

Star Key Wallet Extension Security Audit Report

CDSecurity

Revision: 25 April 2024

Audit Report Composer: Chris

Overview

Background

Star Key Wallet team requested web extension wallet audit to CDSecurity.

Project Dates

2024.4.5 ~ 4.25: Code Review

2024.4.26: Initial Audit Report

Review Team

- Chris : Cryptography Researcher and Engineer, Blockchain Engineer

Coverage

Target Code

The following code repositories are considered in-scope for the review:

<https://github.com/madMax92221/Web-Wallet-extension>

Supporting Documentation

No document was provided.

Areas of Concern

My investigation focused on the following areas:

- Correctness of the implementation and adherence to best practices
- Exposure of any critical information during user interactions with the blockchain and external libraries, including authentication mechanisms
- Adversarial actions and other attacks that impact funds, such as the draining or the manipulation of funds
- Mismanagement of funds via transactions
- Vulnerabilities in the code, as well as secure interaction between the related and network components
- Proper management of encryption and storage of private keys, including the key derivation process
- Inappropriate permissions and excess authority
- Data privacy, data leaking, and information integrity
- Anything else as identified during the initial analysis phase

Findings

General Comments

System Design

The Star Key Wallet Extension system design is constrained by the security challenges inherent to all browser-based wallets. The presence of Chrome browser APIs allow malicious extensions to access critical secrets, which poses a threat to both the security of the application and the end user's assets. While browsers aim at creating secure sandboxes for untrusted code, they are also frequently vulnerable to 0-day exploits that allow malicious websites to escape the code sandbox and achieve Remote Code Execution (RCE) on the user's computer. I acknowledge that these fundamental security challenges are beyond the control of your team and that security should be considered a shared responsibility with the broader community of application developers, in addition to the users. There are currently no known solutions to the inherent vulnerabilities and security challenges resulting from the use of these applications. Nonetheless, development teams are advised to utilize all means possible to provide users with the safest framework for entrusting their assets. In addition, individual teams should educate users on potential, real scenarios in which their secrets might be compromised while using browser-based wallets.

Furthermore, the utilization of browser extensions introduces additional attack vectors, thus increasing the risk of successful attacks on cryptocurrency wallets. For example, browser extensions have suffered from multiple extension hijacking attacks where Google or Mozilla Developer Hub accounts were compromised. Existing extensions uploaded to the Chrome store were then replaced with malicious versions that enabled attackers to gain access to users accounts. These common attack vectors that browser extensions are susceptible to should be incorporated into the threat model, helping the design and building of a robust system with security in mind.

Code Quality

There are some bugs on codes. And there is no comments and documents related wallet.

Documentation

There is no documentation.

Penetration Testing

During penetration testing, two key scenarios were investigated within the scope of this audit: a malicious Stacks application or web page and a malicious browser extension. In both scenarios, we investigated the messaging systems to discover the attack surface and proceeded in attempting to extract secret information from the wallet or initiate malicious transactions.

Specific Issues and Suggestions

Issue 1(High): Private Key and Secret Key Phrase Exposure

Location

<https://github.com/madMax92221/Web-Wallet-extension/blob/main/portal-extension/src/app/pages/account/show-private-key/ShowPrivateKey.tsx#L1-L213>

<https://github.com/madMax92221/Web-Wallet-extension/blob/main/portal-extension/src/app/pages/settings/secret-recovery-phrase/SecretRecoveryPhrase.tsx>

Synopsis

The Star Key Wallet Extension is vulnerable to a clickjacking attack. A malicious web page can load the extension in a hidden iframe while monitoring the navigator.clipboard object, leading to the exposure of a user's secret key phrase or private key when they are logged into the wallet or show private key.

Impact

An attacker can use the secret key phrase or private key to initialize a different extension and take control of all the assets in the wallet.

Preconditions

The victim must be logged into the wallet or show private key while visiting a malicious page. If the victim clicks on a malicious item in the page, the Copy to Clipboard button will be triggered, and the secret key phrase or private key will be available to the attacker in the navigator.clipboard object.

Feasibility

The attack is easy to create, although the user would need to visit the malicious site and perform a single action. Given that an attacker is highly incentivized, they would be motivated to reach as many potential victims as possible.

Vulnerability Details

The Copy to Clipboard button will store the secret key phrase in plaintext in the navigator.clipboard object. A malicious site can run a function that checks the contents of this object periodically, and if it recognizes a valid key phrase or private key, it can then exfiltrate the key phrase or private key to a remote location without the victim being aware.

The clickjacking attack facilitates getting the key phrase or private key into the clipboard. As a result, while it can be considered a stage in the overall attack, mitigating clickjacking attacks will not solve the problem of the key phrase being stored in the navigator.clipboard object.

Recommendation & Mitigations

Protection against clickjacking attacks is traditionally implemented by utilizing the X-Frame-Options HTTP Header. Since the extension pages are not loaded from an HTTP server, research by the Hiro team can be conducted to assess how this could be implemented in the context of a browser extension. Utilizing Content Security Policies (CSP) that prevent framing resources would be a possible solution.

To protect against the clipboard attack, it would be best to prevent the key phrase or private key from ever being accessible to the clipboard available to the browser. One solution is to disable selection of the text and force users to download a file. Another option would be to investigate if the clipboard object can be disabled for that page entirely. This latter option might prove unfeasible, as users will want to be able to copy and paste addresses.

Issue 2(High): Secret Keyphrase Stored in Significant Text in Memory

Location

Browser Memory

Synopsis

The secret key phrase is stored in the memory of the browser in cleartext. Attackers that are able to dump the memory of the extension's process will be able to steal the secret key phrase and take control of a wallet.

Impact

With knowledge of the secret key phrase, an attacker can instantiate a clone of the wallet and gain control of all of its assets.

Preconditions

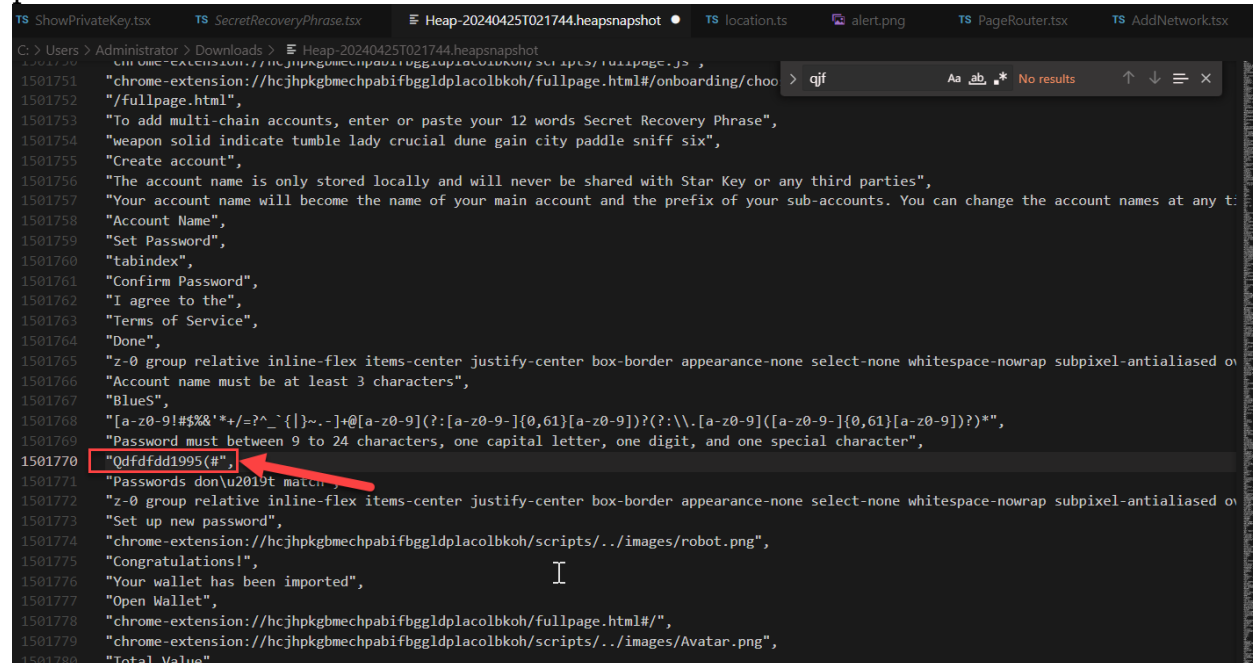
An attacker will need to be able to dump the memory from the extension's process. This would require either physical access to the browser or a post-exploitation condition on the victim's machine.

Feasibility

The attack is trivial to perform if the preconditions listed above are satisfied. Given the incentive for attackers, common malwares could incorporate a check to dump memory from browsers and search for strings in the format of a secret key phrase for exfiltration.

Vulnerability Details

With the extension open in the browser, open Chrome Developer Tools, select Memory, click Take Snapshot, and then Save. With the downloaded file, run the strings command, and for quick results grep for a word in the secret key phrase to view in clear text.



```
TS ShowPrivateKey.tsx TS SecretRecoveryPhrase.tsx Heap-20240425T021744.heapsnapshot TS location.ts alert.png TS PageRouter.tsx TS AddNetwork.tsx
C:\> Users \Administrator\> Downloads \> Heap-20240425T021744.heapsnapshot
chrome-extension://hcjhpkgbmechpabifbggldplacolbkoh/fullpage.html#/onboarding/choo
1501751 "chrome-extension://hcjhpkgbmechpabifbggldplacolbkoh/fullpage.html#/onboarding/choo
1501752 "/fullpage.html",
1501753 "To add multi-chain accounts, enter or paste your 12 words Secret Recovery Phrase",
1501754 "weapon solid indicate tumble lady crucial dune gain city paddle sniff six",
1501755 "Create account",
1501756 "The account name is only stored locally and will never be shared with Star Key or any third parties",
1501757 "Your account name will become the name of your main account and the prefix of your sub-accounts. You can change the account names at any t
1501758 "Account Name",
1501759 "Set Password",
1501760 "tabindex",
1501761 "Confirm Password",
1501762 "I agree to the",
1501763 "Terms of Service",
1501764 "Done",
1501765 "z-0 group relative inline-flex items-center justify-center box-border appearance-none select-none whitespace-nowrap subpixel-antialiased o
1501766 "Account name must be at least 3 characters",
1501767 "BlueS",
1501768 "[a-z0-9!#$%&'*/+=?^_`{|}~.-]+@[a-z0-9]({:[a-z0-9-]{0,61}[a-z0-9])?({:[a-z0-9]([a-z0-9-]{0,61}[a-z0-9])?})?",
1501769 "Password must between 9 to 24 characters, one capital letter, one digit, and one special character",
1501770 "Qdfdfdd1995(#)",
1501771 "Passwords don\u2019t match",
1501772 "z-0 group relative inline-flex items-center justify-center box-border appearance-none select-none whitespace-nowrap subpixel-antialiased o
1501773 "Set up new password",
1501774 "chrome-extension://hcjhpkgbmechpabifbggldplacolbkoh/scripts/../images/robot.png",
1501775 "Congratulations!",
1501776 "Your wallet has been imported",
1501777 "Open Wallet",
1501778 "chrome-extension://hcjhpkgbmechpabifbggldplacolbkoh/fullpage.html#/",
1501779 "chrome-extension://hcjhpkgbmechpabifbggldplacolbkoh/scripts/../images/Avatar.png",
1501780 "Total Value",
```

Wallet Password Exposed in Memory

```
TS ShowPrivateKey.tsx TS SecretRecoveryPhrase.tsx Heap-20240425T021744.heapsnapshot TS location.ts alert.png TS PageRouter.tsx TS AddNetwork.tsx
C:\Users\Administrator\Downloads> Heap-20240425T021744.heapsnapshot
1501735 "onClose",
1501736 "rg",
1501737 "glide",
1501738 "misery",
1501739 "NextUI.Ripple",
1501740 "pactuar",
1501741 "track",
1501742 "(function() {\n'use strict';\nnative function Apply();\nnative function Save();\n\n// Used in the callback implementation, could potential
1501743 'var chrome;if (!chrome) chrome = {};chrome.loadTimes = function() { native function GetLoadTimes(); return GetLoadTimes();};chrome.csi :
1501744 "Refused to evaluate a string as JavaScript because 'unsafe-eval' is not an allowed source of script in the following Content Security Poli
1501745 "Refused to compile or instantiate WebAssembly module because neither 'wasm-eval' nor 'unsafe-eval' is an allowed source of script in the fi
1501746 "chrome-extension:",
1501747 "hcjhpkgbmechpabifbgldplacolbkoh",
1501748 "chrome-extension://hcjhpkgbmechpabifbgldplacolbkoh",
1501749 "#/onboarding/choose-import-method",
1501750 "chrome-extension://hcjhpkgbmechpabifbgldplacolbkoh/scripts/fullpage.js",
1501751 "chrome-extension://hcjhpkgbmechpabifbgldplacolbkoh/fullpage.html#/onboarding/choose-import-method",
1501752 "/fullpage.html",
1501753 "To add multi-chain accounts, enter or paste your 12 words Secret Recovery Phrase",
1501754 "weapon solid indicate tumble lady crucial dune gain city paddle sniff six",
1501755 "Create account",
1501756 "The account name is only stored locally and will never be shared with Star Key or any third parties",
1501757 "Your account name will become the name of your main account and the prefix of your sub-accounts. You can change the account names at any t
1501758 "Account Name",
1501759 "Set Password",
1501760 "tabindex",
1501761 "Confirm Password",
1501762 "I agree to the",
1501763 "Terms of Service",
1501764 "Done",
1501765 "z-0 group relative inline-flex items-center justify-center box-border appearance-none select-none whitespace-nowrap subpixel-antialiased on
1501766 "Account name must be at least 3 characters",
1501767 "Blues",
1501768 "[a-z0-9!#$%&'*/=?^_`{|}~.-]+@[a-z0-9](?:[a-z0-9-]{0,61}[a-z0-9])?(?:\\.\\.[a-z0-9]([a-z0-9-]{0,61}[a-z0-9])?)*",
1501769 "Password must be between 9 to 24 characters, one capital letter, one digit, and one special character",
1501770 "Qdfdfdd1995(#",
1501771 "Passwords don't match",
1501772 "z-0 group relative inline-flex items-center justify-center box-border appearance-none select-none whitespace-nowrap subpixel-antialiased on
1501773 "Set up new password",
```

Secret Phrase Exposed in Memory

Recommendation & Mitigations

While removing the key phrase from memory entirely is not possible, it is recommended that the key phrase be encrypted when not immediately in use, and that neither the encrypted key phrase nor encryption key be stored in a text format that will appear in the output of the strings command. We recommend using a binary object instead. While this would not prevent the key from appearing in a binary search, it would create significant obstacles for an attacker to overcome.

Issue 3(Medium): Use a More Secure Password-Based Key Derivation Mechanism

Location

<https://github.com/madMax92221/Web-Wallet-extension/blob/main/shared/utils/suiHelpers.ts#L1-L62>

Synopsis

The Star Key Wallet Extension currently makes use of PBKDF2, which is a purely CPU-bound key derivation function. This class of algorithms has been advised against for several years due to negative security implications.

Impact

In this instance, the Star Key Wallet Extension may accept messages with a correct signature, created using a key under the control of the attacker.

Preconditions

The attacker needs to be able to circumvent other security restrictions on cross-tab messaging by the browser.

Feasibility

Attacks on browsers are not uncommon, but can usually be quickly patched. The feasibility mostly depends on the users update regimen and information from the development team.

Vulnerability Details

It is feasible to significantly speed up dictionary attacks on CPU bound hashes (e.g., using FPGAs) because the task is easily parallelizable. As a result, using CPU-bound key derivation for passwords has been advised against in favor of memory-hard functions like Argon2. These are more difficult to parallelize, because RAM and fast access to it is expensive to build in hardware.

Recommendation & Mitigations

Argon2 comes in multiple variants optimized for different attack models, but the balanced Argon2id variant would be well suited in this instance. Section 4 of the Argon2 RFC provides a procedure for choosing good parameters for specific use cases. When deciding on the maximum allowed time the derivation is allowed to take place, keeping in mind that it is slower in the browser, there is still a speedup for an attacker who tries to brute-force it using native code. We recommend making use of key derivation functions based on memory-hard functions such as Argon2, which is a more favorable and secure alternative.

Issue 4(Medium): Login Password Invalidation is weakness about dictionary attack.

Location

<https://github.com/madMax92221/Web-Wallet-extension/blob/main/portal-extension/src/app/components/inputs/passowrd-input/PasswordInput.tsx#L1-L63>

Synopsis

Login Password is weakness about dictionary attack.

Impact

The attacker can guess the password of wallet by dictionary attack.

Preconditions

The attacker gains full access to a user's account by brute-forcing an insecure password, which may result in total loss of user funds.

Feasibility

Easy. Password attacks are increasingly common. Furthermore, it may be that Blank's anonymous withdrawals feature is an additional incentive to conduct such an attack.

Vulnerability Details

In case the attacker accessed victim's pc, the attacker can open wallet by dictionary attack.

In case the attacker have a chrome profile which includes wallet info, then attacker can get encryptedPrivateKey from

C:\Users\Administrator\AppData\Local\Google\Chrome\User Data\Profile 1\Local Extension Settings\hcjhpkgbmechpabifbgldplacolbkoh\000003.log.

[illegible]

Encrypted Private Key Exposed

Then the attacker can try dictionary attack to get real private key from encryptedPrivateKey.

After get private key, the attacker can control all assets on wallet.

Recommendation & Mitigation

Make a validation process for password making modal.

I recommend using the `dropbox/zxcvbn` library for a password composition policy intended to prevent the use of passwords obtained from previous breaches, common passwords, and passwords containing repetitive or sequential characters, as recommended by the NIST guidelines. I suggest requiring that all passwords meet `zxcvbn`'s strength rating of 4.

Issue 5(Medium): The account creation logic is wrong

Location

<https://github.com/madMax92221/Web-Wallet-extension/blob/main/portal-extension/src/app/pages/onboarding/create-account/CreateAccount.tsx#L93-L231>

Synopsis

When you add the Bitcoin network and press the `create` button, an Ethereum address is created, but this is because the account creation function is incorrect.

Impact

Users may attempt to transfer funds to the wrong address, which may result in loss of funds.

Vulnerability Details

Users add Bitcoin network through the `AddCustomNetwork`` function.

When the user clicks the 'Create' button, an Ethereum address is created by default.

Users may lose confidence in the system due to the fake address created and may suffer loss of funds if they attempt to transfer funds.

Recommendation & Mitigations

In `CreateAccount.tsx`, you must add the address creation function for various networks as well as Bitcoin. Suggestion 1: Pin Dependencies to Specific Versions.

Issue 6(Low): The logic of adding the newly added Network name as the token name in customNetwork.tsx is incorrect.

Location

<https://github.com/madMax92221/Web-Wallet-extension/blob/main/portal-extension/src/app/pages/wallet/network/add-network/customNetwork.tsx#L151-L158>

Synopsis

The logic of adding the newly added Network name as the token name in customNetwork.tsx is incorrect.

When adding a new network in Wallet, if the function that obtains token information does not obtain the information properly, the name of the token is set to the name of the network, which is incorrect.

Impact

This reduces users' confidence in the system.

Vulnerability Details

```
147: if (network.name !== 'unknown') {
148:     methods.setValue('name', network.name)
149:
150:     const tokenInfo = await getTokenInfoBySymbol(network.name)
151:     if (tokenInfo) {
152:         newToken = {
153:             ...newToken,
154:             image: tokenInfo.image,
155:             shortName: tokenInfo.symbol.toUpperCase(),
156:             title: tokenInfo.symbol.toUpperCase(),
157:             subTitle: tokenInfo.name,
158:             coingeckoTokenId: tokenInfo.id,
159:         }
160:     }
161: }
```

Part of the `handleFetchNetworkInfo` function in `customNetwork.tsx` is as follows. As you can see in the code, when a user adds a new Network, token information is retrieved from the Network name. If there is no information about the token, the name is set to the Network name. This means that the name for the token may be incorrect.

Mitigations

When adding a new network, you must create an input field so that you can add the name of the token used when paying fees on the network.

Issue 7(Medium): There is no validation for Address Input Field in SearchAddress Component.

Location

<https://github.com/madMax92221/Web-Wallet-extension/blob/main/portal-extension/src/app/pages/wallet/token/transaction/send/search-address/index.tsx#L77-L116>

<https://github.com/madMax92221/Web-Wallet-extension/blob/main/portal-extension/src/utls/constants.ts#L17-L29>

Synopsis

When entering the address to which you want to send `token`, the address is not validated.

Impact

Sending a `token` to an invalid address may result in loss of funds.

Vulnerability Details

In the `constant.ts#getWalletAddressRegex` function, validation of three types of addresses, `Supra`, `SOL`, and `default`, is performed, so the wallet is designated as a bitcoin network and coins must be transferred to another bitcoin address. In this case, validation cannot proceed.

Recommendation & Mitigation

Validity checks must be performed on different types of addresses.

Issue 8(Medium): When transmitting, the evaluation of gas fee and expected transmission time is incorrect.

Location

<https://github.com/madMax92221/Web-Wallet-extension/blob/main/portal-extension/src/app/pages/wallet/token/transaction/send/Send.tsx#L1-L396>
[shared/hooks/useGas.ts#L83-L92](https://github.com/madMax92221/Web-Wallet-extension/blob/main/portal-extension/src/shared/hooks/useGas.ts#L83-L92)

Synopsis

When transmitting, the evaluation of gas fee and expected transmission time is incorrect.

Impact

By showing false information to the user regarding token transfer, it makes the user feel rejected towards the wallet.

Vulnerability Details

The `Send.tsx#fetchTransactionCost` function is as follows:

```
79:  useEffect(() => {
80:    const fetchTransactionCost = async () => {
81:      try {
82:        const cost = await estimatedTransactionCost
83:        setEstimatedEthersFees(cost || 0)
```

```

84:     } catch (error) {
85:       console.error('Failed to fetch transaction cost:', error)
86:       setEstimatedEthersFees(0)
87:     }
88:   }
89:
90:   // eslint-disable-next-line @typescript-eslint/no-floating-promises
91:   fetchTransactionCost()
92: }, [estimatedTransactionCost])

```

The `useGas.ts#estimatedTransactionTime` function is as follows:

```

83: const estimatedTransactionTime = React.useMemo(() => {
84:   switch (gasOption) {
85:     case GasOption.Low:
86:       return '30 seconds'
87:     case GasOption.Market:
88:       return '30 seconds'
89:     case GasOption.Aggressive:
90:       return '15 seconds'
91:   }
92: }, [gasOption])

```

As shown in the code, we only get information about the gas fee and block addition time in the Ethereum network.

Recommendation & Mitigations

Depending on which network you are transmitting from, you must obtain and display the exact gas fee and block creation time.

Suggestion 1: Pin Dependencies to Specific Versions

Location

<https://github.com/madMax92221/Web-Wallet-extension/blob/main/Package.json>

Synopsis

Many dependencies are not pinned to a specific version, instead they are pinned to a closed range of releases (e.g. package.json is set to accept major and minor changes). Pinning dependencies to exact versions is a sound approach to retaining more control over when and where upgrades take place. This will prevent unwanted versions from being inadvertently installed, thus minimizing the attack surface. In addition, this will help both developers and reviewers to identify new vulnerabilities discovered in dependencies, which is paramount to the security of the codebase.

Upgrading pinned versions of dependencies in accordance with an internal review to assess whether an upgraded version introduces compatibility issues adheres to best practices. In doing so, the necessary changes can be made to the Stacks Wallet Extension code base to mitigate against potential and

existing issues. In addition, it's critical to check if the upgraded version of the dependency has reported vulnerabilities. This will allow for informed decisions to use a more secure alternative.

Mitigation

I recommend pinning dependencies to specific versions, including pinning build-level dependencies in the package.json file to a specific version and only upgrading dependencies upon careful internal review for potential backward compatibility issues and vulnerabilities.

Suggestion 2: Make Significant Code Comments

Location

All code

Synopsis

All codes have sufficient code comments.

Mitigation

I recommend that code comments coverage is comprehensive and consistent throughout all packages of the code base, as they highlight key information and contribute to easier readability and understanding of the code for both users and reviewers.

Suggestion 3: Make Documentation

Location

<https://www.portalwallet.io>

Synopsis

There is no document so I had a trouble in auditing.

Mitigation

Must make significant documentation for deep audit.

Suggestion 4: Write valid chain id in shared\data\networks.json

Location

<https://github.com/madMax92221/Web-Wallet-extension/blob/main/shared/data/networks.json>

Synopsis

The chain id of SUI was wrong written by 1. Must have to be 21/103(Mainnet/Testnet).

The chain id of Solana and Aptos are wrong written. Must have to be correct.

Mitigation

Mainnet/Testnet

SUI chain id: 21/103

Solana chain id: 900/901

Aptos chain id: 1/2

Suggestion 5: There is no management for NFTs, so this needs to be added

Location

Wallet Interface

Synopsis

There is no management for NFTs.

Mitigation

Should have NFT management interface on code.