A new Token Management System for Local Communities

Fadi Barbàra
fadi.barbara@unito.it
University of Turin
Torino, Italy, Italy

Flavia Fredda
flavia.fredda@unicam.it
University of Camerino
Camerino, Italy, Italy

Claudio Schifanella
claudio.schifanella@unito.it
University of Turin
Torino, Italy, Italy

ABSTRACT

This paper introduces Locale, a novel system that enables cost-effective fidelization processes in local communities while offering enhanced security guarantees compared to traditional methods. By leveraging the unique properties of blockchain technology, Locale provides a solution that is tailored to the specific needs and dynamics of local communities. Locale has a new time-stamping mechanism based on the pay-to-contract method which embeds the commitments inside the public key instead of a separate field in the transaction. This ensures both authenticity and local privacy since the commitment is not visible by external blockchain observers. Nonetheless, this mechanism offers the same security guarantees of traditional timestamping mechanisms. This innovation has independent relevance beyond the context of Locale and opens up new avenues for research and development in the field. By bridging the gap between blockchain technology and local community development, this research contributes to the advancement of practical and efficient solutions for the common good.

CCS CONCEPTS

• **Security and privacy** → *Digital signatures*; **Hash functions and message authentication codes**; • **Applied computing** → **Digital cash**; **Electronic funds transfer**; • **Networks** → Network File System (NFS) protocol; Cross-layer protocols.

KEYWORDS

blockchain, tokens, merkle tree, local communities, local economy, SDGs

ACM Reference Format:
Fadi Barbàra, Flavia Fredda, and Claudio Schifanella. 2023. A new Token Management System for Local Communities. In *ACM International Conference on Information Technology for Social Good (GoodIT '23), September 06–08, 2023, Lisbon, Portugal.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3582515.3609540

1 INTRODUCTION

The Sustainable Development Goals (SDGs) acknowledge the crucial significance of comprehensive efforts at all levels to attain a sustainable future. To effectively contribute to the attainment of SDGs on a national and global scale, it is essential for local communities to concentrate on a subset of goals that are relevant to their specific area and discern the possible future trajectories of significant determinants that impact local sustainability [28].

It is worth to emphasize the difference between *local communities* and *global* ones. We consider local communities those groups of people living in a common area, interacting with each other and organized around shared values. Generally they are organized in social small units, larger than a household, typically the size of a neighborhood. They are commonly characterized by a *local economy*, in the sense that people in it buy from independent, locally owned business. This occurrence leads to a substantial increase in the frequency of monetary transactions of small amounts involving purchases from neighboring businesses, service providers, and agricultural enterprises, thereby fortifying the economic foundation of the local community. Consequently, in a small community there are frequent interactions with the same few local shops. These interactions are characterized by monetary transactions between people who know each other even if only slightly. For this reason, bearing in mind that one should never place unconditional trust in anyone, the level of trust required when consistently purchasing from unfamiliar and perpetually varying sources is significantly lower compared to these alternative procurement practices.

Taking into account the peculiarities of local communities, it is important to create a system that is easy to use and does not rely on large companies for its operations at the same time. Permissionless blockchains satisfy this need albeit inefficiently: while leading blockchains such as Bitcoin and Ethereum offer secure platforms for global transmission of tokens among non-trusting entities, the consequential creation of a worldwide financial infrastructure entails considerable latencies and costs that present significant barriers to the implementation of local (economic) applications that operate within a particular region or community. As mentioned, local economies rely on trust and reputation fostered through recurrent interactions within a participant community and this should be taken into account and leveraged when producing functional applications.

In these cases it is generally suggested to use off-chain methods such as a Layer 2 (or L2) to mitigate costs. An example is Lightning Network [26] whose purpose is to give users the ability to make small payments: since users do not use the blockchain directly, they save transaction-costs and get (confirmations of) payments relatively quickly (seconds instead of tens of minutes), which also improves the scalability (transaction throughput) of Bitcoin's blockchain. Similar value propositions have the L2 methods in the Ethereum ecosystem. For example, rollups (both optimistic [4] and zero-knowledge [27] based) promise better transaction handling, which makes payments and smart contract calls more efficient in terms of speed, costs and privacy.

Unfortunately L2 systems cannot be realistically employed in the case of neighborhood-based communities. In fact, every L2

GoodIT '23, September 06–08, 2023, Lisbon, Portugal

ACM ISBN 979-8-4007-0116-0/23/09.
https://doi.org/10.1145/3582515.3609540

system requires an onboarding process: a user who wants to use an L2 system must first have funds on that L2 in order to make transactions. From a technical point of view, every move from a blockchain to an L2 requires a transaction on blockchain: in Lightning Network it is necessary to "open a channel" in the form of a transaction to a *multisig*, and a similar onboarding process happens in Ethereum's rollups. It follows that even if a user uses an L2, she needs to have the proper type of funds for each different blockchain in order to be able to do transactions. We judge this assumption to be impractical, at least as of now, in a large number of local communities, due to the high level of infrastructure required, both for the availability of Internet connection and the high degree of knowledge required at every social stratum.

In this paper we present a new method for using tokens within local communities. We worked to strengthen the structure of local economies by exploiting the characteristics discussed above, therefore a new system for managing and exchanging digital tokens within local small businesses is proposed. These tokens may be thought of as digital cards, loyalty points or voting shares. Due to the structure of the local economies, existing levels of trust can be leveraged at the design stage to reduce operating costs. Hence we propose a token system with moderate usage of blockchain, which periodically commmits to batches of transactions instead of sending all of them sequentially.

Our contributions. Our contributions are:

- We present Locale and show how it is possible to create fidelization processes in local communities which are *less* expensive than traditional methods but provide *more* security guarantees.
- We propose a new time-stamping mechanism which differently from the previous ones is locally-private and provides authenticity, which can be of independent interest.
- We provide a proof of concept implementation of Locale[1]

2 RELATED WORKS

At least from 2004, with the works by D. Meadows and J. Randers [21] and by M. E. B. Seiffert and C. Loch [29], there has been a growing trend towards the emergence in research about common good applications. Indeed, as stated by P. Ghisellini, C. Cialani et al. [13], to realize the common good in an effective way involves balancing economy, environment and society.

The importance of blockchain in applications for common good has been ascertained. An example of this is provided by A. Upadhyay, S. Mukhuty et al. in [30]. They conducted a comprehensive analysis of the prospective impact of blockchain technology on the circular economy, with a particular emphasis on the domains of sustainability and social responsibility. Unfortunately these suggestions have not been fully exploited yet in the academic literature, as there is still a lack of projects applying most of the aforementioned principles. In fact we are aware of only two projects that rely on the security and integrity properties of blockchains (also referred to as "Layer 1" or *L1*) to contribute to establishing a more balanced and cohesive relationship among the economy, society, and natural environment.

[1]See https://github.com/disnocen/locale/

The first one is by S. Jainand and R. Simha in [16] who propose a blockchain system called *Direct Cash* for the common good in the context of charitable donations. Their system is composed by a digital currency linked to a government-backed currency through a central bank and a permissioned blockchain, with the aim of enbling secure donations. The other is the work by E. Lavoie and C. Tschudin which exploited the peculiar requirements of local communities in [18]. They proposed the design of tokens to be used locally, into intra-community economies, by for example small retailers. Their system however has some shortcomings such the impossibility of verifying the double-spending. In that place, the authors propose the weaker property of double-spending *detection*.

While there are few works based on L1 systems, we can identify scalability proposals on Layer 2 (L2) as works sharing our goals. In fact several L2-based systems aim to enhance the scalability of the underlying Layer 1 (L1) networks, in order to achieve cheaper and frequent transactions. These systems can be broadly categorized into three types: *payment channels*, *sidechains* and *rollups*.

Payment channels provide a mechanism for off-chain token exchanges, enabling faster and more cost-effective transactions [10]. One of the earliest and most well-known payment channel designs is the Lightning Network [26], which serves as the L2 system for the Bitcoin network. The Lightning Network operates by establishing bi-directional payment channels between participants, allowing them to conduct multiple transactions without recording each one of them in the Bitcoin blockchain. These transactions are instead settled on the blockchain only when the channel is closed as sum of the intermediate ones. This approach significantly improves scalability and reduces transaction fees on the Bitcoin network [26]. However, the Lightning Network has some limitations. To utilize this L2 system effectively and securely, users are required to possess bitcoins and interact with the Bitcoin blockchain. This reliance on the Bitcoin blockchain may not be a reasonable expectation for a system aiming to seamlessly integrate with local communities in terms of user experience and costs.

Sidechains have been theorized as an early scaling solution [5]. In Bitcoin, the most prominent sidechain is the Liquid Network [11, 23], while potential sidechain solutions in the EVM-compatible realm have been Plasma [25] and recently one of its concrete realizations, i.e. Polygon [17]. Sidechains operate as a separate blockchain network connected to the mainnet through a set of secure bridges. By leveraging these sidechains, users can offload transactions from the main network, benefiting from faster transaction processing and lower fees.

Finally, Rollups are another scaling solution currently studied for EVM-compatible blockchain [4, 27]. They involve computing transaction data off-chain and then posting the summary to the Ethereum network. There are two primary types of rollups: Optimistic Rollups [4] and Zero Knowledge Rollups [27]. Optimistic Rollups operate on the principle of optimism, i.e. assuming all transactions are correct unless proven wrong. Nodes of an optimistic rollups execute transactions off-chain and submit a summary to the Ethereum network. In the event of an invalid transaction, there's a challenge period where fraudulent transactions can be identified and rolled back.

On the other hand, Zero Knowledge Rollups (also known as ZK-Rollups) also bundle many off-chain transfers into a single

transaction but use zero-knowledge proofs to ensure validity. Zero-knowledge proofs allow one party to prove to another that a statement is true without conveying any additional information. In the context of rollups, they're used to prove that off-chain transactions are valid, without needing to provide the entire transaction data on-chain.

Users of payment channels, rollups or sidechain suffer from the same poor user experience: they need to exchange tokens to and from these L2 systems to utilize either them. This token swapping process introduces additional complexity and potential friction for users. Therefore, it is important to explore alternative approaches that offer a seamless user experience without the need for token transfers between chains.

3 BACKGROUND

3.1 Commitments

Cryptographic commitments play a crucial role in various cryptographic protocols and systems, providing a means to securely and reliably bind information without revealing its content. In this section, we briefly discuss the fundamental properties of commitments, for more information see e.g Chapter 2 of [14].

Cryptographic commitments must satisfy two properties: the *hiding* and *binding* property. Roughly speaking, the hiding property of commitments ensures that the committed information remains hidden from any observer, even if it possess significant computational power or knowledge about the commitment scheme. In other words, this property guarantees that an adversary cannot extract any useful information about the committed value or infer any relationship between different commitments. The binding property of commitments ensures that once a value is committed, it becomes infeasable to alter or substitute it with a different value without detection. This property prevents malicious parties from tampering with the committed information and maintains the integrity and trustworthiness of the commitment scheme.

Cryptographic hash functions are widely regarded as a reliable and efficient mechanism for implementing commitments in various cryptographic applications.

By leveraging them, we can achieve both the hiding and binding properties discussed earlier. The hiding property is achieved by applying the hash function to the committed value, creating a commitment that reveals no information about the original value. The binding property is achieved by incorporating additional cryptographic techniques, such as using a secret key or a random nonce, to prevent malicious parties from tampering with the commitment or providing alternative values that produce the same hash digest.

3.2 Decentralized Storage Systems

In the following we analyze different decentralized storage systems, with the goal of understanding which one may be the best candidate for Locale. More information for the interested reader can be found in the work by Daniel and Tschorsch [9]

Peer-to-peer (P2P) data networks have undergone significant advancements since their inception over two decades ago. The evolution of these networks has led to the development of new technologies and ideas that have shaped the current state of P2P networks. As per the work in [9], we may identify three eras of P2P data networks. The first generation was characterized by the emergence of P2P file sharing and networks like BitTorrent [8] and Kademlia [20] (1999–2002). The second phase brought forth novel ideas, such as information-centric networking and cryptocurrencies. Finally, from around 2014, there has been a rise of a new generation of P2P data networks spearheaded by the invention of InterPlanetary File System (IPFS)[7]. We focus on projects created during the third phase since they generally have the best integration with other decentralized systems and blockchains in particular.

The Interplanetary File Storage (IPFS) is a decentralized storage system. It is composed of two main parts: a name system (IPNS) and the actual storage. Each content in the storage has a unique *content identifier*, or *cid*, based on the hash of the content. It is possible to upload static (e.g. a PDF) or mutable (e.g. a HTML of a webpage) files in the storage. Consequently there are two ways to point to the file. The first one is by simple pointing to the *cid* of the file. This solution works for static files whose update is actually a deletion plus re-upload, but it does not work for mutable files. Thinking about the webpage example mentioned before, it is easy to see how impractical it is to share a new URL for each modification of the HTML file. To solve this problem, it is possible to use a more static link leveraging the IPNS. Similarly to a DNS, the IPNS translate an easy-to-remember name into the latest *cid*. See Figure 1 for a graphical representation of the IPNS system. More information can be found in the documentation [2].

But IPFS is not the only decentralized storage system. Arweave [19], for example, is a data storage protocol that facilitates permanent data storage with a one-time fee. The system is secured using a decentralized network of miners who are incentivized through the exchange of AR tokens. Through Arweave's endowment, the protocol ensures that data can be accessed indefinitely, sustained through an immutable risk model. This system enables the creation of the "permaweb," a digital realm of pages and applications that exist permanently, hosted within Arweave's architecture.

Another decentralized storage system is provided by Storj [3]. Storj is an open-source software that allows users, ranging from individuals with NAS devices, always-on desktop computers, businesses, or data centers, to share their unused disk space and bandwidth with the network. The software aggregates all of the available resources to create decentralized cloud storage (DCS) service that developers can utilize. The distributed cloud object storage service offered by Storj aims to achieve 99.95% availability rate.

Storj's peer network ecosystem encompasses three primary components: the Storage node, Uplink Clients, and Satellites. The Storage Node provides the network its storage capacity and bandwidth by enabling users to share their unused hard drive space, while remunerated for their participation. Data is client-side encrypted, and erasure-coded. The Uplink Clients offer developer tools (both hosted or self-hosted) to upload and download data using Storj DCS. This feature handles end-to-end encryption by default and implements erasure coding, splitting files into 80 parts, each distributed across different storage nodes. The Satellite forms a hosted set of services responsible for managing access control, metadata, ensuring storage node reputation, data audit, data repair, and compensating storage node operators. The satellities are managed by Storj Labs.

For Locale we decided to use IPFS.

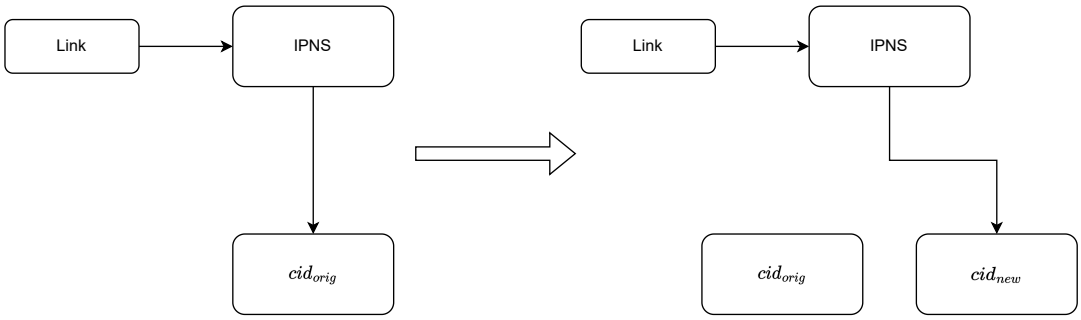

Figure 1: Graphical representation of the flow from a static link to a mutable content in IPFS through IPNS

3.3 Pay to Contract

Blockchains used as time stamping mechanism have a way to store (small) amounts of data. For example, in Bitcoin it is possible to store data using the `OP_RETURN` opcode [24], while in Ethereum it is possible to use the `data` field of the transaction [31]. Unfortunately both methods are costly (especially in Bitcoin since coins sent to a `OP_RETURN` are unspendable) due to higher fees.

Nowadays it is possible to solve this problem by a two steps process: commit to some data in the blockchain and store the original in a decentralized storage service such as IPFS (Section 3.2). While both the `OP_RETURN` opcode and the `data` field can be use to store the commitment, this would not solve the aforementioned problem. Another way to commit to the data is the *pay-to-contract* method as presented by Gerhardt and Hanke in [12] which only leverage elliptic curve cryptography and therefore can be used on any blockchain relying on that, such as Ethereum (on which our implementation is built) or Bitcoin.

The method works as follows. Assume Alice has to pay Bob on public key X (related private key x) for some service or item whose receipt is C, as done for example in [6]. Alice constructs the modified key $X' = (X + \mathcal{H}(X||C) * G)$, where $\mathcal{H}$ is a hash function, $||$ the concatenation symbol, and G the `secp256k1` generator point used both in Ethereum and Bitcoin. Then she sends a transaction to Bob at the address corresponding to X'. To redeem the coins, Bob computes the value $x' = x + \mathcal{H}(X||C)$ and uses x' as private key corresponding to X'.

In our system we use the pay-to-contract mechanism in the following way. Assume the operator has key pair (x, X), and assume at epoch i the the content identifier cid_i, the signature σ_i of Merkle root is $root_i$. Then the operator forwards the coins in the transaction for epoch $i-1$ to the (address related to the) public key $X'_i = (X + \mathcal{H}(X||cid_i||\sigma_i) * G)$. This way the operator commits to both the the cid_i and $root_i$ at epoch i, authenticate $root_i$ via the signature σ_i and prevents forgeries of the current Merkle tree via the double spend mechanism intrinsic in the Bitcoin or Ethereum blockchains.

4 DESIGN AND OPERATIONS

As mentioned in Sections 1 and 2, our goal is to create a system to exchange tokens of small value between citizens in a (at least geographically) local community who are united by their affection for the same store for their purchases. We can imagine these tokens as representing fidelity points, voting shares or discount coupons in a neighborhood store. Since the possible users of this system already trust the store operators, we want to take advantage of this reality to create a tokens exchange system that is easier to use than traditional L2 systems and less expensive than L1 systems for users. At the same time, the system we propose is more secure than the traditional pen-and-paper-based system since the cryptographic processes underlying Locale means that neither users nor operators can forge tokens.

Therefore, in this section, we explain the operations that compose the whole Locale system. We start by presenting the architectural components in Section 4.1 and then we follow with the four operations of Locale: Receiving a token (Section 4.2), Committing the Merkle tree to the blockchain (Section 4.3), Spending a token (Section 4.4) and Transfering a token (Section 4.5).

4.1 Architectural Components

We start by formally introducing the actors in the Locale system:

- **Operator** (O): the one who manages the creation and sending of tokens to users upon purchase of the product he or she sells. Generally, the store cashier or the store owner may act as the Operator. The Operator is also responsible for sending a commitment to the blockchain at the end of each epoch. Multiple times, we identify O with the software client owned by O since any operation will be automatically performed by the software clients.
- **User** (U): generally the customer. The user's goal is to be retained, to receive a new token for each purchase (or in proportion to the value of the purchase) and be able to spend these tokens to get a perk (discount, exercise a voting right or other applications). Multiple times, we identify U with the software client owned by U since each operation will be automatically performed by the clients. Note that the list of users is theoretically an evergrowing list, but there is no way in Locale to keep track of it. We may talk about users U_1, U_2, U_k when we need to address more of them.
- **Blockchain Nodes**: the nodes of a blockchain are those who have write access and create the new blocks. Without going into details, the blockchain nodes we refer to are the "miners" in the case of blockchain PoW (e.g. Bitcoin) or "validators" in the case of PoS blockchains (e.g. Ethereum)
- **Decentralized Storage Nodes**: the nodes of a decentralized storage system are those who maintain the storage. They

have read access to the content. We describe our system by using IPFS as mentioned in Section 3.2.

For the purposes of this paper, we assume that blockchain nodes and decentralized storage nodes are honest: this is a sensible assumption since we assume Locale deals with a blockchain whose rules and incentives force nodes to be honest in the long run and a working incentive system for the decentralized storage nodes. On the other hand, we assume that both the operator and the users can be actively malicious. In particular, we assume that the operator wants to cheat while remaining undetected e.g. by removing tokens belonging to users: this assumption makes sense since the operator is incentivized not to give discounts so as to gain more from a monetary point of view. Similarly, we assume that the users are active adversaries, e.g., that they intend to forge tokens so as to receive more discounts than those to which they are entitled.

We assume the existence of the following structures:

- **Merkle Tree** ($\mathcal{M}$): we assume that there is an accumulator that keeps track of the tokens created by O . Here we use an accumulator in the form of a Merkle tree [22], but other accumulators are possible. $\mathcal{M}$ is not static but changes with each transaction. $\mathcal{M}$ is committed to the blockchain periodically. Following the general terminology, we will say that a transaction is a leaf to the Merkle tree.
- **Decentralized Storage** : we assume that O interacts with a decentralized storage to store the current version of $\mathcal{M}$. Throughout the paper we will assume that the decentralized storage has ability to maintain multiple versions of the same file simultaneously reachable, similar to IPFS/IPNS (Section 3.2). Note that no maintenance of decentralized storage is required by O.
- **Public Board** : we assume O maintains a public boards where some pieces of information are published, such are the initial public key of O pk_O, the epoch number, the IPNS URL where the last version of the Merkle tree is published and other optional data needed for the smooth operations of the actual business. In practice that board may be a page in the shop site or social media page.

We stress the fact that no technical knowledge is required for O to operate with these cryptographic structures: the application we propose will abstract this knowledge for both the operator and the users. In particular, O is not required to set up blockchain nodes or decentralized storage servers.

4.2 Receiving a Token

When a user U buys a product (Step 0 in Figure 2), the operator O creates and stores a token (Step 1), sends pieces of information to the U (Step 2) and stores the Merkle root to the decentralized storage (Setp 3*a*). It is expected that the amount of tokens is proportional to the amount spent, but for easiness of explanation we assume each product amounts to one token only. Each token can be thought of as a fidelity point, voting share, or the digital version of a stamp on a card. We now give a more detailed explanation of how these steps are executed.

We first argue that it is possible to uniquely identify each purchase, and consequently create a uniquely named token: this is necessary to uniquely refer to tokens when exchanged or spent. The assumption of uniqueness stems from the fact that packed products have a bar code that uniquely identifies them. It stands to reason that unpacked products (such as a coffee served at the bar) might be similarly identified, at least within the store that sells them. We call this product identification code $prodID$. Furthermore, we assume it is possible to ascertain the time the product was sold down to a small enough unit of time, e.g. the milliseconds. We denote time as τ [2].

Given these assumptions, it is possible to name a token as $tokenID = \mathcal{H}(prodID||\tau)$ where we used $||$ for concatenation. Since we have chosen a very small unit of time, it is highly unlikely that in a physical, not digital, context it is possible for the same product to be bought in the same millisecond by two people simultaneously. For this reason, we can assume that tokens of the form $\mathcal{H}(prodID||\tau)$ are unique, at least with respect to the micro-economy generated within the store[3].

We now specify how a user may receive a token. When a user buys a product, the owner computes a unique $tokenID$, as explained above and represented in Step 0 of Figure 2, for the transaction and adds it to the Merkle tree $\mathcal{M}$ storing it in the owner's database (Step 1). The owner then sends the $tokenID$ to the user: this action can be part of the application depending on the actual implementation (Step 2). The application then recomputes the whole Merkle tree and posts it on the IPFS decentralized storage (Step 3*a*), with a new cid but the same IPNS-URL (see Section 3.2). Finally, any user can see the updated $\mathcal{M}$ by going to the IPNS-URL, visible in the public board.

4.3 Chain Committing

We suppose time is divided into epochs. For example, each epoch may last two weeks. In Locale, the operator O is responsible to commit the current state of the Merkle tree to a blockchain (of course this operation can be automated too) (Step 3*b*, Figure 2). Recall that O has a keypair (sk_O, pk_O) and that pk_O is public. Also, for the sake of this section, we denote the Merkle tree at the i-th epoch as $\mathcal{M}_i$. Consequently the root is denoted as $root_i$. Similarly, the current cid of $\mathcal{M}_i$ is denoted cid_i. In the following we explain the needed steps.

If $i = 1$, i.e. this is the first time that O commits to a blockchain, then we assume O has funds in an addresses related to the public key pk_O. As mentioned in Section 3.3, the committing is done with a simple transaction differently from other time-stamping systems that use blockchain specific code. The transactions on blockchain are a way to commit the data only: the funds are transferred between addresses owned by O. To do that, O sends a transaction from pk_O to a new address related to the public key pk_O^1 (since we are in the first epoch). The public key is computed in the following way:

$$pk_O^1 = (pk_O + \mathcal{H}(pk_O||root_1||cid_1) * G)$$

[2]To do this, we do not assume a universal clock, but assume that the internal clock of the store's cash register is authentic. This is one of the points where we deviate from general cryptographic assumptions to create a system that can be used in a local setting for the purpose of common good. For these reasons, we decided to move toward the "usability" side in the classic security vs. usability trade-off of applied cryptography.

[3]If one wants to account for the store as well, it is easily possible to change the creation specification to $\mathcal{H}(shopID|||prodID||\tau)$ making the token identifier unique even at the level of the neighborhood economy

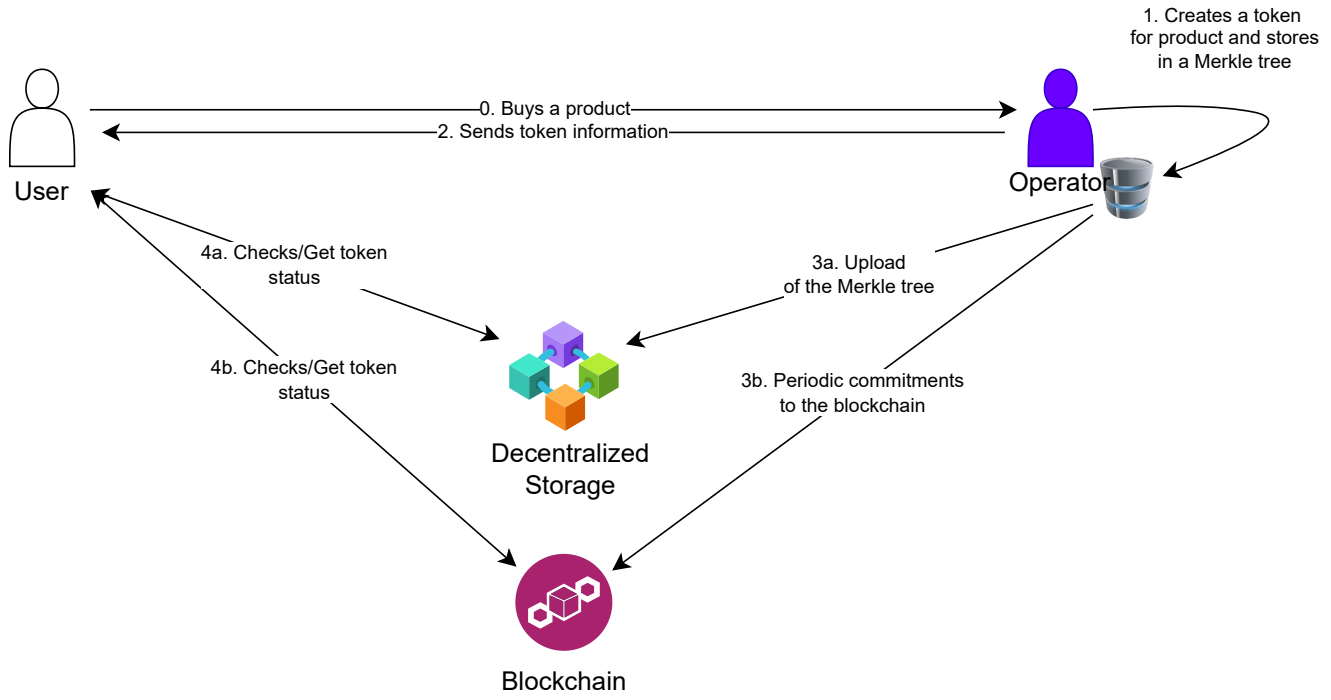

Figure 2: Flow of operations between user and owner in Locale

It is easy to see that the value $sk_O^1 = sk_O + \mathcal{H}(pk_O||root_1||cid_1)$ is the related private key to pk_O^1. Since sk_O is known only by O by assumption, then sk_O^1 is known only by O too[4].

Let's call the address related to a public key pk as addr(pk). Then we may denote the transaction from addr(pk_O) to addr(pk_O^1) as tx_1 since it is the first epoch. Note that the content of the hash is clearly a commitment to: i) the public key, ii) the current state of the Merkle tree and iii) the location of the Merkle tree. Furthermore, only users who have these three pieces of information can check the validity of the commitment by checking the blockchain (Step 4*a* and 4*b*, Figure 2). This way observers from outside the local community are unable to learn about activities, solving the data leakage problem (see Section 6).

It is easy to extend to the case of a general epoch i. In this case O performs a transaction tx_i from the address related to public key pk_O^{i-1} to pk_O^i, where

$$pk_O^i = (pk_O + \mathcal{H}(pk_O||root_i||cid_i) * G)$$

Again, the private key to pk_O^i is $sk_O^i = sk_O + \mathcal{H}(pk_O||root_i||cid_i)$.

4.4 Spending a Token

To spend a list with n token, the first step is for U to prepare the list and send it to O. After receiving the list, O "deletes" the n leaves relative to the n tokens in $\mathcal{M}$ and gives the perk to U. Here the operation of deletion of a token is actually a *substitution* of the leaf content with a default `void` token which represents the deletion. The rationale for substitution instead of deletion is that actually removing a leaf means (in the worst case) recomputing all the hashes in M at all levels, since if a leaf is removed, the others have to be shifted by one. On the other hand if we only substitute one leaf, the system always requires $\log(n) + 1$ recomputations, where n is the number of tokens emitted by O.

Note that the double-spend problem here is inherently "solved", since a Merkle tree with all the spendable tokens is readily available for each user and the operator to see at the IPNS-URL. If a user U tries to double spend a token, the operator O can easily catch the attack since the leaf is not present in the Merkle tree.

4.5 Transferring a Token

Differently from the blockchain model, in Locale a transfer of a token between two users is more similar to a banknote transfer than a blockchain transaction. In fact, if user U_1 wants to transfer a token tx to U_2, U_1 just sends the tx hash to U_2. After receiving it, U_2 can either save it or spend it as part of a n-items list of token as explained in Section 4.4.

5 IMPLEMENTATION

In this section, we present the implementation details of our proposed system, focusing on a specific use case: a café where customers can purchase three kinds of beverages, namely a cup of coffee, a cup of tea, or a glass of water. This particular scenario

[4]We obviously assume that the Discrete Logarithm Problem is hard in the group of the blockchain Locale is used in, as it is assumned in any blockchain related protocol. In practice that means that it is impossible to know sk_O (or sk_O^1) by just knowing pk_O (resp. pk_O^1). Consequently it is impossible to know sk_O^1 even if every Locale users knows $\mathcal{H}(pk_O||root_1||cid_1)$.

allows us to showcase the creation of unique product identifiers and the recording of blockchain transactions, as described in Section 4.

The primary objective of our system is to utilize tokens as loyalty points, enabling customers to accumulate rewards for their purchases and customers can redeem them for a free cup of coffee once they have accumulated ten tokens. This café setting perfectly aligns with our system's goals, as it allows us to illustrate how the proposed solution can be seamlessly integrated into a real-world business environment.

The system is split into two parts: the frontend and the backend. The backend of the application handles the creation of the merkle tree, the update of files in the IPFS storage, and the periodic commitment in the blockchain. This division of functionality ensures that the frontend remains user-friendly and easy to use, while the backend handles the tasks related to Merkle tree creation and blockchain commitments. This design allows for an efficient and effective system that reduces the workload on both the operator and the user while ensuring the security of the system.

The backend of Locale is mainly written in Python and the application is based on the Flask [1] package. By using Python, we leverage the libraries on elliptic curve cryptography (ECC) key manipulations, especially the `ecdsa` and `eth_account` package. As DCS we settled on IPFS for this use case. To deal with the blockchain and IPFS nodes, we used NodeJS. In particular, we use the `w3` package to deal with the upload of the Merkle tree root on IPFS. In this implementation, we chose to use the Ethereum blockchain. In particular we used the `ganache-cli` package to create a test network. This makes the implementation suitable for any EVM-compatible blockchain.

The frontend is divided into two parts, the operator/salesman view (Figure 3), and the user view. The operator view allows the operator to mint tokens based on the product and automatically updates the IPFS file without further intervention. Additionally, the operator can publish the periodic commitments on the blockchain from this view. The user view, on the other hand, enables the user to check if a token has been registered in a specific merkle tree.

Finally, note that since there are no smart contract interaction involved in the implementation, this EVM-compatible instantiation can be repurposed to work on less featureful blockchains such as Bitcoin (see Section 3.3).

6 ANALYSIS

In this section, we analyze the key aspects of our proposed system and discuss how it operates within the local community, while addressing the notion of trust commonly associated with blockchain technology.

Our system, Locale, utilizes the blockchain periodically rather than relying on it for every transaction. While this approach may deviate from the typical "trustless" nature of blockchain systems in the broader community, we contend that there are no additional trust assumptions required beyond those inherent in the local community. We argue that if a user, denoted as U, chooses to patronize a particular shop regularly enough to derive benefits from loyalty, it is reasonable to assume that U likes and trusts the personnel of that shop. Therefore, the existing trust within the local community serves as a fundamental basis for our system.

To reinforce and strengthen this baseline trust, cryptographic techniques are employed, and periodic commitments are made to the blockchain. These commitments serve the purpose of enhancing transparency and accountability within the system. Importantly, these periodic commitments incur minimal costs for the customer/user, U, effectively making them almost free from their perspective.

The operator, denoted as O, bears minimal costs in running the fidelization program, primarily limited to the L1 transaction fees paid periodically. By carefully timing the blockchain commitments, O can keep the costs low. For example, by batching multiple commitments together or leveraging periods of lower network congestion, the operator can optimize the gas fees associated with these periodic transactions. Note that the gas fees are those of simple EOA to EOA transfers since no smart contract is needed for the whole Locale operation.

From the point of view of the user's data, Locale is better suited for local communities than systems relying on blockchains exclusively. In fact, since any transaction complex enough to require a smart contract is currently completely transparent by default, even obfuscation mechanisms can not hide data well enough to block data mining or behaviour patterns [15]. Consequently an adversary would be able to exploit the data and obtain more information than what it is entitled to. The real problem stems from publishing the data in the blockchain in the first place.

This can not happen in Locale, since real data never leaves the physical boundaries of the local community (unless an adversary can compromise the computer of the operator, a scenario which is outside the scope of the paper). In fact, the only data published on the blockchain is a commitment to the sensible data, which by definition has the *hiding* property (see Section 3.1) and no amount of data mining can realistically extract information for the adversary.

7 CONCLUSIONS AND FUTURE WORKS

We presented Locale, a mainly off-chain system that uses a blockchain to time-stamp commitments of a Merkle tree of tokens emitted by small-shop owners which can represent loyalty points, discount coupons or voting rights. The main use case is to create fidelization and incentives for citizens in a local community to use local shops so that the local micro-economy may be invigorated.

Instead of relying on normal timestamping mechanisms and publishing the Merkle tree root on chain, we use the Pay-to-Contract method to never reveal the commitment on chain, so that only the fidelized users and the operator are able to discern the blockchain data, avoiding any data-leakage problem. Despite the periodic (instead of constant) publishing of data on the blockchain, we avoid any double spending problem nonetheless.

We argue that further research and development are necessary to establish realistic assumptions and implementations regarding trust in blockchain-related applications, particularly concerning the common good and local community development. The trust assumptions in our proposed system differ significantly from those applicable to global-scale systems. Therefore, exploring and defining trust in the context of local communities remains an important area for future investigation.

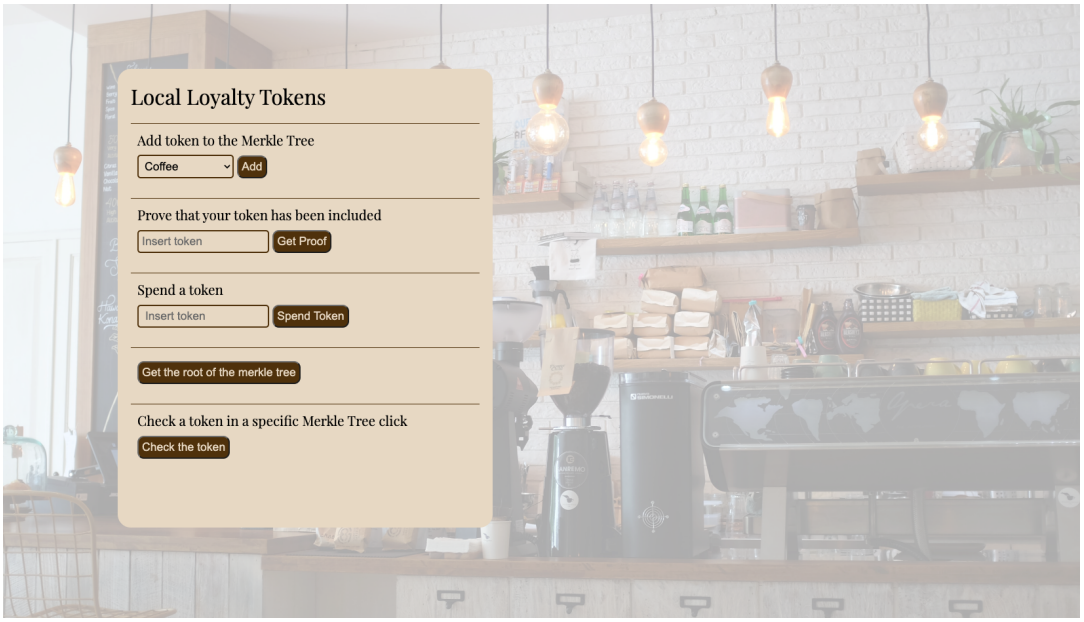

Figure 3: The view of the web application of Locale

REFERENCES

[1] [n. d.]. Flask Userguide. https://flask.palletsprojects.com/en/2.3.x/. Accessed: 2023-05-31.

[2] [n. d.]. InterPlanetary Name System (IPNS). https://docs.ipfs.tech/concepts/ipns/. Accessed: 2023-04-04.

[3] [n. d.]. Understanding Storj DCS. https://docs.storj.io/dcs/concepts/overview. Accessed: 2023-04-26.

[4] John Adler. 2019. Minimal Viable Merged Consensus. EthResear.ch. https://ethresear.ch/t/minimal-viable-merged-consensus/5617

[5] Adam Back, Matt Corallo, Luke Dashjr, Mark Friedenbach, Gregory Maxwell, Andrew Miller, Andrew Poelstra, Jorge Timón, and Pieter Wuille. 2014. Enabling blockchain innovations with pegged sidechains. http://kevinriggen.com/files/sidechains.pdf.

[6] Fadi Barbàra and Claudio Schifanella. 2020. P2T: Pay to Transport. In *Euro-Par 2020: Parallel Processing Workshops - Euro-Par 2020 International Workshops, Warsaw, Poland, August 24-25, 2020, Revised Selected Papers (Lecture Notes in Computer Science, Vol. 12480)*. Springer, 66–77. https://doi.org/10.1007/978-3-030-71593-9_6

[7] Juan Benet. 2014. IPFS - Content Addressed, Versioned, P2P File System. *CoRR* abs/1407.3561 (2014). arXiv:1407.3561 http://arxiv.org/abs/1407.3561

[8] Bram Cohen. 2008. The BitTorrent Protocol Specification. http://www.bittorrent.org/beps/bep_0003.html. Accessed: 2023-04-26.

[9] Erik Daniel and Florian Tschorsch. 2022. IPFS and Friends: A Qualitative Comparison of Next Generation Peer-to-Peer Data Networks. *IEEE Commun. Surv. Tutorials* 24, 1 (2022), 31–52. https://doi.org/10.1109/COMST.2022.3143147

[10] Christian Decker and Roger Wattenhofer. 2015. A fast and scalable payment network with bitcoin duplex micropayment channels. In *Stabilization, Safety, and Security of Distributed Systems: 17th International Symposium, SSS 2015, Edmonton, AB, Canada, August 18-21, 2015, Proceedings 17*. Springer, 3–18.

[11] Johnny Dilley, Andrew Poelstra, Jonathan Wilkins, Marta Piekarska, Ben Gorlick, and Mark Friedenbach. 2016. Strong Federations: An Interoperable Blockchain Solution to Centralized Third Party Risks. *CoRR* abs/1612.05491 (2016). arXiv:1612.05491 http://arxiv.org/abs/1612.05491

[12] Ilja Gerhardt and Timo Hanke. 2012. Homomorphic Payment Addresses and the Pay-to-Contract Protocol. *CoRR* abs/1212.3257 (2012). arXiv:1212.3257 http://arxiv.org/abs/1212.3257

[13] Patrizia Ghisellini, Catia Cialani, and Sergio Ulgiati. 2016. A review on circular economy: the expected transition to a balanced interplay of environmental and economic systems. *Journal of Cleaner Production* 114 (2016), 11–32. https://doi.org/10.1016/j.jclepro.2015.09.007 Towards Post Fossil Carbon Societies: Regenerative and Preventative Eco-Industrial Development.

[14] Oded Goldreich. 2001. *The Foundations of Cryptography - Volume 1: Basic Techniques*. Cambridge University Press. https://doi.org/10.1017/CBO9780511546891

[15] Richard Hobeck, Christopher Klinkmüller, H. M. N. Dilum Bandara, Ingo Weber, and Wil M. P. van der Aalst. 2021. Process Mining on Blockchain Data: A Case Study of Augur. In *Business Process Management*. Springer International Publishing, Cham, 306–323.

[16] Shweta Jain and Rahul Simha. 2018. Blockchain for the Common Good: A Digital Currency for Citizen Philanthropy and Social Entrepreneurship. In *2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*. 1387–1394. https://doi.org/10.1109/Cybermatics_2018.2018.00238

[17] Jaynti Kanani, Anurag Arjun, Sandeep Nailwal, and Mihailo Bjelic. 2021. Polygon Whitepaper. *Whitepaper* (2021).

[18] Erick Lavoie and Christian Tschudin. 2022. Local Crypto-Tokens for Local Economics. In *Proceedings of the 3rd International Workshop on Distributed Infrastructure for the Common Good* (Quebec, Quebec City, Canada) *(DICG '22)*. Association for Computing Machinery, New York, NY, USA, 43–48. https://doi.org/10.1145/3565383.3566113

[19] luckyr13.ar. 2021. What is Arweave? https://arwiki.arweave.dev/#/en/Arweave. Accessed: 2023-04-26.

[20] Petar Maymounkov and David Mazières. 2002. Kademlia: A Peer-to-Peer Information System Based on the XOR Metric. In *Peer-to-Peer Systems*, Peter Druschel, Frans Kaashoek, and Antony Rowstron (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 53–65.

[21] Donella Meadows, Jorgen Randers, and Dennis Meadows. 2004. *Limits to growth: The 30-year update*. Chelsea Green Publishing.

[22] Ralph C. Merkle. 1988. A Digital Signature Based on a Conventional Encryption Function. In *Advances in Cryptology — CRYPTO '87*, Carl Pomerance (Ed.). Springer Berlin Heidelberg, Berlin, Heidelberg, 369–378.

[23] Jonas Nick, Andrew Poelstra, and Gregory Sanders. 2020. Liquid: A bitcoin sidechain. https://blockstream.com/assets/downloads/pdf/liquid-whitepaper.pdf. Accessed: 2023-05-31.

[24] Occupy paul st. 2015. OP_RETURN. https://en.bitcoin.it/wiki/OP_RETURN. Accessed: 2023-04-04.

[25] Joseph Poon and Vitalik Buterin. 2017. Plasma: Scalable autonomous smart contracts. *White paper* (2017).

[26] Joseph Poon and Thaddeus Dryja. 2016. The bitcoin lightning network. https://1bitcoin.ca/s/lightning-network-paper.pdf. (2016). Accessed: 2023-05-31.

[27] sassal. 2020. ZK-Rollups. Ethhub. https://docs.ethhub.io/ethereum-roadmap/layer-2-scaling/zk-rollups/

[28] Jan Schwiderowski, Asger Balle Pedersen, and Roman Beck. 2023. Crypto Tokens and Token Systems. *Information Systems Frontiers* (2023). http://doi.org/10.1007/s10796-023-10382-w

[29] Mari Elizabete B. Seiffert and Carlos Loch. 2005. Systemic thinking in environmental management: support for sustainable development. *Journal of Cleaner*

Production 13, 12 (2005), 1197–1202. https://doi.org/10.1016/j.jclepro.2004.07.004

[30] Arvind Upadhyay, Sumona Mukhuty, Vikas Kumar, and Yigit Kazancoglu. 2021. Blockchain technology and the circular economy: Implications for sustainability and social responsibility. *Journal of Cleaner Production* 293 (2021), 126130. https://doi.org/10.1016/j.jclepro.2021.126130

[31] Gavin Wood. 2016. Ethereum: a secure decentralised generalised transaction ledger. https://ethereum.github.io/yellowpaper/paper.pdf. Accessed: 2023-04-04.