TAKING THE NEXT STEP

WHAT IS THE FUTURE OF STABLECOINS, AND HOW DO WE GET THERE?

KING'S College LONDON
KING'S BUSINESS SCHOOL
QATAR CENTRE FOR GLOBAL BANKING & FINANCE

TAKING THE NEXT STEP:

WHAT IS THE FUTURE OF STABLECOINS, AND HOW DO WE GET THERE?[1]

Rhys Bidder

KING'S College LONDON | KING'S BUSINESS SCHOOL | QATAR CENTRE FOR GLOBAL BANKING & FINANCE

[1] The views expressed in this document and all errors and omissions should be regarded as those of the author and not necessarily those of the Bank of England, the Eurosystem, the Central Bank of Ireland, Qatar Central Bank, or Chainlink Labs. The author is an advisor to Chainlink Labs (Banking and Capital Markets) and thanks Angie Walker, Thomas Chanter, Kostiantyn Dmitriiev, Anthony Butler, Jennifer Peve, Ian Salmon, and Tahreem Kampton for useful comments.

CONTENTS

INTRODUCTION

Stablecoins are often depicted as the first 'killer app' in blockchain-based finance. Born of the need for reliable on-chain money - stable in value, efficient as a medium of exchange and denominated in familiar units of account - the expansion of stablecoin usage has been rapid and appears likely to continue. With advances in tokenization, increased regulatory clarity in important jurisdictions, and further enhancements in blockchain technology, the coming years promise exciting opportunities for the sector. However, digital assets in general, and stablecoins in particular, have experienced false dawns before. Is this time different? Perhaps a better question is to ask: what must be done to ensure that this time *is* different? What can be done to accelerate the hoped-for progress *and* to ensure that it is sustainable?

Many success stories in on-chain finance have stemmed from effective coordination. An excellent example is in the use of well designed 'standards' that provide a focal point for research and development, prevent duplicated effort on multiple similar projects, and promote positive network effects. They allow a common abstraction without imposing implementation details in a centralized manner and without *obliging* adoption in the way a centralized *dictat* might. The ERC-20 standard is an especially good example as it has successfully underpinned the early growth of stablecoin issuance, thanks to its parsimonious and flexible structure that aligns well with the intrinsically fungible nature of stablecoins and the fundamental simplicity of 'cash' on chain. It has also played an important role in promoting the development of composable smart contract-based services that make use of stablecoins - such as decentralized exchanges and lending protocols - in addition to simply enabling P2P payments.

Is it time to propose a 'technical standard' for stablecoins? Perhaps - but in this note we are more agnostic. We advocate for a more *standardized approach* for stablecoins, rather than literally a new technical standard. As we will discuss, in *some* dimensions there is a high degree of consensus on what is best practice. In these dimensions it may be reasonable now to assert technical standards. However, in some of the most important dimensions of stablecoin design - notably in their handling of compliance requirements, or in privacy-enhancing functionality - we are not yet at the point where a technical standard can be confidently asserted. Even in these areas, though, the picture is becoming clearer, and we make some recommendations - or at least predictions - of what are promising avenues for stablecoins to pursue in the future.[2]

Another reason not to take a stance on a specific *technical* standard is that various stablecoin issuers mint their tokens on a wide variety of chains, where many of our abstract recommendations will be

[2] Until key technical and conceptual debates are settled, attempting premature standardization may inhibit adoption of a standard by alienating segments of the developer population, or by mistakenly embedding technical features that ultimately are superseded. As such, we distinguish elements of stablecoin design that arguably are already settled, and ripe for standardization, from those elements that could be standardized in the future but are not yet quite ready.

relevant, but particular technical specifications may be problematic. Clearly, it is vital that a stablecoin should be consistent with multiple chains - enabling the interoperability that will be key in the scaling of on-chain finance in coming years.

Any standardized approach should also be compatible with stablecoins operating in environments ranging from highly decentralized contexts to on-chain centralized finance. As many stablecoins aspire to be used in regulated environments, functionality that promotes trust, good governance and transparency is key. Of course, in many cases the same sort of functionality that a regulator would seek is in fact something that users will typically also demand. Any institution that aims to issue a money (as stablecoins are) must confront the inherent risks and fragility that arise from the possibility of 'bank runs', in addition to various operational and counterparty risks. As such, at the high level of abstraction we will be working at, there is little tradeoff between designing a stablecoin for heavily regulated environments and designing one that could be used in a regulation-lite or even an unregulated context. Defi does not - or should not - mean a lack of safeguards. There should be a common core of functionality that serves both tradfi and defi well, even if that core functionality may eventually be implemented or extended in different ways in particular markets or jurisdictions.

In this note, we will emphasize 7 dimensions of focus, listed as D1-D7 below in box 1. These dimensions can be considered separately, though they are not completely independent, and do overlap and interact in some respects.

D1 **Monetary policy:** Minting/burning, tracking circulating coins, and depositing/withdrawing reserve assets.
D2 **Verifying reserve assets:** Timely and reliable information on the value and composition of reserve assets.
D3 **Compliance:** Functionality focusing on identity, but also more elaborate constraints on business logic.
D4 **Cross-chain interoperability:** Functionality that might be added to stablecoins to promote interoperability.
D5 **Privacy:** Privacy enhancing technologies within the stablecoin, or reliance on third party services/protocols.
D6 **Fees and yield:** Stablecoin business models seem likely to evolve to monetize/incentivize some activities.
D7 **Roles and events:** Assigned roles for key tasks avoid single points of failure, and events promote transparency.

Box 1: Key dimensions for standardization in stablecoins.

Clearly, D1 and D2 are especially important in distinguishing stablecoins from a generic fungible token. Combined with core ERC-20 functionality they help characterize the two sides of the stablecoin balance sheet. What coins are outstanding, and what they are backed by, are intrinsic properties of a collateralized stablecoin. These key dimensions are discussed in section 2 and also feature prominently when we discuss D3 (Compliance) and D5 (Cross-chain interoperability) in section 3 and section 4.

There have been attempts to standardize compliance functionality among a broader class of tokens (see the ERC-3643 standard, for example). Some aspects of these approaches could - and likely should - be embedded in a standard for stablecoins (D3). However, as discussed in section 3, while promising progress in regulatory standardization is being made, in various pilots and jurisdictions, it is still very difficult to anticipate how stablecoins should best implement compliance functionality. In analyzing existing proposed solutions - their strengths and shortfalls - we point the way to possibly improved approaches in the future.

Privacy enhancement and confidentiality (D4) are hot topics across the blockchain space. The ability to transact is core to a stablecoin's functionality, while exposing details of the transaction is not. Exciting work is being done to embed privacy enhancements into stablecoins' core methods, rather than relying on auxiliary services and smart contracts. Section 5 discusses some of the most promising avenues for privacy-enhanced stablecoins. We argue that while there is some commonality across the approaches there is far too much divergence in interfaces and approaches to confidently standardize privacy functionality within stablecoins' own implementations. As such, in the *near* future, protocols that operate on stablecoins, rather than the coins themselves, are likely to provide the desired privacy-enhancing functionality.

Important policy and regulatory bodies emphasize interoperability (D5) in payments - see the BIS' finternet ambitions and the G20 cross border payments roadmap, for example. Furthermore, increasing numbers of financial institutions are deploying their own permissioned blockchains that nevertheless have the capacity to interact with other - possibly public - chains. Interoperability is key to providing a sufficiently large 'pool of liquidity' of the sort that tradfi institutions and their users have come to expect and will continue to demand. Looking beyond tradfi adoption of stablecoins, interoperability is also key to the continued development of defi. Section 4 examines what, if any, functionality is required by stablecoins to promote interoperability, concluding that mint and burn functionality appears to be essentially all that is required for a broad class of cross-chain protocols. In fact, these functions (as implied by D1, discussed in section 2) are defining features of *any* stablecoin even if it were to be issued only on a single chain. We go on to argue that some of the other functions alluded to in section 2 might also be important in making stablecoins particularly easy to incorporate into robust locking protocols, where patterns such as lock/mint are used, rather than burn/mint. A key element of robust cross-chain issuance of a stablecoin is careful accounting of where tokens are locked on one chain, while circulating on other chains.

Section 6 considers yield generation and fees (D6). This is a timely question as yield bearing stablecoins are beginning to emerge as stablecoin business models evolve in response to an increasingly competitive digital money environment. Tokenized money markets funds and tokenized bank deposits may come to provide stiff competition, particularly as the tightening of KYC/AML burdens on regulatory compliant stablecoins may inhibit their appeal to their early adopters and raise compliance costs. The tentative conclusion from this section is that there are already some promising approaches to implementing yield generation, but that these approaches are still fairly novel and as such cannot be confidently said to be 'best practice'. Furthermore, there is still extreme distrust of the yield-generating model among regulators in important jurisdictions. Regarding *fees*, we argue that a generic and parsimonious approach to fee structures could help promote trust and user protection among stablecoins, while allowing some aspects of stablecoin functionality to generate revenue for issuers (or some other designated operator).

Finally, in section 7 we briefly discuss the perhaps secondary - but still important - issue of how to assign responsibilities for certain functionality to addresses with particular roles, and how to use events to further enhance transparency and auditability (D7). Although rarely discussed, these

areas relate to the sort of good governance and user protections that a maturing stablecoin sector will likely need to offer to become the cash leg for institutional tokenized asset trading. For stablecoins intended for regulated environments, clear accountability, responsibility and real-time monitoring are especially important features.

We do not claim that these seven dimensions are exhaustive, or that within these dimensions our suggested functionality is the final word or the most efficient approach. Instead, we hope to start a conversation on standardization that issuers, users, platforms and regulators will participate in.

STABLECOIN BALANCE SHEETS

We begin by considering functions that are core to any stablecoin - minting and burning. We then consider functions that are perhaps less general, but which are key to many collateralized stablecoins - that is, functionality for depositing and withdrawing backing assets. These are likely to become more prominent as high quality backing assets emerge on chain (tokenized money markets funds, natively on-chain sovereign bonds, and possibly wholesale CBDCs). We then go on to consider how reserve assets should be monitored and verified, and discuss how a comprehensive cross-chain and off-chain view of both sides of the stablecoin balance sheet is vital. Converging around standardized best practice in these areas will be pivotal to the success of stablecoins.

MINT AND BURN

Essentially all prominent stablecoins feature methods to mint or to burn tokens, though they vary in who can mint or burn. In some cases, minting can only be done by pre-approved addresses (issuers or approved third parties assigned some sort of 'mint' role). In other cases - with less centralized stablecoin designs - members of the broader community of blockchain users can mint, provided that they can deposit adequate collateral or backing assets in appropriate smart contracts.

It is almost redundant to state that any sensible approach to stablecoin design should require a mint and a burn function, with associated events, as these actions are intrinsic to stablecoins. This is the case even if a stablecoin is designed to operate on a single chain but, as discussed in section 4, mint/burn functionality is also key to allowing common cross-chain interoperability.

DEPOSIT AND WITHDRAW

For stablecoins with off-chain backing assets there is no need for the stablecoin to implement methods for depositing and withdrawing backing assets.[3] However, on-chain assets of the quality

[3] There may be off-chain interaction between a minter and a counterparty wishing to create stablecoins (by paying with bank account funds or posting Treasuries to back the creation, say). This pattern is currently common among

necessary to act as backing assets for regulated stablecoins, or for coins aiming for institutional adoption, are becoming more prevalent. For example, tokenized deposits, tokenized MMF and perhaps even CBDC and natively on-chain sovereign debt may soon offer the sort of backing asset that stablecoins could rely upon for 'on-chain' depositing and withdrawal (see Project Helvetia III and the mooted digital gilt).

The act of depositing or withdrawing backing assets is conceptually related to, but distinct from, the act of minting and burning tokens. Clearly, it is often natural for the actions to be bound together tightly - stablecoins should only circulate if they are backed. However, one can envisage reasonable situations where one might want them to be distinguished. Suppose a regulator or the user community demands that backing assets be adjusted, without any change in the number of outstanding stablecoins. For example, one could envisage precautionary actions being taken by an issuer if there were a concern with a non-trivial fraction of existing reserve assets, perhaps from some concern over a particular custodian of 'existing' reserve assets (somewhat akin to a margin call). It could also be that an issuer mints coins to its treasury but separately deposits backing assets before the coins are actually 'issued' into circulation. One could also envisage a burn of tokens happening separately from withdrawal of assets if an equivalent number of tokens is to be minted on another chain, as part of a bridging protocol - in this case there is no economic reason for any change to reserve assets. These examples all serve to emphasize that burning/minting and depositing/withdrawing can usefully be separated.

Any concern that withdrawal functionality could lead to reserve assets being drained without a burn can be allayed with various approaches. First, it could be that withdrawal methods could be limited to a maximum amount or prevented from being called without an equivalent burn, except under tightly specified legitimate circumstances. Second, access control can be enforced such that only approved addresses are granted 'roles' necessary to call these methods, with those roles being enforced through function modifiers or through require statements in the body of the methods' sourcecode. We discuss access control in greater detail in section 7 where we delve deeper into administrative roles.

CIRCULATING COINS

A key challenge to stablecoins is reliable accounting of their circulating coins in a multi-chain context. We emphasize *circulating* as some methods for cross-chain issuance rely on locking coins in escrow accounts as part of a bridging protocol, while minting or unlocking an equivalent number of coins on another chain - a lock/unlock or lock/mint pattern. A simple example will demonstrate some of the complexities that arise from this situation. Suppose 100 coins are initially minted by the coin 'issuer' on Ethereum. In the background, we assume that there is at least 100 'dollars' worth of

centralized stablecoins, such as USDT and USDC. Tether mints/burns USDT and, ostensibly, ensures that appropriate backing assets are 'deposited'/'withdrawn' accordingly. Circle permits approved third parties to mint, under the Circle Mint scheme.

reserve assets - perhaps with some excess buffer. Let us assume there is $105 worth of reserve assets. For now, the reserve assets are not the focus - except we note that throughout this example, we take them, and their value, as given.

At this point, if one uses the familiar ERC-20 method 'totalSupply', that method will return 100. Now, suppose that the issuer wishes some of its coin - let's say 10 tokens - to be circulating on a different chain, perhaps on Base. Conceptually, there are a few ways in which this might be done. If 10 tokens are bridged to Base with a burn/mint approach then that means that 10 tokens will be destroyed on Ethereum (after which totalSupply called on the stablecoin smart contract deployed on Ethereum will return 90) and 10 will be minted on Base (after which the totalSupply function called on the stablecoin smart contract deployed on Base will return 10). This all makes sense. Under an alternative lock/mint approach, 10 tokens would be sent to an escrow smart contract on Ethereum - perhaps part of a bridge - which (hopefully reliably) 'locks' up a balance of 10 tokens on Ethereum so that it cannot be transferred. In a sense, on Ethereum, these coins are no longer circulating. With this being done, 10 coins can be minted on Base while leaving the total number of coins that are actually circulating unchanged at 100. Note that totalSupply would continue to return 100 on Ethereum (and 10 on Base).

It is key to understand that the reliability of the locking approach is fundamental to the correctness of the stablecoin model. If the coins somehow were not reliably locked and remained circulating on Ethereum, then the stablecoin would no longer be fully collateralized. We would have 110 coins circulating and only $105 assets to back them. Coins that had supposedly been locked could be used to spend, when the coins issued (on Base) under the assumption that the other coins had been locked are *also* available to spend.

It is vital that cross-chain coin issuance is done carefully and *seen* to be done carefully. To some extent this is a matter of process. Prominent issuers will typically use canonical bridges that rely on secure cross-chain messaging and interoperability protocols. As discussed further in section 4, these systems may usefully be abstracted away with the use of cross-chain token standards that give the impression of a single coin operating across multiple chains. The latter model is attractive partly because it avoids some of the pitfalls that other approaches to bridging, such as liquidity fragmentation arising from subtly different versions of the same coin being issued on a given chain, arising from different bridges, even if they remain backed by the same claims on reserve assets.

It is important that it is clear which 'lockers' are being used so that users - and oracle services such as Chainlink Proof of Reserve (PoR) - can easily and efficiently obtain a transparent accounting of the coins that are actually circulating on chain. This makes it easier to ensure that the issuer - or a nefarious party - cannot debase the coin. Best practice for stablecoin protocols is to make explicit in off-chain documentation what addresses they (or parties with roles that have power to bridge assets

with the issuer's blessing) use for 'official' locking of tokens.[4] They would also specify which chains these addresses relate to. Aside from addresses used in cross-chain operations, other types of addresses that control non-circulating coins should also be documented, such as treasury addresses which contain coins that have been minted, but not yet released into circulation.[5]

Arguably, it is not only off-chain where this information should be provided. Stablecoins could - and perhaps should - provide functions in their smart contracts that provide up to date information on what addresses are used as lockers, so that users and oracles can programmatically always obtain a comprehensive view of the circulating coins. Even if someone is calling a function 'locally' (on a particular chain), they should be able to obtain 'global' information about circulating coins on all chains. As noted above, in the lock/mint case, totalSupply called on Ethereum does not provide a complete view of the circulating coins. It is relevant information to know whether 90 coins are circulating on Ethereum or 100 (and in association with that, where else coins are circulating).

One could imagine a new method, 'totalCirculating' that returns the number of 'freely circulating' (not locked, not in Treasury) coins, aggregated across all chains. We might also wish to have an another 'overloaded' version of the function 'totalCirculating(chainID)' returning the number of coins circulating *on a given chain*, identified by the input 'chainID'. These functions should be included in the smart contracts that control the stablecoin, on whatever chain it is issued on.

By combining totalCirculating and totalSupply, a user could quickly obtain an accurate view of the distribution of coins across chains - a comprehensive, global view of coins that are circulating, which is different from coins that are issued. Furthermore, since canonical bridges, escrow contracts and treasury addresses typically are not (and probably should not be) too numerous, it seems reasonable for a stablecoin contract to provide functionality to obtain a complete list of the addresses used, and the balances locked at each address. Of course, there should be consistency between the balances at these addresses and the numbers returned by totalSupply and totalCirculating on a given chain. Naturally, a reliable source of 'off-chain' (in fact, from other chains) data would be required to ensure credible and timely information can be obtained about coins circulating on other chains, either explicitly, through services such as Chainlink's Cross-chain Interoperability Protocol (CCIP) or as part of a cross-chain standard, such as Chainlink's Cross-Chain Token (CCT).

[4] By 'official' we are distinguishing these activities from protocols or parties not affiliated with the issuer bridging the coins according to their own cross-chain protocols. This activity may be entirely legitimate but it is outside the remit of the issuer, and thus it is not their place to take these uses of their coin into account. For an interesting approach to some of these issues, see Bridged USDC Standard and its interaction with Chainlink CCIP.

[5] Although not explicitly in relation to stablecoins, the SEC has recently emphasized the importance of clarity on such features - specifically *'whether any of the supply is reserved for the network's treasury, particular uses or participants, and whether any portion of the supply is subject to vesting and/or lock-ups'*.

COIN VALUATION

Note that totalCirculating should return the notional *number* of coins outstanding, rather than the *value*. There are various reasons for this. First, stablecoins often are denominated in different ways. Second, there are important issues surrounding the pricing of stablecoins - even among those that are ostensibly pegged to fiat - typically pegged to $1.

Due to price variability (which is intended to be small - hence the name **stable**coins!) there is a difference between *value* of outstanding coins and the *number* (see Bidder (2025) for related discussions). We could envisage having a valCirculating method as part of a stablecoin's functionality. However, the price of the stablecoin is not an intrinsic property of the stablecoin, which is a bearer asset that trades (explicitly on exchanges or implicitly as a cash leg of a broader trade) at various prices at any one time, depending on context. The number and distribution of coins, however, *are* intrinsic.

Of course, the defining feature of stablecoins is that they exhibit a high degree of price stability around par. However, owing to the variety of pools and exchanges where they trade, combined with limits to arbitrage, there are deviations from par. Different users may wish to define a price feed in different ways - employing various oracle options, such as Chainlink Price Feeds, or their own preferred approach.

VALUATION, COMPOSITION AND VERIFICATION OF BACKING ASSETS

It is hard to overstate the importance of establishing reliably the adequate backing of collateralized stablecoins by reserve assets. This clearly requires oracle functionality in the case of off-chain reserve assets, but the problem is more general. For example, stablecoins backed by on-chain assets will typically also require some form of oracle service to obtain reliably aggregated price feeds for those assets. This should ensure an accurate valuation of the assets that can fund any redemption requests and, implicitly, maintain confidence in the stablecoin. This promotes its ability to trade at or (very) close to par in secondary markets.

It is striking that in extending ERC-20 functionality, few leading fiat-backed stablecoins include any methods that relate to describing or verifying reserve assets. However, one example of an issuer adopting this functionality is Wenia, the digital asset company from the Bancolombia Group, one of the largest financial conglomerates in Latin America, which uses Chainlink Proof of Reserve (PoR) to bring end-to-end transparency to the Colombian Peso reserves backing its COPW stablecoin. PoR is integrated directly into the stablecoin's minting function, helping to protect users against the risk of infinite mint attacks where additional COPW is issued without sufficient available reserves. Similarly, IDA is integrating this service to verify reserves for its HKDA stablecoin issuance (backed 1:1 by the Hong Kong dollar). True USD (TUSD) also connects its minting to PoR feeds, to enable 'Secure Mint' functionality.

Fundamentally, this functionality is designed to ensure coins can only be minted if there is adequate value of reserve assets to back the total supply of the token. Of course, there are two aspects to this: an accurate awareness of outstanding circulating coins (already discussed above) and the reserves that are implicitly backing them.

One of the defining elements of a collateralized stablecoin is its balance sheet. On the liability side, this is constituted by the circulating coins. On the asset side, the notional value of assets is necessary, but arguably not sufficient, for a full description of the information necessary to evaluate the coin. PoR functionality currently provides 'Proof of Value', which is extremely important. But 'Proof of Composition' is also an important dimension to consider: a notional value of $X, arising from a portfolio of 'risky' or poorly custodied assets, is very different from a notional value of $X, arising from a portfolio of 'safe' and well custodied assets.

The portfolio on the asset side of the 'stablecoin balance sheet' is a structured object - implicitly a mapping from assets to the balance of the assets held, and possibly other metadata (such as what custodian holds it, what ISIN or account number applies to the asset, its accounting treatment, the applicable jurisdiction/laws that apply to it,...). One implementation of this is given by Chainlink's Unified Golden Record - a data container embedded within a stablecoin's smart contract and synchronized across chains. For an example of custody and PoR interactions see the collaboration between Fireblocks and Chainlink Labs.

The bearer asset nature of stablecoins (and their short and somewhat blemished history) makes it especially important that they provide transparency about their assets going forward. In the (for now) absence of the sort of public deposit insurance that bank depositors can rely upon *and* in the presence of a floating secondary market value, there is arguably a greater propensity for 'bank runs' on the coins (see and). Emergency secured credit lines are not (yet) available for stablecoins in any jurisdiction (or indeed anything like the intraday and overnight credit that is offered to banks in less extreme situations). If these are ultimately to be made available to stablecoins then overall transparency will need to improve, to offset the moral hazard concerns that come with such schemes, and to reassure public authorities about the quality of collateral.[6] But this is not only an issue of pleasing regulators - bank runs should be a concern to any stablecoin holder, particularly if backing assets are (as they typically will be) less liquid than the coin itself, and its implicit promise of redemption at par.

How much portfolio information should be on chain, however, is somewhat unclear. Concerns about expensive storage on chain may limit the granularity of information that can be stored, as

[6] The traditional view of central bank or 'lender of last resort' duties in a liquidity, rather than a solvency crisis, is to lend freely against good collateral - possibly at a penalty rate. But often it is unclear whether a crisis is purely a liquidity crisis or arising from more fundamental concerns with solvency. Stablecoins backed by HQLA and well understood assets with low or no credit risk should, in some sense, have no trouble getting liquidity from public authorities in distress - but only if the assets are well audited and custodied. The same might also apply if they are seeking bailout from private sector entities. The digital asset world has seen many examples of camaraderie where private protocols have bailed out other protocols - but to put emergency liquidity provision on a firmer footing, transparency of assets for use as collateral is key.

might the practicalities of updating the information at high frequency. There may also be challenges in protecting reasonable confidentiality about portfolio structure or the identities of custodians, for example. Some proposed standards for stablecoin backing have suggested naming particular custodians or banks where reserves are held in deposits. A perhaps more realistic alternative might be to report characteristics of institutions involved in administering or securing backing assets, rather than identifying them. Indeed, most proposed standards for reporting (see for example) operate at a very aggregated level, so issues of specifying particular accounts or custodians are moot.[7] There is also, of course, the possibility that stablecoin methods could return a URI pointing to resources, that can be automatically accessed and programmatically interpreted if necessary, much as some implementations of ERC-712 do, for example.

Ultimately, some combination of competitive and regulatory pressure will lead to some stablecoins providing more transparency than others. All else equal, if one stablecoin enumerates 80% of their asset holdings with granular detail, and another enumerates 90%, then presumably the latter will be preferred by users and regulators. Stablecoin issuers should account for *all* their backing assets and provide timely updates of this information. Certainly, increased off- *and* on-chain transparency about reserve assets is likely to be required by regulators and users in coming years - the only question is how granular will be the required disclosures. Regulators may set minimum standards for granularity, but this is also an area in which stablecoins will likely compete to attract users.

The BIS' Project Pyxtrial is an interesting insight into the thinking of influential regulators. It is very clear that there is a desire for high frequency (even on-demand), granular reporting on reserve assets, including via automated APIs. This supports the case for at least considering the inclusion of query functions that return such data as part of a stablecoin's smart contract.[8]

A key point to note is that global stablecoins are not only issued all over the world but also may be *backed* by assets that are distributed across various countries and regulatory jurisdictions. This increases the complexity of stablecoins dramatically, relative to existing domestically focused digital moneys, and is one reason why regulators and users may find it difficult to keep track of the risks of a given coin. As such, solutions for obtaining reliable data on backing assets, and simplifying the complexities arising from these assets being held in different jurisdictions, will become increasingly important. These are areas where oracle protocols will obviously be key and stablecoin issuers will likely need to engage increasingly with such protocols.[9]

---

[7] By aggregated, we mean rough characterizations of assets classes such as 'cash and cash equivalents', 'money market funds', 'repo' and 'Treasuries'.

[8] Project Pyxtrial also provides an XML schema for the representation of the asset and liability side of a stablecoin balance sheet, with the former being dramatically more complex. Other institutions, such as the FSB (see FSB (2023)), the New York DFS (see guidance here) and the AICPA (see AICPA (2025)) have all issued proposals for how to account for and represent reserve assets. They are all quite similar, and also consistent with important stablecoin legislative proposals in the U.S. (the STABLE bill) and the GENIUS act).

[9] Indeed, to the extent that public infrastructure such as emergency liquidity lines are sought by issuers, it is doubly important to consider these cross-border issues. A single central bank may not be willing (or able) to lend against assets that are in another jurisdiction with opaque custody and legal treatments.

COMPLIANCE

To the extent that regulated institutions offer stablecoin based products to households and businesses in the future, one must acknowledge the important role of regulation. In fact, even decentralized and regulation-lite defi offerings may ultimately need similar functionality demanded by the user community, rather than by regulators. As such, although 'compliance' is frequently associated with regulations imposed by a government, compliance with guidelines emerging from good governance and industry *self*-regulation is also important (see Carapella (2025) for theoretical analysis of related issues).

Much of the focus of regulation in the context of stablecoins and other digital assets has been on identity: KYC / AML / CFT, white and blacklists, onboarding requirements and so forth. The FATF-inspired 'travel rule' is an important driver of regulation in these areas. In addition, regulators may also wish to distinguish retail users from 'sophisticated' investors and users in certain domains. Nevertheless, regulation does not end with identifying counterparties and basic transaction details. More complex regulatory logic must also be satisfied in many contexts, and these can vary dramatically across jurisdictions. There may also be related *ex post* reporting requirements, and real time monitoring responsibilities.

BLACKLISTS AND WHITELISTS

Blacklists are commonly used to limit the activities of, and possibly wipe the balances associated with, particular addresses. These lists are constructed based on some identity solution that connects addresses to either a specific identity or a class of entity. Alternatively, the blacklist may simply be based on past 'proscribed' activities of the address, such as receiving proceeds of exploits, or making suspicious use of mixers, for example.

The USDT token contract provides a good example of a blacklist. The contract stores a list of addresses of 'evil users' (implemented using a mapping from addresses to a boolean) and includes methods to add and remove addresses to/from the list, check whether a given address is on the list (a method that other token methods can then use to implement transfer conditionality) and even burn (or 'destroy') balances of tokens associated with a blacklisted address. Other prominent stablecoins implement similar functionality, though with different terminology.

Whitelists are essentially the converse of blacklists in that they identify addresses that are approved for some activity. In the stablecoin implementations discussed above, whitelists are less common than blacklists.[10] Whitelists seem more suited to private, smaller-scale, stablecoins, possibly on

[10] Interestingly, EURCV (from SG-Forge) was re-implemented to remove whitelisting functionality, so as to satisfy MiCA requirements for free transferability.

private blockchains, or perhaps in relation to primary market issuance and redemption of large-scale coins, rather than secondary market trading.

Events should be emitted whenever addresses are added and removed to such lists, and public functions should be available to check the status of a given address (is it on a list or not). It should also be possible to call a function to return each list. It should be left to a stablecoin's developers whether to include functions to defund blacklisted addresses, such as USDT's destroyBlackFunds or PYUSD's wipeFrozenAddress, not least as these can be controversial.

What being on a whitelist or blacklist ultimately implies for an address should largely be left up to the implementers. However, some consistency could be desirable. *At a minimum*, addresses on a blacklist should not be able to transfer coins (or have transferFrom used to send their tokens), and that addresses on a whitelist are the only addresses that can transfer coins (or have transferFrom used to send their tokens). A perhaps natural, but opinionated, stance would be to prevent anyone on the blacklist from calling a method that changes the contract state or from holding any administrative roles.

Arguably, white and blacklists are a piecemeal approach to a more elaborate compliance approach that relies on a more comprehensive identity solution, as well as other aspects of compliance. We now turn to regulated token standards, that demonstrate such an approach.

REGULATED TOKEN STANDARDS

Standards for 'regulated tokens' have begun to emerge in recent years. Some of the most prominent and promising examples are the ERC-3643 (or T-REX) protocol, ERC-1400 and the Swiss-focused CMTA token. We discuss these standards in greater detail in the appendix. While there is not yet consensus over how to incorporate compliance into stablecoin design, the standards share many elements. As regulations in important jurisdictions (notably the US) begin to crystallize, it seems that the industry may soon be able to coordinate around a coherent global approach.

IDENTITY

All of the standards feature restrictions on transfers based on counterparty identities, implemented in a more elaborate way than simply using black/whitelist functionality. What identity solution is used (for example, ERC-3643 uses ONCHAINID) remains a difficult issue. For a start, there are various identity solutions that have gained traction within the broader tokenization community. In addition to ONCHAINID, solutions such as the Ethereum Attestation Service (EAS) are also expanding their reach. Underlying paradigms for identity management and attestation are still in flux in various jurisdictions, particularly where decentralized identity (DID) and self-sovereign identity frameworks (SSI) are gaining ground.

As a result of the coexistence of several similar but different identity solutions, there is a lot of complexity and duplication that is difficult for token issuers to navigate in making their products compatible with multiple formats of ID, for example. Complexity may also arise from establishing

the reliability of underlying attestations and the party providing them and keeping this information up to date. These are areas where stablecoin issuers may not have expertize. While some identity solutions have been designed to work efficiently on a *single* chain, cross-chain identity management is a far more difficult design problem - particularly when there may *also* need to be integration with legacy off-chain infrastructure and cross jurisdiction operations. On coordination, one notable issue is how to align the identity solution used by a settlement asset (such as a stablecoin) with that of a tokenized asset that is being paid for. Presumably it will promote market liquidity to have the same - or at least highly interoperable - identity solution used in both legs of a trade, and yet existing solutions do not yet provide this coordination.

RULE ENGINES

For regulatory constraints that are not simply a matter of counterparty identities, but which rely on more elaborate regulatory business logic, standardization is especially difficult. As discussed in the appendix, one emerging model is to allow for a very generic function that checks the compliance of a transaction and returns a yes/no as to whether it should be allowed to proceed (for ERC-3643, this is a 'canTransfer' method). It might also be sensible to allow for informative error codes as auxiliary outputs but, again, the industry is far from agreeing on best practice in this area.

Of course, rule engines are not specific to stablecoins and such 'automated regulation' could apply to any tokenized asset, which is beyond the scope of our analysis. However, in the appendix, we discuss promising work on formalizing standard representations of regulatory requirements - notably the BIS' Project Mandala (and see also the Global Layer 1 and Purpose Bound Money frameworks proposed by the Monetary Authority of Singapore). Further coordination and collaboration between private sector (including stablecoin developers) and institutions such as the BIS/FSB and central banks would likely be necessary before regulatory requirements settle down sufficiently to allow a compliance standard to be created. In the meantime, there is likely a role for services providing middleware to handle compliance rules across the various jurisdictions where stablecoins operate, so that stablecoins can focus on their core design features, while delegating compliance to other services.

FORCED TRANSFERS

More controversially, existing regulated stablecoin (and broader token) standards seem increasingly to accept the need for 'forced transfer' functionality, whereby an address with a particular role, can transfer tokens from an arbitrary account. There are practical reasons why this might be desirable. First, regulators may wish to reserve the right to intervene - to reverse transactions, or to wipe/confiscate the accounts of 'nefarious parties'. Of course, this functionality can also underpin one approach to 'recovering' tokens in the case of a lost/forgotten private key, or to 'reverting' transactions (see the reclaimToken functionality in Paxos' smart contracts, for example). Such functionality is commonly regarded as excessive centralization and a route to censorship among many in defi. As such, many would be hesitant in specifying it as part of a

standardized approach, though it will presumably be a common feature of many stablecoins that seek compliance with regulatory requirements.

DOCUMENTATION AND ENHANCED METADATA

There also appears to be increasing consensus on the need for simple document management functionality within a regulated stablecoin. Under ERC-1643, for example, a document is characterized by a short name, a URI pointing to a document repository and the hash of the document contents (conserving limited on-chain storage). While it may seem trivial, it is arguably important to include document management in a stablecoin standard since various emerging regulatory frameworks demand 'white papers' for investor and user protection (see the EURC white paper required under MiCA, for example). Leading collateralized stablecoins also regularly issue attestation and audits in PDF form. Furthermore, natively off-chain compliance and regulatory requirements will generate documentation that should ideally be available (or at least pointed to) via stablecoin smart contracts. One reason for this is to allow composable services to be developed that can analyze these documents in an automated manner.[11]

Complementary to improved documentation is enhanced metadata, beyond the minimal metadata specified within the ERC-20 standard, say (symbol, name and decimals). Where the issuer is a centralized authority (such as Circle in the USDC case) it would make sense to return an identifier. Arguably, stablecoin standards could mandate the increasingly common Global Legal Entity Identifier scheme, ISO 17442 to be used. Of course, if an identity solution is already being deployed as part of the stablecoin implementation to allow for compliance rules (following the example of ERC-3643, say) then a well-designed implementation of these metadata functions should be consistent with the approach to identity used elsewhere in the token implementation.

A further metadata method that stablecoins might reasonably be expected to implement, is one that reports an identifier for the token itself, which can be used in off-chain platforms. Standards are emerging for such identifiers, such as the ISO 24165 'Digital Token Identifier', and work is ongoing between GLEIF and ANNA to connect ISIN and DTI identifiers in a systematic manner. This work already appears to be being incorporated in important regulatory standards (see ESMA draft technical standards for MiCA and references to DTIs in FCA (2024)).

While the BIS has, as yet, barely engaged with stablecoins, the aforementioned Project Pyxtrial explored technological solutions for monitoring asset-backed stablecoins' balance sheets. As part of that pilot, DTI were also used, for accurate on-chain identification of stablecoins. As noted in Pyxtrial, there are often many similarly- or identically-named tokens on a given chain, or with the same symbol. Sometimes this reflects an innocent name clash and in other cases it may arise from fraudulent tokens mimicking a legitimate token. Regardless of the purpose, this emphasizes that

[11] While there are some attempts to convert the representation of assets into algorithmic form (see the ACTUS standard, for example), for the foreseeable future blockchain-based finance will need to co-exist with legacy technology and processes in other areas of finance, implying a role for document management. We note also that in draft technical standards supporting MiCA implementation, the European Securities and Markets Authority (ESMA) advocates machine readability of white papers.

existing metadata under the ERC-20 standard is inadequate to pin down a token with the precision required for adoption at scale and/or for compliance purposes. Indeed, as one can see by searching for the symbol USDC in the DTI Foundation's online registry service there can be very many securities with the same or similar symbol. As stated in Varrall (2024):

> *ISINs provide economic attributes of the asset and DTIs provide crucial technical implementation details of the token. Where a financial instrument is tokenised on multiple DLTs, each instance gets its own DTI, which are then grouped as a Functional Fungible Group (FFG) and linked to ISIN of the asset. That is, on a given blockchain there will be an 'instance' of a stablecoin - say USDT - which is in some sense the same asset as USDT on another blockchain. As such, there would be one ISIN for 'USDT' but separate DTIs.*

INTEROPERABILITY

Both regulators and private sector developers are keen to enhance cross-chain deployment and (for a time) interoperability with *legacy* systems. Interoperability underpins concepts such as the finternet or 'unified ledger' advocated by the BIS. It is key to maximizing pools of liquidity and promoting applications that can scale across chains and regions. It is also important in allowing permissioned stablecoins to integrate with broader permissionless systems and wider market infrastructure, some of which may use legacy technology. Of course, many cross-chain solutions already exist, including Chainlink CCIP. As such it is not clear that there is a pressing need for functionality to promote interoperability to be embedded within a stablecoin implementation - it can arguably be left to external services. Nevertheless, it is worth considering what sort of features of a stablecoin might promote interoperability.

CROSS-CHAIN DEPLOYMENT

There are already some attempts to develop cross-chain token standards. Many are fundamentally 'bridge abstractions' but two that are more specifically token-focused are the xERC project, Layer Zero's Ominchain Fungible Token (OFT), and the Cross-chain Token standard (CCT) from Chainlink. These solutions emphasize the well-known dangers - particularly damaging exploits - arising from a fragmented development environment for cross-chain functionality.

CCT employs the widely used Cross-chain Interoperability Protocol (CCIP), though is focused on abstracting the cross-chain element to focus on a unified token perspective. The approach allows natively cross-chain tokens to be issued on multiple chains, without relying on liquidity pools that are commonly subject to slippage. This gives the effect of having a single unified pool of liquidity, which is key to the attractiveness of stablecoins. Notably, CCTs do not need to inherit any CCIP-specific code in their implementation, such that any token following the ERC-20 standard is a CCT on EVM blockchains. In contrast, as noted in Vester (2023), OFT is tied to the LayerZero protocol for its cross chain underpinnings, making it less general than ERC-20 tokens, and perhaps somewhat prone to vendor lock-in.

One important element of CCT that seems sensible for any stablecoin standard to allow, is that it accommodates multiple approaches to token transfers across chains. CCT allows for 'burn and mint', 'lock and mint', 'burn and unlock' and 'lock and unlock'. In so doing, CCT enables optional features that go beyond ERC-20. Notably, it expects the implementation of mint and burn methods on whatever chain the token is supposed to circulate on. However, as discussed in section 2.1, mint and burn should be implemented by any reasonable stablecoin *anyway*. There is no need additionally to appeal to their utility in cross-chain deployment to justify their inclusion among the coin's methods.[12]

While mint/burn functions are important to include in a stablecoin standard, an interesting question is whether the standard should mandate some sort of `lock and unlock' functionality. We now turn to this issue.

LOCKING

While mint and burn methods are natural components of any stablecoin, it is less clear that lock and unlock methods should be specified. Conceptually, minting or burning is *intrinsic to the token*, whereas locking/unlocking is perhaps most naturally thought of as extrinsic - something that can be achieved by an additional 'escrow' smart contract to which stablecoins can be sent. No prominent existing stablecoins implement lock/unlock. Instead, the logic is delegated to locking services to which the coins can be sent. OpenZeppelin offers a simple time-based TokenTimelock solution into which ERC-20 tokens can be sent, to be released only once a certain timestamp has been passed. Gnosis offers a somewhat more elaborate TokenLock. In addition, looking beyond cross-chain applications, staking/lending protocols, decentralized overcollateralized stablecoins, deferred token compensation structures, all commonly implement locking functionality that can be leveraged by stablecoins.

Nevertheless, there have been *some* attempts to extend ERC-20 functionality to allow for locking, without relying on external contracts. EIP-1132 is one example (see implementation here) and there have been some more recent proposals along similar lines for a 'Lockable ERC-20' (see here for a sketched interface and implementation). It is very unclear that these attempts have had, or are likely to have, any traction. However, even if locking/unlocking methods are not part of a stablecoin's own smart contract, as discussed above in section 2.3, there is a strong(er) argument to include functions that allow transparent accounting for what coins are locked or circulating across any chains where the stablecoin is officially deployed.

[12] CCT also *suggests* the implementation of an optional burnFrom method though this can presumably be easily implemented by combining allowances/approval, transferFrom and burn.

PRIVACY

One perspective on stablecoins is that they are the 'cash' of on-chain finance, aimed at low value transactions. Thus, one could argue that they should aim to offer a similar degree of privacy. However, another perspective is that stablecoins are a settlement asset for medium to large value payments in the financial system. Such settlement has rarely, if ever, attained the same level of privacy of cash. This tension points to the importance of an approach that is flexible enough to allow for these various perspectives.

The structure of blockchains offers an opportunity to accommodate a wider variety of use cases for a stablecoin than a single 'money' could have accommodated in the past. It is possible that traditional tradeoffs between privacy/confidentiality and transparency or compliance may no longer bind as tightly as they once did, thanks to the sophistication of cryptographic solutions and their incorporation into on-chain finance at a primitive level. Privacy, identity, fraud-prevention were not primitives of the early internet and the awkwardness and fragility of web2.0 protocols in these dimensions is well known (see IBM/Ponemon (2012)).

In this section we briefly consider various privacy solutions that have emerged around blockchain applications in general and some that have been introduced specifically as stablecoin solutions. For deeper discussions we defer the reader to JP Morgan (2024) and Auer *et al* (2025), or for a brief guide see here.

SOLUTIONS FOR CONFIDENTIALITY

Privacy enhancement is close to the heart of many crypto advocates, and various smart contract-based 'solutions' already exist to enhance privacy of transactions using stablecoins (or many other types of transaction). Notably mixers can anonymize transaction details, under some conditions.[13] Other smart contract solutions to which ERC20 tokens can be sent for privacy enhancement also are available, such as Kaleido's Zero Knowledge Token Transfer service and, extending to transactions over multiple chains, Chainlink CCIP Private Transactions.[14]

A 'third party service' could be a single smart contract, or it could be simply a matter of combining a private blockchain with a public blockchain with data being encrypted on (or even before) being added to the private blockchain - such as a chain offered by a bank to its clients - or with the data being encrypted in transit to the public blockchain, before hashed forms of the sensitive data are stored on-chain. Alternatively, two private blockchains might be bridged using a shared interoperability layer. These patterns are adopted in the Chainlink CCIP Private Transaction model and DECO technology, with a recent example of ANZ Bank and ADDX using CCIP Private Transactions to settle tokenized commercial paper across two private blockchains under the

[13] TornadoCash is a well known mixer and see also ErgoMixer.
[14] Recently, interesting work on status verification has emerged from the ERC-3643 standard's working group.

Monetary Authority of Singapore's (MAS) Project Guardian, maintaining confidentiality while meeting regulatory requirements for cross-border institutional finance.

Following a different paradigm, one can deploy tokens on blockchains with privacy enhancements built into the fundamental protocols of the chain (rather than relying on third party services). In the appendix we discuss a (non-exhaustive) list of illustrative projects exploring this approach. Several of these projects show great promise. Privacy-enhanced blockchains will likely continue to gain traction, not only for rollup purposes, but in enabling transaction and smart contract privacy. However, at the moment, the approach is still in its infancy - even allowing for the rapid pace of development in the blockchain space. While zero knowledge proofs and Fully Homomorphic Encryption (FHE) seem likely to be general purpose technological enhancements, experience has taught that today's 'next big thing' in crypto may very well be obsolete within a year or two. Even if such technologies do become entrenched, they are not yet standardized, as we can see from the examples in the appendix (some of which even rely on very particular hardware).

In our view, the lack of standardization (while exciting!) makes it difficult to argue for a particular 'best practice' approach that stablecoins - and tokens more broadly - should coalesce behind. There is not enough agreement yet to design reliable interfaces - even to the extent of knowing what parameter types to assert. This agreement may soon emerge, though it remains far from obvious that a stablecoin's own implementation will handle the privacy enhancing functionality, rather than an external protocol.

CONFIDENTIAL ERC-20

In this subsection we discuss separately the 'Confidential ERC-20' proposals from Circle and Inco (documented in Gai *et* al (2024) and at the github here). We discuss the protocol here, rather than in the appendix, as the project is a recent and especially important contribution in this area. The proposals are currently captured in a Proof of Concept (PoC) implementation, based on FHE technology developed by Zama, exploiting their fhEVM. The aim is to protect transaction privacy, while (like eUSD, discussed in the appendix) enabling a degree of auditability on encrypted data, which will be key for many regulated coins.

While these are not stablecoin proposals, *per se*, the involvement of Circle emphasizes that this will likely be an early application, and burn/mint methods in the standard further emphasize this. The interface sketched out in their proof of concept is clearly designed to be as close as possible to the ERC-20 - essentially identical in fact - but with novel parameter types. Notably, 'mappings' continue to be used to capture balances and allowances, but the values are in terms of an encrypted data type, the 'euint'.[15] Where booleans and addresses are required, the PoC implementation makes use of encrypted versions ('ebool' and 'eaddress'). We observe that every core element of ERC-20 (methods and events) is matched by an identically named element in the confidential

[15] It should be noted that these mappings are not obliged or encouraged by the ERC-20 standard, but they are typically included as part of efficient implementations of ERC-20 tokens (rather than using arrays, say).

ERC-20 standard. The only deviations (in the interface) are the use of euint for token values and, more substantively, the definition of overloaded versions of the approve and transferFrom methods. By overloading, we mean that there is a version of approve that takes spender and value inputs, as in ERC-20, but there is also a version that takes spender, encrypted amount and proof inputs. Similarly, one version of transferFrom takes a from address, a to address and a value, while a second version uses to and from addresses, but also an encrypted amount and another proof.

While this standard is only a PoC, it is important for our analysis for two respects. First, they give credence to the idea that staying close to the ERC-20 standard is possible for a stablecoin framework that is consistent with future advances in FHE and zero knowledge methods. Second, they emphasize that a dominant stablecoin is apparently envisaging a EVM future, though one where the EVM is fundamentally adapted for privacy.

FEES, YIELD GENERATION, AND INTEREST RATES

There is an ongoing debate over stablecoins' business model(s). At the moment, leading stablecoins are very similar in the sense that they issue a zero-yielding money, backed by non-zero yielding reserve assets, and do not collect fees on transactions using their token. Indeed, for many stablecoins, engagement with users is minimal, except at redemption (which may be handled by third parties). While we are beginning to see important collaborations between stablecoins and other institutions, this is also at the embryonic stage (see Stripe's activities here, here and here and those of Visa).

It is likely that stablecoin business models will continue to evolve (Catalini (2024)). Simply buying Treasuries and paying zero yield on a coin seems unsustainable - not simply (as some argue) because the term structure may not always yield such generous margins, but also because the model is extremely simple to replicate. As such, margins are likely to be compressed by competitive pressure, especially as tokenized MMF expand.

Arguably some stablecoins also benefit from a liquidity premium due to the (still) lower KYC/AML requirements that they exhibit, relative to other payment rails and MMF. For stablecoins that seek compliance, though, costs are likely to increase and some of that liquidity premium is likely to dissipate, increasing pressure to seek other revenue generators. Currently it is also surely the case that some stablecoin issuers have benefited from regulatory ambiguity limiting entry by competitors.[16] If stablecoins are to remain as simple as they are currently, then it is unclear that their superior profitability can be sustained.

[16] In the early stages of blockchain and defi's expansion, tribal loyalty to early movers who survived the initial shakeout of flawed (often algorithmic) stablecoin arrangements have surely contributed to implicitly cheap funding for prominent coins. It is unclear this will continue as the industry matures and more competitors emerge.

An important difference with MMFs is the ambition of stablecoins to be a widely used method of payment, not only for domestic financial institutions, but globally and for retail. However, at the moment, *among* stablecoins, there has been barely any effort to differentiate their services. Mainly they are distinguished by transparency, decentralization, compliance and the structure of reserve assets - rather than by the services they offer.[17] The aforementioned competitive pressure will drive such differentiation in the future, so we should expect stablecoin frameworks to evolve. Richer services may be paired with the stablecoins (perhaps by issuers deploying complementary smart contracts, or through partnering with other platforms), but the methods and events intrinsic to the coins themselves will also likely evolve. We will see divergence - with some stablecoins remaining close to the traditional 'cash' model, and others developing more functionality - becoming a richer form of programmable money. It is difficult to anticipate which use cases may become dominant and what methods may be required, but fee structures and yield generation/paying interest seem like natural places to start.

FEES

Tether's USDT perhaps points an interesting direction in that it allows for, though does not yet exploit, transaction fees. Specifically, a proportional fee parameter (basisPointsRate) and a maximum fee parameter (maximumFee) are both set within the contract. The proportional fee is applied to the value to be transfer-ed (or transferFrom-ed) to obtain a candidate fee. The max of this candidate fee and the maximumFee is then deducted from the value 'sent' to the recipient and sent, instead, to the balance associated with the contract owner.[18]

Fees play a prominent role among stablecoins with more complicated internal logic - such as those that rely on some element of algorithmic backing, but also those that use on-chain collateral. For example, under the Zephyr protocol essentially all actions imply transaction fees, which are paid into the token's reserves, and Zephyr documentation alludes to dynamic fees that adjust according to the state of reserves. A similar model is also featured in Djed and in the AgeUSD protocol.

The ability to direct a share of transaction value to another address, as part of a transfer/transferFrom or other core stablecoin methods is a powerful feature that could underpin a variety of use cases. Fees could be based upon the counterparties of a transaction (e.g. capital controls where counterparties are in different countries/jurisdictions), the size of the transaction, membership in a loyalty scheme run by an issuer or a service partnering with the issuer, the time at which a transaction occurs, the context in which the transaction occurs (e.g. when the stablecoin price is trading at a premium or discount to its peg, whether it is interacting with a DEX,...). As

[17] As discussed in section 5 there are also preliminary attempts to distinguish stablecoins on the basis of privacy enhancements.

[18] Both parameters were initially set to be zero at deployment (a 'no fee' situation) and have so far remained so. But they can be reset with the use of a setParams function, which emits an event announcing the values. setParams features the 'onlyOwner' modifier, reflecting the fact that such functionality is reserved for (presumably) Tether.

these use cases emerge it would perhaps be ideal if a fee interface and some core fee-related functions could be standardized.

In the example of USDT given above, the fee structure is extremely simple. The fees apply to only two activities (transfer and transferFrom) and the system is parameterized simply by two numbers (the rate and the max fee) which can be reset by the contract owner. This can be extended in terms of what actions trigger fees, by generalizing from numbers to functions encoding more complex logic. One approach to fees could be to minimally generalize the USDT fee functionality. To do this, we could specify, for each method using a fee, a rate, a fixed absolute value, a max (absolute) fee, and an address to which the fee is to be sent.

It would be natural for a stablecoin's smart contract to include a function ('getFeeParams' - generalizing USDT's getParams) to let people ascertain all the methods involving a fee, together with the current values of the fee parameters (rate, max and fee-recipient address). Generalizing USDT's setParams, we would also expect a setFeeParams function, that takes a method (or the function signature) and a specification of the fee. Changes to fee parameters should perhaps lead to an event being emitted, making clear which method the changes apply to, and which address (with a 'feeAdmin' role) effected the change.[19]

YIELD GENERATION AND INTEREST RATES

Another obvious way in which stablecoins might distinguish themselves is in paying a yield or an interest rate. In this section we first discuss current regulatory opposition to stablecoins paying interest/yield and then discuss some implementation options for adding this functionality. Again, how best to implement yield provision is still somewhat a matter for debate, and not yet ready for standardization, but the industry would likely benefit from a consistent and reliable approach to this functionality, which regulators are particularly suspicious of.

REGULATORY OPPOSITION

At the moment, regulators are typically opposed to yield bearing stablecoins for wide adoption. However, the intellectual justification for this is not entirely obvious. One argument given is that because stablecoins are 'money', they should not pay interest. This is in tension with the treatment of (interest-paying) bank deposits as money and indeed, with the fact that central bank reserves have also frequently paid interest in recent years.[20]

An alternative justification is that if stablecoins pay interest then they will give people the 'wrong' impression that they are bank deposits when - for the time being - stablecoin owners have no recourse to deposit insurance or similar regulatory protections that bank deposits benefit from. It is beyond the scope of this paper to discuss the arguments in favor of such schemes being created or adapted for stablecoins but, even in their absence, it is unclear that two investments both paying

[19] We could envisage *negative* fees to allow the issuer to provide subsidies or incentives/rewards for users.
[20] Similar tensions have been raised in relation to central bank digital currency. See Haldane (2023) for example.

interest will be mistaken for eachother purely because they pay interest. If they *are*, then there are clear investor education responses that are more targeted than banning interest-bearing stablecoins entirely.

Perhaps a stronger justification is the fear that interest paying stablecoins could lead to an unsustainable drive to offer unrealistic 'risk free' rates, that are underpinned by gradually investing in riskier assets, leading to financial fragility - both in terms of solvency and liquidity. Again, there are more targeted responses to this concern. As in the case of government money market funds, very tight restrictions can be applied to the assets that the stablecoin reserves can be invested in. If a class of stablecoins deviate from, say, holding purely HQLA and begin to look more like standard commercial banks in their investments, then the regulatory constraints and oversight applied to them can be tightened.

Another more plausible justification is a concern that there would be disintermediation of the banking system if stablecoins were to compete with banks on the basis of interest paid, as opposed to simply on the basis of transaction efficiency, programmability and other dimensions. This is a concern also raised in relation to CBDCs that could theoretically pay interest. The concern is that because banks play an important role in credit creation, making their funding more expensive (a result of competition between their deposits and novel forms of digital money) could reduce desirable credit creation in the economy. While there is evidence that *some* disintermediation may occur (see Bidder *et* al (2023) and Whited *et al* (2023) for related analyses in the context of CBDC), it remains unclear what the ultimate effect on credit might be, given the likely ability of banks to obtain alternative (e.g. wholesale) funding at rates that are not wildly different from those that they initially had access to through retail deposits. Furthermore, it is well known that many banking systems exhibit 'too big (or complex) to fail' problems and various other market failures. This could mean that substitution from bank deposits to stablecoins for some share of transaction brings some financial stability benefits overall.

Ironically, one criticism of stablecoins sometimes heard is that they may weaken the transmission of monetary policy rates, relative to that based on the traditional banking sector. While the transmission mechanism is multifaceted and the various channels through which stablecoin adoption at scale could affect it are yet to be fully understood, it is somewhat confusing to have these criticisms made by the same institutions that oppose stablecoins paying interest. An interest-bearing stablecoin should presumably pass through interest changes quite directly.

It is unknowable how the debate over interest-bearing *regulated* stablecoins will progress. It seems plausible that they will eventually be permitted for broader adoption, which could justify allowing for it in a stablecoin standard. Indeed, in some jurisdictions, yield generating stablecoins have been approved (notably Paxos' USDL under the Abu Dhabi Global Market and USDM under the Bermuda Monetary Authority). However, many prominent central banks and regulators seem likely to be opposed for some time (for an analysis of MiCA's approach to interest payments, see BCAS (2023) and (2024)). Recent registration of yield-bearing stables in the US (see here) suggests that such coins may be on the horizon.

HOW TO APPROACH YIELD GENERATION

Competitive pressure among issuers and between issuers of stablecoins and MMFs will plausibly lead to yield bearing stables, not least to satisfy institutional demand for a liquidity solution that, while safe, is not completely devoid of return. Decentralized (and largely unregulated) stablecoins also may wish to enable interest payments - and many already do as part of their tokenomics, as a way of passing on earnings from their collateral assets and the staking thereof.

Thus, even if regulators delay the adoption of interest-bearing stablecoins at scale by retail investors and users, there may still be a demand for such functionality in the near term from institutional users. What might be best practice in this area? In fact, the ERC-4626 standard is commonly used to enable coins to offer a yield. The standard leverages ERC-20 functionality, but where the smart contract manages 'shares' in pools, or 'vaults' of an underlying asset, rather than units of the underlying asset itself. By careful accounting of the relationship between the shares and the underlying asset, and by using income from (typically) reserve assets, the contract can replicate the effect of paying interest on a stablecoin (see BCAS (2024) and also the discussions here and here).

Happily, it seems that any stablecoin that extends the ERC-20 in a sensible way should be consistent with existing use of the ERC-4626 vault standard to implement an interest-bearing stablecoin (see USDM, for example). As such, there seems that standardizing approaches to non-yielding stablecoins will go a long way towards standardizing yield-bearing stablecoins also. Delving deeper into how best to approach yield generation is beyond the scope of this note.

ROLES AND EVENTS

Even a superficial glance at most prominent stablecoin implementations shows that certain activities - typically those that are sensitive - are restricted to addresses with particular authority.[21] In some cases, there may be a simple contract 'owner' that undertakes all administrative actions - so that there is no distinction between roles, as such (see USDT for an example of this, where the owner is initially set to be the contract deployer). However, this is increasingly regarded as bad practice in that it does not allow for the distribution of authority for specific tasks and represents a single point of failure.[22]

Note that roles are not typically part of formal token standards, which usually specify only *Methods* and *Events*. As such, there is a risk that in this dimension, the stablecoin industry may evolve with

[21] These addresses are given 'roles' which can then be checked by smart contract methods, either as part of a function modifier or in *require* statements within the body of the method.

[22] Even if some stablecoins assert multiple roles, they will nevertheless retain a dominant 'owner' role. Superficially, retaining such an omnipotent role undermines the avoidance of a 'single point of failure' that multiple roles ostensibly provide. However, the separation of roles in this case might allow differential security measures to be applied - with the overarching role assignment receiving especially careful attention (see OpenZeppelin's AccessControlDefaultAdminRules for an example of extra safeguards for especially powerful roles).

a wide variety of approaches being adopted, making interoperability somewhat more difficult, and risking some fragmenting of liquidity. There are many roles that different stablecoins might require, depending on their business model and implementation. A subset of these are so common among existing coins (though may have subtly different names) and are so intrinsic to typical stablecoin processes, that it seems sensible for the industry to standardize them. By adhering to a familiar and uniform approach in this respect, fungibility between different stablecoins can be even further enhanced.

Common roles observed in prominent stablecoins are shown in box 2. In addition to these, as discussed above, some stablecoins permit methods for 'forced transfers', whereby an address with a particular role can transfer coins from another user's balance, without requiring the approval of the user (as would be the case if transferFrom were used).[23] Such methods are of course restricted to an address or small set of addresses with a specific role for such a powerful action.

- **Owning**: A role with overarching administrative power and which can assign/configure all other roles
- **Minting/Burning:** Roles that can adjust stablecoin supply
- **Blacklisting/Whitelisting:** Roles that can add and remove addresses from lists
- **Pausing/Unpausing:** Roles to suspend ability to call smart contract methods, and to allow resumption
- **Upgrading:** If the coin follows an upgradable/proxy pattern, a role specifically to adjust the address of the implementation contract to which the proxy points, among other tasks

Box 2: Common roles in prominent stablecoins

Frequently the powers assigned to addresses with particular roles may be configured by *another* address with a higher level of authority. For example, as a way of defending against exploits and/or inappropriate coin supply, minting roles may be configured with rate limits. There is a good argument for the industry to coordinate around a standardized approach to associating collections of related roles (e.g. minter and burner) under the umbrella of a specific administrative role (e.g. coin supply administrator). This would make governance structures predictable and consistent, aiding users and regulators in quickly and accurately assessing the structure of governance within the coin. Indeed, although recent SEC communications have apparently put *some* stablecoins outside their remit (coins that do not qualify as securities), it is notable that in relation to other crypto assets, the SEC seems to be leaning strongly towards demanding a high degree of clarity on what parties are responsible for coin supply, and the upgrading of contracts. It is likely that a similar attitude will be taken to stablecoins by whatever regulatory body ultimately oversees them.

While standardization in this area is desirable for stablecoins, different coins should of course be able to assign roles that are idiosyncratic and relate to particular features of their business model and desired functionality. As such, it is difficult to see complete uniformity emerging around more than a handful of completely uncontroversial roles, or in a tightly regulated context where a single authority may demand a specific set of roles. What could perhaps then be a more promising approach to standardization is for stablecoins all to implement a method ('getRoleInfo') that returns

[23] Although such forced transfer roles are controversial, they are one way of implementing coin recovery in the case of lost keys, or providing (perhaps to satisfy regulators) the ability to reverse transactions or confiscate funds.

the roles the coin asserts, along with the addresses that currently hold the role, and the methods to which the roles relate. In this way, a stablecoin issuer is free to assert whatever roles it wants - but all coins must provide functionality to make those roles transparent, on chain and programmatically.

Similarly to roles, *events* have received little discussion in terms of how they may optimally be used to enhance stablecoins and to promote consistency within the industry. Again, greater consistency can promote fungibility between different stablecoins, and as user-facing apps emerge that make use of events for triggering other functionality, it will become more important for developers of such apps to be able to rely on a consistent set of events.

Given the power of administrative roles, it is important that their assignment, reassignment and configuration are transparent. As such, prominent tokens emit events on these occurrences.[24] Events also provide an obvious way to enable real time and *ex post* monitoring by users and regulators of transaction flows, supply events (minting and burning) and perhaps changes in reserve value and composition. Overall, it is therefore surprising how little emphasis there has been on harmonizing best practice for events. Important work in this area will need to be done in coming years to enhance fungibility between coins and composability with other on-chain assets and with on- and off-chain services.

CONCLUSIONS

In the 'old world', different coins were essentially identical in their core functionality - even if they had different names and images stamped into their metal or printed on their paper. The flexibility offered by smart contracts to augment a stablecoin with many features beyond simply being a 'money' is an immense strength. But it can also be a weakness if the scope for creativity leads to excessive variety and reduced fungibility through uncoordinated development. As such, it is important for the industry to seek 'flexible consistency'. Whether this results in a broadly adopted formal stablecoin *standard*, perhaps building on ERC-20, is unclear. But, as stablecoins promise to grow rapidly, it is vital that the sort of incompatibilities and inconsistencies that bedevil much of tradfi do not re-emerge.

A key point to remember is that consistency should not only be sought *among* stablecoins. Consistency *between* stablecoins and other types of token should also be sought, along with consistency across jurisdictions. These issues are beyond the scope of this paper, which has focused

[24] Events related to administrative roles are often delegated to the widely use OpenZeppelin access control contracts, which can be inherited by the core stabelcoin contract (see USDM for a good example of this). One option is to use a relatively simple 'Ownable' contract model, which is intended to standardize a basic access control mechanism. As aforementioned, USDT implements a simple 'ownable' approach where a single authority (the contract 'owner') is invoked for all administrative roles. The default would be for the owner initially to be the deployer of the contract, though there are methods to reassign ownership (see also the Ownable2Step module).

on stablecoins in particular and not given much consideration to multi-jurisdictional concerns, but these are vital issues to consider, going forward.

To promote wider adoption of stablecoins it is important that users and regulators are not confronted with a zoo of different tokens with unpredictable functionality. This note has attempted to identify areas where coordination on consistent design principles is possible now - and other areas where it is desirable and hopefully soon feasible. Standardization has led to many of crypto's greatest successes. Standardization of stablecoins - without excessively limiting innovation - should help them fulfill the immense promise that many believe they offer.

APPENDICES

REGULATED TOKEN STANDARDS AND KEY FUNCTIONALITY

In this section we discuss some existing token standards that are designed for regulatory compliance, focusing on some of their key functionality.

ERC-3643

An important contribution to 'compliance by design' is the ERC-2643 (or T-REX) protocol, which aims to provide a standard for permissioned tokens (tokeny (2023)). The standard embeds functionality that aids user safety (such as enabling token recovery) and it also allows some compliance requirements to be associated with the token. The core of ERC-3643, however, is that it invokes an on-chain identity solution - ONCHAINID.

ONCHAINID allows users to make use of proofs of 'claims' issued by trusted third parties. The proofs are associated with users' addresses and can be accessed on chain by token issuers and holders. Restrictions on who might be permitted to hold, or transact with, the token can then be incorporated into the token design. Token methods (such as transfer and transferFrom) can then consult a registry of identities associated with addresses on the blockchain and prevent the execution of the method if there is no proof of a suitable claim.

Identity is a key element of travel rule requirements, so some solution akin to ERC-3643 seems a natural element of any stablecoin that aims for institutional adoption. However, it is less clear that the particular ONCHCAINID identity solution incorporated in ERC-3643 need necessarily be prescribed in a standard. Given the fluid situation around digital ID in Europe and diverse approaches across multiple jurisdictions, there is perhaps value in a more agnostic approach to incorporating features from ERC-725 and ERC-735 (see here and here for discussions of these two standards). The growing need for cross-chain solutions may also require a more flexible approach to identity than ONCHAINID requires. Identity of a person or an institution is something that should be handled consistently across chains and it is not obvious that the ONCHAINID solution can be deployed consistently and coherently on multiple chains.

For compliance rules beyond pure white or blacklisting the standard allows 'compliance modules' to be associated with the token (tokeny gives examples such limiting the number of investors per country of distribution, and the number of tokens that can be held by a single investor). ERC-3643 is very agnostic and less developed in this respect - arguably by design - than in the dimension of identity management. It gives essentially no guidance about how a compliance rule should be expressed, beyond that it ultimately can generate a boolean that can be used to prevent a non-compliant method call. As discussed below, it is an important open question how to standardize compliance for stablecoins as it is this area that is the source of the main frictions in payments.

ERC-1400

Another prominent standard designed for tokenized assets is the ERC-1400 standard. In fact, ERC-1400 is effectively a collection of standards aimed at tokenization applications where compliance and regulatory requirements are prominent.

CANTRANSFER

ERC-1400 is more elaborate than ERC-3643 though does not explicitly standardize identity management in the way ERC-3463 does. Instead it allows the token developer to implement identity checks within a generic 'canTransfer' function - relying on the incorporation of some form of identity solution (this is covered in the ERC-1594 component of the ERC-1400). In fact, ERC-3643 *also* includes a 'CanTransfer' method in its compliance interface, though its specification is somewhat different.[25]

First, we note that the inputs to the two functions are not nested. The ERC-3643 version includes a 'from' input, allowing the function to be used without assuming msg.sender is the 'from', which presumably increases its generality. On the other hand, the ERC-1400/ERC-1594 version allows for a bytes input for additional data to be passed. In the associated ERC-1594 documentation they state that if this bytes argument is supplied, a different type of transfer function will be assumed when checking for the validity of that transfer. The ERC-1594 transferWithData method does also allows for a bytes input, but with no guidance on what the passed variable relates to. Quoting their documentation (emphasis added):

> *Transfer restrictions can take many forms and typically involve on-chain rules or whitelists. However for many types of approved transfers, maintaining an on-chain list of approved transfers can be cumbersome and expensive. An alternative is the co-signing approach,* ***where in addition to the token holder approving a token transfer, and authorised entity provides signed data which further validates the transfer.***
>
> ***The bytes _data allows arbitrary data to be submitted alongside the transfer****, for the token contract to interpret or record. This could be signed data authorising the transfer (e.g. a dynamic whitelist) but is flexible enough to accommodate other use-cases.*

It is arguably an unattractive feature of a standard to propose a method that leaves a generic data input in a function to allow arbitrary inputs. As such, a stablecoin standard might wish to offer a more parsimonious but still flexible third argument to a canTransfer function (noting that this would - obviously - be consistent with the arbitrary third argument that ERC-1594 suggests).

In terms of outputs, both functions return a boolean, indicating whether a transfer would be successful, if attempted. Note that this allows canTransfer to be used to check a hypothetical transfer, but also to act as conditionality within a richer 'compliant transfer' function. The ERC-1400/ERC-1594 allows for a return code, which is good practice. The standard alludes to ERC-1066 as a possible source of standardized error codes for functions to return, either on failure or, as in canTransfer's case, as the response to a query as to whether a particular transaction will be successful (or, at least, whether it might succeed or fail for compliance reasons). Again, perhaps

[25] In the case of ERC-1400/1594, there is also a canTransferFrom, with the obvious analogous structure to canTransfer.

reflecting a lack of clarity about ultimate use cases, there seems to be an extra output argument that again allows arbitrary data to be returned.

TOKEN RECOVERY

Other similarities with ERC-3643 include making provisions for recovery of tokens (such as through forced transfers following an exploit of after the loss of keys). Recovery functionality reflects the incorporation of the ERC-1644 standard within ERC-1400. ERC-1644 allows 'controller' addresses to execute transfers between other users' accounts, reflecting the expectation that such controller roles may be demanded by regulators to enable 'forced transfer for legal action or fund recovery'.[26]

Clearly this is an area where tensions between decentralization and regulatory oversight are likely to emerge. Arguably, a *regulated* stablecoin standard should specify a consistent way in which regulator intervention can be managed through role allocation and through transparent event emission, to promote regulator accountability.

The documentation for ERC-1644 somewhat limited and it is rather unclear what token 'redemption' means in this context. The essence of the approach, though, is that a controller may make transfers from one account to another without the use of 'allowance' and, in so doing, must use a special 'controllerTransfer' function that includes additional inputs for data about the transfer and the operator. The documentation of the latter two arguments is (apparently?) non-existent, but presumably they are to enhance transparency about the purpose and justification for the controlled transfer and allow fairly arbitrary data to be included in the method call. Events must be emitted if these methods are used, adding the address of the controller to the function inputs, for logging. As a further nod to transparency, it is suggested that the isControllable() method be included to make clear whether a token is subject to such controlled transfers. There is also the stipulation that if a token returns FALSE for isControllable() then it MUST always return FALSE in the future, so that the controllability cannot be 'sneaked' into an initially uncontrolled token.

DOCUMENT MANAGEMENT

Other 'user safety' features include requiring document management functionality based on ERC-1643. Under ERC-1643 a document is characterized by a short name, a URI pointing to a document repository and the hash of the document contents (conserving limited on-chain storage).

The interface proposed under ERC-1643, which is self-explanatory other than perhaps the timestamp variable, which captures the time at which the document was last modified via setDocument. Overall, the interface and rationale seems very sensible to the extent that it could perhaps be legitimately included as part of a general stablecoin standard. Two minor critiques should perhaps be considered, though. First, one might wish to index more of the event parameters.

Second, the getDocument and removeDocument methods seem perhaps to risk an indeterminacy in the case where two documents have the same name. On the latter point, of course one could rely on the user taking precautions, or even an implementation of setDocument (or an overlayed front

[26] As documented here, ERC-3643 also specifies functionality for 'forced transfers'.

end UX) that could reject a duplicate name. Presumably there will be a Documenter role (perhaps held by the issuer, or the issuer's legal department) that can ensure that the (presumably small number of) documents uploaded avoid a name clash - but setDocument should still check this and return an informative error code.

CMTAT AND ERC-1404

Switzerland has been a leader in establishing a legislative framework for digital assets. Reflecting the relative regulatory clarity in Switzerland, and the thriving digital asset ecosystem springing up there, there has been a drive by the Capital Market and Technology Association (CMTA) to develop a standard for compliant tokens. These standards (somewhat confusingly - as they are not, *per se*, a token) are referred to as the CMTA token, or CMTAT.

CMTAT is designed to accommodate a broad class of tokenized assets, and thus allows for more functionality than a stablecoin would likely need. However, the token standard allows for much functionality that stablecoins should have, and there are close parallels with elements of ERC-3643 and ERC-1400. Much as ERC-3643 allowed for extensibility through compliance modules, CMTAT features a RuleEngine contract (a reference solidity implementation is provided here) that can be customized to require transactions to respect various restrictions.[27] While it seems reasonable for a standard to include for conditional transfers, it seems unwise to require a stablecoin standard to take a stance on what the format should be used for underlying rules. However, as discussed above, there is likely to be progress in these areas in the near future.

RULE ENGINES

The incorporation of elaborate regulatory requirements - beyond simply KYC-based constraints - is an open area for research. It is of course much more difficult to standardize these, beyond simply demanding a function that returns a boolean for whether a transfer is to be allowed. Again, the now familiar question arises of whether this needs to be in a standard, rather than delegated to the smart contracts and protocols through which the stablecoins are used.

In the standards discussed above (ERC-3643, ERC-1400 and CMTAT) we have seen some attempts to allow for compliance by design, but it is reasonable to ask whether more structure can be embedded in a standard. Indeed, if a stablecoin standard prescribes functionality to provide information on reserve assets or the issuer, or to execute actions such as minting and burning, then it is natural to design a generic regulatory component that assumes the presence of such functionality. This functionality can then be exploited to check regulatory requirements that many stablecoin use cases will require. Given the somewhat anti-regulatory instincts of some defi participants, it is important to note that at least some elements of regulatory requirements are likely to be similar to due diligence and user-safety protocols that would emerge from private interactions. As such, 'compliance' functionality should not be seen as being in tension with

[27] A reference implementation for the core token contract (aside from the rule engine) can be found here.

adoption of the standard in the defi system, though the ultimate use of the functionality will of course diverge to an extent.

Project Mandala - run by the BIS in collaboration with the RBA, BoK, BNM and MAS - is a promising pilot that explores the possibility of distilling regulatory requirements from various jurisdictions into code. The aim is to formally automate complicated logic and enhance scope for straight-through-processing and real-time monitoring of (large value) cross-border transactions subject to compliance requirements. Notably, the system leverages zero knowledge proofs (ZKP) and security multiparty computation (MPC) to create proofs of compliance that can be associated with a transaction without revealing sensitive customer or counterparty details. A rules engine is used that can retrieve a formally stated compliance rule and pre-validate a checklist of conditions that the rule implies. Given this, a proof is generated (see Mandala (2024)). Some of the applications considered are in line with the white or blacklisting previously discussed (sanctions list checks, for example) while others are more elaborate (checking for MPC thresholds and also compliance with capital flow restrictions). It is worth noting that elements of Mandala are apparently being incorporated into MAS' 'Global Layer 1' initiative, which features Euroclear as a participant.

The FATF 'travel rule' provides a good example of the sort of compliance that could be, in part, automated within the logic of a stablecoin. The travel rule (so far) has been debated mainly in the context of institutions that enable payments (in some jurisdictions they may be referred to as Virtual Asset Service Providers) rather than in the context of P2P transactions. However, here is an open question as to whether stablecoin issuers might be drawn into similar compliance frameworks in the future. Even if issuers are spared this oversight, it may also be useful for stablecoins to be equipped with functionality so as to promote the support of stablecoin transactions by VASPS who must respect the travel rule and by regulated institutions transact on behalf of clients.

To begin with, a simple test based on the size of a transfer (or transferFrom) could be incorporated, which could then determine what other information is included with the transfer. If an identity solution (such as the ONCHAINID component of ERC-3643) is included within the stablecoin standard, it might be possible for all the data elements required of a travel rule-compliant transaction to be gathered together and for their presence to be tested as a condition for the transfer being permitted.[28]

[28] For small value transfers it typically will suffice for only the names of the originator and beneficiary of a payment to be collected, along with a wallet addresses or unique transaction reference. The latter will essentially be automatically implied by inclusion of the transaction in a block. For larger value transactions, there are additional information required, such as the originator's geographical location and other metadata.

PRIVACY ENHANCED STABLECOINS

In this section we discuss existing solutions - or experimental solutions - to enhancing the privacy of stablecoins.

ZEPHYR STABLE DOLLAR (ZSD)

Zephyr Stable Dollar (ZSD) is based on the Zephyr protocol and operates on its native blockchain.[29]. ZSD can be best understood as extending Djed to feature privacy enhancements akin to those of Monero, based on ring signatures and Bulletproofs.

ZKERC20 - EIP 1724

The zkERC20 Confidential Token Standard (EIP 1724), associated with the Aztec blockchain, offers an interesting example to consider. While the standard relates to general fungible tokens, it is relevant to consider for stablecoins as a particular case. The Aztec blockchain is an Ethereum L2 zero knowledge rollup. However, its zero knowledge elements are not simply limited to proving the correctness of transactions computed off (L1) chain (underpinning the proposed state evolution), but are applied in enhancing privacy of transactions on Aztec. Indeed, in their online documentation they emphasize 'privacy-preserving programmable digital money' as a key use case of the Aztec chain. Furthermore, the claim is that the zero knowledge elements could be implemented using proof approaches that are more general than their Aztec-focused implementation.

An interesting aspect of this project (or suite of projects) is that an example interface is provided at the associated EIP page (see here). Somewhat intuitively we observe various methods analogous to standard ERC-20 methods but prefixed with 'confidential' and with arguments whose types deviate from the standard variable types expected in ERC-20, anticipating the use of zk circuits.[30] However, it is unclear if the approach is backwards compatible with ERC-20 - through the ability to wrap an ERC-20 or to inherit. Discussions at the github suggest not (see here). Indeed, in more up to date documentation for Aztec (see here) it seems that even deploying an ERC-20 is non-trivial, to a large degree because of their 'private state' architecture. This of course reflects ongoing (though likely eventually surmountable) difficulties of porting projects from EVM to zkEVM environments.

While Aztec appears to be an Ethereum L2 it apparently is *not* a zkEVM (see here). Of course, a stablecoin standard should aim to be more general than simply applying to EVM deployment but, given the dominance of ERC-20 and the Ethereum or EVM/zkEVM compatible ecosystem, being

[29] It is a little unclear if this stablecoin is to be called ZSD or ZephUSD. See here for a fairly recent 2024 blog post, though one which again refers to ZephUSD rather than Zephyr Stable Dollar. The github associated with the protocol is here

[30] A primitive variable type in the protocol is the 'zero-knowledge note' and we observe proofs and signatures being passed as parameters in the interface. More detail is provided here though the page is dated and may not reflect the current situation.

incompatible with it, is a concern, from the perspective of developing a broadly applicable stablecoin standard.

EUSD

Marketed as 'the first data-protecting stablecoin', eUSD uses zk technology to hide all transactional information, while allowing for KYC/AML at on- and off-ramps. A white paper is available at MobileCoin (2023) and more documentation is available from Sentz (formerly MobileCoin).

While eUSD can be collateralized using tokens sent from other blockchains, it appears currently to be deployed only on the MobileCoin blockchain, which has various features embedded in the protocol, such as native ring signature capabilities, and zero knowledge proofs. Interestingly, there also appears to be the assumption that validator nodes feature trusted execution environments (Intel's SGX), which is perhaps problematic from a standardization perspective.

OTHER APPROACHES

The xUSD algorithmic stablecoin exploits the Haven Protocol, burning XHV to mint the stablecoin on a fork of Monero - and perhaps THORchain (more details here).[31] As such, it depends on the well known inbuilt privacy enhancements of Monero.

The zero knowledge blockchain, Mina, is also seeing development activity around a fungible token standard (with a recent beta implementation at this git repository). The interface is discussed here and although not developed in standard smart contract programming languages such as Solidity, or for a EVM deployment (it uses TypeScript), many of the methods appear closely related to those of the ERC-20. It also features obvious extensions for stablecoins (burning, minting, role administration). There do, however, appear to be elements that seem quite domain specific such as 'actions and reducers' and some other idiosyncratic elements. For example, it is not clear that recursive ZK proofs would be compatible with interfaces for systems employing different proof approaches.

[31] Also note an ostensibly similar model in the context of stablecoin development on the Aleo blockchain