2021. AMR: autonomous coin mixer with privacy preserving reward distribution. In *Proc. of ACM AFT'21*. ACM, 142–155.

[88] Rujia Li, Qin Wang, Qi Wang, David Galindo, and Mark Ryan. 2022. SoK: TEE-assisted Confidential Smart Contract. arXiv:2203.08548

[89] Yannan Li, Guomin Yang, Willy Susilo, Yong Yu, Man Ho Au, and Dongxi Liu. 2021. Traceable Monero: Anonymous Cryptocurrency with Enhanced Accountability. *IEEE Trans. Dependable Secure Comput.*. 18, 2 (2021), 679–691.

[90] Junkai Liang, Daqi Hu, Pengfei Wu, Yunbo Yang, Qingni Shen, and Zhonghai Wu. 2025. SoK: Understanding zk-SNARKs: The Gap Between Research and Practice. arXiv:2502.02387

[91] Chao Lin, Debiao He, Xinyi Huang, Muhammad Khurram Khan, and Kim-Kwang Raymond Choo. 2020. DCAP: A Secure and Efficient Decentralized Conditional Anonymous Payment System Based on Blockchain. *IEEE Trans. Inf. Forensics Secur.* 15 (2020), 2440–2452.

[92] Joseph K. Liu and Duncan S. Wong. 2005. Linkable Ring Signatures: Security Models and New Schemes. In *Proc. of ICCSA'05*. Springer, 614–623.

[93] Yi Liu, Xingtong Liu, Chaojing Tang, Jian Wang, and Lei Zhang. 2018. Unlinkable Coin Mixing Scheme for Transaction Privacy Enhancement of Bitcoin. *IEEE Access* 6 (2018), 23261–23270.

[94] Yunke Liu, Jianbing Ni, and Mohammad Zulkernine. 2022. AT-CBDC: Achieving Anonymity and Traceability in Central Bank Digital Currency. In *Proc. of IEEE ICC'22*. 4402–4407.

[95] James Lovejoy, Madars Virza, Cory Fields, Kevin Karwaski, Anders Brownworth, and Neha Narula. 2023. Hamilton: A High-Performance Transaction Processor for Central Bank Digital Currencies. In *Proc. of NSDI'23*. USENIX, 901–915.

[96] Qinghua Lu, Xiwei Xu, H.M.N. Dilum Bandara, Shiping Chen, and Liming Zhu. 2022. Patterns for Blockchain-Based Payment Applications. In *Proc. of EuroPLoP'21*. ACM, Article 28, 17 pages.

[97] Shunli Ma, Yi Deng, Debiao He, Jiang Zhang, and Xiang Xie. 2021. An Efficient NIZK Scheme for Privacy-Preserving Transactions Over Account-Model Blockchain. *IEEE Trans. Dependable Secure Comput.*. 18, 2 (2021), 641–651.

[98] G. Maxwell. 2013. CoinJoin: Bitcoin privacy for the real world. https://bitcointalk.org/index.php?topic=279249.0.

[99] Sarah Meiklejohn and Rebekah Mercer. 2018. Möbius: Trustless Tumbling for Transaction Privacy. *Proc. Priv. Enhancing Technol.* 2018, 2 (2018), 105–121.

[100] Sarah Meiklejohn, Marjori Pomarole, Grant Jordan, Kirill Levchenko, Damon McCoy, Geoffrey M. Voelker, and Stefan Savage. 2013. A fistful of bitcoins: characterizing payments among men with no names. In *Proc. IMC'13*. ACM, 127–140.

[101] Sarah Meiklejohn, Marjori Pomarole, Grant Jordan, Kirill Levchenko, Damon McCoy, Geoffrey M. Voelker, and Stefan Savage. 2016. A fistful of Bitcoins: characterizing payments among men with no names. *Commun. ACM* 59, 4 (2016), 86–93.

[102] Ian Miers, Christina Garman, Matthew Green, and Aviel D. Rubin. 2013. Zerocoin: Anonymous Distributed E-Cash from Bitcoin. In *Proc. of IEEE S&P'13*. 397–411.

[103] Malte Möser, Kyle Soska, Ethan Heilman, Kevin Lee, Henry Heffan, Shashvat Srivastava, Kyle Hogan, Jason Hennessey, Andrew Miller, Arvind Narayanan, and Nicolas Christin. 2018. An Empirical Analysis of Traceability in the Monero Blockchain. arXiv:1704.04299

[104] Satoshi Nakamoto. 2008. Bitcoin: A peer-to-peer electronic cash system. *Satoshi Nakamoto* (2008).

[105] Neha Narula, Willy Vasquez, and Madars Virza. 2018. zkLedger: Privacy-Preserving Auditing for Distributed Ledgers. In *Proc. of NSDI'18*. USENIX, 65–80.

[106] C. Andrew Neff. 2001. A verifiable secret shuffle and its application to e-voting. In *Proc. of the ACM CCS'01*. ACM, 116–125.

[107] Shen Noether, Adam Mackenzie, et al. 2016. Ring confidential transactions. *Ledger* 1 (2016), 1–18.

[108] Mohammad Reza Nosouhi, Shui Yu, Keshav Sood, Marthie Grobler, Raja Jurdak, Ali Dorri, and Shigen Shen. 2023. UCoin: An Efficient Privacy Preserving Scheme for Cryptocurrencies. *IEEE Trans. Dependable Secure Comput.*. 20, 1 (2023), 242–255.

[109] George Papadoulis, Danai Balla, Panagiotis Grontas, and Aris Pagourtzis. 2024. AQQUA: Augmenting Quisquis with Auditability. IACR ePrint 2024/1181.
[110] Torben Pryds Pedersen. 1992. Non-Interactive and Information-Theoretic Secure Verifiable Secret Sharing. In *CRYPTO '91*, Joan Feigenbaum (Ed.). Springer, 129–140.
[111] Li Peng, Wei Feng, Zheng Yan, Yafeng Li, Xiaokang Zhou, and Shohei Shimizu. 2021. Privacy preservation in permissionless blockchain: A survey. *Digit. Commun. Netw.* 7, 3 (2021), 295–307.
[112] Maharage Nisansala Sevwandi Perera, Toru Nakamura, Masayuki Hashimoto, Hiroyuki Yokoyama, Chen-Mou Cheng, and Kouichi Sakurai. 2022. A Survey on Group Signatures and Ring Signatures: Traceability vs. Anonymity. *Cryptography* 6, 1 (2022).
[113] Nadia Pocher and Andreas Veneris. 2022. Privacy and Transparency in CBDCs: A Regulation-by-Design AML/CFT Scheme. *EEE Trans. Netw. Serv. Manag.* 19, 2 (2022), 1776–1788.
[114] Andrew Poelstra. 2016. *Mimblewimble.* Technical Report. The University of Rostock.
[115] David Pointcheval and Olivier Sanders. 2016. Short Randomizable Signatures. In *Proc. of CT-RSA'16.* Springer, 111–126.
[116] Xianrui Qin, Shimin Pan, Arash Mirzaei, Zhimei Sui, Oğuzhan Ersoy, Amin Sakzad, Muhammed F. Esgin, Joseph K. Liu, Jiangshan Yu, and Tsz Hon Yuen. 2023. BlindHub: Bitcoin-Compatible Privacy-Preserving Payment Channel Hubs Supporting Variable Amounts. In *Proc. of IEEE S&P'23.* 2462–2480.
[117] Chaitanya Rahalkar and Anushka Virgaonkar. 2024. Summarizing and Analyzing the Privacy-Preserving Techniques in Bitcoin and other Cryptocurrencies. arXiv:2109.07634
[118] Reserve Bank of Australia. 2023. *Australian CBDC Pilot for Digital Finance Innovation.* Technical Report. RBA.
[119] Ronald L. Rivest, Leonard Adleman, and Michael L. Dertouzos. 1978. On Data Banks and Privacy Homomorphisms. In *Foundations of Secure Computation.* Academic Press, 169–179.
[120] Ronald L. Rivest, Adi Shamir, and Yael Tauman. 2001. How to Leak a Secret. In *Proc. of ASIACRYPT'01.* Springer, 552–565.
[121] Dorit Ron and Adi Shamir. 2013. Quantitative Analysis of the Full Bitcoin Transaction Graph. In *Proc. of FC'13.* Springer, 6–24.
[122] Tim Ruffing and Pedro Moreno-Sanchez. 2017. ValueShuffle: Mixing Confidential Transactions for Comprehensive Transaction Privacy in Bitcoin. In *Proc. of FC'17.* Springer, 133–154.
[123] Tim Ruffing, Pedro Moreno-Sanchez, and Aniket Kate. 2014. CoinShuffle: Practical Decentralized Coin Mixing for Bitcoin. In *Proc. of ESORICS'14.* Springer, 345–364.
[124] Tim Ruffing, Pedro Moreno-Sanchez, and Aniket Kate. 2016. P2P Mixing and Unlinkable Bitcoin Transactions. IACR ePrint 2016/824. https://eprint.iacr.org/2016/824
[125] R. Sai Anand and C.E.Veni Madhavan. 2000. An Online, Transferable E-Cash Payment System. In *Proc. of INDOCRYPT'00.* Springer, 93–103.
[126] Hamza Saleem, Amir Ziashahabi, Muhammad Naveed, and Salman Avestimehr. 2024. Hawk: Accurate and Fast Privacy-Preserving Machine Learning Using Secure Lookup Table Computation. arXiv:2403.17296
[127] Tomas Sander and Amnon Ta-Shma. 1999. Auditable, Anonymous Electronic Cash. In *CRYPTO'99.* Springer, 555–572.
[128] Amirreza Sarencheh, Aggelos Kiayias, and Markulf Kohlweiss. 2022. PEReDi: Privacy-Enhanced, Regulated and Distributed Central Bank Digital Currencies. IACR ePrint 2022/974.
[129] Amirreza Sarencheh, Aggelos Kiayias, and Markulf Kohlweiss. 2025. PARScoin: A Privacy-preserving, Auditable, and Regulation-friendly Stablecoin. In *Proc. of CNS'25.* Springer Nature Singapore, 289–313.
[130] István András Seres, Dániel A. Nagy, Chris Buckland, and Péter Burcsi. 2019. MixEth: efficient, trustless coin mixing service for Ethereum. IACR ePrint 2019/341.
[131] Srinath Setty. 2019. Spartan: Efficient and general-purpose zkSNARKs without trusted setup. IACR ePrint 2019/550.
[132] Adi Shamir. 1985. Identity-Based Cryptosystems and Signature Schemes. In *CRYPTO'84.* Springer, 47–53.
[133] Ruisheng Shi, Yulian Ge, Lina Lan, Zhiyuan Peng, Shenwen Lin, and Lin Li. 2024. Deanonymizing Transactions Originating from Monero Tor Hidden Service Nodes. In *Companion Proc. of WWW'24.* ACM, 678–681.
[134] Alberto Sonnino, Mustafa Al-Bassam, Shehar Bano, Sarah Meiklejohn, and George Danezis. 2020. Coconut: Threshold Issuance Selective Disclosure Credentials with Applications to Distributed Ledgers. arXiv:1802.07344
[135] Xiaoqiang Sun, F. Richard Yu, Peng Zhang, Zhiwei Sun, Weixin Xie, and Xiang Peng. 2021. A Survey on Zero-Knowledge Proof in Blockchain. *IEEE Network* 35, 4 (2021), 198–205.
[136] Alin Tomescu, Adithya Bhat, Benny Applebaum, Ittai Abraham, Guy Gueta, Benny Pinkas, and Avishay Yanai. 2022. UTT: Decentralized Ecash with Accountable Privacy. IACR ePrint 2022/452.
[137] Muoi Tran, Loi Luu, Min Suk Kang, Iddo Bentov, and Prateek Saxena. 2018. Obscuro: A Bitcoin Mixer using Trusted Execution Environments. In *Proc. of the ACSAC'18.* ACM, 692–701.
[138] Luke Valenta and Brendan Rowan. 2015. Blindcoin: Blinded, Accountable Mixes for Bitcoin. In *Proc. of FC'15.* Springer, 112–126.
[139] Nicolas Van Saberhagen. 2013. CryptoNote v 2.0. https://cryptonote.org/whitepaper.pdf.

[140] Friedhelm Victor. 2020. Address Clustering Heuristics for Ethereum. In *Proc. of FC'20*. Springer, 617–633.
[141] Dan Wang, Jindong Zhao, and Yingjie Wang. 2020. A Survey on Privacy Protection of Blockchain: The Technology and Application. *IEEE Access* 8 (2020), 108766–108781.
[142] Zhipeng Wang, Marko Cirkovic, Duc V. Le, William Knottenbelt, and Christian Cachin. 2023. Pay Less for Your Privacy: Towards Cost-Effective On-Chain Mixers. IACR ePrint 2023/1222.
[143] Karl Wüst, Kari Kostiainen, Vedran Čapkun, and Srdjan Čapkun. 2019. PRCash: Fast, Private and Regulated Transactions for Digital Currencies. In *Proc. of FC'19*. Springer, 158–178.
[144] Karl Wüst, Kari Kostiainen, Noah Delius, and Srdjan Capkun. 2022. Platypus: A Central Bank Digital Currency with Unlinkable Transactions and Privacy-Preserving Regulation. In *Proc. of ACM SIGSAC CCS'22*. ACM, 2947–2960.
[145] Liang Xue, Dongxiao Liu, Jianbing Ni, Xiaodong Lin, and Xuemin Sherman Shen. 2023. Enabling Regulatory Compliance and Enforcement in Decentralized Anonymous Payment. *IEEE Trans. Dependable Secure Comput.*. 20, 2 (2023), 931–943.
[146] Qiming Yu, Shilei Liao, Lianhai Wang, Yong Yu, Lingyue Zhang, and Yanqi Zhao. 2024. A regulated anonymous cryptocurrency with batch linkability. *Comput. Stand. Interfaces* 87 (2024), 103770.
[147] Lingyue Zhang, Huilin Li, Yannan Li, Yong Yu, Man Ho Au, and Baocang Wang. 2019. An efficient linkable group signature for payer tracing in anonymous cryptocurrencies. *Future Gener. Comput. Syst.* 101 (2019), 29–38.
[148] Tao Zhang. 2023. Privacy Evaluation of Blockchain Based Privacy Cryptocurrencies: A Comparative Analysis of Dash, Monero, Verge, Zcash, and Grin. *IEEE Trans. Sustain. Comput.* 8, 4 (2023), 574–582.
[149] Sheng Zhong and Abdullah Mueen. 2024. BitLINK: Temporal Linkage of Address Clusters in Bitcoin Blockchain. In *Proc. of ACM SIGKDD KDD'24*. ACM, 4583–4594.
[150] Jan Henrik Ziegeldorf, Roman Matzutt, Martin Henze, Fred Grossmann, and Klaus Wehrle. 2018. Secure and anonymous decentralized Bitcoin mixing. *Future Gener. Comput. Syst.* 80 (2018), 448–466.
[151] Guy Zyskind, Oz Nathan, and Alex Pentland. 2015. Enigma: Decentralized Computation Platform with Guaranteed Privacy. arXiv:1506.03471

Supplementary Material to:

A Hitchhiker's Guide to Privacy-Preserving Cryptocurrencies: A Survey on Anonymity, Confidentiality, and Auditability

MATTEO NARDELLI, FRANCESCO DE SCLAVIS, MICHELA IEZZI, Bank of Italy, Italy

A Appendix

This appendix provides supplementary material for *A Hitchhiker's Guide to Privacy-Preserving Cryptocurrencies: A Survey on Anonymity, Confidentiality, and Auditability*, which complements the main analysis but is excluded from the core sections to preserve narrative flow. Specifically, it includes: *(i)* a breakdown of publication venues for the works reviewed in this survey, and *(ii)* a summary of mixing protocols referenced in Section 4.3.5. These materials are presented as tables and offer additional context and technical detail for readers interested in the broader research landscape and protocol-level design trade-offs. Furthermore, in Section A.3, we briefly present interesting yet underexplored cryptographic tools that can be used to design privacy-preserving digital currencies.

A.1 Most Popular Publication Venues

Table 4 summarizes the most common publication venues among the reviewed works. We report only the most frequent venues (with more than 2 works), including journals (e.g., IEEE Transactions on Dependable and Secure Computing, Future Generation Computer Systems), international conferences (e.g., Financial Cryptography and Data Security (FC), ACM Conference on Computer and Communications Security (CCS), IEEE Symposium on Security and Privacy (S&P)), and other (non-peer reviewed) sources, such as preprints and whitepapers.

Table 4. Most popular publication venues among the reviewed works.

Venue Type	**# of Publications**	**Frequent Venues** ($\geq$ 2 works)
Journal	13	IEEE Trans. Dependable Secur. Comput (5) Fut. Gen. Comp. Sys. (2)
Conference	30	FC (7) ACM CCS (5) IEEE S&P (3) AFT (2) ASIACRYPT (2) CRYPTO (2) ESORICS (2)
Other	16	IACR Cryptol. ePrint Arch. (7) arXiv (2) Whitepapers (7)

A.2 Reviewed Mixing Protocols

Similar to Table 2, which surveys key privacy-preserving digital currencies, Table 6 focuses specifically on mixing protocols. The corresponding legend in Table 5 extends the notation introduced in Table 1. These mixing protocols are discussed in Section 4.3.5, and the table serves as a reference to facilitate comparison across existing approaches in the literature.

Table 5. Additional labels used in Table 6 beyond those defined in Table 1.

Group	Legend			
Protocol				
No-Trust Assumption	●	Accountable		
Other				
Cryptographic	DC-net	Dining cryptographers network	HDC-net	Harmonized DC-net
↪ Primitives	MPC	Multi-party computation	TEE	Trusted Execution Environment
	VS	Verifiable shuffling		

Table 6. Classification of existing mixing protocols (see Table 5 for a legend).

		Protocol					*Privacy*				*Other*
Paper	**Year**	**Decentralized**	**Blockchain Compatibility**	**No-Trust Assumptions**	**Implemented**	**Different Denominations**	**Sender Anonymity**	**Receiver Anonymity**	**Transaction Value Confidentiality**	**Unlinkability**	**Cryptographic Primitives**
CoinJoins [98]	2013	✓	✓	✓	×	×	×	×	×	◐	DS
Xim [24]	2014	✓	✓ (B)	✓	×	✓	×	×	×	✓	PE
Mixcoin [30]	2014	×	✓ (B)	●	×	×	×	×	×	✓	DS
CoinShuffle [123]	2014	✓	✓ (B)	✓	×	×	×	×	×	✓	DS, PE
Blindcoin [138]	2015	×	✓ (B)	●	×	×	×	×	×	✓	BS
TumbleBit [76]	2016	×	✓ (B)	✓	×	✓	✓	✓	×	✓	MPC, ZK
CoinShuffle++ [124]	2016	✓	✓ (B)	✓	✓[30]	×	✓	✓	×	✓	DC-net
SecureCoin [79]	2017	✓	✓ (B)	×	✓	×	×	✓	×	✓	SA
ValueShuffle [122]	2017	✓	✓ (B)	✓	✓	✓	✓	✓	✓	✓	PE, SA
Liu et al. [93]	2018	×	✓	✓	×	×	×	×	×	✓	RS
Möbius [99]	2018	✓	✓ (E)	✓	✓	×	✓	✓	×	✓	RS,SA
Obscuro [137]	2018	×	✓ (B)	✓	✓[31]	×	×	×	✓	✓	TEE
CoinParty [150]	2018	✓	✓ (B)	✓	×	×	×	×	×	✓	DS^t, Mixnets, MPC
MixEth [130]	2019	✓	✓ (E)	✓	✓[32]	×	×	×	×	✓	VS
AMR [87]	2021	✓	✓ (E)	×	✓	×	×	×	×	✓	ZK
UCoin [108]	2023	✓	✓ (B)	✓	×	×	×	×	×	✓	HDC-net
BlindHub [116]	2023	×	✓ (B)	✓	✓[33]	✓	×	×	×	✓	BS*
Wang et al. [142]	2023	✓	✓	✓	✓	×	×	×	×	✓	MT

A.3 Emerging Cryptographic Tools for Future Digital Currencies

In addition to the cryptographic tools discussed in the main survey, several advanced primitives offer promising tools for future research in privacy-preserving digital currencies. Although not yet widely adopted in practical systems, these techniques may enable more flexible or robust privacy features in regulated or decentralized contexts.

[30] https://github.com/decred/cspp
[31] https://github.com/BitObscuro/Obscuro
[32] https://github.com/seresistvanandras/MixEth
[33] https://github.com/blind-channel/blind-hub

A.3.1 Anonymous Credentials. Systems like Idemix [154] and Coconut [134] allow users to prove possession of attributes (e.g., age, residency, KYC compliance) without revealing their identity. These systems support selective disclosure, enabling users to present only the minimal information required for a given interaction; for example, proving age above 18 without disclosing birthdate. This selective transparency could be very important for digital currencies subject to regulatory constraints (e.g., AML, KYC), as users can demonstrate compliance without full deanonymization. Moreover, advanced anonymous credential schemes could support unlinkability, preventing different credential uses from being correlated. However, at the time of writing, efficiently revoking credentials (e.g., after misuse or expiration) while preserving privacy is nontrivial and typically requires accumulator-based techniques or dynamic revocation protocols. As digital currencies increasingly intersect with real-world identity frameworks, anonymous credentials offer a promising path for balancing privacy and regulatory needs.

A.3.2 Cryptographic Accumulators. Accumulators provide compact representations of sets along with efficient membership proofs. They are useful for constructing privacy-preserving mechanisms such as spent coin tracking, revocation, and anonymous group membership verification. Although some systems already use Merkle trees to prove coin or user inclusion in a list of valid or authorized entities (e.g., [56, 72, 127, 142, 145]), we believe that accumulators remain underutilized—particularly in the design of revocation mechanisms, where they could offer efficient and privacy-preserving solutions.

A.3.3 Functional Encryption. Functional encryption [153] allows specific functions to be computed on encrypted data without revealing the underlying inputs. In contrast to traditional public-key encryption, where decryption reveals the entire message, functional encryption supports function-restricted decryption keys. In the context of digital currencies, this could enable selective disclosure, e.g., allowing regulators to verify policy compliance or transaction conditions without full access to transaction details. For instance, a regulator could decrypt only whether a transaction exceeds a policy-defined threshold without learning the actual value. Potential applications may include: *(i)* allowing a central bank to verify that a transaction is below a predefined limit, without revealing the amount; *(ii)* enabling eligibility verification for financial services based on encrypted KYC attributes; and *(iii)* constructing privacy-preserving tax collection mechanisms that expose only aggregate (or policy-relevant) outcomes. Despite its promise, functional encryption remains largely theoretical for high-complexity functions and faces efficiency and key management challenges in practical deployments.

A.3.4 Time-Lock Encryption. Time-lock encryption [156] delays access to encrypted information until a predefined time has elapsed. Potential applications include time-delayed auditability, sealed transactions, or rate-limited disclosure of account (sensitive) data.

A.3.5 Witness Encryption. In its canonical form, witness encryption [155] allows one to encrypt a message with respect to an instance x of a (nondeterministic polynomial time NP) language, $L \in NP$, such that decryption is possible if and only if the decryptor possesses a valid witness w for the statement $x \in L$. This capability opens the door to a number of expressive access control mechanisms that could be beneficial in the context of digital currencies. One promising application is *signature-based witness encryption*, where the NP statement encodes a threshold condition over a set of public keys. For example, the instance x could describe a set of n public keys, and the corresponding witness would be any subset of k valid signatures under distinct keys. This effectively enables a form of threshold encryption—where decryption requires authorization from at least k participants—without any setup phase for correlated key generation or key sharing, unlike traditional threshold crypto-systems. While a downside is that ciphertext size can grow with the

complexity of the underlying relation (e.g., linear in n), recent constructions have demonstrated ciphertexts of size $O(\log n)$, under specific assumptions (i.e., indistinguishability obfuscation for Turing machines), making this approach increasingly practical [152]. Another line of research could explore *witness encryption with hidden statements*, where the encrypted message can only be decrypted by someone knowing a witness for a secret statement, while the statement itself remains concealed. This property could offer a powerful tool for regulatory compliance. For example, policy enforcement could be realized by requiring users to provide ZKPs of compliance (i.e., knowledge of a valid witness), without revealing the rule being enforced. Such a mechanism may be appealing to regulators who wish to enforce certain constraints (e.g., AML/KYC policies, tracing) without disclosing the exact criteria (e.g., [157]). To this end, users must include a secret witness with each transaction, attesting to their compliance with the policy. To conclude, witness encryption could provides a rich foundation for designing advanced privacy and access control features, especially in contexts where minimal trust assumptions, fine-grained control, and privacy coexist.

These emerging primitives highlight exciting directions for foundational research in the design of privacy-preserving digital currencies. While most remain experimental, their potential to address regulatory, architectural, or usability gaps makes them worth further exploration.

Additional References

[152] Gennaro Avitabile, Nico Döttling, Bernardo Magri, Christos Sakkas, and Stella Wohnig. 2024. Signature-based Witness Encryption with Compact Ciphertext. IACR ePrint 2024/1477.

[153] Dan Boneh, Amit Sahai, and Brent Waters. 2010. Functional Encryption: Definitions and Challenges. Cryptology ePrint Archive, Paper 2010/543. https://eprint.iacr.org/2010/543

[154] Jan Camenisch and Els Van Herreweghen. 2002. Design and implementation of the idemix anonymous credential system. In *Proc. of ACM SIGSAC CCS'02*. ACM, 21–30.

[155] Sanjam Garg, Craig Gentry, Amit Sahai, and Brent Waters. 2013. Witness Encryption and its Applications. Cryptology ePrint Archive, Paper 2013/258. https://eprint.iacr.org/2013/258

[156] Jia Liu, Tibor Jager, Saqib A. Kakvi, and Bogdan Warinschi. 2018. How to build time-lock encryption. *Designs, Codes and Cryptography* 86, 11 (2018), 2549–2586.

[157] Changrui Mu and Prashant Nalini Vasudevan. 2024. Instance-Hiding Interactive Proofs. IACR ePrint 2024/776.