SmartSign

Signable Non-fungible Tokens
(based on TON)

White paper

by Serge P.,
@sergey_msu, spmathf@gmail.com

Created: 2022.04.01
Version #2.0: 2022.08.17

Contents

Disclaimer

The information in this paper specifies the general architecture of signing non-fungible tokens based on the TON blockchain – SNFT – as indicated below. This paper is intended to describe the general concept of signing digital assets on the TON blockchain, as well as the implementation of the described architecture and its benefits. The paper shall not be considered as a guarantee or commitment regarding the future availability of services associated with its use or the future value of non-fungible tokens signed in this way. The paper is not an offer for the sale of shares, tokens, other securities, or digital assets. The paper is not a purchase recommendation or financial advice and is created for educational purposes only.

Graphs, diagrams, and other visuals are intended for educational purposes and describe the basic idea behind the process of signing non-fungible tokens based on the TON blockchain. None of these graphs, charts, and visuals per se shall be used to make investment decisions.

The authors shall not be responsible for the use of this technology to the detriment of anyone. The proposed architecture per se is not intended to protect a digital asset owner from malicious actions of third parties if the owner loses his/her private keys from his/her wallets where his/her purchased SNFT tokens are registered.

Abbreviated terms

TON	- The Open Network blockchain, see ton.org.
SC, SMC	- Smart contract on the TON blockchain
NFT	- Non-fungible token.
NFTaaS	- NFT-as-a-Service concept.
Obj	- An object to sign: NFT or NFT collection.
Sign	- Signature. A special smart contract linking an object and a signatory.
Prv	- Signature Provider. A special smart contract that ensures the integrity of signature opening/cancellation requests and serves to index signatures in the blockchain.

Summary

The concept of signing tokens is a new paradigm that allows, among other things, to validate digital property and confirm ownership of both digital and real assets. The scope of potential applications of this idea is impressive: from usual autographs on collectibles to verification of ownership of real things, types of real estate, etc. Digital property such as NFT, after signing, can acquire additional collection value and increase its investment attractiveness.

For the time being, the so-called electronic digital signatures are widely used. They allow to confirm ownership but require the storage and protection of a special key physically placed at best on an external medium, and at worst – centrally on an e-signature service provider website [EsW, EsA, EsT]. In case of loss of an electronic key or cyberattack on a provider (which has happened many times), the owner of the signature loses control over it, therefore he/she will have to block it, reissue it and, in the worst case, deal with the consequences of misuse of the signature by third parties.

With blockchain, the situation has somewhat improved since decentralization is the core idea of blockchain technology. Encryption of external requests to user wallets is built into most blockchains by default. Therefore, there is no need to constantly have an external medium with a secret key on hand or use a third-party service to store it [Dsb, hsB, esB].

In the TON blockchain, each non-fungible token is a separate smart contract in the network, which is also rather convenient for the correct implementation of the mechanism for signing such objects [NsT].

The main idea we propose for signing NFTs in the TON blockchain is to implement the signature as a separate smart contract, which serves as a kind of handshake between the owner of the token and its signatory. In one of the possible schemes of implementing this idea, the NFT owner deploys a signature smart contract and indicates a person whose signature he/she expects (push-scheme). Upon the signatory confirmation, the signature is considered successful. In another implementation, the signatory can request to sign a given NFT, and the owner can confirm this request (pull-scheme). Both schemes allow further variations and generally cover almost all applied needs for signing non-fungible tokens on the blockchain.

This paper discusses the very idea of signing NFTs on the TON blockchain. Details of possible implementations of various signature architectures, and their strengths and weaknesses are also included herein. The paper is intended both to give a general perception of the capabilities for tokens signing in the TON blockchain and to introduce a reader to the technical details of the project, the project team, and further plans.

1. NTF as a New Type of Property

NFT means non-fungible token. Unlike ordinary tokens-coins in the blockchain, NFTs are unique, they cannot be copied, but they can change their owner [NFT]. Each NFT object exists in a single copy and has a fixed address in the blockchain where you can get all the information about it. From a technical point of view, NFT serves as a kind of abstraction-container put over a more real object like an image, a file, etc. Thus, an ordinary file that can be unlimitedly locally copied is placed in the blockchain at a unique address and becomes unique in a sense. For the time being, the concept of non-fungible tokens has firmly entered the blockchain paradigm and serves as an integral part of the ecosystem of almost any blockchain. The TON blockchain is no different although the NFT token standard has a short history here so far [NsT].

NFT tokens can be created inside the blockchain, can be changed, sold, and in fact, they are in demand. Thus, NFT objects are de facto items of property of a new type, they are purely digital and stored in a distributed decentralized database – blockchain. This type of digital property is quite specific and gives a number of advantages and disadvantages to its owners.

Technically, an NFT token can be easily created by any user. In many blockchains (recently in the TON blockchain), there are special marketplace services that make it easy to create your own NFT and immediately put it up for sale. The price of the first NFTs created in this way reached several million dollars. But now it is almost impossible to sell them at that price. The price of an NFT, even if it is quite high up on NTF creation, usually drops significantly after a while, and there are reasons for this.

The main reason that objects like NFTs have some value is the paradigm of pseudo-uniqueness described above, namely, the uniqueness in terms of blockchain address, but perhaps not in terms of content. It was the uniqueness that historically fueled interest in the NFT market on various blockchains. It would not be an exaggeration to say that many people perceive and keep perceiving NFTs as an exotic toys and purely speculative tools. It is unlikely that anyone purchased an NFT intending to keep it forever. Rather, this type of asset, until recently, was viewed as to be sold as soon as its price slightly increases. For those reasons, it is not surprising that NFTs were and continue to be considered a rather risky asset. The reasons for the recent crisis in the NFT market, which began in 2021 and continues to this day (summer 2022 at the time of writing this paper), are also quite clear. Investors and ordinary buyers tended to be disillusioned with the investment potential of NFTs due to the impossibility to evaluate it even approximately, its strong volatility, and dubious value in general.

However, not that long ago, the situation around NFT began to change in a positive direction. All the mentioned disadvantages of a naive understanding of NFT do not cancel the fact that NFT objects are real digital property, which can provoke not only speculative interest but also bring real benefits to its owner. Today, it is already clear to many people that the previous perception of NFTs as a usual image in the blockchain does not bring any benefit, and just the fact of owning such an asset does not give anything to its owner. Gradually, there comes the realization that NFT can and should be used profitably and develop not only the NFT market itself but first of all the blockchain infrastructure as such by searching for more and more new applications of the non-fungible token concept.

Below are the most common examples of NFT applications outside of pure speculation for the time being [Naf, Nbm, Ntt, Nmm]:

- confirmation of ownership of other digital or real assets (music, art, real estate, etc.);
- various events tickets;
- metaverse objects, or parts of compound objects;
- certificates;
- loyalty points or rewards for any actions;
- decentralized voting, in particular in DAO;

Recently, there has been a clearer trend towards linking NFT with the world of real objects rather than virtual ones.

For example, NFTs are increasingly being used as proof of ownership of real estate.

2. NFT-as-a-Service

As previously stated, the scope of NFT as an object of the digital property has recently expanded significantly compared to the purely speculative trends of earlier times. The main areas of application of NFT in the modern world are art, music, the gaming industry, and recently, material objects of the real world [Nbm, Ntt, Nmm]. Due to the growing number of applications, the most massive, bright, and useful of them are usually united by the term NFTaaS – NFT as a service or service [NaS, Naa, Naf]. This emphasizes the mass character and importance of this or that NFT application.

In our project of signable NFTs – SNFT – the idea of additional verification of digital tokens plays a key role. Let's review the most striking examples of the NFTaaS concept where this kind of verification could be most useful. Omitting the details for now, let's assume that NFT as an object in the blockchain can have a signature (one or many) – some additional entity that serves as a kind of visa of someone stamped on this object. What can it give to business and people?

In almost any of the above areas of NFT applicability, additional verification with a signature at least would not harm. So, for example, if NFT is used as a ticket or pass for some event, then access to different zones can be organized by issuing different NFTs. However, it is more natural to do this by putting various visas on NFT invitations, which guarantee access, for example, to the VIP zone. Thus, signable NFTs turn out to be a more convenient and natural tool that can be presented as a third-party service and thus fully complies with the NFTaaS concept.

No less impressive are the modern applications of NFT in the gaming industry, in particular, in the metaverses. It has now become almost the standard to include NFT objects in new games or to build the whole game entirely around a blockchain. In metaverses, almost every game object is an NFT token. Thus, NFTs become natural and integral parts of virtual universes, with which in-game services must be able to work correctly. Here, the NFTaaS concept is central, although it deals with objects of not a real world, but a game one. The additional idea of signing NFT objects would also play a significant role here, for example, adding additional rarity to the objects of the game world or collecting visas of top gamers on the legendary objects of the game, therefore making them truly unique.

There are a lot of examples of using NFT as a service and more will come up in the coming years. Global digitalization and decentralization will undoubtedly deeper penetrate all areas of our life, and NFTs will take a special place in it. Gradually, the most useful and socially significant applications of this kind will be presented as services following the general concept of NFTaaS [Nap], and the signable NFT service – SNFT will surely take its place in it.

3. Digital Signature Conception

Electronic digital signatures serve the same purpose as physical signatures, namely, to verify the identity of a person who signed a certain document [EsW]. Digital signatures work based on cryptographic methods that exclude forgery by direct copying [Dsb, esB].

There are several approaches to EDS implementation. One of the standard and closest to the blockchain concept is an approach of asymmetric encryption – the use of a pair of keys, a private and a public [EsW, Dsb, esB]. A private or closed key is created by its owner and must be kept confidential. The owner openly shares the public key with everyone who needs to confirm his/her authorship on any documents, messages, etc. The ability to decrypt a document/message with a public key automatically confirms authorship.

Technically, the process of signing a document consists in obtaining a cipher from it using a private key on the side of the key owner. This cipher is deemed an electronic signature. Further, the document and the cipher shall be sent to the addressee, who has the public key of the signatory and can easily make sure that it was the owner who signed the document. It is a purely mathematical reason for having all this worked, namely, knowing the public key, it is impossible to compute the private key in a reasonable amount of time, even with the most powerful computers. At the same time, the encryption methods themselves are known, open, and sufficiently reliable [esB].

The signing process implemented in this way has many advantages and weaknesses. Being almost ideal from a mathematical point of view, the process of asymmetric encryption can be significantly affected by the human factor or imperfectly implemented by third-party e-signing services. The loss of the private key by its owner means that an attacker can sign any document on the owner's behalf. Moreover, some online digital signature service providers generate and store their users' private keys on their side creating unnecessary centralization, which poses great risks to their customers.

4. Message Signing in the TON Blockchain

The concept of the electronic signing of digital documents described above plays a key role in the technical implementation of any blockchain [Dsb, hsB]. Before describing the NFT signing process, it is necessary to understand the message encrypting process in the blockchain and why a signature can be reliably associated with the individual who put it.

In the TON blockchain, each NFT object, as well as each user wallet, is a separate smart contract [NsT]. Moreover, each smart contract allows coin acceptance. In this sense, NFT as an entity in the blockchain and a usual user wallet are very similar. The main difference between a user wallet and any other smart contract lies in their operation specifics – it is possible to withdraw coins from a user wallet, and transfer them to other wallets and smart contracts, but generally speaking, it is impossible to withdraw funds from an arbitrary smart contract.

As a matter of fact, in the TON blockchain there are three types of messages that smart contracts can process: external, internal, and get-methods [Tsm]. The latter usually serves to obtain information about the state of the smart contract and does not change it in any way. Internal messages between smart contracts are initiated by the blockchain, have a valid sender and some service information. Thus, the sender of the internal message does not have to additionally identify himself – the blockchain guarantees that his/her address is reliable and can't be forged. External messages come to the smart contract from outside the blockchain. It is messages of this type that are used in wallets for transactions to other smart contracts, and verification of their sender is a must. It's usually processed as below:

```
() recv_external(slice in_msg) impure {
    var signature = in_msg~load_bits(512);
    var ds = begin_parse(get_data());
    var public_key = ds~load_uint(256);
    ds.end_parse();
    throw_unless(34,
                 check_signature(slice_hash(in_msg),
                 signature,
                 public_key));
    ;; business logic
}
```

Here, a signature is obtained from the outside message, then from the state of the smart contract ds the known public key of its owner is read, and then, using the check_signature method, it is checked that the message is signed by the owner's private key. If this condition is not met, the contract request fails.

This scheme is similar to the scheme of the electronic signing of documents described above. Indeed, the principle underlying both actions – electronic digital signature and external requests to TON smart contracts – is the same. When creating a TON wallet, a pair of keys is generated – a private and a public. The private key is uniquely determined by a seed phrase – 24 words that must be remembered or stored in a safe place and not disclosed to anyone. If the seed phrase is not lost, only the wallet owner can send transaction requests to it.

It means a lot. Firstly, the blockchain itself acts as a provider of electronic signing and verification of transactions and does not require the storage of private wallet keys on its side. Secondly, a user does not need to store the private wallet key separately on an external medium and constantly use it for each transaction. The wallet itself is uniquely associated with its owner and managed by him/her through official wallet applications.

Thus, the TON blockchain itself initially provides all the necessary infrastructure for transaction validation and, by default, protects a user as much as possible from the illegal actions of third parties. The risk of losing the seed phrase, of course, remains, but nothing can be done about it at the blockchain level. Both in the classical scheme of signing electronic documents and in any blockchain, it is the user who shall be responsible for the safe storage of the private keys of his/her wallets.

5. Token Signing in the TON Blockchain

Now let's review the problem of signing non-fungible tokens in the TON blockchain. The TON NFT standard itself does not imply such functionality by default, it should be implemented separately. The proposed solution shall successfully solve many problems and meet some mandatory requirements. This paper discusses in detail some implementation methods leading to two types of NFT signing architectures, conditionally pull- and push-approaches.

5.1 Signing with a transaction

The mechanism of signing external messages in the TON blockchain described above can be considered as the simplest option for signing any smart contract. Simply put, we can try to consider as a signature of some person a simple transaction from his/her wallet to a sign-to-be object (for example, NFT). The only advantage of this approach is its simplicity. Unfortunately, there are far more disadvantages:

- Possible spamming of the object with microtransactions makes detection of real conscious signatures rather difficult.
- Smart contract phishing. A wallet owner can be tricked into sending a transaction to some object which is being "signed" without the knowledge of the signatory.
- The complexity of indexing signatures in the blockchain for their subsequent validation.
- The signature may contain additional information about the object and the signatory that cannot be included in a normal transaction (graphical representation, the ID of a linked social network account, etc.).

Thus, simple transactions, even if they are provided with special comments, cannot serve as a sufficiently reasonable mechanism for signing NFT objects.

5.2 General architecture of token signing

Analyzing the above disadvantages of signing objects with a simple transaction, we can conclude that the conceptually correct solution would be the storage of each signature in a separate smart contract. This will allow any additional information to be included in it and also fully comply with the general concept of smart contracts in the TON network, namely, any object is a smart contract.

It is worth adding one more smart-contract – a provider that functions as a guarantor of deployment of the very expected verified smart contracts of signatures, and functions as a convenient indexer of signatures in the blockchain for their subsequent validation.

So, the general scheme of signing objects in the TON blockchain should include the following actors:

1. The owner of the NFT object.
2. The NFT object itself.

3. A signature is opened for a given object and a signatory.
4. The signatory whose signature is expected on the object.
5. The provider that controls the issuance of signatures.

5.3 Decentralization and trust

With the five object types described above, it is quite possible to build a signing architecture in several qualitatively different ways. This approach, however, has a number of disadvantages. For example, the signatory has to trust the signature smart contract and the provider has to trust some entity responsible for issuing them. However, if you allow an object to issuing signature contracts on its own, such contracts will not be standardized. In this case, the object may include in them any information about the signatory, and the latter will not always take the trouble to figure out what kind of smart contract is presented to him/her for signing and whether it can be trusted.

Therefore, it seems more natural to standardize signature smart contracts at least at the level of a signature provider. Of course, there can be many providers, but even in this case, their validity will be much easier to check than the validity of each individual signature of each new object. The provider also serves as a convenient entity for collecting all necessary fees for all signing objects operations.

Another advantage of having a provider is the natural indexing of requests to open, cancel, and erase signatures.

6. Solution requirements

From the above discussion, it is clear that transactions by themselves, despite their cryptographic security, cannot serve as a justified and reliable solution for signing objects on the blockchain. Additional smart contracts, on the one hand, give more flexibility and allow to implement any requirements and business logic. On the other hand, additional objects in a solution architecture raise additional requirements on its security and stability.

The main requirements that an object signing architecture shall meet are as follows:

R1. Providing the functionality of signature opening and cancellation at the request of the object owner (NFT or collection), or the signatory.

R2. Providing possibilities to the object owner and the signatory for seeing all the information about the signature.

R3. The impossibility of spamming the object with the signatures of people from whom the owner does not expect a signature.

R4. The impossibility of obtaining a signature of a person by fraud against his will.

R5. The impossibility of validating a signature with a smart contract other than expected by the owner and signatory.

R6. The impossibility of validating a signature issued by a provider other than expected by the owner and signatory.

R7. The possibility to easily check the availability of a signature directly through blockchain transactions (avoiding any third parties and services).

R8. The possibility to quickly find all signatures for a given object.

Below we introduce two conceptually different architectures able to solve problems **R1**-**R8** to one degree or another.

7. Different Schemes of NFT Signing

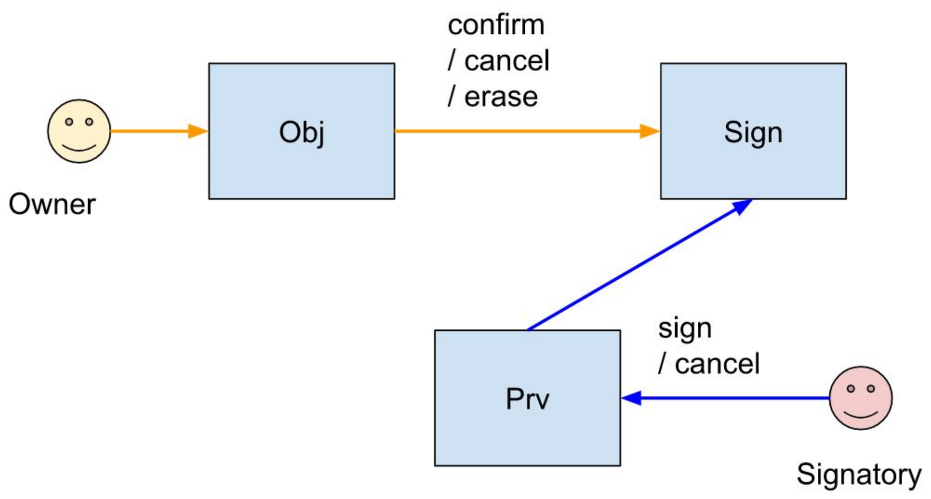

Fig. 1.: *Pull scheme of object signing: the signing process is initiated by the signatory.*

Above we have specified the main requirements to be met by a solution for NFT signing on the TON blockchain, as well as the main operating components of that solution. The first and main question that needs to be answered is who the initiator of the signing shall be. Is that the object owner or a potential signatory? Based on the answer to this question, there can be two qualitatively different architectures that satisfy all the stated requirements but have their own advantages and disadvantages.

7.1 Pull scheme of signing

In this signing scheme, the process is initiated by the signatory. He or she, when seeing someone's object, expresses the desire to sign it, and the NFT owner decides later whether he/she wants this signature.

This is reflected in Fig. 1. The signatory gives the provider a command to sign an object. This action is automatic confirmation of the signatory of this action.

The provider deploys a signature smart contract that includes:

- Address/DNS of the signatory.
- Address of the NFT object to be signed.
- The provider's address.
- Optional signatory's comment to the signature.
- Date of signing.
- Additional information about the signatory: links to social networks, a graphical representation of the signature, etc., if available to the provider.

Next, the signature smart contract waits for one of the possible actions:

1. Confirmation from the object owner in the form of a simple transaction originating from the NFT object itself.
2. Cancellation of the signing request on the NFT object side, if by this time the signature has not been confirmed.
3. Cancellation of the signing request on the signatory side, if by this time the

NFT owner has not confirmed the signing.

4. Erasure of the existing signature on the object owner side (if such functionality is allowed by the current solution).

Here it is necessary to pay attention to some points. Deployment of the signature smart contract shall occur on the part of the signatory. For usual document signing in the real world, it is a somewhat unusual scenario. Rather, it is more suitable for petitions, when it is not so important for the owner of the document who exactly will sign his/her document. In our case, signing will still not happen without a subsequent confirmation from the object owner, so this scenario is also eligible. Moreover, it looks more convenient in terms of UX than the push scheme of signing described below.

Let's check if the given signing scheme meets conditions **R1**-**R8** specified above.

R1. Opening a signature is possible only on the side of the signatory. Cancellation of an open but not yet confirmed signature is possible on the part of the signatory and the owner of the object. Erasure of a verified signature is possible only on the part of the current owner of the NFT object – **met**.

R2. All information about the signature is available to any user by a simple get-request to its smart contract – **met**.

R3. Technically, anyone can request to open their signature on a given NFT. However, this includes some commission/fees, plus the signature will not be considered valid until validated by the object owner. Thus, direct spamming of objects with signatures is practically excluded – **met**.

R4. Obtaining someone's signature fraudulently or without their knowledge is extremely difficult since the signature request transaction must include a special payload – the address of the object being signed and some other information. Further, the provider can imitate someone else's signature intention only if it is not checked. The contract code of official providers is not able to forge signatory requests and will be placed in the public domain. However, given the risk of use by unofficial and unverified providers, there remain some risks to all parties – **partly met**.

R5. The signature smart contract is weaved into the provider's smart contract. Therefore, as in **R4**, the risk of signature contract spoofing remains only if the signatory uses unofficial and unverified providers – **partly met**.

R6. Provider spoofing is exactly what was discussed in **R4** and **R5**. Under the condition of using proven and open providers, there is no spoofing risk – **partly met**.

R7. To confirm the signature being put on the object is possible through corresponding transactions in the blockchain. First, it is needed to make sure that the addresses of the object, the signatory and the provider are correct.

Knowing these addresses, it is possible to track the opening transaction from the signatory and the confirmation transaction from the NFT – **met**.

R8. The necessity to repeat actions described in P7 for all potential signature transactions in the entire transaction history of this object hides the possibility of simply finding all signatures for a given object. This is possible, but technically not quite trivial – **partly met.**

Thus, the pull scheme of object signing is free from all disadvantages of a naive signature with a simple transaction, however, it may not be suitable for all cases. To prevent spoofing, it is necessary to monitor the correctness of the addresses from which the request for signature came and to which their confirmations are sent or use trusted signing service providers. Other than these minor inconveniences, the pull scheme is fully viable and meets all the above requirements for a secure NFT object signing architecture.

7.2 Push scheme of signing

In this signing scheme, the process is initiated by the owner of NFT. He/she opens the object to be signed by a specific individual. In this case, the signatory must be somehow notified that a signature for some object awaits his/her confirmation. Upon signature confirmation by both parties – the owner of the object and the signatory – it is deemed verified. Note that signature confirmation by the NFT owner, in this case, may occur both before and after its confirmation by the signatory; it may not even be necessary if the owner fully relies on the provider. In particular, it means that the provider can only open an NFT for signature at the owner's request.

The described process is shown in Fig. 2. The owner of NFT gives the provider a command to open the signature of some signatory for some object. The provider deploys a signature smart contract that includes:

- Address/DNS of the signatory.
- Address of the NFT object to be signed.
- Address of the provider itself.
- Additional information about the signatory: links to social networks, a graphical representation of the signature, etc., if available to the provider.

Next, this signature smart contract waits for one of the possible actions:

1. Confirmation from the signatory in the form of a simple transaction. The time of signing and an optional comment are to be fixed in the signatory contract.
2. Cancellation of the signature request on the side of the NFT object or provider, if by this time the signature has not been confirmed.
3. Erasure of the existing signature on the NFT owner side.
4. (optionally) Confirmation of the signature on the object owner side.

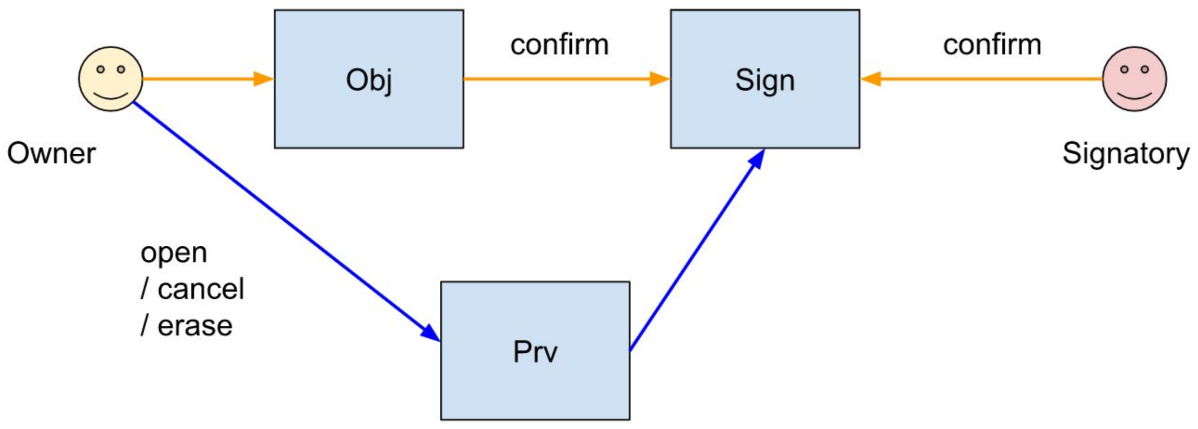

Fig. 2.: *Push scheme of object signing: the process of signing is initiated by NFT owner.*

Here, it is necessary to pay attention to some points. Deployment of the signature smart contract occurs on the part of the owner of NFT. For standard NFT contracts, it is quite difficult to implement this mechanism. Most likely, this task will have to be shifted for the provider, as shown in Fig. 2, which means additional centralization and trust between the NFT owner and the provider. Further, the NFT owner will have to confirm his/her ownership at the time of the execution of the signing request. Finally, the NFT object gets opened for signing not by all signatories at once, but by the specific one wanted. These features of the push scheme are quite natural for the usual signing of documents in the real world; however, they can cause some inconvenience for specific NFT signing cases in the blockchain.

Let's check if the push signing scheme meets the **R1**-**R8** prerequisites above.

R1. Opening, canceling, and erasing of a signature is possible only on the part of the object owner. However, this will most likely have to be implemented via an intermediary – a provider – **met**.

R2. All information about the signature is available to any user through a simple get-request on its smart contract – **met**.

R3. Object spamming with signatures is technically impossible since the action of opening it for signature is a conscious one on the part of the object owner – **met**.

R4. Obtaining someone's signature fraudulently or without their knowledge is possible if confirmation is understood as a simple transaction on the signatory side. This can be avoided by requiring, for example, a special prefix in the text message to the transaction – **partly met**.

R5. The signature smart contract is weaved into the provider's smart contract, so the risk of spoofing the signature contract remains only if the signatory uses unofficial and unverified providers – **partly met.**

R6. Provider spoofing is exactly what was described in **R5**. Under the condition of using proven and open providers, there is no spoofing risk – **partly met**.

R7. Corresponding transactions in the blockchain make it possible to confirm a signature on an object First, it is required to make sure that the addresses of the object, the signatory and the provider are correct. Knowing these addresses, it is possible to track the opening transaction from the signatory and the confirmation transaction from the NFT – **met**.

R8. The necessity to repeat actions described in P7 for all potential signature transactions in the entire transaction history of this object hides the possibility of simply finding all signatures for a given object. This is possible, but technically not quite trivial – **partly met.**

It is obvious that the push scheme is similar to the pull scheme generally but differs significantly from it in a number of details. In the push scheme, the object owner acts as the initiator of signing, which is natural for usual document signing, but may be inconvenient for a number of NFT signing cases. Both schemes are generally functional and solve the problem with the proviso that transaction control remains the responsibility of all parties involved in the signing.

8. SNFT – Technical Implementation

It follows from the discussion of push and pull architectures that the former is more resistant to spam attacks and is generally more similar to the classic document signing process while the latter is more convenient in terms of user experience and is generally more attractive for NFT signing. We will proceed from the pull architecture as more interesting and user-friendly.

There are three smart contracts involved in the signing pull scheme: the object to sign (NFT or collection), the provider, and the signature contract. Let's discuss the main points of implementation of each.

Object to sign
As it follows from the diagram in Fig. 1, the only thing that a to-sign object should be able to do besides its standard implementation is to send messages to confirm, cancel and erase (if required) a signature. If the object's default implementation does not send such messages, it should be added. It can be done with a special mixin – a file that is compiled together with the contract of the source object and supplements it with the necessary functionality. This separate *.fc file will contain all specific handlers, and the corresponding call should be added to the source smart contract code. Based on this scheme, there can be added the signing functionality to both the NFTs themselves and the collection contract in a uniform way:

```
() recv_external(int my_balance,
                 int msg_value,
                 cell in_msg_full,
                 slice in_msg_body) impure {
    ;; object-specific code

    ;; ------------------ Signable block ------------------
    int handled? = recv_internal_sign_mixin(
        my_balance, msg_value, in_msg_body,
        op, query_id, sender_address, owner_address, signature_params);
    signature_params = pack_sign_params();
    if (handled?) {
        save_data({- params to save -});
        return ();
    }
    ;; ------------------ Signable block ------------------

    ;; object-specific code
}
```

Signature provider
Signature Provider The presence of the signature provider as a separate smart contract has the following goals:

- standardization of signature contracts;
- centralized deployment of signatures;
- collection of statistics required for fast signature validation;
- collection of operation fees;

To achieve condition **R4** – the impossibility of obtaining a person's signature fraudulently against his/her will – the contract should not accept simple transactions with an arbitrary message as a request to open signatures. It is possible to protect against this kind of fishing by, for example, requiring the signatory to forward messages with a special expected prefix, or by setting an opcode other than 0 (simple transaction). For example, a transaction with a comment

sign~from Pavel with love

will confirm the fact of a non-occasional opening of the signature, and the text without the service prefix will be written in the commentary of the signature itself:

from Pavel with love

For complete transparency, a provider's smart contract shall return all available information about its current state and the contract code of the signatures it emits.

In a certain sense, the provider can be viewed as a marketplace contract in the standard scheme for putting NFTs up for sale. The latter also issues a special sale contract, but, by contrast, the signing contract does not require the transfer of ownership of the NFT object to it.

Signature
A signature smart contract is created by the provider at the request of the signatory. The first transaction deploys the smart contract itself, the address of which depends only on the three addresses (the address of the object, the address of the signatory), and besides that provides some additional information about the signature: links to the social networks of the signatory, a graphical representation of his/her signature, etc. Whether to include the provider's address in the initial state of the signing contract is up to the implementation process. Perhaps this is worth doing to avoid the possible third-party deployment of the same contract by third parties, bypassing the general scheme. However, even such a deployment will only affect the statistics collected by the provider but will not affect the quality of the signature itself.

The signature is considered validated after the corresponding transaction from the address of the object to the address of the signature. The transaction from the object also allows cancellation or, if necessary, erasing the signature. In these cases, the signature contract is deleted, and its funds are returned to the object.

9. Application Scenarios

Signing NFTs and their collections is a new feature that has no direct analogs for the time being. The signature, like the object being signed, is a smart contract that is also stored in the blockchain and is reliably linked to the owner who registered it. The potential of such a solution is great – from a simple increase in the collection value of an object to a reliable anti-scam verification. Initially, an empty and virtual NFT object acquires real value, which increases in multiples with time and new signatures of celebrities on it.

Autograph
The first and most natural application of object signing is autographs of famous people on NFT which significantly increase their market value and value to the owner. It is important to understand that while the value of an NFT usually falls quickly over time, just one signature can well anchor the price at a high level, and it will only increase over time, as it happens to autographed collectibles in the real world.

Attraction
The release of NFT collections with their subsequent signing can serve as a quick and bright way to attract celebrities to the TON blockchain and increase its recognition in the world.

Signatures
Technically, any user can put an autograph as a gift to another, for example, for any holiday. If NFT is used as an object that represents some real document, then the signatures on it can be considered a certifying visa of all actors.

Verification
Anti-scam verification by signing an object is one of the most striking applications of the concept of signatures. For the time being, marketplaces have to manually track and ban scam collections that are copies of the real ones. If the author of the collection puts his/her signature on it, then this will automatically indicate it as the authentic one, thereby simplifying its anti-scam verification. It is easy to imagine even the case when a well-known person puts his/her signature on a scamming object and then the object becomes normal with an increased market value by orders of magnitude.

Charity
Having a celebrity sign someone's NFT is great, but what if the NFT or collection is signed by multiple celebrities at the same time and auctioned off for charity? It is possible to mint entire collections in support of some goal, and the signatures of famous people on them will multiply the value and allow to raise more funds.

Event invitations and tickets
NFTs are even now being used as tickets and invitations. Signatures on them open up a whole new potential for such application of NFTs. For example, VIP guests can receive signatures from key speakers and event organizers, so such a used NFT “ticket” will not become useless after the end of the

event. Moreover, the signatures on the NFT ticket can flexibly regulate the participant's access to certain zones of the event. If an event has for example 5 zones and 3 levels of access, then (25 - 1) * 3 or about 100 different types of NFTs would have to be used to cover all alternatives to attending it. Instead, the same task is solved by NFT with 6 slots for the corresponding visas – 5 for zones and one for an access level. It is not difficult to come up with more complex scenarios for such use of signable NFTs.

Metaverses and the game industry
One of the central NTF applications in recent times has been games and metaverses. Almost every object in the metaverse is an NFT, and the very interaction of gamers is based on the exchange of use of such objects. Adding the possibility to sign NFTs in metaverses and games will provide the same benefits as in the real world. Thus, everything that is described above for the real world will turn out to be quite true for the metaverses: autographs, visas, verifications, invitations, etc.

It is easy to come up with other scenarios for using token signatures in the real world. We are firmly convinced that NFT as a concept has come forever to our world and will only expand the range of applications in various fields over time. The concept of signable NFTs, in turn, will also be inextricably linked to it and will probably soon be universally included in NFT standards of many blockchains.

Conclusion

The purpose of this paper is to demonstrate the idea and discuss possible implementations of the concept of signing NFT objects. The TON blockchain is a fast blockchain of the next generation and is ideal for the high-quality implementation of this idea. We believe that the concept of NFT signing proposed here will be widely accepted and greatly expand the scope of NFTs per se. With its great potential, the described technology makes it possible to successfully solve many specific business tasks – from attracting famous people to TON and getting their autographs to charity and competent organization of events.

References

[NFT] - https://en.wikipedia.org/wiki/Non-fungible_token
[NaS] - https://www.taskus.com/insights/nft-as-a-service-industries-trends/
[Naa] - https://opengeekslab.com/blog/nft-services/
[Nap] - https://www.leewayhertz.com/nft-apis/
[Naf] - https://mn2s.com/our-services/nft-services/
[Nbm] - https://moralis.io/what-are-nft-based-memberships-full-guide/
[Ntt] - https://www.binance.com/en/blog/nft/what-is-nft-ticketing-and-how-does-it-work-421499824684904022
[EsW] - https://en.wikipedia.org/wiki/Electronic_signature
[EsA] - https://www.adobe.com/sign/electronic-signatures.html
[Dsb] - https://www.coinbase.com/cloud/discover/dev-foundations/digital-signatures
[hsB] - https://101blockchains.com/hashing-and-digital-signature-in-blockchain/
[esB] - https://academy.binance.com/en/articles/what-is-a-digital-signature
[NsT] - https://github.com/ton-blockchain/TIPs/issues/62
[Tsm] - https://ton.org/docs/#/smart-contracts/messages
[Nmm] - https://www.coindesk.com/layer2/metaverseweek/2022/05/23/how-the-metaverse-could-be-a-game-changer-for-nft-gaming/
[EsT] - https://www.techtarget.com/searchcontentmanagement/ElectronicSignatures/Why-Electronic-Signatures-and-Why-Now