CSCI530: Computer Secrity Systems

Term Paper

Non-Fungible Token: Security Concerns and Thoughts

Yining Liu
6168529797
liuyinin@usc.edu

I have read the Giude to Avoiding Plagiarism published by the student affairs office. I understand what is expected of me with respect to properly citing sources, and how to avoid representing the work of others as my own. The material in this paper was written by me, except for such material that is quoted or indented and properly cited to indicated the sources of the material. I understand that using the words of others, and simply tagging the sentence, paragraph, or section with a tag to the copied source does not constitute proper citation and that if such materiel is used verbatim or paraphrased it must be specifically conveyed (such as through the use of quotation marks or indentation) together with the citation. I further understand that overuse of properly cited quotations to avoid conveying the information in my own words, while it will not subject me to diciplinary action, does convey to the instructor that I do not understand the material enough to explain it in my own words, and will likely result in a lesser grade on the paper.

Signed: 

Non-Fungible Token: Security Concerns and Thoughts

YINING LIU

abstract - The concept of non-fungible token (NFT) and its marketplaces have been mushrooming for the past years, growing tremendously in trading volumes. As NFT is on its way going viral and the number of participants is exploding, people start to hear about NFT everywhere, but the underlying technologies of this industry remains mysterious in the public eye.
The first half of this paper aims to introduce implementation of NFT system, by elucidating underlying technologies and implications behind their design. The second half introduces main functionalities of NFT and discusses possible security concerns in detail.

1. PRELIMINARIES OF NFT

In this section, we will go through several concepts that are tightly related to NFT. They are the preliminaries to understand life cycles of NFT and then to discuss security properties of NFT on each phase. Background information such development history, working mechanisms, security and privacy related characteristics are in our range of discussion.

A. Ethereum

Ethereum is a decentralized, open-source blockchain with Ether (ETH) as its native cryptocurrency. It was initially conceptualized by programmer Vitalik Buterin [1] and went live in 2015 with participation of addtional founders. Ethereum is a new generation of distributed ledgers which inherits Bitcoin's incentive scheme for miners, mechanism of cryptographic proof of work (PoW) challenges (Ethereum moved to proof of stake (PoS) in September, 2022[2]), and also reserves the storage structure of Bitcoin blockchain. With the similar design of linked blocks and Merkle tree, the transactions stored on the Ethereum is immutable and undeletable once uploaded, and can be verified through public access at any time. Being referred as the 2.0 era of Bitcoin blcokchain, Ethereum achieves properties such as decentralization, transparency, traceability and tamper-proofing by nature, which provides its users with verification functionality and data integrity.

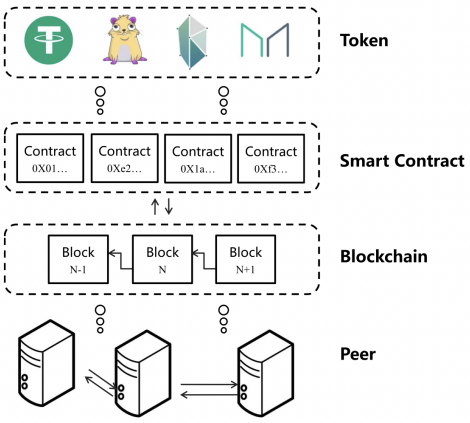

Fig. S1. Overview of ethereum blockchain. Source [3]

Moreover, while Bitcion is more of a currency, or a store of value and Bitcion blockchain is nothing other than an unhackable but naive ledger, Ethereum enables building and deploying smart contracts and decentralized applications with its implementation on a turing-complete language – *Solidity*, which gives it the capability to handle complex codes and exploiting the very best of computing power. In other words, Ethereum is a programmable blockchain. *"Ethereum is the community-run technology powering the cryptocurrency ether (ETH) and thousands of decentralized applications"*, is what we can find as the headlines on the official homepage of Ethereum [4].

B. Smart Contract

As stated in the above section, smart contract functionality is one of the main differences between Bitcoin blockchain and Ethereum. A smart contract *"can be described as lines of code that are stored on a blockchain and are automatically executed when predetermined or predefined terms and conditions are met"*[5], so that all participants can immediately be certain of the outcome, without any time loss or involvement of a middleman. Because smart contracts are self-executed and no paperwork or manual work is involved, they are expected to be way more efficient and accurate than regular contracts.

Note that on original conceptual level, smart contract is not exclusive to blockchain or Ethereum, the outcome of execution could be any action which has an analogy in the real business world, such as registering a vehicle, or issuing a check. However, with the astonishing development of Ethereum in the past years, it is fair to say that the concept of smart contract is now bound to Ethereum and blockchain. Also, properties of blockchain enhance smart contracts and bring the best out of them in many ways. Other than being efficient and accurate, smart contracts are transparent and tamper-proofing under the application scene of blockchain, because contracts are shared across participants once uploaded cannot be deleted by default, records on ledgers can not be altered, or to say, interactions with smart contracts are irreversible.

There are specialized contract-oriented high-level languages to enable developers to program smart contracts. For example, *Solidity* with syntax similar to that of *JavaScript* and *Serpent* with syntax similar to that of *Python*. *Solidity* is currently the most popular language while *Serpent* is fading out. In real practice, Ethereum business usually provide templates, web interfaces, and other online tools to simplify of process of programming smart contracts.

Smart contracts that deployed to the Ethereum exist in a form of Ethereum accounts, which means they have balances and work as they are programmed when becoming targets of transactions. When a user account hopes to interact with a smart contract, it submits a transaction that execute a function defined on the smart contract, the outcome will be enforced automatically.

C. Non-Fungible Token (NFT)

Tokens refer to transferable digital assets in the scene of Ethereum (or blockchain). Before non-fungible tokens, there are fungible tokens (FT) that act as a secondary currency (other than native cryptocurrency Ether) in Ethereum. Implication of the word "fungible" is the reason why FT can be seen as a kind of currency: any two FT of the same value can be swapped and no difference will be caused because they are identical and interchangeable.

Now without the fungibility, NFT cannot be supplanted one for another, considering no two NFTs are identical. This is the key difference between NFT and FT. Also, when we go with terms like "one for another" and " no two NFTs are identical", a secondary difference between the two can be inferred: FT is divisible while NFT is not. Those characteristics of NFT further makes it suitable for identifying a specific asset and representing the ownership of it. According to Lavrova and Iakushkin's work [6], except for common uses such as representing digital art, music, and game assets, attempts are also being made to link NFT to virtual real estates, attendance receipts, domain names and even physical objects. On implementation level, an NFT always contains a reference pointing to the asset it links to, which is what we call metadata. matadata are usually URIs that link to JSON files other medias which are descriptions of NFT or NFT itself.

We often find the uses of term "NFT", "smart contract" and "ERC-721" (an NFT standard to be discussed in the next section) are confused and mixed up in real practice and even in many academic articles. This is because those terms together serve a common purpose while putting into use, and their concepts overlap. It is fair to say that NFT would not be brought to life without smart contracts, and this is why we need to go through the concept of smart contract before this section. Some may argue that an NFT and a smart contract can be looked as one whole unit [7], but a more precise way to describe their relations should be: If a smart contract implements certain methods and events that conformed to ERC-721 token standard, it can be called an ERC-721 NFT contract, and it is bestowed the ability to represent an NFT, and once deployed, it will be responsible to keep track of the NFT it created on Ethereum.

To further clarify, there are smart contracts who serve various purposes and have various outcomes other than representing FT and NFT, including proxy smart contracts, access smart contracts, crosschain smart contracts, to name a few [8]. In this paper, we focus on ERC-721 NFT smart contract and may mention ERC-20 FT smart contract as a comparison. In fact, both fungibility and non-fungibility can be realized by implementing smart contracts, as long as appropriate interface standard is conformed for each condition.

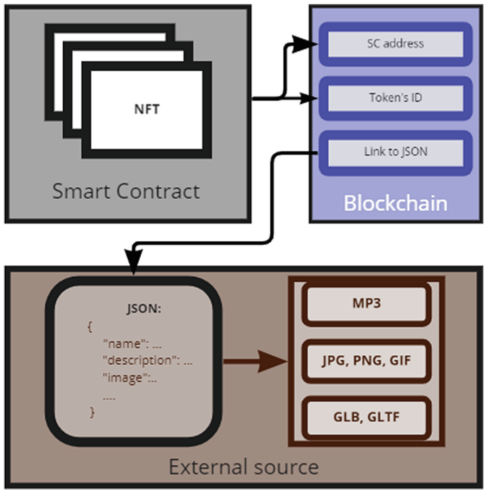

Fig. S2. Architecture of NFT token components' links. Source [6]

D. ERC-721

ERC-721 [9] is the mainstream interface standard for NFT on Ethereum. The standard itself defines a minimum interface a smart contract must implement to allow NFT to be managed, owned, and traded. However, the standard does not impose any restrictions on the metadata of NFT or forbid the addition of supplemental functions. Currently, ERC-721 contracts that are popular in use is provided and supported by *Open Zeppelin*, a security products library, who who has also implemented ERC-20, the interface standard for FT. If adding a few more comments to *Open Zeppelin* on its security level, it is built on a solid foundation of community-vetted code and has been audited by leading security firms and, all blogs for past audits are open to public on their website.

Since ERC-721 was moved out of beta after projects like CryptoCelebrities[10], CryptoKitties[11], and EtherTulips[12] and, brought to official release in Solidity. It now defines nine methods (or to say, functions) and three events (the pre-defined conditions that contracts listen to). In manner of Solidity, function execution must always begin with an external caller. Until one of its functions is called, a smart contract will simply sit on the Ethereum doing nothing. The address of the external caller who called the current function, in this case, is referred as `msg.sender`, which is one of several global variables in Solidity that are accessible to all functions.

To get some intuitive impressions on how ERC-721 manages tracking and transferring functionalities of NFT, we take a closer look at the design of ERC-721, by way of comparison with that of ERC-20 when necessary.

Within the ERC-721 smart contract, each NFT is identified by a distinct ID of datatype `uint256`, and this identification number will not change for the life of the contract. `uint256` is chosen to represent NFT token ID because it is compatible with `UUIDS` and `Keccak256` hashes, which allows a wider variety of applications. For a particular asset on Ethereum blockchain, we use the `address` that refers to the smart contract and the `tokenId` of this asset to create an identifying pair `(contract address, uint256 tokenId)`, which will then be seen as a global and fully-qualified identifier.

As for the asset linked to NFT, ERC-721 provide a mechanism to associate NFTs with its metadata, i.e., URIs. Recall that NFT started its journey from digital pictures, so ERC-721 also suggested a recommended image size from `Instagram`, assuming that they have a wealth of knowledge on the usability of images. The return value of metadata retrieval function is an URI as a string, and calls to the retrieval function can only be made from web3 applications. Alternatives under consideration include putting all asset metadata on the blockchain, but some argue that it is too expensive for current stage. Those URIs are now mutable under implementation of `Openzeppelin`, given that in cases like representing the ownership of real estates, information about them (e.g., occupants, appearance) may naturally change. This, however, make us wonder how they deal with tampering issue of metadata.

The owner of an NFT, the approved address of an NFT, or an authorized operator of the

current owner of an NFT may initiate a transfer. Three transfer methods are standardized in ERC-721: an unsafe function `transferFrom` and two safe transfer functions `safeTransferFrom`. Three main components in a transfer, the sender, the receiver and the NFT, are input to functions as parameter "`_from`" address, "`_to`" address and "`_tokenId`". In the unsafe function, the responsibility to confirm that "`_to`" address is available and is capable of receiving NFTs is left to the caller `msg.sender`, failure to confirmation might lead to permanently loss of the NFT. On the contrary, safe transfer functions check if "`_to`" address is indeed a smart contract and call function `onERC721Received` after confirmation to handle the receipt of the NFT. Versions of function `safeTransferFrom` with and without additional data as input are provided for flexible use.

Compare to transfer functions in ERC-20 for FT, ERC-721 does not suffer from later-modified-allowance problems[13]. An `allowance` feature is defined in ERC-20 to represent the quantity of FT, which may become target of modification after transfer and cause loss to the sender. However, NFT is indivisible and have no such feature as amount, thereby can hardly be victims of this issue.

As mentioned before, an authorized operator of the current owner of an NFT can also initiate a transfer. Under the circumstance that an authorized operator is applied, it may set the an approved address for an NFT as well. This gives wallet, broker, and auction apps a potent set of capabilities for quickly utilizing a huge number of NFTs. Also, they must have a strong need to identify which NFTs an owner owns. The good news is, function `owenerOf` in ERC-721 can help them verify the owner for any possible token at anytime as they wish. But what about privacy? Unfortunately, ERC-721 cannot ensure privacy by original design, and leave it developers to improve. Develops may then add extensions that can be later applied to situations like private ownership registry, or other occasions that privacy is emphasized.

2. SECURITY CONCERNS OF NFT

In this section, security challenges at the pace of an NFT's life cycle will be discussed. We will revisit main events happening on each phase of an NFT's life, and then arise our security concerns. Current solutions to these problems or prevention methods and mechanisms will be mentioned if there is any.

A. Minting

After a piece of digital art is created, an NFT need to be `minted` before it can be written to Ethereum and can establish its immutable record of authenticity and ownership. This is achieved by calling the appropriate method of ERC-721 and passing appropriate parameters to it. Specifically, a transfer function is called and "`_from`" address set to 0-address, denoting the creation (minting) of an NFT in association with its digital asset.

In real world, the concept of "collection" is naturally raised because s series of arts often come in similar forms or share a common theme. This "collection" concept is inherited on Ethereum to apply to digital arts and their NFTs. A single contract conforming to ERC-721 is able to manage a series of NFTs, which are referred as an NFT project sometimes. Each NFT in the collection is assigned a "`_tokenId`" so that we can then identify it with pair `(contract address, uint256 tokenId)` on the blockchain.

A.1. Integration of External Contracts

Das, Bose, Ruaro, Kruegel, Christopher and Vigna's work[14] summarized minting of NFT in three ways: **(a)** Creators deploy default contracts that are pre-deployed and pre-designated by NFT marketplaces, and no customized parameters allowed in this case. **(b)** NFT marketplaces deploy replica contract on behalf of creators and therefore conduct the management functionality. **(c)** Creators independently deploy custom contracts to manage their own collections and later import them to the marketplace.

(a) and **(b)** together are called internal contracts while **(c)** is called external contracts. Effective integration and verification methods for external contracts are significant because they can be malicious or malfunctioning. To address this, some NFT marketplaces do not allow external contracts at all. As for others such as OpenSea[15] and Rarible[16] who allow this type of customisation, it is ideal for them to mandate external contracts to be open to public scrutiny and verify their conformed smart contract standards before they are integrated into the marketplace's own contract, and to set corresponding "take-down" mechanisms if external contracts are found malfunctioning afterwards.

A.2. Storage of Digital Asset

There are three different locations to store digital assets: **(a)** Personal file server (HTTP server) hosted by the creator of NFT. **(b)** IPFS (Interplanetary File System) that provides the user with unique links containing a hash of data. **(c)** On Ethereum blockchain.

Table S1. Different storages of digital assets

-	**(a)** Personal Server	**(b)** IPFS	**(c)** Ethereum Blockchain
Data Modification	*Yes*	*No*	*No*
Data Deletion	*Yes*	*Yes*	*No*
Data Distribution	*No*	*Yes*	*Yes*

Management of digital assets in case **(a)** is not under supervision of blockchain ecosystem and responsibility to verify the asset while or before minting is left to artists and buyers alone. Attack methods to HTTP server are out of discussion range of our paper, but still make **(a)** the least secure and least recommended way.

(b) is better than **(a)** because URLs stored in IPFS including the hash of their content cannot be modified once uploaded. One can always mint after retaining the URL recorded in IPFS and previewing the digital asset. But still, data storage remains off-chain and data deletion becomes possible when the whole storeage node is shut down. In this scenario, owners won't have access to their assets and will just hold a empty record on Ethereum. Also, some NFT projects promise the rarity of certain NFT collections but then abuse the commitment by minting more NFTs than the promised amount, thus harming the benefits of the buyers.

(c), as an alternative, is not considered much currently due to its high cost of management. However, it is the best practice to verify digital asset while or before minting to ensure data integrity. This might come into effect in future when blockchain technologies become cheaper, hopefully.

B. Transferring

This subsection is named "transferring" because this is what included in ERC-721's original design. That being said, actions such as listing, bidding, and selling are more precise to describe behaviors in real world NFT trading marketplaces. However, those functions are not included in ERC-721 by default, therefore, NFT marketplaces or dApps are always in charge of implementations of their own auction or trading logic.

B.1. Bidding

As mentioned above, bidding is likely to happen before an offer is accepted or an auction is settled. It is interesting to discuss the manner of bidding on Ethereum because more than one buyers may offer bids at the same time, and bid withdrawals may be available at marketplaces or buyers' wish, which indicates multiple active bids on the same NFT. These features show a tendency of concurrency. However, it is known that Ethereum does not support concurrency. Also there is debate about where we should store bids, on-chain or off-chain.

Currently, NFT marketplaces' solutions to address this bidding problem vary greatly[14]. Marketplaces like `CryptoPunks` store their bids on-chain, and therefore disallow multiple active bids at the same time. When a bidder outbids the current top bidder, the latter gets automatically refunded. Others keep bids off-chain for for gas efficiency, which means Ethereum is not involved until an offer is settled and transfer of an NFT is about to happen.

The former on-chain mechanism hands the duty of ensuring the fairness and integrity of bidding to Ethereum. Again, with properties like transparency, we will be able to verify. Under the off-chain circumstance, cryptographical methods are used to verify bidding offers so that malicious bids are prevented.

B.2. Transaction Privacy

"*Don't trust, verify*" is a famous proverb to describe blockchain technologies when Bitcoin first went virus. It implies that transactions (i.e., transfer details of NFTs on Ethereum) should be available to public for verifying. This may violate privacy of buyers and sellers when sensitive information are involved and adversary may collect this kind of identity information for further attacks.

Thoughts to protect privacy are raised from different levels of the Ethereum structure. On the level of smart contract, Kosba[17] designed a privacy-preserving tool to add to current ERC-721, while Watanabe[18] suggested to encrypt smart contracts before deploy then to Ethereum. The former one does not store sensitive transactions on the blockchain with they smart contract framework and the latter one made smart contracts invisible to public. These are great ideas to address our problem, but also leave us in dilemma between transaction privacy and transparency.

From another perspective, Oasis[19] proposed confidential NFT (cNFT), which makes use of a so-called cNFT Oracle and a trusted execution environment (TEE) linked to an underlying blockchain. The Oracle contains a full permissioning and computing layer which decrypt data related to the NFT and parse blocks to track ownership of NFTs. Developers can interact with this cNFT Oracle using an SDK. Furthermore, the entire parsing procedure takes place inside a secure enclave that is running inside of the TEE to verify newly generated blocks securely. Using the remote attestation feature of TEEs, users can verify that the enclave runs the correct code at any time.

B.3. Risks on Cross-Chain Transfer

The current blockchain systems are fragmented. Public blockchains like Ethereum and Bitcoin blockain, along with private and permissioned blockchains for various purposes are isolated from each other. As a result, it is challenging to transfer digital assets to another blockchain if they were originally purchased on Ethereum. This is also the key barrier to the introduction of the NFT ecosystem into crypto games, since identical games may be hosted on different blcokchains but transfers of gaming equipments are considerably frequent[20].

Right now, we're limited to using cross-chain bridges to transfer user assets between different blockchains. A cross-chain bridge is a third-party application that simultaneously monitor the network the user is on and the network the user wish to connect to. For example, the bi-directional bridge Wormhole[21] allows connectivity between the Ethereum and Solana blockchains by wrapping tokens. When user requests transfer from Solana to Ethereum, *Wormhole* will lock NFT on Solana and then mint a wrapped version of it on Ethereum. Unwrapping means bring token back to Solana and the wrapped version will be destroyed in this case.

Fig. S3. Cross-chain bridge scheme. Source [22]

However, assets transferred on a different chain will not be nearly as secure as assets on their native blockchain and if protocol inter-dependency go beyond two bridged blockchains, the risk becomes exponentially greater. This year in August, The cross-chain token bridge *Nomad* was exploited, with attackers draining the protocol of virtually all of its funds[23]. The total value of cryptocurrency lost to the attack totaled near $200 million USD.

Unfortunately, improvement proposals on security of cross-chain bridges mostly focus on non-technical level: people are suggesting decentralized validators, monitored transfers, audits and insurances.

C. Managing

Expect for minting and transferring of NFTs, there are other actions (conducted by users) or scenarios (passively encountered) in NFT systems. In this subsection, we discuss security concerns during daily management of NFTs.

C.1. Burning

Burning an NFT means destroying it. Once an NFT is uploaded to Ethereum, it is impossible to delete it, so burning is realized by sending a token to an unspendable address that no one has access to. There are numerous burn addresses spanning across Ethereum, but the official burn address is `0x000000000000000000000000000000000000dEaD`.

The most frequent reason for NFT burning is value management. A decrease in supply raises an asset's price, luring traders and investors to participate with urgency. Others may burn NFTs simply because the token has an error or flaw, or too many were minted.

Burning functionality is not mandatory for an NFT smart contract, but a smart contract with a burn mechanism might be authorized to burn your NFT, or even authorize others to burn your NFT. Given consideration to that, marketplaces like OpenSea set guidelines for how an NFT project must be applied for user trust and protection, including the declaration of the existence of the burn mechanism, price range of to-be-burned NFTs, and consensus or event that triggers the burning action.

C.2. Spam NFTs

Spam NFTs are unsolicited NFTs that are sent to user's wallet address to attempt phishing attacks. Generally, spam NFTs prompt the receiver to click a link to mint a free NFT, in where users ends up losing funds. Alternatively, the link will ask the receiver to input their seed phrase, resulting in the same outcome. Because anyone can send and receive tokens to and from a wallet address on Ethereum, unwanted NFTs occasionally appear. Sometimes, marketers send spam NFTs to users, aiming to market their newly launched NFT projects. Despite this type of spam NFTs means no harm to users, they lower user Experience as well.

Tools are developed as extension to prevent spam NFTs. For example, Alchemy's[24] `getNFTs` spam filter can be used to filter spam NFTs that have been classified as spam. Their complete design also includes the `isSpamForContract` endpoint, which checks if a particular contract is spam or if a particular NFT by owner by a sole owner; and the `getSpamContracts` endpoint, which returns the list of all classified spam contracts on the selected blockchain.

C.3. Ethereum Scaling

Over a million transactions take place on the Ethereum network every day. However, because of this popularity, the network's limited computing resources are becoming more in demand. Ethereum scaling is here to improve network performance without sacrificing security and decentralization.

Ethereum scaling can be on-chain, which could be accomplished by dividing transactions into segments and allowing each node of the main Ethereum to process just a few components of the transactions. Or, off-chain scaling is under consideration as well, which implies the development of a separate solution from the main Ethereum.

Security of on-chain opinion is not influenced and will remain as before, while off-chain opinion may be in demand of assistance from separate solutions. There is Plasma[25], a separate blockchain, which helps user to create chains within the main Ethereum, thereby achieves scaling. And there are side-chains, whose functionalities are compatible with Ethereum, allowing digital assets to move between blockchains. Side-chains are essentially cross-chain solutions.

3. CONCLUSION

In this paper, technologies centered on NFT, including blockchain, Ethereum, smart contracts, non-fungible token and ERC-721 standard, are discussed. Our discussion goes from upper-level big concepts to lower-level small concepts with an emphasis on security properties.

By dividing NFT system functionalities into three classes, minting, transferring, and managing, we discuss security concerns part by part. Current solutions are included if applicable, and opinion on better practice scheme are suggested if there is any.

The main purpose of this paper is for the readers to understand some popular NFT-related concepts and relations among them, to have a grasp on how NFT system realize security, and to be aware of some of remaining security challenges for NFT and Ethereum.

REFERENCES

1. V. Buterin, "A next generation smart contract & decentralized application platform," (2013).
2. "Ethereum switches to proof-of-stake consensus after completing the merge," https://techcrunch.com/2022/09/15/ethereum-switches-to-proof-of-stake-consensus-after-completing-the-merge/.
3. P. Zheng, Z. Zheng, J. Wu, and H.-N. Dai, "Xblock-eth: Extracting and exploring blockchain data from ethereum," IEEE Open J. Comput. Soc. **1**, 95–106 (2020).
4. "Ethereum website," https://ethereum.org/en/.

5. D. Chirtoaca, J. Ellul, and G. Azzopardi, "A framework for creating deployable smart contracts for non-fungible tokens on the ethereum blockchain," in *2020 IEEE International Conference on Decentralized Applications and Infrastructures (DAPPS)*, (2020), pp. 100–105.
6. A. Lavrova and O. Iakushkin, "Nft performance and security review," in *Computational Science and Its Applications – ICCSA 2022 Workshops*, O. Gervasi, B. Murgante, S. Misra, A. M. A. C. Rocha, and C. Garau, eds. (Springer International Publishing, Cham, 2022), pp. 217–228.
7. "What is an nft smart contract," https://hedera.com/learning/smart-contracts/nft-smart-contract.
8. "Openzeppelin - the standard for secure blockchain applications," https://openzeppelin.com.
9. "Erc-721 offical site," https://erc721.org.
10. "Crypto celebrities," https://cryptocelebrities.org.
11. "Crypto kitties," https://www.cryptokitties.co.
12. "Ether tulips," https://ethertulips.com.
13. "Implementation of approve method violates erc20 standard 438," https://github.com/OpenZeppelin/openzeppelin-contracts/issues/438.
14. D. Das, P. Bose, N. Ruaro, C. Kruegel, and G. Vigna, "Understanding security issues in the nft ecosystem," (2021).
15. "Open sea," https://opensea.io.
16. "Rarible," https://rarible.com.
17. A. Kosba, A. Miller, E. Shi, Z. Wen, and C. Papamanthou, "Hawk: The blockchain model of cryptography and privacy-preserving smart contracts," in *2016 IEEE Symposium on Security and Privacy (SP)*, (2016), pp. 839–858.
18. H. Watanabe, S. Fujimura, A. Nakadaira, Y. Miyazaki, A. Akutsu, and J. J. Kishigami, "Blockchain contract: A complete consensus using blockchain," in *2015 IEEE 4th Global Conference on Consumer Electronics (GCCE)*, (2015), pp. 577–578.
19. "Oasis foundation- a new era for blockchain," https://oasisprotocol.org.
20. J.-T. Chen, "Blockchain and the feature of game development," in *Frontier Computing*, J. C. Hung, N. Y. Yen, and J.-W. Chang, eds. (Springer Singapore, Singapore, 2020), pp. 1797–1802.
21. "Wormhole offical docs," https://docs.wormhole.com/wormhole/.
22. "What are cross-chain bridges: And their importance for defi," https://www.coinbureau.com/education/cross-chain-bridges/.
23. "Crypto bridge nomad drained of nearly $200m in exploit," https://www.coindesk.com/tech/2022/08/02/nomad-bridge-drained-of-nearly-200-million-in-exploit/.
24. "Spam nfts and how to fix them," https://www.alchemy.com/overviews/spam-nfts.
25. "What is plasma?" https://docs.ethhub.io/ethereum-roadmap/layer-2-scaling/plasma/.