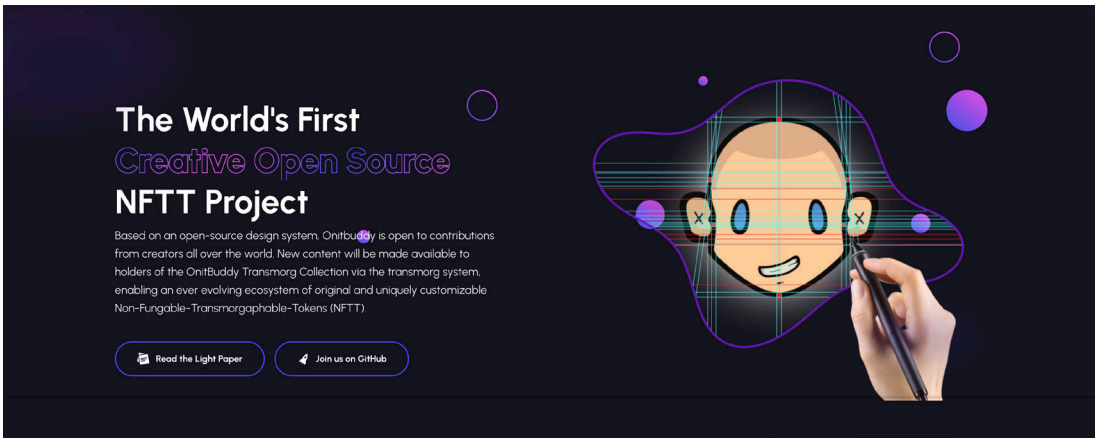
ONITBUDDY

+ + +

++

Lightpaper



OnitBuddy

PFP Project to produce the world's first Non-Fungible Transmorg Token (NFTT)

Join our Discord Community

https://discord.gg/RWs3DP2

Contribute on GitHub

For Developers:
github.com/onitbuddy/transmorgprotocol

For Designers:
github.com/onitbuddy/OnitDesignSystem

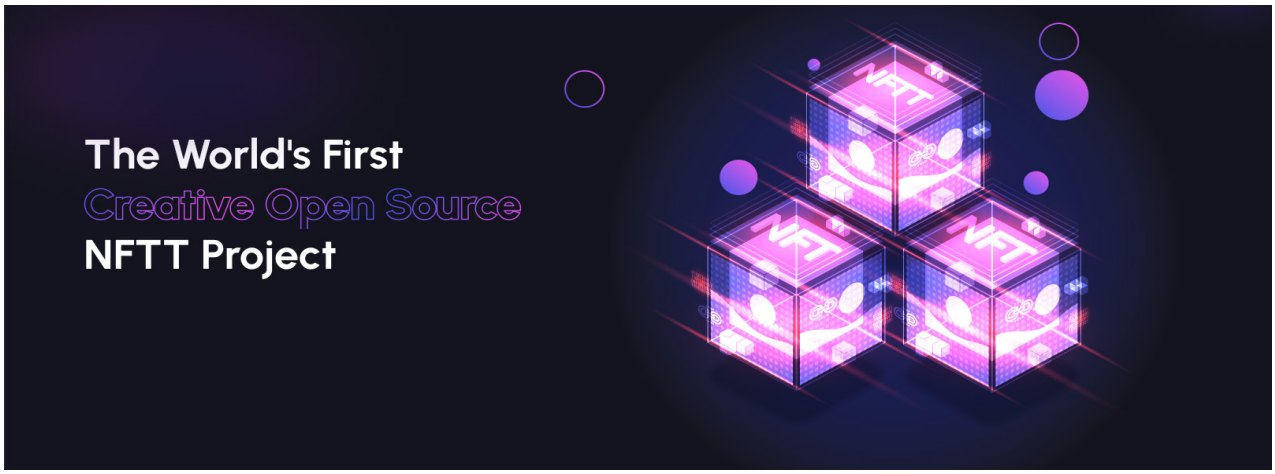
Online

Mint & Presale:
onitbuddy.com/mint

Website:
onitbuddy.com

Table of Content
Lightpaper

Overview
Using NFTs to Evolve the NFT-Space



Any act of creativity is an exercise in derivative originality.

We use technology to create more technology, music to inspire music and art to create art. Our greatest societal, technological and creative achievements have been enabled by our ability to build on each other's ideas, and use inspiration to evolve entire genres of movies, music and all forms of artistic expression.

While the NFT space adequately delivers a decentralized transaction database, it is lacking a platform that enables, tracks and protects this very way in which creators and artists combine IP to create original works.

OnitBuddy is an NFT project that intends to change that. Its PFP collection of 8080 unique, hand-crafted character illustrations is based on an open-source design system that encourages others to contribute and evolve each NFT, via a secondary market of component-based contributions. The collection‘s pre- and post-sale will then support the development of a new kind of smart contract that governs how IP gets captured, combined and rewarded across the NFT space.

These components are made available via a new ‘transmorg smart contract’ that is capable of combining two existing NFTs to augment their visual aesthetic, while protecting and enhancing its holder’s investment and their creators’ commissions.

Project Outline
Two Problems in Digital Art

> Art, as with most endeavors of human creativity and ingenuity, is not valued based on production-based variables. A piece of music, game or art is not valued based on the materials or time it took to produce the work.
>
> Instead, their value derives from its novelty and originality.

Acquiring an original idea, brand or canvas is a function, not of the material's value, but the one true expression of a unique act of creation that resulted in a piece so novel, so unique, that it has never been seen before. Art commerce is more about buying and selling an artist's original way of approaching their work, and the material manifestation of these ideas than it is about technique, skill or visual appeal. Owning an idea is only worth something if a certified record of its purchase irrefutably proves the buyer's owning privilege.

The NFT space has successfully provided a credible record to certify the transactions of digital art, but lacks two major features:

1. NFT art in its current form needlessly continues to mirror the static and immutable properties its material counterpart is limited by. This inability is producing pump-and-dump patterns, as there is little incentive to invest in a digital piece of art beyond speculative gains.

2. Although digital art is much easier to copy, the NFT space has not provided adequate mechanisms to help creators not just protect but instead embrace derivative or additive works, due to inadequate attribution, commission and profit sharing.

This unnecessarily limits the use-cases NFTs can be considered for. Rather than protecting IP-ownership in a single-file, immutable format, a new approach should permit the organic evolution, combination of NFT, regardless of whether its IP reflects an artistic, technical or otherwise relevant intellectual or creative expression worth protecting.

The Solution
The Non Fungible Transmorg Token (NFTT)

The NFTT intends to address the above challenges by enabling the combination of two NFTs through the introduction of an authenticity-chain and smart contract.

ONITBUDDY is the world's first NFTT project and platform, designed to bring this innovation to life.

To accomplish this, the talented team and contributors of the OnitBuddy project intend to deliver an NFTT protocol that delivers the following capabilities:

2.1 Easy combination of two or more existing NFTs into one new NFT.

2.2 Adequate value and authorship protection.

2.3 Equitable commission sharing amongst NFT creators wishing to add to each other's project to keep the art/ IP fresh, original and ever evolving.

OnitBuddy's PFP collection intends to support the development of this new 'transmorg protocol' capable of changing the way creators contribute to their and each other's projects.

It is our intent for NFTTs to serve as a transformative catalyst to stimulate a paradigm shift in how any creative endeavor is created, built and traded in an open, decentralized marketplace.

The following document contextualizes this goal by providing a comprehensive treatment of this project's underlying macro dynamics, market gaps, corresponding opportunities, value proposition and technical execution.

Market Dynamics

If the picture changes, is the art worth less?

‘Why can’t I just screenshot an NFT?’ The answer to that question goes back to the way humanity has valued art since the world’s most polite, yet game-changing, fist-fight at a Sutherby’s auction in 1973.

Robert Rauschenberg shoves Scull at Sutherby’s auction in New York.

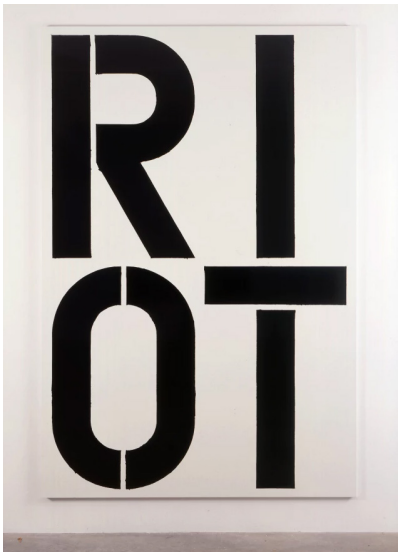
It was the first time living artists sold for large sums of money. Robert Rauschenberg, an artist who had sold his work for $900 to Robert Scull, felt ripped off and aggressively shoved Scull, the organizer, who auctioned the work for $85,000. “I have been working my ass off for you to make all that profit?!”, he shouted at Scull. Scull turned, erupted in laughter and responded: “[But] how about yours? The ones you’re gonna sell now?” (see video).

As soon as Scull’s point sunk in, Robert’s aggressive stance softened. The men hugged, both chuckling at the realization that this conversation, this moment, had changed the artworld forever. Not only did it explode art-pricing to ever increasing heights, it also marked the start of an era where living artists who continued to create more supply of their own art as time progressed were celebrated. It was this time in history that artists like Warhol, amassed a level of celebrity equal if not exceeding the celebrity of the famous personas he captured in his works.

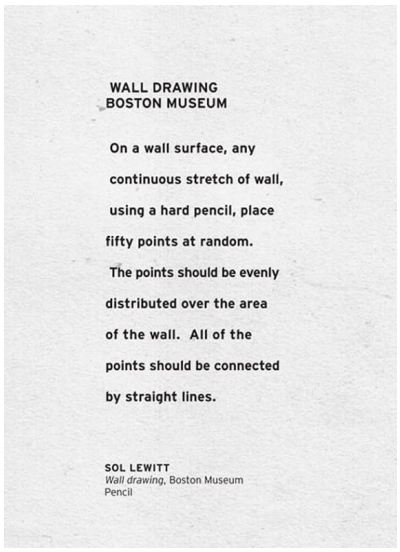
It was at that point that the art itself became more about the artist’s celebrity, their way of expressing themselves and the novel ideas behind their works. This untethered an artist’s work from other benchmarks of value. Rather than the cost of materials to create the art, or the time it took to create it, the value of art became a proxy for valuing the novelty of the idea that brought it to life. And while art critics continue to debate the finer details of how art ought to be valued, this general understanding has led the value of art to move uniformly in one direction: Up!

Since then, the market for art has completely exploded. And so has the idea of what art is.

Picture of Red Yellow Looming by Jenny Holzner.



Picture of Red Yellow Looming by Jenny Holzner.



Wall Drawing instructions, Sol Lewitt

Red Yellow Looming by Jenny Holzner, a collection of text phrases shown in public places, was sold for $583,000. One of Christopher Wool's pieces resembling a large word-document with the word RIOT stenciled in black letters on white paper auctioned at Sutherby's for $29.9M.

Even instructions to make the art itself, called 'Instructions' by Sol LeWitt sold for $437,000.

The art itself is not measured by the complexity of its artistic expression but rather the idea it expresses. One of the things LeWitt is famous for are wall drawings which are site-specific, painted and drawn on site. When valuing LeWitt's 'Instructions', instead of looking at what appears to be nothing but a poor copy of a dusty patent application, the work embodies how LeWitt saw 'originality' and how he wanted for his art to exist. In publishing his instructions, he disconnected the very way his art is created from 'the hand of the artist'.

LeWitt wanted for others to create derivatives and evolve his artform, keeping it fresh and novel through the many contributions of dedicated inviduals.

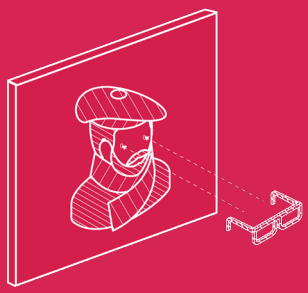
His instructions still exist, although he does not, and yet his works can be created anew and the resultant drawings can be attributed to him now as they ever were. The value of his work, is therefore not derived from its physical manifestation of 'paint being arranged in a certain way on a wall' but from the concept of transcendent authenticity and originality it embodies.

`Derivative art' became an entire artistic subgenre featuring so called 'appropriation artists' such as Elaine Sturtevant whose entire body of work comprises of copies of other artists work. Warhol even gave her the silkscreen he used for his work so that she could make appropriations of his work.

This notion confirms that actual drawing on the wall is not the artwork. The reason people buy art at such astronomical prices is that its physical manifestation, in whatever form, permits a person to own the concept behind it. What they own is not the actual drawing on the wall. They own the fact that they bought it. The artwork's value is its certificate. Without the certificate, the artwork has no value.

Its physical manifestation, what it looks like, isn't material. Art is valued based on the Intellectual Property it stands for. Physical art only exists in the same shape it was conceived in, only due to the immutability of our physical world. But digital art isn't subject to the same limitations. This realization opens up tremendous opportunities for rethinking the form, shape and dynamic manifestations of digital creations. Similar to LeWitt's work they can evolve through the many contributions of a dedicated community.

Market Gaps
Digital art doesn‘t need to be static!



Let‘s buy your art designer shades!

The blockchain has delivered an effective mechanism to certify true ownership of digital art in ways that weren’t possible before. However, there are three major differences between NFTs and the traditional artworld. The blockchain, smart contracts and the dynamic properties of a digital rather than static asset.

Two are well understood and used. The third remains underutilized.

The first difference is ‘who decides what art is worth’. Similar to how the global economy believes that the value of a cup of coffee sits between 3 to 8 dollar-bills that were issued by a government authority, an NFT’s value is based on the shared belief that the blockchain represents a singular authoritative database to determine authenticity and its chain of ownership.

The second difference is an NFT’s ability to leverage smart contracts to introduce perks, allowing the NFT to serve as tickets to special events, or membership to a club etc.

The third is the dynamic nature of digital assets. Opposed to their traditional counterparts, digital assets don’t have to be rigidly static.

While it is difficult to change a wall-painting without destroying it, NFTs can have the luxury to change their look without losing their link to the certificate of authenticity that is stored on a decentralized blockchain.

They could even be reverted to their original state at the click of a button if necessary. Instead of subscribing to the pre-modern idea that art must remain unchanged once published because it derives its value from its connection to the artist itself, NFTs can take a page from LeWitt’s instructions.

Not only doesn’t their value have to be connected to the hand of the artist (most digital artists are anonymous anyways) NFTs can evolve because they derive their value from what makes them original, not the way pixels were arranged on a jpeg at the moment it was minted.

NFTs are digital and the decentralized nature of its underlying blockchain technology gives rise to a new opportunity the NFT and artworld has yet to embrace:

Complementing its digital transferability with its inherently digital flexibility to evolve, change, transform and morph.

Opportunity
NFTT – Non Fungible Transmorg Token

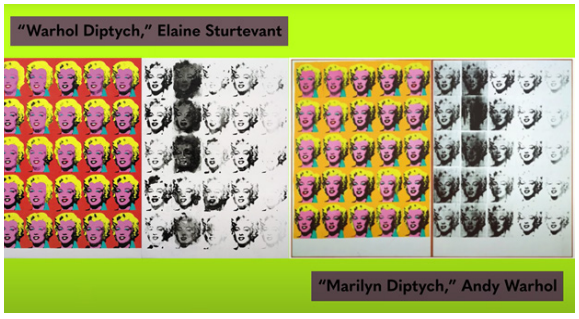
This is where OnitBuddy as the world's first NFTT, Non Fungible Transmorgrable Token comes in.

Transmorgriphication or transmorg is an expression borrowed from the world of video games. It provides players with the ability to augment and replace the appearance of your in-game items such as armor or weapons by combining them with other items.

A transmorgrable NFT is an NFT whose appearance and contents can be augmented by combining it with another NFT using a so called Transmorg Protocol. This Protocol evaluates and combines creative assets along pre-defined attributes.

This gives rise to a never-before-seen lifecycle that tracks, and accounts for the creative contributions to existing projects by passionate creators. This transmorg-protocol, not only allows an NFT to change but has the potential to solve a problem fundamental to any and all forms of human creative expression in the process. This creates a system that permits intellectual property and to be be built on to of each other by tracking an infallible chain of authorship.

Not only are most NFTs relegated to looking the same from the moment they are issued, its potential for capturing a uniform chain of authenticity and preventing IP theft is largely under-utilized.



As we have seen, it is an artwork's natural inclination to be changed by a community of contributors, to morph, to be added to and subtracted from to evolve into something new.

With traditional art, modifying the work restarts its chain of authenticity without attributing its origins or commissions appropriately. When Elaine Sturtevant, a derivative artist, sold her 'Warhol Diptych', an appropriation of Warhol's 'Marilyn Diptych', Warhol didn't see a cent of the sale.

In a more modern example, there is no mechanism at the moment that prevents a user from appropriating a stock vector image and using it as a component in their NFT collection, without recognizing the value the 'pair of glasses' or 'hat' contributed to their artwork.

OnitBuddy is an NFT project designed to fund and support the development of the NFTT protocol. This aims to not only enable the art-world but expand the utilization of NFTs by leveraging its capability, to capture, combine and evolve creative contributions to create new IP, whatever form it may take.

Value Proposition
Open Source that Pays Creators

Non Fungible Transmorgrable Tokens (NFTT) are defined as an add-on system designed to solve the problem of IP-attribution and reward distribution for creators wanting to use, build on and contribute to each other's work.

6.1 Value retention and protection

It intends to permit NFTs to be augmented, integrated, modified and evolved while securing its value by creating an infallible authenticity-chain that retains the creation's value.

6.2 Governing IP : The artistic use case

Based on a portfolio of smart contracts, it delivers a governance mechanism that allows any creative work to be opened up to creative contributions, and use those contributions to change the art's appearance, without breaking the chain of authenticity and commission attribution each contributor deserves.

Imagine a world where owners get to add aesthetic ornaments, shaders, colors or lighting to their NFTs, and spruce up an existing accessory with a new look.

6.3 Expanding the NFT Market

This and more is possible with the addition of the 'transmorg protocol', enabling NFTTs to expand the existing NFT-market via two additional secondary markets:

1. Primary market for publishing, selling and buying NFTs.

2. Secondary market for publishing, selling and buying NFT augmentations that change an existing NFT's look, perks and feel.

3. An attribution system that permits the integration and attribution of existing NFT's smart contract attributes into new works of art, while capturing, securing and disseminating the chain of authenticity, commission rewards and attribution credentials.

The transmorg protocol constitutes a new mechanism to expand the block-chain's decentralized consensus mechanism to enable any project capable of benefiting from the contributions of smart and engaged individuals to do so.

Goals and Objectives
Transmorg Smart Contract

Transmorgraphy describes the practice of combining two NFTs to create one new on-chain token, which is generated and deployed alongside a transmorg smart contract designed to protect the authenticity, scarcity and value of its originating token. These smart contracts endeavor to accomplish the following objectives:

7.1 Track the combination of two existing NFTs.

Generating an authenticity chain and audit trail capable of tracking the combination of two previously separate parent NFTs, we call them NFTx and NFTy, into one new NFT-Omega.

7.2 Deployment of smart-contract to integrate and govern attribution and commission configurations.

Deploying an intermediate smart contract capable of facilitating the integration of NFTx and NFTy smart contracts, their attribution and commission configurations into a single proxy contact capable of implementing a keep storage – keep address paradigm (see details below).

7.3 Enforcing collection constraints to protect NFT value.

Capturing and enforcing collection constraints to protect the value of both parent NFTs (NFTx and NFTy) including attribute-rarity, collection limitations, token compatibility, NFT and meta-data requirements.

7.4 Protect authorship of original NFTs that get combined.

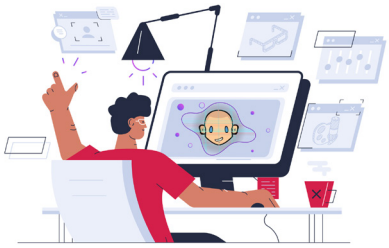
Effectively protect the authorship and scarcity of NFTx and NFTy's collections that are to be transmorged/ combined with each other.

7.5 Create visual combination of NFTx and NFTy into a new NFT.

Automatically generate a new visual representation of NFTx based on both NFTy's visual properties.

User Experience
The Transmorg Process

The above objectives intend to solicit the below user experience. The following represents an illustrative example based on the OnitBuddy collection:

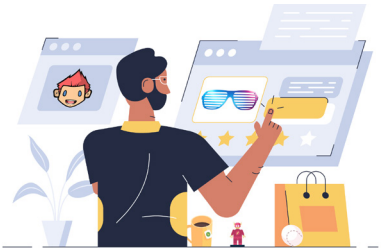


Utilizing the open sourced OnitBuddy design system, a creator decides to participate in the project.

They do so by creating a series of visual augmentations for existing traits, such as appealing new skins for existing glasses or hats. These augmentations are called 'trasmorgs' and are collection-specific.

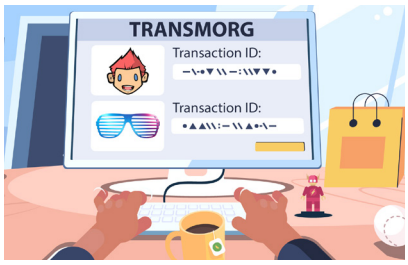
These creations are submitted and approved for publication by the project's governing body. In OnitBuddy's case this is a DAO.

Once approved, each transmorg is made available publicly as an NFT in its own right. Their creator also determines its selling price and commission percentage.



The next day, an existing OnitBuddy holder discovers the new sunglasses NFT and wants to add them to their OnitBuddy NFT. They proceed to purchase them.

The holder subsequently visits the Transmorg website to access the transmorg-interface.



They enter two transaction ID's:
(1) The transaction ID for when they obtained the OnitBuddy NFT.
(2) The transaction ID for the transmorg NFT.

This triggers the transmorg-process, It augments the owner's OnitBuddy NFT by adding the new sunglasses.

In the background, the transmorg protocol kicks off six procedures to facilitate this process. These are explained in detail in the next section.

Transmorg Smart Contracts

A portfolio of smart contracts automatically govern the following transaction processes:

8.1 Transfer of NFTs to Transmorg Smart Contract

Both the original OnitBuddy NFT and the sunglasses transmorg NFT are transferred to the transmorg smart contract and held in escrow. Both parent NFTs are also removed from relevant exchanges. This change is permanent unless the owner decides to roll back the NFT to before the transmorg, which is possible.

8.2 New NFT and its corresponding smart contract is deployed

In exchange a new NFT, combining the properties of its parent NFTs is created and transferred to the owner.

8.3 Authenticity chain: The Transmorg Hash

During this process, the newly created NFT receives a transmorg hash: A cryptographically encoded combination of its parents' unique identifiers. This serves as an additional mechanism to track the NFTs' chain of authenticity prior to and after the transmorgification process.

8.4 Protect authorship of original NFTs that get combined

Simultaneously the transmorg smart contract is deployed, which serves as a proxy-contract through which any future transactions of the newly created NFT will be handled.

8.5 Creator commission structures are respected and paid via transmorg smart contract

The transmornb smart contract ensures that further transactions respect and enforce both commission structures and perks. Commission percentages for both NFTs are paid out to their respective creators.

Milestones
Roadmap

While technical implementation details are provided below, the following roadmap is a living and breathing document that summarizes the delivery-milestones the OnitBuddy team will drive towards:

2022

December 23rd - 2022

OnitBuddy Origin Collection: Pre-Sale Commences

Presale and whitelisting commences on Discord channel.

February 20th - 2023

OnitBuddy Origin Collection Launches

OnitBuddy‘s main release makes the Origin collection available for trading on public exchanges.

Funding Goal 20% of NFT Supply sold

OnitBuddy Open Source Design System launches:

OnitBuddy‘s Open Source Design System starts accepting community contributions.

Funding Goal 60% of NFT Supply sold

NFTT Transmorg Protocol V1 Release

The first version of the NFTT Transmorg Protocol is released and available for deployment with NFT projects. This V1 will be capable of completing the transorg process.

Funding Goal 70% of NFT Supply sold

Onit Transmorg Collection Launches

The Onit Transmorg Collections is the first NFTT collection capable of integrating the dedicated contributions of OnitBuddy‘s transmorg creators is made available.

Funding Goal 100% of NFT Supply sold

Transmorg Portal V1

The Transmorg portal complements the abovementioned Transmorg Protocol by delivering a UI-enabled platform permitting users to combine NFTs via the NFTT Transmorg Protocol.

0014

Technical Implementation Details

The following pages outline a concrete proposal for the technical implementation of the transmorg system. It intends to propose and provide guidance for the NFTT-project‘s technical execution. This includes an auditable and non-fungible authenticity chain that protects the NFT’s authorship, originality, interests, scarcity and value of all relevant parties, including NFT creators and holders.

Non-Fungible Authenticity Chain and Proxy Contracts

The NFTT systems introduces an auditable and non-fungible authenticity chain called the Transmorg Hash. It protects the NFT’s authorship, originality, interests, scarcity and value for all relevant parties and respect creator-entitlements and commission structures via a portfolio of Smart Transmorg Contracts.

While the technical implementation of this process is explained in further detail below, it is enabled by introducing two well established technologies already subject to wide adoption in the crypto space:

1. The Transmorg Hash: Feistel Network block encryption algorithm.

2. Transmorg Smart Contract: Automated deployment of Proxy Contracts.

The Transmorg Hash
Using Feistel Networks to generate authenticity chains

An infallible chain of authenticity is created by default through the chain of records secured on the blockchain. However, in order to minimize unnecessary compute, and/or transaction fees, the NFTT system generates a so called 'transmorg hash' from which its parent NFTs and governing smart contract addresses can be inferred.

While based on a cryptographic algorithm, the transmorg hash is not a 'hash' in the classical sense. It is intentionally designed to be reversible so that it can serve as an information-repository from which anyone can infer its input variables.

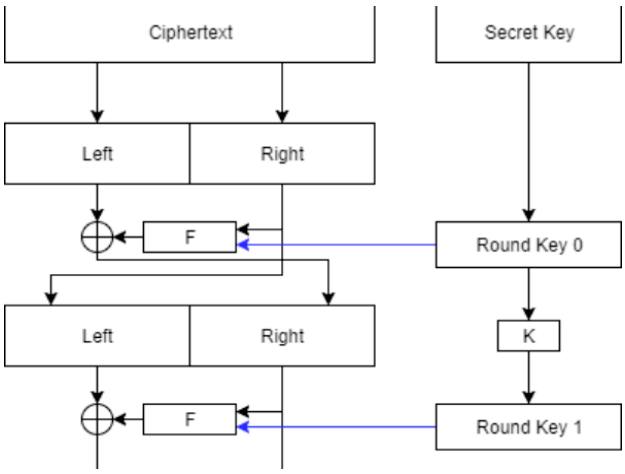
The 'Feistel Network' serves as the underlying cryptographic algorithm enabling both: the encoding and reversible decoding of an NFTTs transmorg hash.

A Feistel Network is a cryptographic technique used in the construction of block cipher-based algorithms and mechanisms. Designed by IBM employees Horst Feistel and Don Coppersmith.

A Feistel Network implements a series of iterative ciphers on a block of data and is generally designed for block ciphers that encrypt large quantities of data. A Feistel network works by splitting the data block into two equal pieces and applying encryption in multiple rounds. Each round implements permutation and combinations derived from the primary function or key. The number of rounds varies for each cipher that implements a Feistel Network. In the case of the NFTT, the number of rounds is determined by the number of parent NFTs (NFTx & NFTy) that get combined into one new NFT (NFT-Omega).

Its most significant property however, which lends this algorythm to the creation of the transmorg hash, is that it is a reversible algorithm. A Feistel Network produces the same output so long as the input is the same.

A Feistel cipher is a multi-round cipher that divides the current internal state of the ciphertext into two parts and operates only on a single part in each round of encryption or decryption. Between rounds, the left and right sides of the internal states switch sides. The image below shows a notional Feistel cipher with two rounds.



Feistel Netgwork Illustration

The left side of the image shows the encryption of the plaintext to the ciphertext and the right side shows the creation of round keys via a key scheduling algorithm, K.

9.1 Feistel Network: Inputs in the context of the NFTT

In the case of NFTTs, a collection of NFT's smart contract addresses serve as the input variables for the Feistel Network:

Input Variables	Feistel Network Attributes
1. The smart contract address for the purchase transferring ownership of the original NFTx to its current holder.	These values are concatenated to make up the cyphertext.
2. The smart contract address for the creation of the transmorg NFTy, owned by the creator (i.e. new sunglasses skin for OnitBuddy NFTx)	
3. The address of the new NFT's governing transmorg smart contract	'Round Key 0'

The function used for the encryption of the first round should be defined as a generating function that considers a randomly chosen, yet persistent secret key as its input.

If the function is used to generate a uniquely identifiable combination of the two parent-NFTs (NFTx and NFTy) only two rounds of Feistel encryption should be applied.

This algorithmic treatment of the input variables is further extensible to accommodate larger numbers of parent NFTs, in which case more than 2 NFTs be combined. With every additional variable an additional two rounds are added.

9.2 Encryption in a Feistel Network – The process

The following outlines the steps for encrypting using an example Feistel Network:

1. Alice and Bob exchange a secret key, S, through a secure channel.

2. Alice selects a plaintext, P, to send to Bob and breaks it into blocks of the length that the cipher accepts. For each block, the following steps are followed:

3. Alice splits the plaintext into a left piece and a right piece, L_0 and R_0.

4. Alice sets the value of the 'Round Key 0' to the initial secret key. That is, $RK_0 = S$. Here, RK stands for 'Round Key'.

5. The left side of round i+1 is set as the right side of round i. That is, $L_{i+1} = R_i$

6. Alice evaluates the function $F(R_i, RK_i)$, where R_i is the right side of the current round. The result is exclusive-ored with L_i and stored as the right side of round i+1. That is, $R_{i+1} = L_i \oplus F(R_i, RK_i)$

7. If not the last round, Alice evaluates the function $K(RK_i)$ and stores the result as round key i+1. That is, $RK_{i+1} = K(RK_i)$.

8. Alice repeats steps 5-7 for n rounds (one in the case of the diagram above and two in the case of the transmorg protocol)

9. L_n and R_n are combined to create the ciphertext block for this plaintext block.

10. All ciphertext blocks are combined to a single ciphertext C. Alice sends the result C to Bob.

In the next couple of sections, we will talk about the encryption part (steps 5 and 6) and the key scheduling part (step 7) of a Feistel structure in more detail.

9.3 The Encryption Phase

A video tutorial illustrating Feistel En- and Decoding is available:

- Encoding (see here).

As shown, the plaintext is split into two pieces. The right piece of one round becomes the left piece of the next. The left piece is exclusive ordered with the result of performing the function F on the right piece. The result of this is placed

on the right side for the next round. This means that the two pieces of the original plaintext will be transformed in alternate rounds (since whatever is on the right side during a round moves on to the next round unchanged).

The F function is what makes a Feistel cipher unique from other Feistel ciphers. The F function is where the „encryption“ happens and its security is vital to the security of the cipher.

For example, an F function that completely discards the round key input and operates only on the plaintext can be trivially broken since all an attacker has to do is guess the plaintext and confirm that the ciphertext matches.

9.4 Key Scheduling

Feistel ciphers also have what is called a key schedule that acts as an input to each round of the cipher. In the classical Feistel Network, there are two possible options for a key schedule.

The first is that the key for each round of the cipher (or „Round Key“) is included in the secret key shared by the two parties (resulting in a long shared secret key).

The other option is that the shared secret key is used as an input to a „key expansion“ function (shown in the diagram as K), which performs some operation on the previous Round Key or the current internal state of the key generation algorithm to generate the next Round Key.

In the context of the NFTT, it is the address of the new transmorg smart-contract that governs the newly created NFT-Omega.

The K function transforms the initial secret key into Round Keys for each round of encryption. This function must be selected to maintain the key space of the encryption function. If a 64-bit secret key is required, but at some point the effective key space shrinks to 32 bits, then an attacker only has to search a space of 32-bit keys in order to decrypt the ciphertext.

9.5 Decryption in a Feistel Network

A video tutorial illustrating Feistel En- and Decoding is available here:

- Decoding (see here).

The major benefit of the Feistel Network is that the same structure can be used for encryption and decryption. For a ciphertext encrypted with the Feistel Network shown in the diagram, we can use the exact same structure to decrypt. The only difference is that, in decryption, we use the Round Keys in reverse.

The steps for decryption in a Feistel Network are as follows:

1. Alice and Bob exchange a secret key, S, through a secure channel and Alice sends Bob a ciphertext, C.

2. Bob calculates the Round Keys for all rounds using the key scheduling functions K, i.e. $RK_{i+1} = K(RK_i)$

3. Bob breaks C into blocks of the length that the cipher accepts. For each block, the following steps are followed.

4. Bob splits the ciphertext block into a left piece and a right piece. These are L_n and R_n.

5. The right side of round i is simply the left side of round i+1. That is, $R_i = L_{i+1}$

6. L_i is calculated as follows. $L_i = R_{i+1} \oplus F(R_i, RK_i)$. This works because the xor-function ($\oplus$) has the property that $A = B \oplus C$ implies $B = A \oplus C$. More details below.

7. Alice repeats steps 5-6 for n rounds (one in the case of the diagram above and two in the case of the transmorg protocol).

8. L_0 and R_0 are combined to create the plaintext block for this ciphertext block.

9. All plaintext blocks are combined to a single plaintext P !

9.6 Why decryption works

It may seem odd that the same operation can be used to perform and undo itself. To understand this, we need to take another look at the diagram of the Feistel cipher shown above.

1. Seeing how the right half undoes itself is easy, all that happens is that something switches sides.

2. Undoing the left half depends on two crucial things:

a. First, we have all the information that was available to the function F during the encryption phase, i.e. R_i and RK_i.

b. Second, the xor-function ($\oplus$) has the property that $B \oplus B = 0$ for all values of B. This implies that if $A = B \oplus C$, then $A \oplus C = B \oplus C \oplus C = B \oplus 0 = B$. That is, the xor-function, $A = B \oplus C$ implies $A \oplus C = B$. Hence, we have $L_i = R_{i+1} \oplus F(R_i, RK_i)$.

Notice that this will only work for an operation that is its own inverse (like exclusive-or), which is why Feistel Networks always use exclusive-or for changing the transformed half of the state in one round.

9.7 Making strong Feistel ciphers

The Feistel structure does not clearly map to the cryptographic principles of confusion and diffusion. Confusion requires that each bit of the ciphertext is based upon several bits of the shared secret key. Diffusion means that a change of a single bit of the plaintext should change roughly half of the bits of the ciphertext after encryption (and vice versa after decryption). All of these properties must be handled within the round function, F, which is not specified as part of the Feistel structure.

9.8 Advantages of Feistel Ciphers

Feistel ciphers have two main advantages:

Structural reusability: As we discussed previously, the same structure can be used for encryption and decryption, as long as the key schedule is reversed for decryption.

This is extremely useful for hardware implementations of ciphers since all of the encryption logic does not have to be reimplemented in reverse for decryption.

Ability to use one-way round functions: The other major advantage of Feistel ciphers is that the round function, F, does not have to be reversible. Most ciphers require that every transformation of the plaintext performed in encryption be reversible so that they can be undone in decryption. Since this is not a requirement for ciphers using the Feistel structure, it opens up new possibilities for round functions.

9.9 Disadvantages of Feistel Ciphers

One disadvantage of Feistel ciphers is that they are limited in their ability to be parallelized as compared to other ciphers. In other ciphers, the entire internal state of the cipher changes with each round, while Feistel ciphers only change part of the internal state each round.

.

The Transmorg Contract
Proxy Smart Contracts

The second critical element used to enable NFTTs are proxy contracts. Proxy contracts are smart contracts that can be deployed in front of other smart contracts, with the ability to delegate transactions (aka requests) to a portfolio of other contracts.

The target contracts then provide return values in their response to the proxy contract, which enables it to deliver its intended functionality.

The NFTT's MVP intends to utilize the implementation of the so called 'Diamond Standard' in order to combine the smart contracts of two previously separate NFTs (NFTx + NFTy = NFT-Omega).

This permits the combination of the logic contained in NFTx and NFTy's governing smart contracts and facilitate the combination of perks and most importantly commission-structures.

When a combined NFT is sold (NFT-Omega = NFTx + NFTy) instead of calling the smart contract that used to govern NFTx, the NFTT system deploys a proxy contract that governs the sale so that both artists can receive their fair share of the sale's commission.

We call this proxy contract the transmorg contract. This ensures that artists have an incentive to create both, new NFTs (like NFTx) as well as NFT transmorgs (like NFTy's sunglasses skin) that augment the look and feel of an existing NFT (NFTx).

This is designed to enable the transmorgrification in the NFTT workflow mentioned above (see here).

10.1 Background - Proxy Smart Contracts

The primary use case for proxy contracts was the augmentation of existing contracts. The designed behavior of smart contracts is that they are immutable once deployed.

This immutable property means that redeploying a smart contract usually results in losing both: the smart contract's address and storage.

In order to circumvent this behavior, a series of proxy patterns have been developed. The Etherium ecosystem already relies on a rich ecosystem of proxy contracts, ranging from simplified implementations to established standards not currently available on the Solana Blockchain.

Pattern Name	Pattern Dynamic	Summary	Resources	Pros/ Cons
Eternal Storage Pattern	Keep storage - lose address	A pattern that comprises of two smart contracts: 1. A storage smart contract that only stores values. 2. A logic smart contract that executes logic only.	Source: here. Tutorial: here.	**Pros:** • Easy to understand • Eliminates Storage Migration **Cons:** • Contract address changes • Difficult access patterns • Doesn't work out of the box with existing smart contracts
Proxies	Lose storage – keep address	First proxy pattern that introduced the 'delegate call', which uses a callback function to another smart contract that is on a target address that is passed into the function. A 'delegate call' uses the logic that is running on a target address and executes it in the scope of the contract that calls it.	Source: here. Proxy Tutorial: here. Understanding storage Collisions: here.	**Cons:** • The proxy smart contract needs to inherit the target smart contract's storage pointers and slots • If it doesn't inherit the upgradable storage it gets a storage collision.
Diamond Standard	Keep storage – keep address	Using the many sides of a diamond as a metaphor, the idea is to have function calls go somewhere else based on the function signature. This is explained in more detail below.	Source: here. Tutorial: here. Sample implementation: here.	**Pros:** • Allows for interesting separation of code, logic and storage • Permits large constructs of smart contract combinations
Metamorphosis Smart Contract	Keep storage – keep address – replace byte code	A pattern permitting the deletion of an existing smart contract for it to be replaced with a different smart contract (replaces bytcode entirely).	Source: here. Tutorial: here. Sample implementation: here.	**Con:** • Hard to audit.

In order to maximize the decentralized nature of the NFTTs underlying blockchain technology, we will refrain from considering the metamorphic pattern and instead focus on implementing a version of the **Diamond Standard smart contract.**

It must be highlighted that this standard currently only exists on the Etherium blockchain. It is a part of this project's objective to transpose this standard to the Solana blockchain as part of the project's NFTT MVP.

10.2 EIP-2535: Diamond Standard

The NFTT MVP (Minimum Viable Product), intends to utilize an implementation of the Diamond Standard Proxy Pattern in order to combine the smart contracts of two previously separate parent NFTs: NFTx and NFTy.

It is necessary to mention that the 'Diamond' nomenclature was chosen deliberately. As such we will spend a few paragraphs on contextualizing the concepts that make up a so-called Diamond Standard Proxy.

At a high level the concept of a 'diamond' (aka: a Diamond Standard Proxy) represents a smart contract that can contains an entire portfolio of function calls. Similar to how a diamond reflects light from a multitude of sides, a Diamond Contract proxies requests to a different smart contract based on a provided function signature.

The combination of a selected path comprises of the combination of a function (a logic contract that is being called) and a function signature (which identifies which logic contract ought to be called). This combination is called a 'Facet'.

Here is a simple example of a diamond's fallback function:

```
// Find facet for function that is called and execute the
// function if a facet is found and return any value.
fallback() external payable {
  // get facet from function selector
  address facet = selectorToFacet[msg.sig];
  require(facet != address(0));
  // Execute external function from facet using delegatecall and return any value.
  assembly {
    // copy function selector and any arguments
    calldatacopy(0, 0, calldatasize())
    // execute function call using the facet
    let result := delegatecall(gas(), facet, 0, calldatasize(), 0, 0)
    // get any return value
    returndatacopy(0, 0, returndatasize())
    // return any return value or error back to the caller
    switch result
      case 0 {revert(0, returndatasize())}
      default {return (0, returndatasize())}
  }
}
```

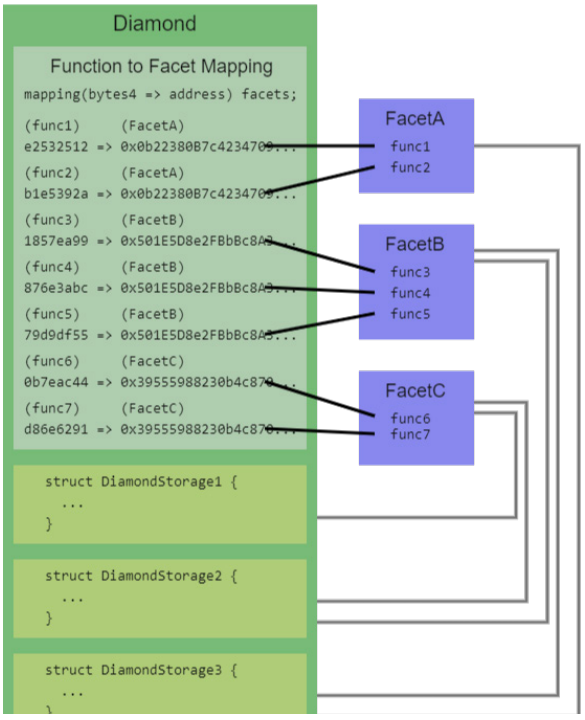
Here the Facet that is being called in the logic contract, is coming from whatever is contained in the message signature **selectorToFacet[msg.sig]**. Here msg.sig contains the function signature. Based on the function that gets selected, a delegate call is made to execute that function's logic within the scope of the proxy contract.

A diamond uses a so called **diamondCut function** to add/replace/remove any number of functions from any number of Facets in a single transaction. The diamondCut updates the mapping of a function selector to the Facet address.

The functionality to view what functions a Facet has is called „Loupe": It returns the function signatures, addresses and everything else you might want to know about a Facet.

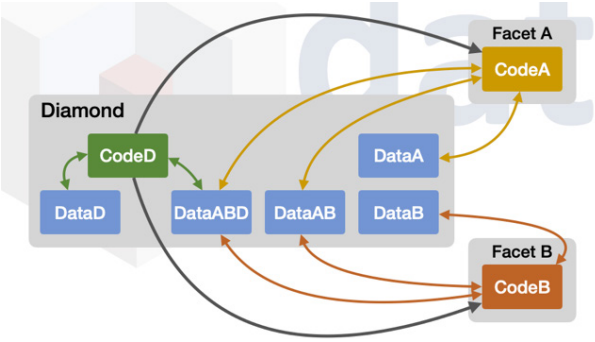
The Diamond Smart Contract (aka the proxy) then maps the functions to the Facets' functions.

These Facets look like a normal logic smart-contract, but instead of only one, we can have many. Each Facet is responsible for defining the storage location, which lives inside the Diamond smart contract.



This enables interesting ways for the separation of code storage and logic.

While the storage will always remain part of the diamond storage, we can share different storage structs that are residing on specific storage slots, with different Facets.



10.3 Diamond implementation for NFTTs

In order to faciltate the use of NFTTs, the OnitBuddy team will proceed with the development of the Diamond Standard on the Solana blockchain. During the transmorg step in the NFTT-process (see here), a transmorg smart contract is deployed. This smart contract is a Diamond Standard Proxy and accomplishes two things:

A. It acts as the primary governance mechanism for the administration of the newly created NFT-Omega, which was born from the combination of two previously independent NFTx and NFTy parent NFTs.

B. It also implements the Diamond Standard to serve as a proxy for any future transactions. The newly created NFT-Omega will be subject to the combined attributes of NFTx and NFTy's comission structures and NFT benefits.

When combined, NFTx and NFTy's smart contract-addresses are combined into transmorg hash which is utilized to create a new Facet, that will be added (diamondCut) to the Transmorg Smart-Contract's Diamond (proxy).

This Diamond then:

A. Becomes the principal authority for any future transactions of NFT-Omega,

B. introspects the commission structures of both NFTx and NFTy.

C. Combines and normalizes those commission structures into the transmorg smart contract for NFT-Omega.

D. Governs the sale and attribution of any commission to both NFTx and NFTy's creators

Any future additions to NFTx will be handled similarly through the introduction of new Facets to the diamond, permitting the automatic integration of each combined NFT's commission requirements into an equitable and transparent commission structure that is managed in a single smart contract (the transmorg contract), inside the Diamond smart contract.

10.4 The Diamond Cut Process

The updating of an existing diamond follows the below mentioned steps:

1. A migration file that defines the deployment of a new Facet must be created.
2. This migration adds the new Facet A to the Diamond proxy via the diamondCut function.
3. Once added the Diamond can execute the function defined in Facet A based on the function signature.

In the context of the NFTT this translates to the following process:

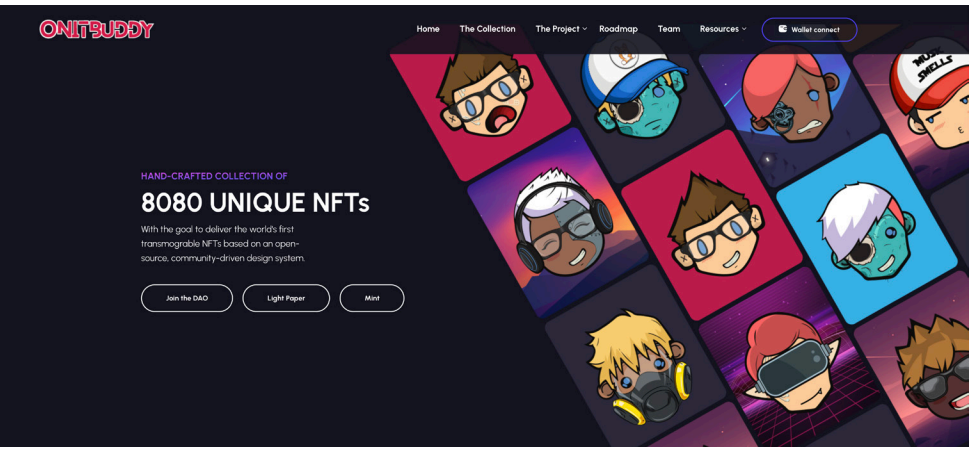
1. User visits the NFTT portal to add a transmorg NFTy to their NFTx.
2. They configure the transmorg via the transmorg interface.
3. Said interface then checks the NFT meta-data to confirm the compatability of both NFTs.
4. Once confirmed the transmorg smart-contract is created. This smart contract does the following:

- Automatically generates the visual combination of the NFTs into a new visual representation for NFT-Omega.
- Stores the location of the corresponding JPEG in the newly minted NFT's meta.
- Receives the old NFTs and hold them in escrow.
- Creates a diamond with a new Facet that contains dispatch calls to the NFTx's and NFTy's smart contracts
- Integrates their perk and commission logic into one unified smart contract and creates an equitable commission structure based on normalized commission percentages.

ONITBUDDY

Join us on Discord:
https://discord.gg/RWs3DP2

Join us on GitHub:
github.com/onitbuddy/OnitDesignSystem
github.com/onitbuddy/transmorgprotocol



Lightpaper - v1

www.onitbuddy.com