Non-Fungible Tokens: New-age Profits? Or just a gimmick.

Joshua Tan Yin Feng[A0199502Y], Wu Yifan[A0242690E], Foong Yan Kai, Brandon [A0135261N], and Donavan Lim Jia Hui[A0206088A]

National University of Singapore
joshua.tyf@u.nus.edu
yifan.wu@u.nus.edu
brandon.foong@u.nus.edu
donavanlim@u.nus.edu

Abstract. Non-fungible tokens (NFTs) are transferrable, unique, and non-duplicatable units of data registered on a blockchain digital ledger. NFTs represent rights to digital assets such as digital art and in-game collectables. Despite the zealous adoption and unrelenting trust shown by the NFT user base towards the safety of exchanging NFTs for economic gains, recent cases of users being deceived or losing their cryptocurrency due to social-engineering tactics have raised salient risks on using NFTs. We investigate two broad security implementations that underpin the implementations used to mint NFTs on third-party marketplaces: a mint-impersonation scheme and a lottery scheme. By devising schemes, constructions, and proofs, we aim to demonstrate how analysts can qualify the security of such NFT-based implementations, and as such realizing the relevance and gravitas of these security qualities in a system that holds the monetary stake of an expanding user base.

Keywords: Non-Fungible Tokens; Cryptography; Cryptocurrency.

1 Introduction

1.1 The Fervid Adoption of NFTs

Non-fungible tokens (NFTs) have seen a rise in popularity in recent times. NFTs can be considered a proof of ownership of a digital asset, a much-needed feature in today's highly digitalised world. Prior to NFTs, it was difficult for anyone to verify their ownership over digital assets since exact duplicates could be made without difficulty. Premised on the blockchain, NFTs enable verifiable ownership by associating digital assets to a digital blockchain wallet [1]. Association could be as simple as storing the pair of unique ids of an asset and wallet on the blockchain. The owner of an NFT can easily prove his ownership by the private key that gives access to the digital wallet. Coupled with the immutable property of the blockchain, these associations are safe from malicious modifications and hence protect the ownership rights of an NFT [7].

NFTs are a great solution to introduce the concept of scarcity in a digital world. Scarcity promotes the appraisal of digital assets and successfully creates a marketplace for them. Instead of trading these assets directly, NFTs are traded on their behalf. The non-fungibility of NFTs means that they are unequal in value and cannot be interchanged with each other, unlike fiat currency or cryptocurrency [9]. To trade NFTs, traditional cryptocurrencies such as Ethereum are often used in NFT-exchange on third-party marketplace platforms [1].

Investors have hailed NFTs as a potentially disruptive innovation that is paving the way for wealth preservation through digital asset ownership [12]. As such, NFTs have experienced a meteoric rise in sales volume and attention since its inception in 2015, with increasing traction from the digital art and gaming industries. NFT-exchange skyrocketed from 1.8 billion USD in Q2 2021 to 10.7 billion USD in Q3 2021, signifying the volumetric adoption of NFTs mostly in the domain of personal and non-commercial use [11].

It is important to note that an NFT, however, does not confer copyright protection over the digital asset it represents. As such, a buyer of an NFT merely obtains a license to use the art piece and cannot prevent the original author from creating more NFTs of the same work. In short, the owner of an NFT does not own nor have the right to the actual digital asset the NFT represents; the owner instead owns a certificate of authenticity, often represented by a hash code that can be exchanged on marketplaces [4].

1.2 Identifying Insecurities of NFTs

While many adopt a positive outlook for NFTs as the future of digital assets amidst fast adoption, issues pertaining to security that leave users vulnerable exist.

NFTs are not a standalone technology. Instead, they are built upon the blockchain, smart contracts, the internet, etc. Security of NFTs is thus dependent on these technologies as well. With digital signatures and hash functions underscoring the blockchain, transactions made on it can be proven to be secure. However, the security of the blockchain is only a *necessary* condition for the security of NFTs, not a *sufficient* one. There have been several instances of artists who had discovered their works sold on NFT platforms without their consent, breaching copyright. Yet, these transactions do not contradict the security of the blockchain. The blockchain is only concerned with ensuring the verifiability of the person making the transaction and the immutability of its entire transaction history. In the former case, a thief selling a piece of art not belonging to him does not contradict the verifiability of the person making the transaction.

Even though platforms have taken measures to curb fraud, many scammers still successfully deceive users to believe that they are the original creator, and illegally resell artists' works on their behalf to profit off their credibility and reputation [3]. On the other hand, there also exists the issue of theft, where hackers manipulate people to give up their account information via ransomware or use malware to steal their accounts. Once their accounts are compromised, the victims' NFTs in them are also

stolen or resold for profits [2]. All these issues further motivate the importance of creating secure schemes for NFTs.

2 The Mint-Impersonation Scheme

2.1 Impersonating an NFT mint

As alluded to in the previous section, fraud can be committed when NFTs for digital assets under an artist's name can be minted without their consent and then sold on the marketplace. The profits are pocketed into the accounts of the fraudster who minted without the artist's consent. This tactic is known as sleepminting [5].

2.2 The ERC721 Standard

NFTs are created through smart contracts, which are essentially just pieces of code that dictate the actions that can be done with regards to the NFTs. As a form of standardisation, the NFT community collectively decided on a standard known as ERC721 [6]. Being an interface (in the object-oriented programming sense), ERC721 is simply just a set functions that the NFT smart contracts must have. This standard specifies how the tracking and transacting of NFTs created by the smart contract must be done.

ERC721 however does not specify secure or good implementations for minting and burning (destroying) NFTs [6]. Furthermore, there is no security provided by the ERC721 standard itself. Being just mere pieces of code, the smart contract can be modified such that it still follows the ERC721 standard and yet allow for fraudulent mints. This is possible because NFT marketplace service providers, such as *OpenSea* or *Rarible*, do not dictate the full implementation of smart contracts based on the ERC721 standard. Finlow-Bates [8] provides a concrete example of how this could be achieved. Here, a fraudster can create an insecure smart contact that provides backdoor access to the NFTs minted into other accounts. Afterwards, using the backdoor access, they can approve transactions and claim profits despite not being the rightful owners of these NFTs.

This underscores the relevance of understanding the minting process of an NFT and creating secure minting schemes to protect users from fraud.

2.3 Mint Scheme

In this section, we introduce an authenticated minting scheme. The idea is that we want a way to prove that any person intending to lay claims on an NFT, is indeed the rightful owner.

Definition 1.1 Mint: *A* $Mint$ *consists of 3 probabilistic polynomial-time algorithms* $(Gen, AssociateOwner, VerifyOwner)$ *such that*:

- *The key-generation algorithm* Gen *takes as input the security parameter* 1^n *and outputs a random key* k *and a random string* r*, with* $|k|, |r| \geq n$
- *The owner-association algorithm* $AssociateOwner$ *takes as inputs secret key* k *as well as strings* $itemId$ *and* r*, and outputs the string* tag.
- *The owner-verification algorithm* $VerifyOwner$ *takes as inputs key* k *as well as strings* $itemId$*,* r *and* tag*. It then outputs a bit* b *with* $b = 1$ *meaning valid and* $b = 0$ *meaning invalid.*

It is required that for every n, every (k, r) generated by $Gen(1^n)$, and every $itemId$, it holds that $VerifyOwner_k(itemId, r, tag) = 1$, where tag is generated from $AssociateOwner_k(itemId, r)$.

To model after the scheme, we create a mint impersonation game. The intuition and inspiration behind it are briefly described as follows: prior to minting, a message authentication code tag, corresponding to the asset whose NFT is to be minted, would be first created using the key k belonging to the owner of the asset. The user is given the $itemId$ of the asset and a random value r. $itemId$ serves as a representation of the asset and the tag will serve as an irrevocable proof of the original owner. Both $itemId$ and tag will be included publicly in the blockchain during the minting process. After successful minting, the NFT should be safe from an adversary attempting to claim that they are the owner. Verification of the NFT would involve the adversary producing a tag and checking the validity of it. Logically, the tag must belong to an asset that has been minted. The difficulty lies in creating a valid tag without having knowledge of the key k and random value r.

More formally, we define the mint impersonation game with adversary A, a security parameter n and a mint scheme $\Pi = (Gen, AssociateOwner, VerifyOwner)$ as follows:

The mint impersonation game $Mint - Impersonate_{A,\Pi}(n)$**:**

1. The owner's key k and a random value r are generated by running $Gen(1^n)$.
2. The adversary A is given oracle access to $AssociateOwner_{k,r}(\cdot)$.
3. A can query its oracle $AssociateOwner_{k,r}$ with any string $itemId$ of its choice. With each query, the oracle will produce and return the value tag to A.
4. Oracle also stores the i^{th} tag associated with the $itemId$ in its i^{th} query in a set Q as a pair $< itemId, tag >$.
5. A is allowed to query its oracle $AssociateOwner_k$ as many times as it wants.
6. Eventually, A outputs the values $< itemId', r', tag' >$
7. A succeeds if, and only if:
    a. $VerifyOwner_k(itemId', r', tag') = 1$; and
    b. the pair $< itemId', tag' > \in Q$.
8. On success, the output of the experiment is defined to be 1. Else, the output is 0.

A Mint is non-impersonate-able if no efficient adversary can succeed in the above experiment with non-negligible probability.

Definition 1.2: *A mint scheme* $\Pi = (\text{Gen}, \text{AssociateOwner}, \text{VerifyOwner})$ *is non-impersonate-able if, and only if, for any probabilistic polynomial time adversary* A*, there exists a negligible function* $negl$ *such that:*

$$\Pr\left[Mint - Impersonate_{A,\Pi}(n) = 1\right] \leq negl(n)$$

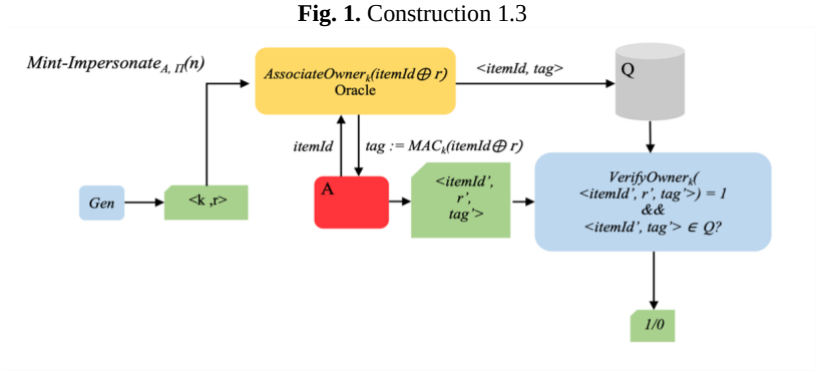
2.4 Constructing a secure Mint scheme

Our construction revolves around the usage of a secure MAC scheme. Intuitively, we can use a MAC to create the tags; if the MAC is secure, then it should not be possible for any efficient adversary to create another valid tag.

Construction 1.3: Let $\Pi' = (Gen, Mac, Vrfy)$ be a secure MAC scheme. We define a $Mint$ with security parameter n as follows:

- Gen: On input 1^n, run $Gen(1^n)$ to obtain (k, r), with key $k \in \{0,1\}^n$ and random value $r \in \{0,1\}^n$.
- $AssociateOwner$: On input $(itemId, r)$, output $tag := Mac_k(itemId \oplus r)$.
- $VerifyOwner$: On input $(itemId, r, tag)$, return $Vrfy_k(itemId \oplus r, tag)$.

Fig. 1. Construction 1.3



Theorem 1.4: If Π' is a secure MAC scheme, then Construction 1.3 is *non-impersonate-able*.

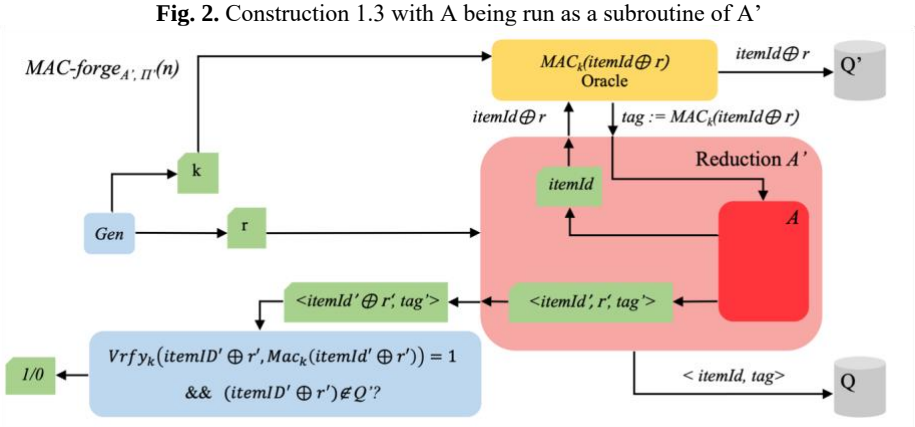
Proof: We seek to prove the above theorem by contradiction. Let Π be our construction. To aid our proof, we construct another adversary A' that seeks to create a valid forge on the MAC scheme Π'. A' runs A as a subroutine and works as follows:

1. Run $Gen(1^n)$ to create random value r and secret key k
2. A' is given r and has access to an oracle $Mac_k(\cdot)$
3. A' runs A and serves as its oracle $AssociateOwner$. A will query A' with $itemId$. On every query that A makes, answer it as follows:
4. Query $Mac_k(itemId \oplus r)$ and get tag back in return

5. Return $tag\ to\ A$
6. Eventually, A outputs $(itemId', r', tag')$
7. A' then outputs $(itemId' \oplus r', tag')$. The forge experiment on Π' outputs 1 if 1) $itemId' \oplus r'$ and tag' is valid, and 2) $itemId' \oplus r' \notin Q'$ where Q' represents the set of all messages queried on $Mac_k(\cdot)$

One can see that A' is emulating Π for A.

Fig. 2. Construction 1.3 with A being run as a subroutine of A'



By the law of total probability,

$$\begin{aligned}&\Pr\left[Mint - Impersonate_{A,\Pi}(n) = 1\right]\\ &= \Pr\left[Mint - Impersonate_{A,\Pi}(n) = 1 | r' = r\right] \cdot \Pr[r' = r]\\ &\qquad + \Pr\left[Mint - Impersonate_{A,\Pi}(n) = 1 | r' \neq r\right] \cdot \Pr[r' \neq r]\end{aligned}$$

Case 1: $r' = r$

From the definition of $Mint - Impersonate_{A,\Pi}$, the adversary will win with probability 1 if he can output $itemId'$, r' and tag' such that $r' = r$ and $< itemId', tag' > \in Q$. The second condition is trivial to achieve since A knows this knowledge.

Thus,

$$\Pr\left[Mint - Impersonate_{A,\Pi}(n) = 1 | r' = r\right] \cdot \Pr[r' = r] = \Pr[r' = r]$$

However, by the randomness of r,

$$\Pr[r' = r] = 2^{-n} \leq negl(n)$$

Case 2: $r' \neq r$

To evaluate $\Pr\left[Mint - Impersonate_{A,\Pi}(n) = 1 | r' \neq r\right]$, it suffices to reduce the evaluation to simply the probability of $Mac - forge_{A',\Pi'} = 1$. Logically, since $r' \neq r$ and if $itemId'$ has been seen before in one of the i^{th} queries from A, then it must be that A' outputs $itemId' \oplus r' \notin Q'$. Further, if $Mac_k(itemId' \oplus r')$ produces a valid tag, then both experiments Π and Π' output 1. Else, both output 0.

Hence,

$$\Pr\left[Mac - forge_{A',\Pi'}(n) = 1\right] = \Pr\left[Mint - Impersonate_{A,\Pi}(n) = 1 |\ r' \neq r\right]$$

Since Π' is a secure MAC scheme, then

$$\Pr\left[Mac - forge_{A',\Pi'}(n) = 1\right] \leq negl(n).$$

Thus,

$$\begin{aligned}&\Pr\left[Mint - Impersonate_{A,\Pi}(n) = 1 | r' \neq r\right] \cdot \Pr[r' \neq r] \\ &\leq \Pr\left[Mint - Impersonate_{A,\Pi}(n) = 1 | r' \neq r\right] \times 1 \\ &\leq\ negl(n)\end{aligned}$$

Having evaluated both cases, we can now determine the probability of success for A on Π.

$$\begin{aligned}&\Pr\left[Mint - Impersonate_{A,\Pi}(n) = 1\right] \\ &=\ \Pr\left[Mint - Impersonate_{A,\Pi}(n) = 1 | r' = r\right] \cdot \Pr[r' = r] \\ &\qquad\qquad +\ \Pr\left[Mint - Impersonate_{A,\Pi}(n) = 1 |\ r' \neq r\right] \cdot \Pr[r' \neq r] \\ &\leq\ negl_1(n) + negl_2(n) \\ &\leq negl(n)\end{aligned}$$

This completes the proof.

3 The Lottery Scheme

3.1 NFT Lotteries

On August 28th, an NFT-based project known as “Loot Project” was introduced to the public [10]. The project offered 8,000 NFTs known as loot bags, with each bag associated with a bunch of items of varying rarity. Anyone could simply claim these bags for free and only needed to pay the standard Ethereum gas fee associated with the minting of these bags. The process for minting a particular bag was as simple as inputting any user-chosen bag id into the system and paying the minting fee. If the bag had not been minted by someone else, the bag would be successfully minted and owned by the user who minted it. The owner can then check all the items in the minted bag, especially the rare items which are of high value. These items are hierarchically classed on the factors of value and rarity in the same manner as items in a game like Dungeons & Dragons, where an example of a rare item would be “Divine Robe of the Fox”.

The aforementioned project is akin to a simple number lottery system with predetermined winning numbers. Participants would then pick a random number and hope that it was a winning number. These lottery-based giveaways are widely used in generating hype and attracting users to new NFT based projects. With such a demand, it would be beneficial for such lottery-based systems to be designed securely. The

potentially high valuation of the prizes also serves as a further motivation in creating secure online lottery systems. Typically, the actual implementation of an offline lottery scheme is hidden from the public. While this may be attributed to security reasons, it also allows for the conducting authority to make unfair adjustments to the lottery at their own discretion. For example, according to Rule 8.3 of the Singapore Pools TOTO Game Rules, the company may adjust the prizes offered at any time [14]. Kuacharoen proposed an online lottery system that claims to prevent this issue but also acknowledges that it is not perfect as there is still an element of trust needed in its design [13]. Having a lottery system based on the blockchain circumvents this by allowing the public to always monitor the system and verify the correctness of it. Any discrepancies because of, but not limited to, duplicated awards, fraudulent numbers, and any unannounced modifications to any part of the lottery system, would be easily discovered. However, such transparent systems are vulnerable against malicious participants who will attempt to discover the winning numbers through any means necessary. If a participant can discover that a particular number has a slightly higher odds of winning than the other numbers, then he is considered to have an advantage and the lottery would be deemed unfair.

3.2 Lottery Scheme

We now define the scheme called $Lottery$. Taking inspiration from a simple real life lottery scheme where the winning numbers are predetermined, $Lottery$ consists of a string called $prize$ and each of its bits is analogous to a lottery number. If the i^{th} bit is a 1, then it is considered a winning bit and is analogous to a winning lottery number. Just like how any participant in a real lottery doesn't know which number is a winning number, we need to mask $prize$ to hide information about the winning bits. To do this, we only reveal a string called $hiddenPrize$, that is generated after masking $prize$, to the adversaries. An adversary's job is to guess the winning bits in $prize$ with only knowledge about $hiddenPrize$. With this, we now provide the full definition of our $Lottery$ scheme:

Definition 2.1 Lottery: *A $Lottery$ consists of 3 probabilistic polynomial-time algorithms (Gen, $GenPrize$, $HidePrize$) such that:*

- *The key-generation algorithm Gen takes as input the security parameter 1^n and outputs a key k with $|k| \geq n$.*
- *The prize-generation algorithm $GenPrize$ that takes as input a winning probability w such that $0 \leq w \leq 1$ and outputs a string $prize$ such that the fraction of 1s in $prize$ is equal to w.*
- *The prize-hiding algorithm $HidePrize$ that takes as input a key k and a string $prize$ and outputs a hidden-prize string $hiddenPrize$.*

For any string $hiddenPrize$, security of the lottery scheme means that an adversary shouldn't be able to guess the value of any i^{th} bit of $prize$ better than the initial winning probability w set for the lottery. This is analogous to real-life lotteries where

participants try to guess which numbers are the winning numbers. Ensuring that any adversary does not have such an advantage makes the lottery *fair*.

We now present the formal definition on the security property of *fairness* for a lottery scheme. Let $\Pi = (Gen, GenPrize, HidePrize)$, and consider the following experiment for an adversary A and parameters n, w:

The fair lottery game $Lottery - Fair_{A,\Pi}(n, w)$:

1. A key k is generated by running $Gen(1^n)$.
2. A string $prize$ is generated by running $GenPrize(w)$.
3. A string $hiddenPrize$ is generated by running $HidePrize(k, prize)$.
4. The adversary A is given $hiddenPrize$. The adversary eventually outputs i where i represents the i^{th} bit in $prize$ (indexed from 0).
5. A succeeds if and only if the guessed i^{th} bit in $prize$ is equal to 1. In event of success, the output of the experiment is defined to be 1.

A lottery is fair if no efficient adversary can succeed in the above experiment with non-negligible probability more than the wining probability w.

Definition 2.2: *A lottery is fair if, and only if, for any probabilistic polynomial time adversary, there exists a negligible function* $negl$ *such that:*

$$\Pr\left[Lottery - Fair_{A,\Pi}(n, w) = 1\right] \leq w + negl(n)$$

3.3 Constructing a Secure Lottery Scheme

At this juncture, a natural construction to use would be that of a one-time pad to mask $prize$. The indistinguishable property of a one-time pad encryption scheme implies that the ciphertext does not leak any information about individual bits of the plaintext. In our case, the $hiddenPrize$ is akin to a ciphertext and should not leak out any information about $prize$. Further, the function of a pseudorandom generator and the security it provides in a one-time pad encryption scheme would be useful in constructing our secure lottery scheme as well due to the similarities in both schemes.

Construction 2.3: Let G be a pseudorandom generator with expansion factor l. Define a $Lottery$ with security parameter n and a winning probability w as follows:

- Gen: On input 1^n, output a key $k \in \{0,1\}^n$
- $GenPrize$: On input w, output the string $prize \in \{0,1\}^{l(n)}$
- $HidePrize$: On input key $k \in \{0,1\}^n$ and $prize \in \{0,1\}^{l(n)}$, output the string $hiddenPrize = G(k) \oplus prize$.

Theorem 2.4: If G is a pseudorandom generator, then Construction 2.3 is *fair*.

Proof: Let Π denote the scheme in $Construction$ 1. We use a reduction approach to prove the fairness of Π; specifically, we construct a separate distinguishing game using an efficient distinguisher D that receives a string s and emulates the lottery game for an efficient adversary A. Let A be a probabilistic polynomial time adversary and let D be an efficient distinguisher. D emulates the lottery game for A in the following manner:

Distinguisher D:
D is given input 1^n, $s \in \{0,1\}^{l(n)}$ and string $prize \in \{0,1\}^{l(n)}$. D works as follows:

1. Create $hiddenPrize := s \oplus prize$.
2. Give $hiddenPrize$ to A and eventually receive i.
3. Output 1 if the i^{th} bit of $prize$ is equal to 1, else output 0.

D's ability to distinguish whether s was truly randomly generated or generated from a pseudorandom generator is directly related to A's ability to guess if the i^{th} bit of $prize$ equals to 1. If A can guess correctly with a non-negligible probability greater than the winning probability w, then it implies that A is somehow able to distinguish the origin of randomness for the string s. With this information, D should be able to likewise distinguish s with a non-negligible probability. This way, the security of the pseudorandom generator implies the fairness of Π.

Before analysing the probability of success of D in its distinguishing game, we construct a second game Π′ that is identical to Π except that $prize$ is XORed with a truly random string s of length $l(n)$ instead of the output from a pseudorandom generator to create $hiddenPrize$. In this game Π′, the probability of success for A is simply just the probability of winning w. To see why, consider the proof below:

i^{th} bit of $prize$	i^{th} bit of s	i^{th} bit of $hiddenPrize$ = i^{th} bit of $prize \oplus i^{th}$bit of s
0	0	0
1	0	1
0	1	1
1	1	0

Since s is randomly generated, then $\Pr[i^{th}\ bit\ of\ s = 0] = \Pr[i^{th} bit\ of\ s = 1] = 0.5$. For $prize$, $\Pr[i^{th} bit\ of\ prize = 1] = w$. Note that strings $prize$ and s are generated independently. Given the i^{th} bit of $hiddenPrize$ equals 1, the probability the i^{th} bit of $prize$ equals to 1 is:

$$\Pr[i^{th} bit\ of\ prize = 1 | i^{th} bit\ of\ hiddenPrize = 1]$$

$$= \frac{\Pr[\mathrm{i^{th}} bit\ of\ prize = 1 \cap i^{th} bit\ of\ hiddenPrize = 1]}{\Pr[i^{th} bit\ of\ hiddenPrize = 1]}$$

$$= \frac{w \times 0.5}{w \times 0.5 + (1-w) \times 0.5}$$
$$= w$$

A similar proof holds when the given i^{th} bit of $hiddenPrize$ equals to 0. In both cases, the probability of guessing whether the i^{th} bit of $prize$ equals to 1 is just w.

Now, if we observe the distinguishing game for D which runs A as a subroutine, we note that:

1. If the input s to D is a truly random string, then the probability of D outputting 1 in its own distinguishing game running A as a subroutine is equivalent to the probability of success of A in Π'. Thus, we have:

$$\Pr_{s \leftarrow \{0,1\}^{l(n)}}[D(s) = 1] = \Pr\left[Lottery - Fair_{A,\Pi'}(n, w) = 1\right] = w \quad (1)$$

2. Else if the string s provided to D is generated from a pseudorandom generator G on an input k, that is $s := G(k)$, then the probability of D outputting 1 in its own distinguishing game running A as a subroutine is equivalent to the probability of success of A in Π. Thus, we have:

$$\Pr_{k \leftarrow \{0,1\}^n}\left[D\left(G(k)\right) = 1\right] = \Pr\left[Lottery - Fair_{A,\Pi}(n, w) = 1\right] \quad (2)$$

Since G is a pseudorandom generator, then there must be a negligible function $negl$ such that:

$$\left| \Pr_{s \leftarrow \{0,1\}^{l(n)}}[D(s) = 1] - \Pr_{k \leftarrow \{0,1\}^n}\left[D\left(G(k)\right) = 1\right] \right| \leq negl(n)$$

Together with equations (1) and (2), we get:

$$|w - \Pr\left[Lottery - Fair_{A,\Pi}(n, w) = 1\right]| \leq negl(n)$$
$$\Rightarrow \Pr\left[Lottery - Fair_{A,\Pi}(n, w) = 1\right] \leq w + negl(n)$$

This completes the proof.

Further discussion. We acknowledge that our proposed $Lottery$ scheme and its security definition of *fairness* are not applicable to all varying lottery systems in the world. Other conventional and popular lotteries include Toto and Lotto, which do not fit the mould of our defined $Lottery$ scheme. These separate lotteries will require their own lottery schemes and security definitions which we do not go into.

4 Conclusion

We have investigated two broad security implementations that underpin the implementations used in minting NFTs on third-party marketplaces: an impersonation

scheme and a lottery scheme. By devising the above schemes, constructions, and proofs, we demonstrated how analysts can qualify the security of such NFT-based implementations based on the qualities of whether an implemented scheme can be impersonated (*impersonate-able*), and whether the scheme implemented is *fair*.

NFTs have been hailed as a salient tool for wealth accumulation in the new age. It is a matter of time before the adoption of NFTs becomes mainstream. However, NFT implementations made by third-party marketplace service-providers in other areas beyond minting, such as transactions, fee deductions, and bidding, must continue to be hardened to minimize the risk of such attacks negatively impacting a burgeoning user base. This could result in devastating monetary losses to a significant portion of NFT-users. For NFTs to continue being reputable and embraced in accumulating wealth, the increasingly vital and relevant security qualities that make up NFTs must be thoroughly explored and refined.

References

1. Ante, Lennart: The non-fungible token (NFT) market and its relationship with Bitcoin and Ethereum (2021). https://ssrn.com/abstract=3861106.
2. Bonderud, D.: *Token Resistance: Tackling the New NFT Threat Landscape*. Security Intelligence (2021). https://securityintelligence.com/articles/new-threat-landscape-nfts/, last accessed 2 November 2021.
3. Crow, K., *Scammers and Hackers See New Frontier in NFT Art*. The Wall Street Journal (2021). https://www.wsj.com/articles/scammers-see-new-frontier-in-nft-art-11629896400, last accessed 2 November 2021.
4. Daniele, D.: *NFTs' Nifty Copyright Issues*. Mondaq (2021). https://www.mondaq.com/canada/copyright/1067734/nfts39-nifty-copyright-issues-, last accessed 30 October 2021.
5. Daubenschütz, T.: *What is sleepminting and will it ruin NFT provenance?* (2021). https://timdaub.github.io/2021/04/22/nft-sleepminting-beeple-provenance/, last accessed 3 November 2021.
6. Entriken W., Shirley, D., Evans, J., Sachs, N.: *EIP-721: Non-Fungible token standard. Ethereum Improvement Proposals (2018)*. https://eips.ethereum.org/EIPS/eip-721, last accessed 3 November 2021.
7. Fairfield, J.: *Tokenized: The law of non-fungible tokens and Unique Digital Property*. SSRN (2021). https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3821102, last accessed 3 November 2021.
8. Finlow-Bates, K.: *How to sleepmint NFT tokens*. Medium (2021). https://kf106.medium.com/how-to-sleepmint-nft-tokens-bc347dc148f2, last accessed 3 November 2021.
9. Frankenfield, J.: Fungibility: *When interchangeability matters*. Investopedia (2021). https://www.investopedia.com/terms/f/fungibility.asp#:~:text=Fungibility%20is%20the%20ability%20of,equal%20value%20between%20the%20assets, last accessed 30 October 2021.
10. Hoffman, D.: Loot- randomized adventurer gear- no images or stats. intentionally omitted for others to interpret- no fee, just gas- 8000 bags totalopensea: Https://t.co/qsnrj1fd0netherscan: Https://t.co/bf9p0rshx2available via contract only. not audited. mint at your own risk pic.twitter.com/ulukzfayuk. Twitter (2021).

https://twitter.com/dhof/status/1431316631934967815?s=20, last accessed 2 November 2021.

11. Howcroft, E.: *NFT sales surge to $10.7 bln in Q3 as Crypto Asset Frenzy hits New highs*. Reuters (2021). https://www.reuters.com/technology/nft-sales-surge-107-bln-q3-crypto-asset-frenzy-hits-new-highs-2021-10-04/, last accessed 30 October 2021
12. Mashayekhi, R.: *Where the smart money in crypto investing is going next*. Fortune (2021). https://fortune.com/2021/07/29/crypto-investing-where-smart-money-going-next-andreessen-horowitz-nfts-cathie-wood-ark/, last accessed 2 November 2021.
13. Kuacharoen P.: *Design and Implementation of a Secure Online Lottery System*. In: Papasratorn B., Charoenkitkarn N., Lavangnananda K., Chutimaskul W., Vanijja V. (eds) Advances in Information Technology. IAIT 2012. Communications in Computer and Information Science, vol 344. Springer, Berlin, Heidelberg (2012).
14. Toto Game Rules (general). *TOTO Game Rules (General) | Singapore Pools.* (2021). https://www.singaporepools.com.sg/en/rules/pages/toto-game-rules-general.html, last accessed 2 November 2021.