How can run risk in digital asset markets be reduced?

Received (in revised form): 14th July, 2023

Greg Hopper
Senior Fellow, Bank Policy Institute, USA

Greg Hopper is a senior fellow at the Bank Policy Institute and a Principal at Enterprise Risk Economics. He is also on the advisory committee of the Office of Financial Research. Previously, he was a managing director at Goldman Sachs, where he was Head of the Office of New and Emerging Risks and Global Head of Enterprise Risk Management. In those roles, he oversaw the Sovereign and Economic Risk Group, Firmwide Risk Identification, Firmwide Limits and Risk Appetite, ESG Quantitative Analysis, Firmwide Stress Testing and the Risk Economics Group.

Bank Policy Institute, 1300 Eye St., NW Suite 1100 West, Washington, DC 20005, USA
E-mail: gregory.hopper@er-econ.com

Abstract This paper reviews the proof of reserve methodologies employed by crypto exchanges that attempt to demonstrate cryptographically that assets exceed liabilities so that there is no reason for participants to run on the exchange. The paper suggests a number of enhancements to these methodologies using a cryptographic statistical proof, proof of knowledge methods and other techniques. Although this paper reviews the proof of reserve methodologies of crypto-native institutions, its goal is not to address the risk management challenges of crypto exchanges alone, but, rather, to illustrate how enhanced versions of these methods might be used to mitigate run risk of digital assets on the nascent regulated crypto platforms being developed by banks and other financial institutions. A simplified version of the techniques that could be used to reduce the run risk of stablecoins is presented as an example.

Keywords: *crypto, digital assets, risk management, run risk, cryptography*

INTRODUCTION

The crypto sector is currently in its second crypto winter. The first crypto winter, which lasted from 2018 to 2020, was caused by the collapse of the Initial Coin Offering (ICO) market. In contrast, the most recent crypto winter, which started in 2022, was caused by widespread risk management failures in the crypto markets that led to classic runs. The crisis in the crypto markets began in May 2022 with the collapse of the crypto currencies Luna and TerraUSD. Those events helped to bring down Voyager Digital, a crypto broker, and Celcius Network, a crypto lender. Ultimately, FTX, a prominent crypto exchange, failed and BlockFi, essentially a crypto bank, went bankrupt soon after. In 2023, the crypto winter spilled over into the conventional financial sector with the failure of two banks that served the crypto sector: Silvergate Bank and Signature Bank. The effect of the crypto winter on the conventional financial system has nonetheless been limited, since the crypto markets to date have not found many conventional financial applications.

The next phase of crypto development is moving quickly in the direction of tokenisation of conventional financial assets, connecting crypto with the conventional financial markets. JP Morgan, for example, has launched Onyx, a blockchain application for wholesale payments. Canton, a blockchain network for financial institutions, is being developed by a group of firms including Goldman Sachs, Microsoft and Deloitte. A consortium of banks including Citi, BNY Mellon, PNC, Truist and Wells Fargo have demonstrated the feasibility of a regulated digital asset settlement platform. HSBC

and the European Investment Bank recently launched a digital bond denominated in pound sterling while Franklin Templeton launched a tokenised Government Money Fund, available on the Stellar and Polygon blockchains. The FDIC has recently included the management of digital asset risk in its 'Top Challenges', noting that 52 million Americans have digital asset investments and 136 FDIC-insured banks have current or future digital asset plans.[1]

Because crypto is on course to becoming more integrated with the conventional financial system, it is becoming increasingly important to develop appropriate risk management techniques for crypto assets. It is tempting to believe that conventional risk management procedures can be straightforwardly employed in the crypto sector, since many of the risks look the same. For example, crypto exchanges and banks were the victims of classic runs that ended in their collapse. However, run risk in the crypto sector should be managed in a completely different manner than in the conventional financial sector, since the cryptography, transparency and low reliance on trust inherent in crypto assets can be used to mitigate run risk.

In this paper, the nascent crypto-native risk management techniques that crypto institutions, particularly crypto exchanges, are implementing to manage run risk are reviewed. Run risk arises when market participants suspect that a financial institution does not have sufficient assets to cover its liabilities, and run on the institution to remove their assets first. The crypto-native risk management practices, 'proof of reserves' (POR), are designed to provide cryptographic proof to the market that assets exceed liabilities and, therefore, there is no incentive for the market to run on the institution. While no cryptographic proof or other method can provide perfect certainty about a financial institution's solvency, we will suggest how current POR methodologies can be enhanced to provide much more comfort to the market, reducing run risk.

Improving POR methodologies will reduce run risk in purely crypto-native financial institutions, but they cannot eliminate it. It is important to recognise that POR methodologies are not a substitute for a robust legal and regulatory structure. Rather, crypto native financial institutions should follow legal and regulatory rules that safeguard conventional financial assets in the countries in which they do business. The POR methodologies, then, would provide even more protection for end users and markets than conventional financial institutions could.

Although this paper reviews the POR methodologies of crypto-native institutions and suggests a series of improvements, its goal is not to address the risk management challenges of crypto exchanges alone, which are much deeper than run risk, but, rather, to illustrate how enhanced versions of these methods might be used for risk management of digital assets on the nascent regulated crypto platforms being developed by banks and other financial institutions. They could also be used to reduce the run risk of stablecoins.

PROOF OF RESERVES

A number of crypto institutions have developed POR methodologies. These methodologies are not standardised as of yet and so there are a wide range of practices currently. Centralised stablecoins tend to employ periodic independent attestations to demonstrate proof of reserves while the crypto exchanges tend to use either a combination of cryptographic methods along with an independent auditor or pure cryptographic techniques without third-party review. First, the POR methodologies of Kraken and BitMEX will be reviewed, since they provide examples of each type. Kraken combines cryptographic methods with an independent auditor while BitMEX does not rely on an auditor. The purpose of choosing these exchanges for discussion is not to argue that one exchange's POR is better than the other, but, rather, to assess the strengths and weaknesses of alternative POR methodologies that are considered best-in-class. Although both methods have a lot of value, POR methodologies are still in an early stage of development and can be improved.

Kraken's proof of reserves methodology

Kraken is a crypto exchange formed in 2011. It allows market participants to trade over 200 cryptocurrencies in over 190 countries and has over US$200bn of quarterly trading volume. Kraken's

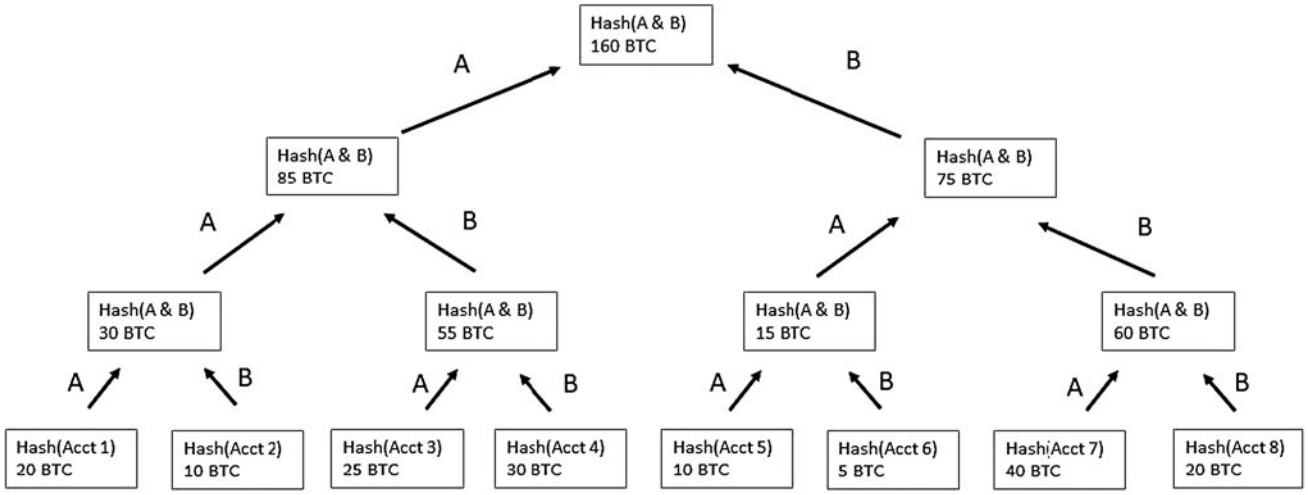

Figure 1: Merkle sum tree

POR combines an independent audit with a Merkle sum tree, a cryptographic device to verify that every user's balance is included in the audit, no balance is negative, and that the sum of the balances is less than, or equal to, the assets owned by the exchange.

Before describing the Merkle tree, it will be useful to define a cryptographic hash function. This is a function that takes a message, m, of arbitrary length and maps it to a number with a constant number of digits. For example, the SHA-256 hash function maps a message of arbitrary length to a binary number with 256 digits. The mapping is one-way in the sense that it is relatively easy to go from the message to the 256-digit number but effectively impossible to go from the number back to the original message. Hash functions have many uses but for our purposes we will focus on their ability to make commitments.

To illustrate how a hash function is used, suppose a sealed auction online was to be implemented. Each participant would provide a hash of their bid. Using SHA-256, the hash of the message $m =$ 'X will bid US$1000' would be the hexadecimal number

a6b3eb1a08c325b91bf5865b61fe8
b1f1ec4326c4ab6bfc3d0539ec0c9aba3f1.

Then each participant could unveil their bids by showing that the hash of the bid is identical to the hexadecimal number they committed to.

Hash functions are used to make commitments in a Merkle sum tree. Figure 1 depicts such a tree.

Each of the bottom leaves of the Merkle tree incorporates two numbers: the hash of the user's account information and the number of bitcoins the account owns. Bitcoins are used in these examples, but any digital asset could be substituted.

The next layer of leaves above the bottom is constructed by hashing the concatenation of the previous leaves and then adding the bitcoin amounts. The process is repeated until it terminates with the top leaf, which holds one hash value — a 256-digit binary number — and the total number bitcoins held on the exchange. The top leaf is publicly disclosed.

In Kraken's POR, an independent auditor starts with all accounts and balances and then constructs a Merkle sum tree, as in Figure 1, of all balances of the exchange's customers. The auditor then collects the digital signatures of the public blockchain accounts that Kraken owns, which proves that Kraken does own the crypto currency in those accounts. The ability to sign digitally using the public key of a blockchain account proves that the signer knows the private key and therefore owns the crypto currency associated with the public account. Finally, the auditor verifies that the sum of the crypto currency in the accounts that Kraken owns equals or exceeds the balances represented in the Merkle tree.

Using the Merkle tree, any user can verify that his balance was included in the audit by performing a

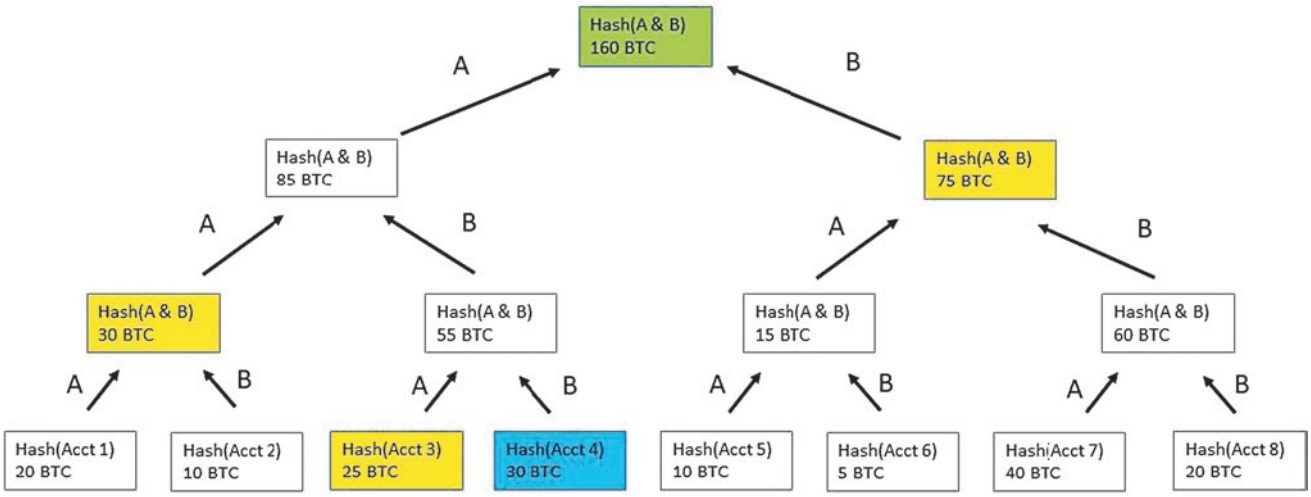

Figure 2: Merkle sum tree proof

Merkle tree proof. Figure 2 shows how this would be done. Suppose the holder of account 4, coloured in blue, wants to verify his account was included in the audit. Kraken would send the three leaves coloured in yellow to the holder of account 4, who would use the leaves to construct the proof. Since the owner of account 4 knows his own account information and balance, he can construct his own leaf by hashing his account information and adding his bitcoin (BTC) balance. Once his own leaf is defined, he then hashes it with the leaf to the left coloured in yellow to obtain the next leaf up in the tree. He then hashes that leaf with the leaf to the left in yellow to get the next leaf up, which he then hashes with the leaf coloured in yellow on the right to produce the top leaf. If user 4's calculated top leaf equals the previously disclosed top leaf, then his account must have been included in the audit.

How could this POR methodology fail?

Using an auditor implies infrequent audits

Relying on an auditor limits how often a POR can be done. Kraken's last POR report was performed on positions dated 30th June, 2022.[2] At that time, the auditor confirmed that Kraken controlled the private keys of cryptocurrency accounts that exceeded the liabilities owed to customers of the exchange. Thus, the proof of ownership of sufficient crypto assets to pay all liabilities is only valid on the specific day the audit was conducted. Moreover, the auditor verified that all customer liabilities were included as of 30th June, 2022 but, again, that attestation is only valid on that particular day. The risk that audits are stale can be mitigated by performing a more frequent audit. However, auditors must be available to perform the audits; at the time of writing, the two auditors that had been providing POR attestations have withdrawn from the market.

Were all liabilities included in the Merkle sum tree?

The POR methodology attempts to mitigate the risk that all liabilities may not have been included by providing tools for exchange participants to verify for themselves that their balance is included in the Merkle sum tree, using the proof methodology described above. If a market participant finds that his account has not been included, he could alert other market participants. This methodology, of course, depends heavily on individual market participants taking the time to verify their accounts. The methodology would be stronger if the entire Merkle tree were disclosed so that independent parties could verify the inclusion of arbitrary accounts. However, there are privacy constraints that prevent disclosing the entire Merkle sum tree. A glance at Figure 1 shows that if all leaves were disclosed, then market participants, competitors, and potentially malicious actors could learn the

amount owed to each account on the exchange. Moreover, an adversary could open a large number of very small accounts in order to do a series of Merkle tree sum proofs to learn about the balances on the tree.

Were assets borrowed?

The Merkle sum tree may contain a complete list of all accounts to which the exchange owes crypto assets. However, there may be hidden off-chain liabilities in which the exchange borrowed assets in order to pass a POR audit.

BitMEX's POR methodology

Founded in 2014, BitMEX is a crypto spot and derivatives exchange that caters to institutional and professional traders. BitMEX was one of the first crypto exchanges to create a POR. Introduced in 2021, BitMEX's POR is a fully automated system that does not require an independent auditor. As such, the POR is produced at a much higher frequency, currently twice per week. As in Kraken's methodology, BitMEX uses a Merkle sum tree to prove liabilities. On the other hand, BitMEX proves ownership of crypto assets directly, without an external auditor. BitMEX's POR addresses some of the issues in Kraken's POR identified above but also introduces some new problems.

BitMEX's proof of control of crypto assets is very simple. It simply discloses the full list of accounts it owns on various blockchains and then transfers crypto from those accounts to prove ownership of each account. Thus, any participant can observe the total amount of crypto assets the exchange owns at the time the POR is published and then verify that it is greater than, or equal to, the total liabilities recorded in the Merkle tree.

On the liability side, BitMEX's POR uses a Merkle sum tree with some additional features to allow it to disclose the entire tree to the public while still maintaining privacy. First, each leaf at the bottom of the tree is split into two leaves randomly. For example, in Figure 1, account 3 could be split randomly into one leaf with a balance of 15 BTC and one account with a balance of 10 BTC. All other leaves would be split randomly in two as well and then the leaves shuffled. The new tree would then have 16 leaves on the bottom row rather than eight and the tree would be constructed as before, terminating in one top leaf. By obscuring the accounts in this way, it is possible to disclose the whole tree while still preserving significant privacy, since no observer can track the spending of particular accounts over time.

Each user can verify his account's inclusion in the Merkel tree as before. To do the proof, BitMEX will provide enough information for each user to locate the two leaves that correspond to his account at the bottom of the tree and the node hashes that will allow him to reconstruct the top node of the tree. But this scheme also provides some additional information:

- anyone can verify that all user balances add up to the claimed top balance;
- third parties can verify that other accounts are included in the Merkle tree, provided that the account holders give out their leaf information.

The BitMEX POR resolves some of the issues that arose with the Kraken methodology. Because the POR is conducted twice per week, there are only a few days over which the proof is no longer valid. Insolvency issues generally do not happen quickly and so would be discovered relatively quickly. However, the methodology cannot ensure that assets were not borrowed to pass the audit. Also, the methodology will not detect collusion between an exchange and another institution to pass collateral back and forth to pass a POR. Although the BitMEX POR does not guarantee that all liabilities are included, since the full Merkle tree is disclosed twice per week, it could provide a lot of assurance since it would be possible for market participants to coordinate to spot-check a large number of accounts in the event that questions arise. The higher frequency does not solve the problem of determining whether assets were borrowed, but it does mitigate the risk since borrowed assets would have to show up or be removed in on-chain accounts which are compared with liabilities twice per week.

Although the BitMEX POR addresses some of the problems inherent in the Kracken POR, it does so by introducing some new risks. Proving ownership of assets is simple if the exchange uses a relatively small number of blockchain accounts to hold its assets, but that small number can conflict with private key risk management. Using a small number of accounts implies that large balances are kept in each

account. The loss or theft of the account's private key would mean that a large proportion of assets could be lost. It might appear that this problem could be easily solved by using a large number of on-chain accounts and proving ownership by moving assets in each account during a POR. But that strategy also conflicts with good private key risk management. To protect private keys, the procedure for using them must be hard. Keys should be held offline in cold storage, using careful protocols that limit and log internal access. Private keys may also be protected by holding only pieces of each key in cold storage, with the entire key assembled offline using a secure multiparty computation that hides knowledge of the entire key from each participant.[3] To maintain high private key security, only a limited number of keys should be used each day, making it unfeasible to prove ownership of assets by moving crypto in all accounts.

The modified Merkle sum tree employed in the liability side of BitMEX's POR increases transparency of the total liabilities at a cost of some potential loss of privacy, although the randomisation scheme may provide an acceptable level of privacy. But the technique does not resolve two fundamental problems of proving liabilities in a POR:

- the methodology does not prove that all liabilities have been include;
- the methodology does not prevent a malicious actor from falsely claiming that his account was excluded from the Merkle tree.

HOW CAN POR BE IMPROVED?

In this section, deficiencies found in the review of current POR methodologies are examined and some enhancements suggested. The problems that must be resolved are:

- How do we prove asset ownership without using an independent auditor and without compromising key security?
- How do we lower the risks that assets were borrowed to pass the POR?
- How can we allow any account holder to check that his balance is included in the POR without leaking private information?
- How do we prevent a malicious actor from falsely claiming that his account was excluded from the Merkle tree?
- How do we prove that all liabilities were included?

We begin with a technique to prove the ownership of assets and then turn to the liability side.

Proof of assets (POA)

As has been seen above, proving ownership of assets is relatively simple if just a few accounts are kept and access to those accounts is proved. However, the risk management problem was that having few accounts can produce significant losses if a private key is lost or stolen. On the other hand, if a greater number of accounts is used to diversify the risk, it is necessary to use a large number of keys to prove ownership of those accounts, which conflicts with the requirements of good private key risk management. What is needed is a way to prove assets in a large number of accounts by proving ownership of a small subset of them.

One way to accomplish this goal is to implement a probabilistic proof mechanism. Intuitively, the proof proceeds as follows: suppose there are N accounts where N is large and that each account has some constant amount of BTC, such as 100 BTC. Thus, total assets are $N \times 100$ BTC. Suppose that some fraction of the N accounts, p, are fraudulent in the sense that the exchange does not really possess their private key and therefore does not own them. If m accounts were chosen at random and the exchange was challenged to prove ownership of them, how large would m need to be to have some high probability of discovering fraud, such as 95 per cent?

To proceed, it is assumed that the m accounts are chosen randomly from a Poisson distribution with mean λ. Thus, m is a random variable drawn from a Poisson distribution with mean λ. Let k denote the number of fraudulent accounts that are discovered. Assuming that the number of accounts, N, is large, it can be approximated as infinite. Then the probability of discovering k fraudulent accounts when choosing m to test randomly is

$$P(k) = \sum_{m=k}^{\infty} \binom{m}{k} p^k (1-p)^{m-k} \frac{\lambda^m e^{-\lambda}}{m!}$$

This expression can be simplified to

$$P(k) = \frac{(p\lambda)^k e^{-p\lambda}}{k!}$$

Thus, the probability of finding no fraudulent accounts is $P(0) = e^{-p\lambda}$ and the probability of finding at least one fraudulent account is

$$P(k > 1) = 1 - e^{-p\lambda}$$

To see how this would work concretely, suppose it is assumed that 5 per cent of the accounts are fraudulent. Then, $p = 0.05$. Suppose further that it is desired to detect fraud with a 99 per cent probability. Then, λ can be inferred as

$$0.99 = 1 - e^{-.05\lambda}$$

$$\lambda = -\frac{\ln(0.01)}{0.05} = 92$$

Thus, it would be necessary to sample, on average, 92 accounts to test for fraud at the 99 per cent confidence level if the rate of cheating is 5 per cent. Alternatively, if it were desired to test at the 99.9 per cent confidence level, it would be necessary to test, on average, 138 accounts.

The simplest way for the exchange to prove ownership of the randomly selected 92 accounts would be to sign a message from each account's public key, since that would prove knowledge of the account's private key. There are other methods as well, such as zero knowledge proofs to prove knowledge of the private keys, that are discussed further below.

To use this scheme, p, the probability of cheating, as well as $P(k > 1)$, the fraud detection probability, would need to be specified. p could be chosen to be low enough such that in the small chance that fraud is not detected, it would not be large enough to threaten the exchange.

Preventing the prover from cheating

The methodology above cannot be used as is since if the exchange were allowed to choose the random accounts, it could cheat by selecting random numbers corresponding to accounts that it knows it controls. For the methodology to work, the accounts must be chosen randomly and it is thus necessary to guarantee that the exchange selected the accounts randomly. Using a verifiable random function, the exchange can prove that the accounts were selected randomly.

A verifiable random function (VRF) is a cryptographic method that generates a random number along with a proof that the number was drawn randomly.[4] The methodology consists of three functions and four data inputs. The exchange would first run a Prove() function that takes as input the exchange's private key as well as a string, the 'alpha_string' that is known to everyone and is generated in some way the exchange cannot predict. The exchange uses the Prove() function to generate a proof string, 'pi_string'.

```
Prove(alpha_string,private_key) - > pi_string
```

The exchange then runs Proof_to_Random() which takes pi_string as an input and then generates a hash, which is effectively a random number.

```
Proof_to_Random(pi_string) - > hash
```

The hash is a binary number with a pre-defined number of digits. The hash is then converted to a uniformly distributed random number by dividing the hash by the largest value of the integer possible. For example, if the hash has 16 digits, then the simulated random number would be $\frac{hash}{2^{16}-1}$.

The validator would run a Verify() function that takes the alpha_string and the pi_string as inputs, and then produces a hash as output. If the hash equals the hash that was provided by the exchange, then the random number was produced randomly.

```
Verify(alpha_string,pi_string) - > hash
```

To see how this would work concretely, a public and private key of length 32 bytes is generated using elliptic curve cryptography and hashes with 512 digits created, leveraging the implementation of the VRF standard draft[5] from Schorn.[6] Writing the keys in base58, the standard used by bitcoin, assume that the private key is set to be

DtJ2Wq8MyyqfRfhoBQu3GjBimLNmbCKn4bRw JmBN4iyu.

The corresponding public key would then be

C3beZyJwEGQvqGvKnUBNbLWkpXcKPaV1
GQYUvBgo7Xwe.

The alpha_string should be known by everyone and not predictable by the exchange. It could, for example, be derived from some data from the most recent block of the blockchain that precedes the proof, which would not be predictable by anyone. Suppose the alpha_string is set to

Ixd$tuhIrRt6588@!

Then

Prove(Ixd$tuhIrRt6588@!, DtJ2Wq8MyyqfR
fhoBQu3GjBimLNmbCKn4bRwJmBN4iyu) - >
2MBXzjrS4VePKXz3Fcb9qP2SPquvXuRgDcmc
FQjxE9fAMtDxpuqxBbD5gUaUdTr7Uzea
EJmdxZQv9bxt7ASRyDqa5hwWogvkqK65tz
77KVT7MR (this is the pi_string).

Now, the random number can be generated by

Proof_to_Random(pi_string) - >
2GxByELambKs9qX5VcmJJ752BRrCbm5nurL35
MtJQ9kjtYTVZjRhMPhr2LkVwdCsdii1Q4GA
fpy5pfHF5smK4iki (this is the beta_string).

Dividing the beta_string by $2^{512} - 1$, we get the random number 0.24911327397232452.

The validator can verify the random number was truly produced randomly by running the Verify() function:

Verify(Ixd$tuhIrRt6588@!,pi_string) - >
2GxByELambKs9qX5VcmJJ752BRrCbm5nur
L35MtJQ9kjtYTVZjRhMPhr2LkVwdCsdii1
Q4GAfpy5pfHF5smK4iki (this equals the original
beta_string)

which validates that the random number was, in fact, chosen randomly, since the beta_strings agree.

Generating a Poisson variable

The VRF generates a provable uniformly distributed random number. To generate a Poisson random variable is simple: a method is selected to simulate a Poisson random variable from a series of uniformly distributed random numbers and then that method is disclosed to everyone. For example, we could use the Poisson deviate algorithm from Numerical Recipes.[7] The exchange would disclose the simulated Poisson random variable and the set of uniformly distributed random variables used to construct the Poisson variable. The validator would verify that the Poisson variable was produced by the supplied uniform random variables and would also check the proof that each uniform variable was generated randomly.

Summary of proof of assets

1. Exchange pre-commits in Merkle tree to the set of accounts the exchange owns before issuing POA.
2. Exchange draws N uniform random variables from VRF to simulate λ from Poisson random variable.
3. Exchange draws λ additional uniform random variables from VRF and, based on those, selects λ accounts.
4. Exchange sends $N + \lambda$ random variables, $N + \lambda$ proofs, Merkle tree inclusion proofs for each random account selected, and λ digital signatures corresponding to the random accounts.
5. The verifier validates that: all uniform random variables were drawn randomly; that λ is a valid random draw from a Poisson distribution given a pre-agreed algorithm; that each account was included in the Merkle tree; each account was correctly selected given the λ random numbers; and the λ signatures are valid.

The validation could be done on chain by a smart contract or off-chain by an independent oracle.

Proof of liabilities (POL)

To prove liabilities, it is necessary to demonstrate that all accounts that have a claim on the assets have been included, that no negative balances have been included, and that the sum of the liabilities is less than, or equal to, the assets. In addition, it should be possible to execute a POL without revealing any account details and no participant should be able to

falsely claim that their liability was not included. Thus, it is necessary to prove the following:

1. All account balances are positive.
2. The sum of the account balances is less than, or equal to, the sum of the assets.
3. No one can falsely claim that their account balance was not included.
4. All account balances have been included.

The technology that can be used to prove 1 and 2, while revealing no other user information, has been rapidly advancing over the past few years. Zero knowledge proofs (ZKPs) can show that an arbitrary calculation has been carried out using some public inputs and some inputs that are kept private. As an example, one such statement that could be proved by a zero knowledge proof might be 'I know a message *m* such that Hash(*m*) = X, where Hash() is a known hash function, X is publicly known, but *m* is private, only known to the prover'. By the properties of hash functions, it is not possible to determine *m* from X. However, it could be proved that X is the hash of *m* without revealing *m*, using a zero knowledge proof.

A zero knowledge proof could be used to prove points 1 and 2 above.[8] A zk-SNARK[9] or a zk-STARK[10] could be used for this purpose. The advantage of a zk-SNARK is that proof sizes are very small, but the tradeoff is that the setup of a zk-SNARK requires a trusted environment. zk-STARKs do not require a trusted set up, but proof sizes are much larger. Which technology an exchange will select will depend on the set up. If a smart contract is verifying the proof, then a zk-SNARK will be preferred but if an oracle is validating the proof off chain, a zk-STARK may be preferable. Whichever version is selected, the proof would be executed by including the liabilities in a Merkle sum tree as above and the exchange would provide a ZKP that no balance is less than zero and that the sum of the balances is less than, or equal to, the assets. Moreover, account holders, on request, could receive a ZKP that their balance was included in the Merkle tree, without revealing their private information. Third parties could also check on behalf of account holders that their balances were included, without disclosing private information.

Point 3 can be handled by issuing each account holder a digitally signed receipt for their balances. No participant could challenge the inclusion of their balances in the Merkle sum tree without disclosing the receipt. Regarding point 4, it is not necessary for a full proof of inclusion under ordinary market conditions, but it could be very useful during a stressed market environment. The Merkle sum tree could be disclosed daily with an attestation that all liabilities were included. Accounting examinations would verify that attestation periodically. Any daily exclusion of liabilities in the Merkle sum tree would constitute a material misstatement and would be subject to legal and regulatory liability. The exchange could also implement a proof of inclusion (POI) protocol that could be used at any time, but most probably would only be used in stressed market conditions. Under that protocol, account holders would be allowed to carry out a simulated run in which they simultaneously use a smart contract or an oracle to check that their liabilities were included, with the aggregate results fully disclosed to the market. Only account holders with digitally signed receipts could use the POI mechanism. The POI could be repeated every day that market stress continues, reducing run risk by giving confidence to the market.

The POI does not require that every participant in the exchange understands how the cryptographic proofs work or, in fact, participates in the protocol. Instead, only the most sophisticated agents would need to monitor the safety of the exchange, similar to how bank solvency is currently monitored. In the conventional banking system in the US, deposit insurance is set at a limited level, US$250,000, such that small participants are protected and, thus, are not expected to surveil the bank. Uninsured depositors, on the other hand, have an incentive to monitor the health of a bank. In a POI protocol, the agents with larger deposits at risk would be expected to take the leadership in conducting a POI. Smaller depositors could easily delegate to them (or to an outside institution) the information necessary for their deposits to be included and verified in any POI.

Random daily POR

Because the proposed POR methodology requires no independent auditors (except for the normal

periodic audit process of the institution), it should be repeated daily on average. To increase security, the timing of the POR should be chosen independently and randomly. To accomplish this, a smart contract could implement a VRF that chooses the time when the POR should be conducted, with an average interval of 24 hours between PORs. Randomising the timing of the POR is important to make it more difficult to borrow assets to pass a POR.

HOW THESE ENHANCEMENTS RESOLVE SOME OF THE DIFFICULTIES WITH CURRENT POR SCHEMES

Recall that the desirable features to include in an enhanced POR methodology were:

1. prove assets without using an independent auditor and without compromising key security by using too many private keys;
2. lower the risks that assets were borrowed to pass the POR;
3. allow any account holder to check that their balance is included in the POR without leaking private information;
4. prevent a malicious actor from falsely claiming that their account was excluded from the Merkle tree;
5. prove that all liabilities were included.

For (1), assets are proved automatically every day using a statistical method and VRF that only requires a limited number of private keys. For (2), the timing of the POR is randomised to occur, on average, daily using a VRF. For (3), ZKPs are supplied so that an account holder or a designee can verify inclusion of the account in the POR without revealing private information. For (4), it is required that only account holders who have a digitally signed receipt can request a ZKP of account inclusion. For (5), a mechanism is provided, most probably used during times of market stress, for account holders to perform a simultaneous check for account inclusion in the Merkle tree and then publicise the aggregated result to the market.

APPLICATIONS TO STABLECOINS

A simplified version of the POR protocol described above can be applied to stablecoins. Gorton and Zhang[11] note that stablecoins are essentially private money, similar to the private notes issued by unregulated banks during the free banking era in the 19th century. Stablecoins may, therefore, be expected to trade below par as well as be vulnerable to bank runs. Gorton and Zhang discuss three options to implement stablecoins: (1) allow only banks to issue stablecoins; (2) require that stablecoins be backed one-to-one; and (3) create new legislation that transforms stablecoins into public money. At the time of writing, option 3 is being actively discussed in Congress.[12]

However, the POR methodology provides another option. Suppose that Treasuries and other fixed income instruments were tokenised. For example, a bank could issue a token for a three-month Treasury instrument. One token would correspond to a specific notional size of a portfolio of three-month Treasuries. Another token might correspond to some other fixed income instrument. The tokens would be held in an account on a blockchain or regulated distributed ledger, with a public key denoting the account name and the private key proving ownership. These tokens, which are referred to as atomic tokens, could be the building blocks of a range of stablecoins, each with different properties. Importantly, these stablecoins could be issued by a financial institution different from the bank that created the tokens.

A stablecoin could be issued that could always be exchanged for some combination of the atomic tokens. For example, the stablecoin could be an ERC-20 instrument along with a smart contract that always allows each stablecoin to be exchanged on demand for the atomic tokens. As long as the public is convinced that the issuing institution can always exchange all stablecoins on demand for the atomic tokens, there will be no need to exchange the tokens and no need to run on the stablecoin. The stablecoins could then function as a medium of exchange that could be used in the conventional payments system. The stablecoins would also possess all the advantages of digital assets, such as programmability. In essence, such a stablecoin would be equivalent to a tokenised money market mutual fund that could also function as a means of payment.

Run risk can essentially be eliminated by a simplified version of the POR. In this case, proof of liabilities is simple, since the stablecoins are the liabilities that can be publicly observed on the blockchain. The POL techniques discussed are therefore unnecessary in this case.

For proof of assets, the issuing institution could implement the probabilistic proof of assets described above to prove that assets equal liabilities to any desired confidence level. Essentially, the issuing institution would need to prove ownership of a subset of the atomic tokens backing the supply of stablecoins every day. Interestingly enough, the probabilistic proof of assets protocol is a modern cryptographic version of the methods used by Scottish banks during the free banking era in 19th century Scotland. As described by Kroszner,[13] Scottish banks hired 'note pickers' to acquire the notes of rival banks; they would then randomly go to those banks to demand that the notes be redeemed. This market mechanism functioned as a controlled run, disciplining banks not to reduce reserve ratios excessively. The probabilistic proof of assets is a similar controlled run, but it allows much higher confidence in the results.

CONCLUSIONS

In this paper, some methods are suggested to improve the POR methodologies currently being deployed by crypto exchanges. These techniques should reduce run risk substantially when coupled with effective internal risk management as well as legal and regulatory compliance. Developing better POR techniques is important not only for making crypto-native institutions less risky, but also for reducing the run risk of banks and money market funds as digital markets continue to develop.

A simplified version of the POR could be used to mitigate run risk of stablecoins. A fully tokenised bank could have all tokenised assets and liabilities on chain, with the real assets corresponding to the tokens held by custodians. POR for banks and money market funds could be verified daily. In addition, since zero knowledge proofs can validate any computation, banks could issue daily ZKPs of adherence to capital standards as well as ZKPs of performance of stress tests. Tokenisation combined with POR and other cryptographic methods could reduce systemic risk while increasing flexibility, efficiency and access to financial products.

References and notes

1 Federal Deposit Insurance Corporation (2023) 'Top Management and Performance Challenges Facing the Federal Deposit Insurance Corporation', available at https://www.fdicoig.gov/sites/default/files/reports/2023-02/TMPC%20Final%202-16-23_0.pdf (accessed 5th July, 2023).

2 Kraken (2022) 'Proof of Reserves Agreed-Upon Procedures Report', available at https://proof-of-reserves.trustexplorer.io/clients/kraken?utm_source=content+announcement&utm_medium=blog&utm_campaign=por+announcement&utm_content=content+link (accessed 5th July, 2023).

3 Lindell, Y. (2020) 'Secure Multiparty Computation', available at https://eprint.iacr.org/2020/300.pdf (accessed 5th July, 2023).

4 Micali, S., Rabin, M. and Vadhan, S. (1999) 'Verifiable Random Functions', IEEE, 40th Annual Symposium on Foundations of Computer Science, available at https://ieeexplore.ieee.org/document/814584 (accessed 5th July, 2023).

5 The Draft Implementation is available at https://datatracker.ietf.org/doc/html/draft-irtf-cfrg-vrf-06 (accessed 5th July, 2023).

6 Code available at https://github.com/nccgroup/draft-irtf-cfrg-vrf-06 (accessed 5th July, 2023).

7 Press, W., Teukolsky, S., Vetterling, W. and Flannery, B. (2007) 'Numerical Recipes: The Art of Scientific Computing' (3rd edition), Cambridge, Cambridge University Press.

8 Berentsen, A., Lenzi, J. and Nyffenegger, R. (2023) 'An Introduction to Zero Knowledge Proofs in Blockchains and Economics', *Federal Reserve Bank of St. Louis Review*, available at https://files.stlouisfed.org/files/htdocs/publications/review/2023/05/12/an-introduction-to-zero-knowledge-proofs-in-

blockchains-and-economics.pdf (accessed 5th July, 2023).
9 Boneh, D. (2022) 'Using zk-SNARKS for Privacy on the Blockchain', Lecture 14, CS 251, Stanford University, available at https://cs251.stanford.edu/lectures/lecture14.pdf (accessed 5th July, 2023).
10 Ben-Sasson, E., Bentov, I., Horesh, Y. and Riabzev, M. (2018) 'Scalable, Transparent, and Post-Quantum Secure Computational Integrity', available at https://eprint.iacr.org/2018/046.pdf (accessed 5th July, 2023).
11 Gorton, G. and Zhang, J. (2023) 'Taming Wildcat Stablecoins', *University of Chicago Law Review*, available at https://live-chicago-law-review.pantheonsite.io/sites/default/files/2023-04/03_Zhang%20%26%20Gorton_ART_Final.pdf (accessed 14th July, 2023).
12 US House of Representatives (2023) 'A BILL to Provide for the Regulation of Payment Stablecoins, and for other Purposes', available at https://docs.house.gov/meetings/BA/BA00/20230613/116085/BILLS-118pih-Toprovidefortheregulation.pdf (accessed 14th July, 2023).
13 Kroszner, R. (1995) 'Free Banking: The Scottish Experience as a Model for Emerging Economies', Policy Research Working Papers, Washington, DC, World Bank Group.