Tampereen ammattikorkeakoulu

Potential Use Cases for Non-fungible Tokens in Combination with Physical Art

Mikko Vinnari

OPINNÄYTETYÖ
Toukokuu 2021

Tieto- ja viestintätekniikan tutkinto-ohjelma

TIIVISTELMÄ

Tampereen ammattikorkeakoulu
Tieto- ja viestintätekniikan tutkinto-ohjelma

VINNARI, MIKKO:
Mahdollisia käyttötapauksia ei-korvattaville poleteille fyysisessä taiteessa

Opinnäytetyö 31 sivua
Toukokuu 2021

Lohkoketju-pohjaisia poletteja käytetään tällä hetkellä digitaalisen taiteen kaupankäyntiin. Tässä opinnäytetyössä tutkitaan millaisia käyttötapauksia vastaaville poleteille voisi olla fyysisessä taiteessa.

Kolme käyttötapausta tunnistettiin. Poletteja voitaisiin käyttää taiteen omistajaketjun selvittämisessä sekä taiteen omistajat voisivat varmentaa omistuksensa niillä. Lisäksi ehdotetaan järjestelmää, jolla voisi kumota sekä vahvistaa taideteoksen aitouden.

Kaikki tunnistetut käyttötapaukset tarjoavat huomattavia etuja eikä niiden käyttöönotto ole teknisesti erityisen vaativaa. Tosin polettien käyttö vaatii käyttäjiltä jonkin verran uusien käytäntöjen opiskelua. Lisäksi, ennen käyttöönottoa, käyttöön otettavan lohkoketjun kelpoisuus täytyisi arvioida erikseen.

Asiasanat: ethereum, lohkoketju, poletti, taide

ABSTRACT

Tampereen ammattikorkeakoulu
Tampere University of Applied Sciences
Degree Programme in ICT Engineering

MIKKO VINNARI:
Potential Use Cases for Non-fungible Tokens in Combination with Physical Art

Bachelor's thesis 31 pages
May 2021

Blockchain-based tokens are currently used as a method of buying and selling digital art. This thesis investigates use cases for such tokens in combination with physical art.

Three use cases are identified. Tokens could be used in establishing provenance and allow owners to verify their ownership of an artwork. In addition, a system is proposed for refuting as well as proving the authenticity of an artwork.

All identified use cases offer significant advantages and their implementation is not exceedingly difficult technically, though using the tokens will require users to learn some new practices. Also, before being implemented, a more thorough investigation should be conducted on the viability of any blockchain which would host the tokens.

Key words: ethereum, blockchain, token, art

CONTENTS

ABBREVIATIONS

DeFi	decentralized finance
ECDSA	elliptic curve digital signature algorithm
ERC	Ethereum request for comments
EVM	Ethereum virtual machine
IEC	Internation Electrotechnical Commission
IPFS	Interplanetary File System
ISO	International Organization for Standardization
JSON	JavaScript object notation
PBFT	practical byzantine fault tolerance
QR	quick response
URI	uniform resource identifier
URL	uniform resource locator

1 INTRODUCTION

A token in the context of blockchains is a transferable digital asset or a representation of an asset. Tokens are widely used, for example, to create cryptocurrencies. Tokens are also used for selling and buying digital art and assets. In the former use case the token is called fungible and in the latter non-fungible. This thesis explores what use cases there might be for non-fungible tokens in combination with physical art.

In the case of digital art, the token typically contains information about the art or asset. This thesis is based on the idea of creating a similar token for physical art. Such a token would contain relevant information about the artwork and could be transferred along with the art during a sale or a transfer. The token could exist solely on the blockchain, or the artwork could also contain a reference to the token associated with the artwork (using a sticker to achieve this will be discussed in section 4.3.5).

As transactions on a blockchain are verifiable and timestamped and tokens can be securely transferred between owners, the use of tokens in combination with physical art creates some interesting use cases. Section 4.1 will discuss the possibility of using tokens in verifying artwork ownership, section 4.2 will cover using tokens to help establish the provenance of an artwork and section 4.3.2 explores using tokens to help prevent art forgeries.

Section 2 will cover requisite information on topics discussed later and section 3 will give an overview of blockchains in general and the Ethereum blockchain in particular. In order to use specific terminology, the thesis concentrates on the Ethereum blockchain, but the basic ideas are applicable on other blockchains as well. All mentions of blockchains are references to permissionless blockchains.

2 PRELIMINARIES

2.1 Provenance

The provenance of an artwork is its historical record of ownership and transference. Ideally it contains details such as the names of owners, dates of ownership and methods of transference. (Provenance Guide, n.d., 1.)

Provenance is established due to several reasons. Provenance is an integral part of establishing the authenticity of an artwork. Provenance impacts the value of an artwork, as poorly documented provenance can lead to suspicions of the artwork being a forgery. Provenance is also important in showing ownership if the ownership of an artwork is contested. (Provenance Guide, n.d., 2.)

The process of establishing provenance involves things such as seeking information from libraries and exhibition catalogues, reviewing inscriptions and collector's marks from the artworks and interviewing gallery owners, collectors and heirs of owners (Provenance Guide, n.d., 1—3).

2.2 Relevant cryptography

2.2.1 Cryptographic hash function

A cryptographic hash function is a one-way mathematical function which takes an arbitrary, but finite, size input and outputs a fixed size output (also called digest). Such a hash function must be easy to compute, have pre-image and second pre-image resistance and also have collision resistance. (Judmayer, Stifter, Krombholz, & Weippl, 2017, 10—11.)

The requirements stated above mean, that a cryptographic hash function cannot be reversed efficiently and finding two inputs with the same output is very unlikely. When combined with ease of computation, cryptographic hash

functions become a very useful security tool and are extensively used in blockchains.

2.2.2 Asymmetric cryptography

Symmetric cryptography refers to techniques used to encrypt a message with a password. The same password is used to both encrypt and decrypt a message (Bashir, 2020, 77). But transferring such a password over a network, where it can be intercepted, is not safe. A solution to this is asymmetric cryptography.

In asymmetric cryptography, a key pair is generated; a private key and a public key. The public key can be shared, while the private key must be kept secret. The public key can be used to encrypt a message, but the encrypted message can then only be decrypted with the private key. This allows two parties to pass encrypted messages between each other in public, without having to exchange passwords in some non-public way. (Bashir, 2020, 77.)

This process is used in many blockchain solutions to derive the account address. In Ethereum, a random private key is used to derive a public key and the account address is derived from the public key (Bashir, 2020, 320). This has the benefit that an account is very secure, as there is no centralized server which can be compromised, but it does mean that if the private key is lost, access to the account is lost. This threat exists for all the use cases discussed in section 4.

2.2.3 Digital signatures

A digital signature is a way for a receiving party to verify the authenticity of the message. In terms of blockchains, this typically means that the receiving party can verify that the transaction was signed by whomever controls the private key of a particular account.

The main properties of digital signatures are authenticity, unforgeability and non-reusability. Authenticity means that the signature is verifiable, unforgeability means that only the holder of the private key can sign the message and non-reusability means the signature cannot be extracted from one message and used to sign another message. (Bashir, 2020, 115.)

Ethereum uses the elliptic curve digital signature algorithm (ECDSA for short) to sign transactions (Bashir, 2020, 117). Evaluating the security of ECDSA is beyond the scope of this thesis, but it should be pointed out that Ethereum's implementation of ECDSA has potential security problems (Mayer, 2016, 8—9).

2.3 Quick response code

A quick response code (QR code for short) is a 2-dimensional barcode, which has become popular due to its ability to encode considerably more data in a format that is easier to scan than traditional 1-dimensional barcodes (Vidas et al., 2013, 52). QR codes also have error correction and can recover between 7 to 30 percent of the data, depending on the error correction settings (ISO/IEC 18004:2015(E), 2015, 36).

QR codes are categorized into versions according to their module size (module being a reference to a single white or black rectangle in the code image). Version 1 is the smallest version and has a total of 441 modules in it, while the largest version, version 40, has 31,329 modules (Information capacity and versions of the QR code, n.d.).

QR codes are used in this thesis, as they make reading long, random strings much easier. Manually copying such strings would be error-prone and tedious for the user. QR codes used in this thesis are generated by using the libqrencode library[1].

1 https://fukuchi.org/works/qrencode/

2.4 JavaScript object notation

JavaScript Object Notation (JSON for short) is a data format used to exchange data between platforms (Bassett, 2015, 1). JSON allows data to be represented in a manner that is easy to parse programmatically and still be very readable to humans.

In JSON, data is represented as name-value pairs. A name and a value are distinguished by a colon between them, with the name always being on the left and the value on the right. Multiple name-value pairs are separated by a comma and all name-value pairs are finally declared as an object by enclosing them with curly brackets. Names can also be paired with objects, which then contain name-value pairs of their own. An example of this is in FIGURE 1. (Bassett, 2015, 6—8)

```
{
  "name1": "value1",
  "name2": "value2",
  "object1": {
    "name3": "value3"
  }
}
```

FIGURE 1. JSON example.

JSON is referenced in a very limited scope in section 4.3.5. JSON has much more functionality than presented here, but understanding what happens in FIGURE 1 is sufficient for understanding its use in this thesis.

3 BLOCKCHAIN

3.1 History, application and potential

The blockchain as a full-fledged concept was introduced to the world along with the Bitcoin cryptocurrency in 2008 by Satoshi Nakamoto (Bashir, 2020, 12). It is a commonly held belief that Bitcoin and the blockchain concept originated from outside of academia, being solely the brainchild of the possibly pseudonymous Nakamoto, but this is not wholly accurate (Narayanan & Clark, 2017, 1).

For example, the concept of blocks of timestamped documents chained together with hashes, which serves as the basis of the blockchain, came out of the work of Stuart Haber, Scott Stornetta and Dave Bayer. Fault tolerance, which is required by a distributed public system such as Bitcoin, has been studied since the 80's. Leslie Lamport presented an early solution, called Paxos, in 1989 and the PBFT algorithm was introduced in 1999 by Miguel Castro and Barbara Liskov. (Narayanan & Clark, 2017, 4—10.)

There were also attempts to create a digital currency prior to Bitcoin, such as DigiCash by David Chaum in 1990, B-money by Wei Dai in 1998 and Bit Gold by Nick Szabo in 1998 (Judmayer et al., 2017, 16—17). Bitcoin and the blockchain have a decades-long intellectual history and Nakamoto's significant contribution was to take these disparate concepts and put them together in a way that created Bitcoin.

The blockchain has been overshadowed by the success of Bitcoin and other cryptocurrencies. It is often seen only as ledger for cryptocurrency transactions, but even Haber and Stornetta were originally attempting to use their proto-blockchain as a digital notary service (Narayanan & Clark, 2017, 4).

In many ways the blockchain concept is far more important than the cryptocurrency it was used to create. A blockchain is a distributed store of data which can be operated in a trustless environment. The implications of this go far beyond cryptocurrencies. It has already been leveraged to add an execution

layer to the blockchain, which can be used to store and run code on the blockchain (Bashir, 2020, 15). The execution layer in turn has been leveraged to create myriad financial instruments, grouped together under the umbrella term decentralized finance (DeFi for short) and there is an effort to leverage it further to create a decentralized web by creating blockchain-based internet protocols and file storage (Bashir, 2020, 57—59).

3.2 Basic concept of the blockchain

The term blockchain can be confusing, as it can be understood as a reference to the entire ecosystem of protocols which make up a blockchain, as well as a reference to the fundamental way data is organized in a blockchain (Narayanan & Clark, 2017, 20). To avoid this confusion, the underlying data structure of a blockchain is here referred to as a chain of blocks, which is what Satoshi Nakamoto (2009, 7) originally called it.

A chain of blocks at its most basic level is a type of reversed linked list[2] where each node contains a hash of the previous node (Judmayer et al., 2017, 24). This is illustrated in FIGURE 2.

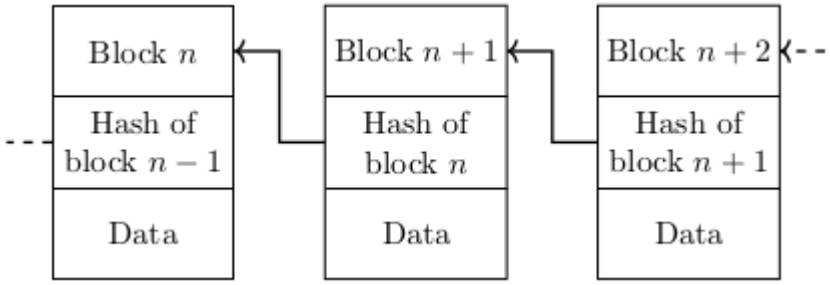

FIGURE 2. A chain of blocks.

The blocks in FIGURE 2 contain the hash of the previous block, thus creating a chain where the blocks preceding any particular block can be verified based on the hash contained in that particular block. Such a chain also has the property that, in order to change one block, the hashes of all successive blocks have to be recalculated as well.

2 A linked list is a data structure where each node contains the address of the next node, until the terminating node.

The data portions of the blocks in FIGURE 2 represent the varied data which can be included in the blocks. This can be protocol-related information or cryptocurrency transactions or code to be run in a virtual machine, among other things. And, as this data is included in the hash, the blockchain effectively becomes a verifiable, sequential database (Bashir, 2020, 13). Once this property is combined with a consensus protocol, we have the basic building blocks of a public transaction ledger.

3.3 Ethereum

3.3.1 Overview

Bitcoin was never intended to be anything more than a cryptocurrency transaction blockchain. Bitcoin does contain a limited scripting language, but most it can do is set conditions for when cryptocurrency coins are spent. (Van Hijfte, 2020, 89).

Ethereum expanded on this concept by allowing arbitrary programs to be run in a virtual machine (Bashir, 2020, 311). Ethereum is described as a ”world computer” and its basic premise is to offer a globally available computer which is nigh impossible to censor or take down. Ethereum can also be seen as a transaction-based state machine, where the state can include any data which can currently be represented by a computer (Woods, 2021, 2). Though perhaps with the caveat, that all programs on Ethereum have to be deterministic, so, for example, generating random data is not possible.

Accounts are the main building blocks of Ethereum and are defined by a pair of private and public keys, as discussed in section 2.2.2. Accounts are updated as transactions are processed, which can be considered the state transitions in the state machine. (Bashir, 2020, 321.)

Ethereum has two types of accounts; externally owned accounts and contract accounts. Externally owned accounts are user accounts and contract accounts

are accounts with executable code and storage. Both types of accounts have an address by which they can be located on the blockchain. The code in contract accounts can be invoked by both externally owned accounts and other contract accounts by sending an appropriate transaction to the contract account's address on the blockchain. (Bashir, 2020, 321—322; Van Hijfte, 2020, 156.)

3.3.2 Smart contract

The term ”smart contract” is widely used, but is somewhat redundant. A smart contract is simply the code that is stored on the Ethereum blockchain and executed in the Ethereum virtual machine (Van Hijfte, 2020, 148).

The term is used in this thesis mostly for brevity. It is sufficient to understand it as the code on the blockchain.

3.3.3 Token

Discussions of blockchains often involve the terms coin and token. A coin is a blockchain's native cryptocurrency, while a token is a digital asset or a cryptocurrency built on top of an existing blockchain (Chittoda, 2019, 189). In other words, a token is a smart contract.

Tokens typically conform to interface standards to help with interoperability. This interoperability allows websites and other services to easily interact with various tokens. Common interface standards on Ethereum are ERC-20 and ERC-721.

Tokens can be roughly divided into fungible and non-fungible tokens. Fungible tokens are interchangeable and indistinguishable from similar tokens. Non-fungible tokens on the other hand are unique and can be distinguished from tokens of the same type. (Van Hijfte, 2020, 28—29.)

The ERC-20 interface describes a smart contract for fungible tokens and is commonly used for building cryptocurrencies on the Ethereum blockchain. The ERC-721 interface describes a smart contract for non-fungible tokens and is used for things such as selling and trading digital assets and collectibles.

Both the ERC-20 and ERC-721 contracts manage several tokens in a single contract instance. That is to say, once a contract is deployed on the Ethereum network, it can be used to create several tokens of the same type. Token details are kept in the contract's storage, which in the case of ERC-20 tokens means details such as how many tokens each account owns, while the ERC-721 contract can store more detailed information about each individual token.

Non-fungible tokens, such as ERC-721 tokens, are commonly used to buy and sell digitals assets. Popular platforms include NBA Top Shot, where users can buy and sell highlights from NBA games, Decentraland, a game which uses tokens to manage in-game land rights and the in-game currency, and platforms such as OpenSea and Rarible, which are marketplaces for digital art (Trade, 2021).

There are also other token standards. One example is the ERC-1155 standard, which is designed to manage several different types of tokens, both fungible and non-fungible as well as semi-fungible (Radomski et al., 2018). Semi-fungible tokens are fungible tokens which can be redeemed, similarly to a coupon.

4 USE CASES

4.1 Ownership verification

Tokens are used to sell digital art and assets, but thus far it remains untested in courts if tokens represent actual legal ownership. This discussion is beyond the scope of this thesis. Though tokens do offer one potentially very powerful property in claiming ownership: a chain of digital signatures ideally starting from the original artist (digital signatures are covered in section 2.2.3).

On Ethereum, each ERC-721 token is created by an account on the blockchain. This creator account signs the transaction which creates the token. When ownership of the token is transferred to a new account, the transaction is signed by the current owner account (Woods, 2021, 4). With a few additional assumptions, such as the signing algorithm being secure and the blockchain not being compromised, this verifiable chain of signatures is difficult to refute.

One issue with this chain of signatures is that Ethereum accounts are just 160-bit values with no intrinsic information about the account owner (Woods, 2021, 24). Without identifying information, the associated signatures are of questionable value. Ideally, the original artist or seller should publicize their account address in some manner.

Ownership of the token could also be seen as compelling evidence for ownership of the artwork, even without considering the chain of signatures. Assuming a token for the artwork was created by a reputable source, such as the original artist or a gallery, having control of the token in combination with the artwork would bolster a claim of ownership.

4.2 Help in establishing provenance

An Ethereum node can contain the entire blockchain starting from the first block (called a genesis block) and the current state is retrieved and verified by

processing the entire blockchain starting from the genesis block (Bashir, 2020, 351). As a token is transferable, ownership can be tracked over time based on the historic data in the blockchain. With some caveats, this data could be used to help establish provenance all the way to the origin of the token. As discussed in section 2.1, more work goes into establishing provenance than a token can hope to fully replace, but it could be a valuable investigative tool. For example, even without identity information attached to accounts which have owned the token, a researcher could still see that a transfer occurred. In situations where records aren't necessarily produced, such as a sale between two private individuals, the blockchain would reveal the transfer.

Using the token for establishing provenance requires that the token is transferred at each sale to the new owner. Furthermore, should an owner lose his private key, the token cannot be transferred to a new owner. For this scheme to work, all buyers of the artwork should be aware of the token and require it to be transferred with the artwork and take adequate steps to secure their private key.

This scheme also suffers from the same problem with semi-anonymous accounts which was discussed in section 4.1. This issue could be alleviated by galleries and auction houses openly publishing their account addresses, thus creating points of reference for provenance researchers. Auction records are already a valuable tools for provenance researchers (Provenance Guide, n.d., 8—9).

4.3 Refutation system

4.3.1 Difficulty of proving authenticity

One way of attempting to prove the authenticity of an artwork would be to embed information in or on the artwork, which can be compared to information on the token. But any easily retrievable information on the artwork can be retrieved and copied by anyone with access to the artwork. Such a system

would rely on security by obscurity and thus could at best only be considered weak evidence of authenticity.

Information could be hidden, but unless it becomes irretrievable, the system would still rely on security by obscurity. And obviously irretrievable information defeats the purpose of having the information on the artwork.

4.3.2 Refuting authenticity

A system for refuting authenticity appears to be viable. If a hash digest is included in the token associated with the artwork and the input message is attached to the artwork itself, then a mismatch in the digest retrieved from the token and the digest generated from the input message on the artwork could be argued to be strong evidence for the artwork not being the original artwork.

This process is illustrated in the flowchart in FIGURE 3. For the sake of simplicity, we'll assume the input message is encoded into a QR code (which is discussed in section 2.3) and attached to the artwork with a sticker.

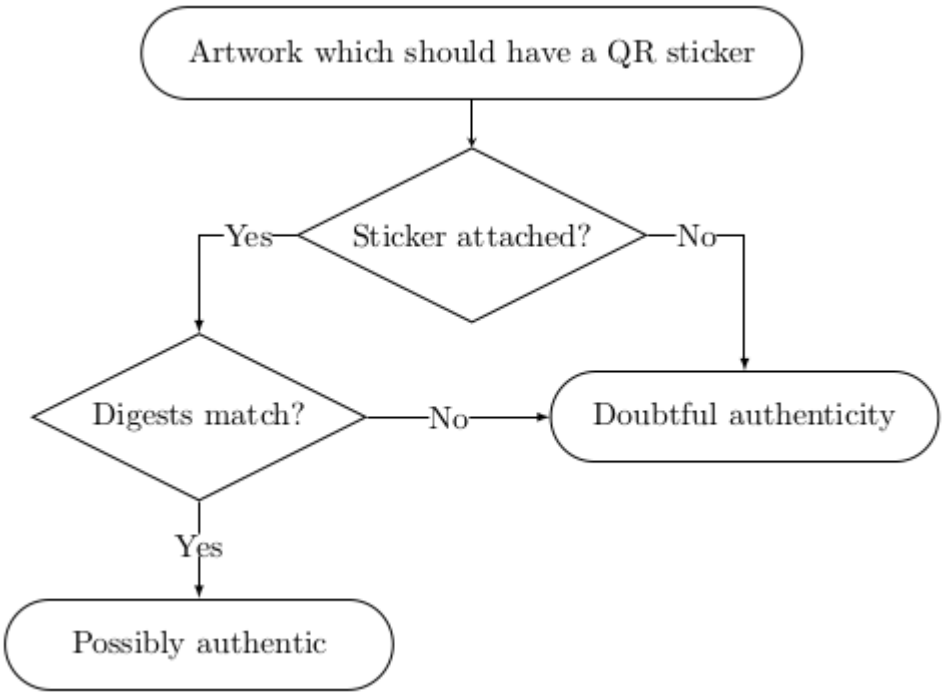

FIGURE 3. Refutation system flowchart.

The system in FIGURE 3 has at least two issues. First, anyone inspecting the artwork would have to know that the artwork is supposed to have a QR sticker attached. Second, someone might have created a new token and a new sticker to go along with it, leading to a mismatch.

The first issue can be solved by attaching metadata to the token, such as the artist's name, name of the artwork, etc. This way tokens become searchable and existing platforms, such as OpenSea, can be used for creating and searching for tokens. Though there are potential issues with metadata and ERC-721 tokens, which are covered in section 4.3.8.

The second issue is a bit more complicated, as there are legitimate reasons for creating a new token and sticker, such as the old sticker falling off or becoming damaged and the input message being lost. To give credibility to any new token, it would have to be created by a very trustworthy source, such as a reputable art broker. As discussed in section 4.2, blockchain address owners are typically anonymous, but owners can publicize their ownership of an address and thus can attach their credibility to any token they create. Similarly, if the token is transferred from owner to owner as described in section 4.2, one can inspect the blockchain to see if the old and new tokens have overlapped. If the old token has been transferred when the new token has already been created, that's a good indication that the new token might be fraudulent.

4.3.3 Input message processing

To create the system described in section 4.3.2, one has to first generate a random input message. This input message is then hashed to create a digest and the digest is included in the token. The input message is also included as such in the QR sticker, which is attached to the artwork in a location that is not easily accessible, such as the back of the canvas of a painting. This process is presented in the simple flowchart in FIGURE 4.

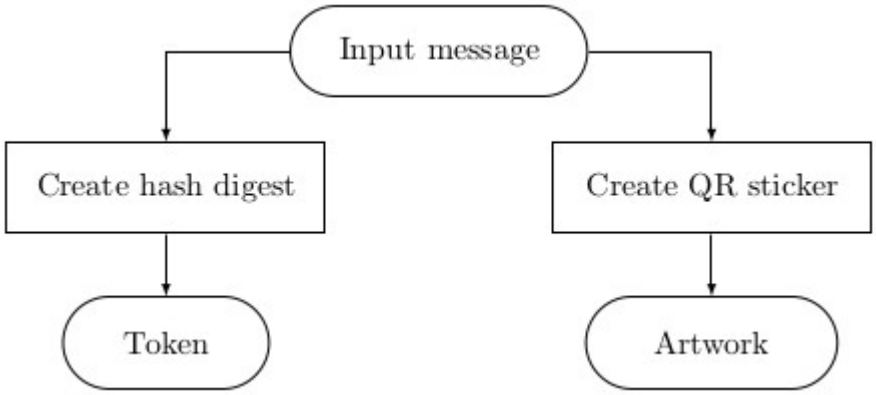

FIGURE 4. Input message processing.

How the digest is stored on the token involves details which need to be taken into consideration. To save on costs, metadata for a token might be stored off-chain, which is not ideal. This issue with regards to the ERC-721 standard is discussed in section 4.3.8.

4.3.4 Digest comparison procedure

To compare the digest in the token and the input message in the QR sticker attached to the artwork, the input message is read from the QR sticker and it is hashed with the appropriate hash function to produce a digest and this digest is compared to the digest found in the token. This procedure is represented in FIGURE 5

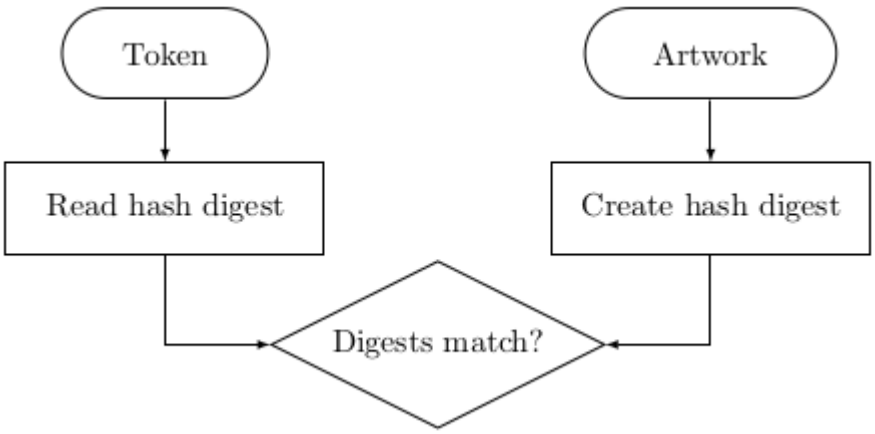

FIGURE 5. Digest comparison procedure.

4.3.5 QR sticker example

The minimum information the QR sticker should contain is the input message, the hashing function used to create the digest and all relevant location information on the token, such as the blockchain where it's located, the contract address and token ID. Additional information, such as who added the token and when, are possibly beneficial as well.

An example QR sticker will be made based on the token for the physical painting 'White Skull' by Javier Martinez[3]. The token is technically based on the ERC-1155 standard, but, for the purposes of addressing, it is similar to ERC-721. The main difference is the terminology; a token ID in ERC-721 is referenced as 'tokenId', while in ERC-1155 is is ' id' (Entriken, Shirley, Evans, & Sachs, 2018; Radomski et al., 2018).

The token is located on the Ethereum blockchain, the token's contract address is 0x495f947276749Ce646f68AC8c248420045cb7b5e and the token's ID is 501802790394530593530600493615820354532664795583132496752950782 52488886845441. We'll choose ZYpzjCQC47aylI6fgzpZZlKc253wWTe1MWR-3kKj as the random input message, which will produce the digest 60d9-aa3ad6df27c3a4dd8822e3e4746a7ac151bd4917a1cf7c94a6de6c43cfbe when hashed with the SHA-256 hashing algorithm. This digest would be included with the token, while the rest of the information will be encoded into QR codes. An example of a QR sticker with this information can be found in FIGURE 6.

3 Token on OpenSea platform at https://bit.ly/3bGKYuN. URL shortener used for convenience.

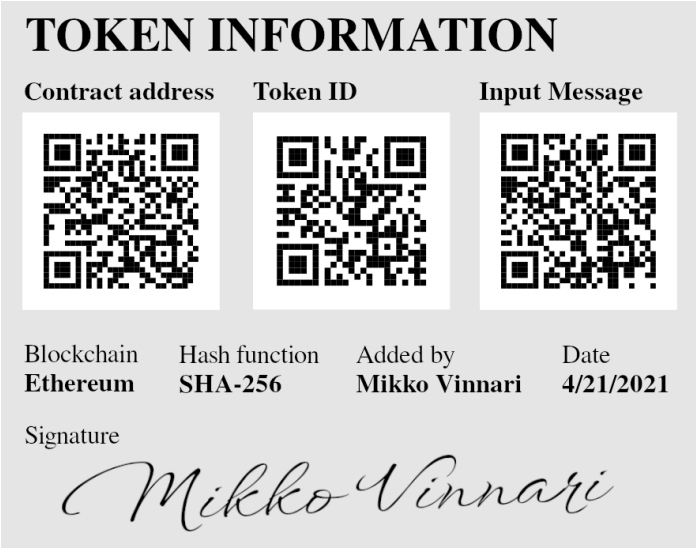

FIGURE 6. An example of a QR sticker with token information.

The QR codes are separated in FIGURE 6, but the information could also be joined into a single QR code. A downside of this would be that the information read from the QR code would need parsing, either manually or in some automated way. One potential way for joining the QR codes is looked at in section 4.3.6.

The contract address and token ID in FIGURE 6 could also be joined into one QR code by using an OpenSea URL, as the URL contains both the contract address and token ID in the URL. In this case, even if the OpenSea platform shuts down some day, the contract address and Token ID are still recoverable from the URL.

4.3.6 JSON standardization

The QR sticker in section 4.3.5 could be simplified by standardizing the data in some fashion. This section contains an example of doing so with JSON data, which can be easily handled by web browsers and mobile apps once read from the QR code (JSON is briefly covered in section 2.4). FIGURE 7 contains information about the artwork and token used in section 4.3.5 (contract address and token ID have been shortened to display them correctly).

```
{
  "artwork": {
    "artist": "Javier Martinez",
    "name": "White Skull"
  },
  "token": {
    "token standard": "erc-1155",
    "blockchain": "ethereum",
    "contract address": "0x495f...7b5e",
    "token id": "5018...5441",
    "input message": "ZYpzjCQC47aylI6fgzpZZlKc253wWTe1MWR3kKj",
    "hash function": "SHA-256",
    "added by": "Mikko Vinnari"
  }
}
```


FIGURE 7. Artwork and token information as JSON data.

The data in FIGURE 7 is minimized and encoded into a QR code. An example of what a sticker might look like with this QR code is presented in FIGURE 8.

FIGURE 8. QR sticker with information condensed into one QR code.

An issue with this approach is that it will require significantly more information to be encoded in a single QR code, as evident in FIGURE 8. A QR code will theoretically hold around 3 kilobytes of data at the lowest error correction setting and, in byte mode, ISO/IEC 8859-1 character encoding can be chosen, which means that technically a QR code could hold around 3000 characters (Focardi, Luccio, & Wahsheh, 2019, 8; ISO/IEC 18004:2015(E), 2015, 21). In practice this is too optimistic, as the sticker would have to survive decades or more, so

choosing the highest possible error correction seems prudent. This will lower available data space.

A further issue with increased data in a QR code is that scans of the QR code will fail more often. One study found that encoding more than 1200 bytes of data with medium error correction will result in read failures half of the time or more (Focardi et al., 2019, 4—5). The study refers to data amounts prior to encoding, so it doesn't include error correction data, but we can roughly approximate that 1200 bytes at medium error correction results in a version 28 QR code (Information capacity and versions of the QR code, n.d.). Assuming we use version 28 with high error correction, we would be left with 658 bytes of space to use in byte mode (Information capacity and versions of the QR code, n.d.). With ISO/IEC 8859-1 encoding, this would mean the QR code can host 658 characters, so it appears feasible to include token information in JSON format into a QR code without sacrificing readability or error correction.

The approach of using JSON or a similar encoding scheme would require a website or an application for processing the information. Though creating the token and the sticker would require some technical expertise to begin with, so offering a service for creating them would be necessary in any case from an ease-of-use perspective.

4.3.7 Potential way to prove authenticity

It would be possible to divide the input message into two and attach one half to the artwork and the other half to any sales or provenance documentation which may accompany the artwork. In this case, generating the digest would require both the artwork and the documentation, thus leading to forgeries being very unlikely.

This method would increase the likelihood of the system failing; the token could be lost or not transferred and the documentation could be lost. But it would provide a robust way to verify the authenticity of both artwork and associated documentation.

4.3.8 Notes on using ERC-721

The base ERC-721 standard interface is fairly limited and only includes functionality regarding ownership and transfer of tokens. The base interface can be extended with the ERC721Metadata extention interface, which makes it possible to store a uniform resource identifier (abbreviated URI) for each individual token's metadata (Entriken et al., 2018). An issue with this approach is that the metadata is stored somewhere else than the blockchain and the security of the off-chain storage has to be ensured. If the metadata is stored on a single server, the security and the longevity of the metadata are questionable.

One way to store metadata securely off-chain is to use the Interplanetary File System (abbreviated IPFS). IPFS is a peer-to-peer protocol for storing data in a decentralized manner (Van Hijfte, 2020, 177). ERC-721 metadata is recommended to be stored in a ERC721 Metadata JSON schema and details about the artwork, the digest and other relevant information can be added to this schema (Entriken et al., 2018). As this schema is the standard way to store metadata for tokens, many platforms dedicated to trading tokens will display the information correctly. For example, the OpenSea platform supports this schema (NFT Metadata Standard , n.d.).

There are at least two concerns with using off-chain storage. First, if the off-chain storage fails in some way, then the digest is lost along with other metadata. Such a situation can be remedied by making the URI mutable (Entriken et al., 2018). The EIP-721 documentation does not mention an extention for making the URI mutable, but the OpenZeppelin ERC-721 documentation does include the ERC721URIStorage extension interface with an internal _setTokenURI function for setting the URI (Entriken et al., 2018; ERC 721 , n.d.). As the function is internal and thus not accessible from outside the contract, it would seem to indicate there is no platform-agnostic way to change the URI after the creation of the token.

The second concern with off-chain metadata storage is that token creation platforms aren’t necessarily open about how they store metadata. It might be difficult for a non-technical user to determine how the metadata is stored. There are also concerns with the IPFS protocol with regards to longevity and pricing, but these go beyond the scope of this thesis. Storing at least the digest on-chain would seem to be the most prudent choice, although this will require creating a custom smart contract and the digest will not be visible on existing token platforms without modifications to the platforms.

4.3.9 Security considerations

The utility of the refutation system hinges on the input message remaining a secret. To this end, it would be pivotal that the input message never leaves any device which hashes the input message. Transmitting the input message over a network would risk it being intercepted or the receiving party leaking it intentionally or unintentionally. This is something users of the system would have to recognize and make sure the tools they use are secure. If someone wanted to gather input messages from numerous artworks, one way to do it would be to create an application dedicated to reading the QR codes and secretly send the input messages to an external server.

Measures to obfuscate the input message on the artwork could also be considered. For example, if a QR sticker is used, it could be covered up in some manner. This wouldn’t make it impossible for someone to read the input message, but it would make it slightly more difficult, as it couldn’t be read with just a quick scan of the QR code.

5 CONCLUSIONS

Three use cases were identified for tokens and physical art. Owner verification and provenance are inherent traits of tokens and would provide value when used in combination with physical art. A refutation system was proposed, which is a more novel idea and would give art buyers confidence in the authenticity of the artwork they are buying. An augmentation to the refutation system was also briefly mentioned, which could be used to practically prove the authenticity of an artwork.

With the full aid of the use cases mentioned here, forgers would require ownership of the token associated with the artwork and access to the artwork and its documentation in addition to the artistic skills required to create a forgery. While not completely impossible, forging an artwork under suck conditions would be exceptionally difficult.

The valuation of art is very dependent on the authenticity of the art and using tokens as evidence of authenticity would bring financial value to owners of art. The decreased likelihood of forgeries would also give buyers confidence in the authenticity of the art they are buying and improve the image of the art world in general.

All use cases mentioned here are relatively simple to implement. Creating a token can be done with existing platforms, but the refutation system would require a new platform or for an existing platform to offer it as a service. As tokens are already used for digital art, the threshold for adopting them for physical art might be lowered.

The use cases suffer from some potential issues which are related to the way blockchains function. As decentralized systems, there is no central authority on a blockchain to fix user errors. For example, if a user forgets an account’s password, the token associated with the account is lost. It is unknown to what degree this would be a problem.

This thesis concentrated solely on potential use cases and ignored the viability of blockchains for long-term storage of tokens. This is something which should be considered before any implementation of the use cases. Proposed criteria for viability would be accessibility, affordability, security, data longevity and environmental effects. Accessibility and affordability would be crucial for making sure the tokens are created as early as possible in the lifespan of the artwork, ideally by the original artist. Security and data longevity determine whether the blockchain itself will stand the test of time. Environmental effects should be considered, as negative environmental effects could lead to usage of the blockchain drastically decreasing or the blockchain being abandoned altogether.

Despite the obstacles involved, tokens have the potential to drastically change the art market. With a fairly minor effort in learning to use tokens, buyers of art could be confident that they are buying an authentic artwork, sellers would receive better prices due to the increased confidence and owners could use their control of the token as evidence of their ownership of the artwork itself.

REFERENCES

Bashir, I. (2020). Mastering blockchain (Third ed.). Birmingham: Packt Publishing Ltd.

Bassett, L. (2015). Introduction to javascript object notation. Sebastopol: O'Reilly Media, Inc.

Bez, M., Fornari, G., & Vardanega, T. (2019). The scalability challenge of ethereum: An initial quantitative analysis. In 2019 ieee international conference on service-oriented system engineering (sose) (p. 167-176). IEEE.

Chittoda, J. (2019). Mastering blockchain programming with solidity. Birmingham: Packt Publishing Ltd.

Entriken, W., Shirley, D., Evans, J., & Sachs, N. (2018). Eip-721: Erc-721 non-fungible token standard. Ethereum Improvement Proposals. Retrieved May 19th 2021, from https://eips.ethereum.org/EIPS/eip-721

Erc 721. (n.d.). OpenZeppelin. Retrieved May 20th 2021, from https://docs.openzeppelin.com/contracts/4.x/api/token/erc721

Focardi, R., Luccio, F. L., & Wahsheh, H. A. (2019). Usable security for qr code. Journal of information security and applications, 48 , 102369-.

Information capacity and versions of the qr code. (n.d.). Denso Wave Incorporated. Retrieved May 22nd 2021, from https://www.qrcode.com/en/about/version.html

Iso/iec 18004:2015(e). (2015). International Organization for Standardization.

Judmayer, A., Stifter, N., Krombholz, K., & Weippl, E. (2017). Blocks and chains: introduction to bitcoin, cryptocurrencies, and their consensus mechanisms. Synthesis Lectures on Information Security, Privacy, & Trust, 9 (1), 1–123.

Mayer, H. (2016). Ecdsa security in bitcoin and ethereum: a research survey.

Monte, G., Pennino, D., & Pizzonia, M. (2020). Scaling blockchains without giving up decentralization and security: a solution to the blockchain scalability trilemma. In Proceedings of the 3rd workshop on cryptocurrencies and blockchains for distributed systems (p. 71-76). ACM.

Narayanan, A., & Clark, J. (2017). Bitcoin's academic pedigree: The concept of cryptocurrencies is built from forgotten ideas in research literature. Queue, 15 (4).

Nft metadata standard. (n.d.). OpenSea. Retrieved May 19th 2021, from https://docs.opensea.io/docs/metadata-standards

Provenance guide. (n.d.). International Foundation for Art Research. Retrieved April 9th 2021, from https://www.ifar.org/Provenance Guide.pdf

Radomski, W., Cooke, A., Castonguay, P., Therien, J., Binet, E., & Sandford, R. (2018). Eip-1155: Erc-1155 multi token standard. Ethereum Improvement Proposals. Retrieved May 20th 2021, from https://eips.ethereum.org/EIPS/eip-1155

Thomson, G. (2020). Ethereum 2.0 will walk and 'roll' for two years before it can run. Decrypt. Retrieved May 27th 2021, from https://decrypt.co/34204/ethereum-2-0-will-walk-and-roll-for-two-years-before-it-can-run

Trade, C. (2021). Top 5 nft crypto marketplaces for 2021. Publish0x . Retrieved May 20th 2021, from https://www.publish0x.com/christrade/top-5-nft-crypto-marketplaces-for-2021-xompnyp

Van Hijfte, S. (2020). Blockchain platforms: A look at the underbelly of distributed platforms. Synthesis Lectures on Information Concepts, Retrieval, and Services, 8 (1), i–239.

Vidas, T., Owusu, E., Wang, S., Zeng, C., Cranor, L. F., & Christin, N. (2013). Qrishing: The susceptibility of smartphone users to qr code phishing attacks. In A. A. Adams, M. Brenner, & M. Smith (Eds.), Financial cryptography and data security (pp. 52–69). Berlin, Heidelberg: Springer Berlin Heidelberg.

Woods, G. (2021). Ethereum: A secure decentralised generalised transaction ledger. Retrieved April 19th 2021, from https://ethereum.github.io/yellowpaper/paper.pdf