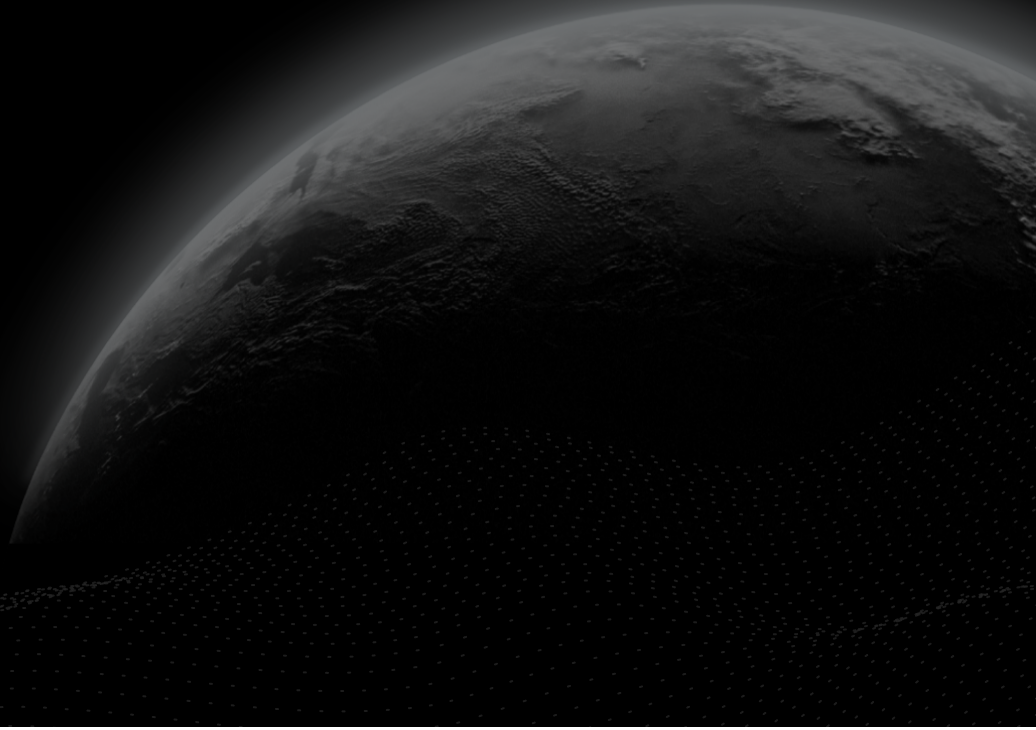




Security Assessment

OKX Marketplace of Solana

CertiK Verified on Jul 27th, 2022





CertiK Verified on Jul 27th, 2022

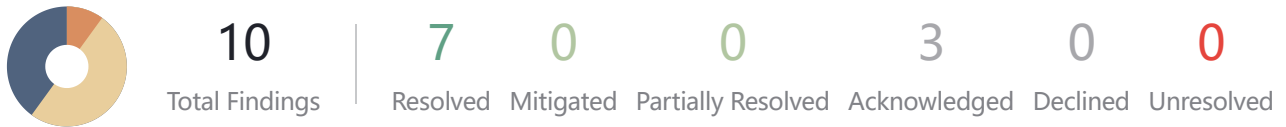
OKX Marketplace of Solana

The security assessment was prepared by CertiK, the leader in Web3.0 security.

Executive Summary

TYPES NFT, Trading	ECOSYSTEM Solana	METHODS Manual Review, Static Analysis
LANGUAGE Rust	TIMELINE Delivered on 07/27/2022	KEY COMPONENTS N/A
CODEBASE https://github.com/okex/solana-nft ...View All	COMMITTS 17113edb9a51fa4800ef22e88dac67e0344995d9 a376d4c662832c54043a3fec84b94ca84e695263 ...View All	

Vulnerability Summary



■ 0 Critical

Critical risks are those that impact the safe functioning of a platform and must be addressed before launch. Users should not invest in any project with outstanding critical risks.

■ 1 Major

1 Resolved

Major risks can include centralization issues and logical errors. Under specific circumstances, these major risks can lead to loss of funds and/or control of the project.

■ 0 Medium

Medium risks may not pose a direct risk to users' funds, but they can affect the overall functioning of a platform.

■ 5 Minor

4 Resolved, 1 Acknowledged



Minor risks can be any of the above, but on a smaller scale. They generally do not compromise the overall integrity of the project, but they may be less efficient than other solutions.

■ 4 Informational

2 Resolved, 2 Acknowledged



Informational errors are often recommendations to improve the style of the code or certain operations to fall within industry best practices. They usually do not affect the overall functioning of the code.

TABLE OF CONTENTS | OKX MARKETPLACE OF SOLANA

I Summary

[Executive Summary](#)

[Vulnerability Summary](#)

[Codebase](#)

[Audit Scope](#)

[Approach & Methods](#)

I Review Notes

[System Overview](#)

[Review Notes](#)

[Upgradability](#)

[Testing and documentation](#)

[Centralization](#)

I Findings

[CCK-01 : Outdated packages](#)

[PRO-01 : Anyone can update the protocol global configuration](#)

[PRO-02 : Order delegate payment is not updated during `update_price` instruction](#)

[PRO-03 : NFT metadata doesn't have sufficient validations](#)

[PRO-04 : System program is not validated](#)

[PRO-05 : Native mint is not validated](#)

[CKP-01 : Typos](#)

[PRO-06 : Missing `rent` account validation](#)

[PRO-07 : Missing rent-exempt validation when update `protocol_fee_recipient`](#)

[PRO-08 : Missing metadata validations on `make_order`](#)

I Optimizations

[CKP-02 : Useless `token_program` field on order account](#)

[PRO-09 : Inefficient location of `find_wsol_vault_address` statement](#)

[PRO-10 : Inefficient use of two `if` statement instead of one `if else` statement](#)

I Appendix

I Disclaimer

CODEBASE | OKX MARKETPLACE OF SOLANA

Repository










<https://github.com/okex/solana-nft>

Commit

[17113edb9a51fa4800ef22e88dac67e0344995d9 a376d4c662832c54043a3fec84b94ca84e695263](#)

AUDIT SCOPE | OKX MARKETPLACE OF SOLANA

9 files audited ● 2 files with Acknowledged findings ● 3 files with Resolved findings ● 4 files without findings

ID	File	SHA256 Checksum
● PRO	 program/src/process or.rs	f35fadd94c9d35605c89780c75e885fe88430a59708af48056002 45a956c0747
● STA	 program/src/state.rs	ec215b87b535d47318c8975c1ab09765fcab72e7adc35cdd821c 9e5870130347
● ERR	 program/src/error.rs	324fa8b6aef736250622efdf3fea9e3b01887251c5dfffb1323c92fc 38ae6badb
● INS	 program/src/instruct ion.rs	f3836e579280e1bc3e8fecf499ddb47e31d62ea27cea9a8250bf4 c20f12922e1
● UTI	 program/src/utills.rs	9d425b9739f4ccdebb8115c448622b3db844d5622e2fd1a8c152 3dd5636871c3
● CON	 program/src/consta nts.rs	0d016eff693cd0f0dd2cbc0e1a7a459022a449f17036055bae004 0016318c23c
● ENT	 program/src/entryp oint.rs	6e26acdb519adb9c52c45188744d06531991fd649c61d674d2d 225961bdabbbd
● LIB	 program/src/lib.rs	53ab7423eca6d2ee18ee66f5a3fa6bd0cfa3fe8e2aec1934cce45d 2b7b1be6b8
● PDA	 program/src/pda.rs	ef6591601d64f461d1aa632fa4981825f2b549e490db74221286 20017fc3c24b

APPROACH & METHODS | OKX MARKETPLACE OF SOLANA

This report has been prepared for OKX Marketplace of Solana to discover issues and vulnerabilities in the source code of the OKX Marketplace of Solana project as well as any contract dependencies that were not part of an officially recognized library. A comprehensive examination has been performed, utilizing Manual Review and Static Analysis techniques.

The auditing process pays special attention to the following considerations:

- Testing the smart contracts against both common and uncommon attack vectors.
- Assessing the codebase to ensure compliance with current best practices and industry standards.
- Ensuring contract logic meets the specifications and intentions of the client.
- Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders.
- Thorough line-by-line manual review of the entire codebase by industry experts.

The security assessment resulted in findings that ranged from critical to informational. We recommend addressing these findings to ensure a high level of security standards and industry practices. We suggest recommendations that could better serve the project from the security perspective:

- Testing the smart contracts against both common and uncommon attack vectors;
- Enhance general coding practices for better structures of source codes;
- Add enough unit tests to cover the possible use cases;
- Provide more comments per each function for readability, especially contracts that are verified in public;
- Provide more transparency on privileged activities once the protocol is live.

REVIEW NOTES | OKX MARKETPLACE OF SOLANA

I System Overview

OKX NFT Solana Marketplace aims to provide a straightforward NFT marketplace where users can buy and sell NFT using orders. The project aims to simplify the process of buying and selling NFTs by reducing everything to four actions:

- Make a buying order to get someone' s NFT or a selling order to offer your NFT
- Take a buying order to sell your NFT or a selling order to buy someone' s NFT
- Cancel your order
- Update the price of your order

The marketplace only accepts NFTs using the [Metaplex Metadata account format](#).

I Review Notes

Upgradability

Solana's design allows the on-chain program to be upgradeable after the initial deployment without alerting the users or modifying the program' s address by the upgrade authority, who is by default the account that uploads the program. The management of this account allowed to upload the contract should be analyzed carefully considering options like cold wallets, multisig and/or governance to reduce the potential threats.

Testing and documentation

The project only has best-case scenario tests. For each instruction there' s one or two tests that validate the execution of those instructions using correct and precise data. However, this doesn' t cover even all the best-case scenarios, for example: the execution of update price on a buying order.

It' s important to analyze the codebase and increase the quantity and quality of the tests. It' s suggested to introduce no best-case scenarios tests to detect automatically unexpected behaviors of the program. This will not only increase the quality of the code, but it will make it more robust and comprehensible.

Centralization

At its current stage, the project has certain degrees of centralization because the protocol manager has high privileges according to the design of the program. The protocol manager can change the fee recipient, the base fee points, himself, and the protocol status. This means that the protocol manager can freeze the program at any moment and the user could only cancel their orders. It's suggested to analyze options of multisig, governance and cold wallets to manage the protocol manager as well as the program's upgrade authority if it's needed. During the audit is not possible to cover the nature of those accounts because they are created when the program is deployed

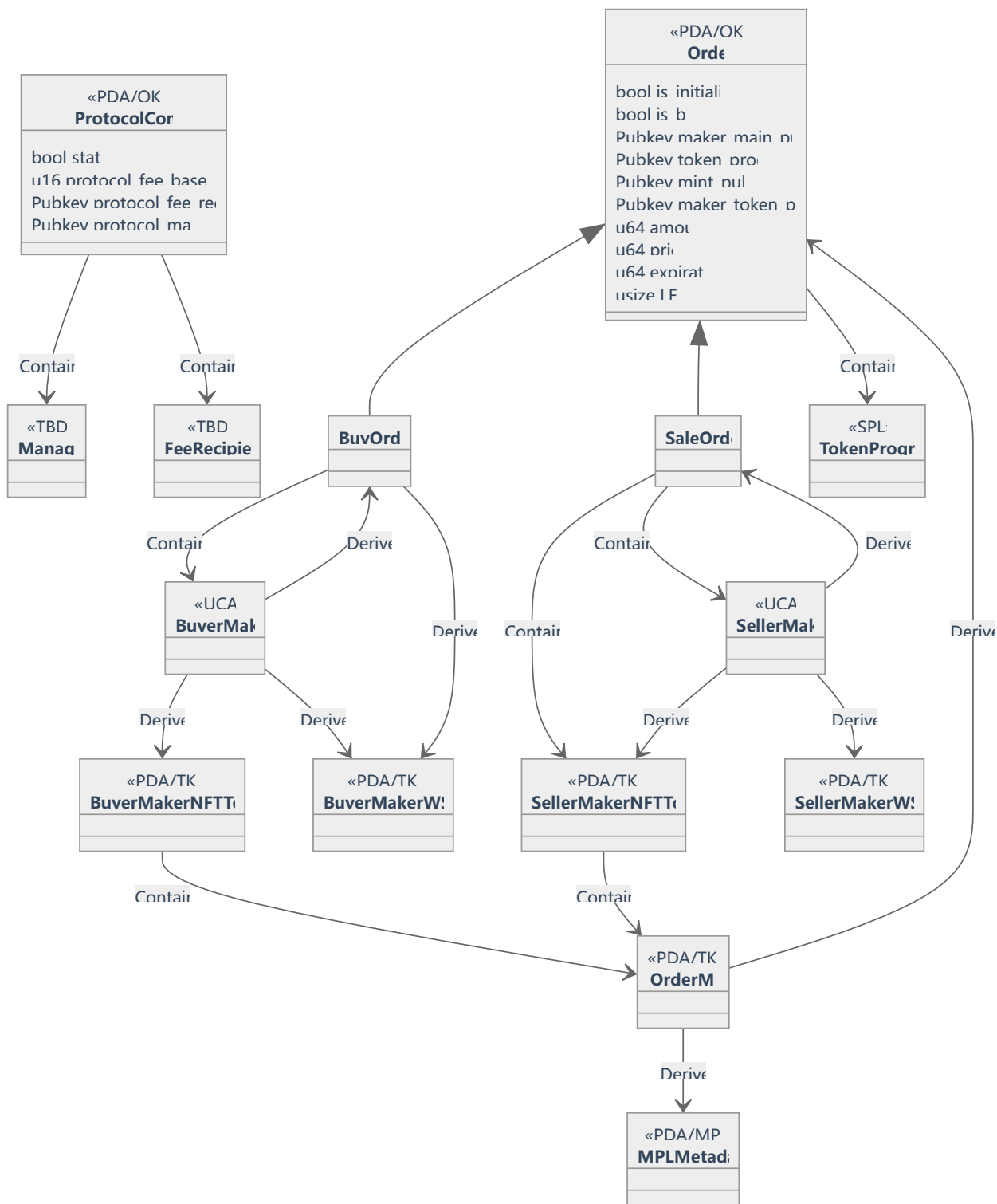
on-chain, so we incentivize the project developers to utilize decentralized and secure options to manage their wallets and authority accounts.

DIAGRAMS | OKX MARKETPLACE OF SOLANA

I Accounts Architecture

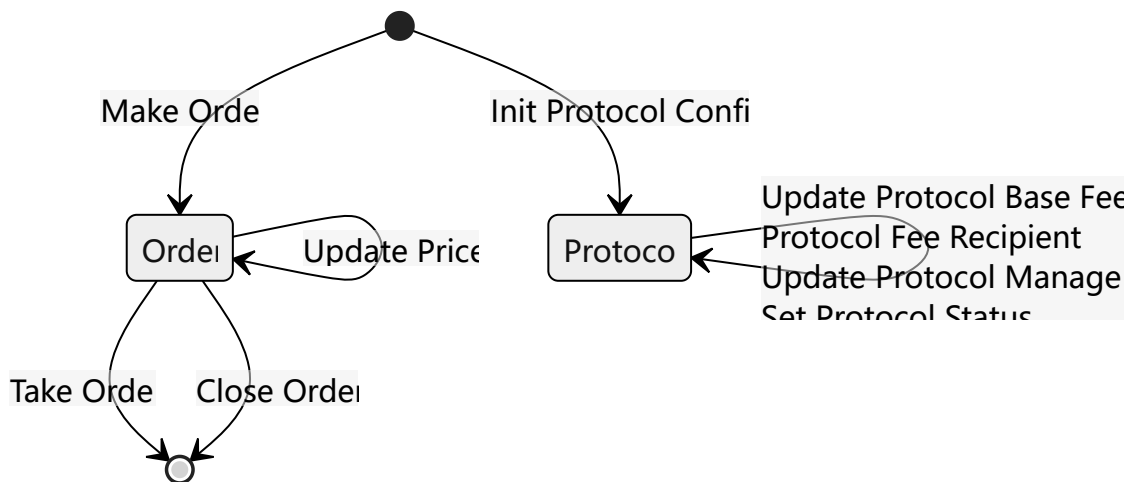
This graph describes the relationships among the internal accounts used by the audited program. The classes only represent the fields and methods implemented on the scope of the audit, and it doesn't include fields or methods implemented by external packages' traits. The methods with an underscore `_` at the beginning are methods that were implemented but never called by the program.

- `TBD` (To Be Determinate): account that is not possible to determinate its nature during the audit and that can be defined in multiple ways by the project.
- `PDA` (Program Derived Address): as defined in the Solana architecture and documentation. PDAs are managed through audited program logic. This allows the program to sign transactions with these accounts even if they don't have private keys.
- `UCA` (User Controlled Address): an account that is not controlled by the audited program, so it can be either a `PDA` by a different program or an address generated by a private key.
- `PCA` (Program Controlled Address): an account that is owned and controlled by a program. It differs from a `PDA` because it has a private key created by the user that paid for the account's rent and transfer it to a program. Even if the creator user can sign transactions with this account and the owner program can't, the account can't be modified or debited on external programs. This can only be done by the program that owns the account.
- `MPM` (MPL Metadata Program): official program of the Metaplex Metadata Program.
- `SPL` (SPL Program): official program of the Solana Program Library.
- `TKN` : account owned by the Solana Program Library (SPL) Token Program. This program is part of the seeds used to generate the PDA account.
- `OKX` : account owned by the OKX NFT Marketplace Program. This program is part of the seeds used to generate the PDA account.
- `Contains` : an account can contain structures that are stored in the same place.
- `References` : an account can reference other accounts that are not stored in the same place. Other accounts can be of whatever type, and they are referenced using their address.
- `Derives` : states that the account address from which the arrow goes out is part of the seeds that generate the pointed PDA.



Accounts Lifecycle

This graph describes the lifecycle of the instructions used by the audited program. The end point (circle with black border) means that the account is closed, if there's no end point, it means that the account is never closed. In the left side is possible to visualize the Order account while in the right side is possible to visualize the Protocol account.

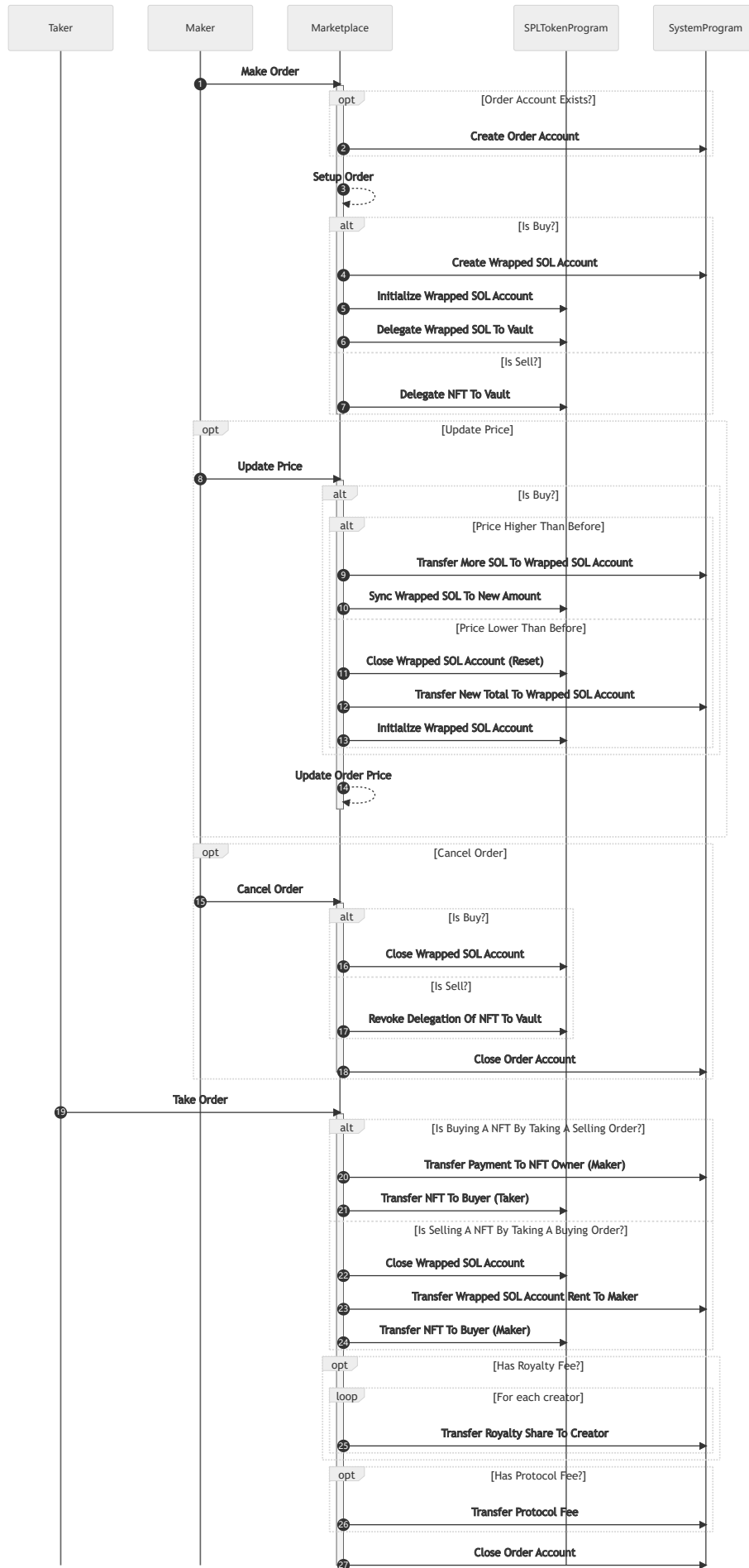


Instructions Sequence

The following graph describes the flow of each instruction called by the client and the CPI instructions (calls to external programs) and actions executed by the program itself during the respective instructions. The SPL Token program, the System Program and the MPL Metadata Program (not called but involved in the logic) were considered as black boxes during this audit and they were not analyzed in detail.

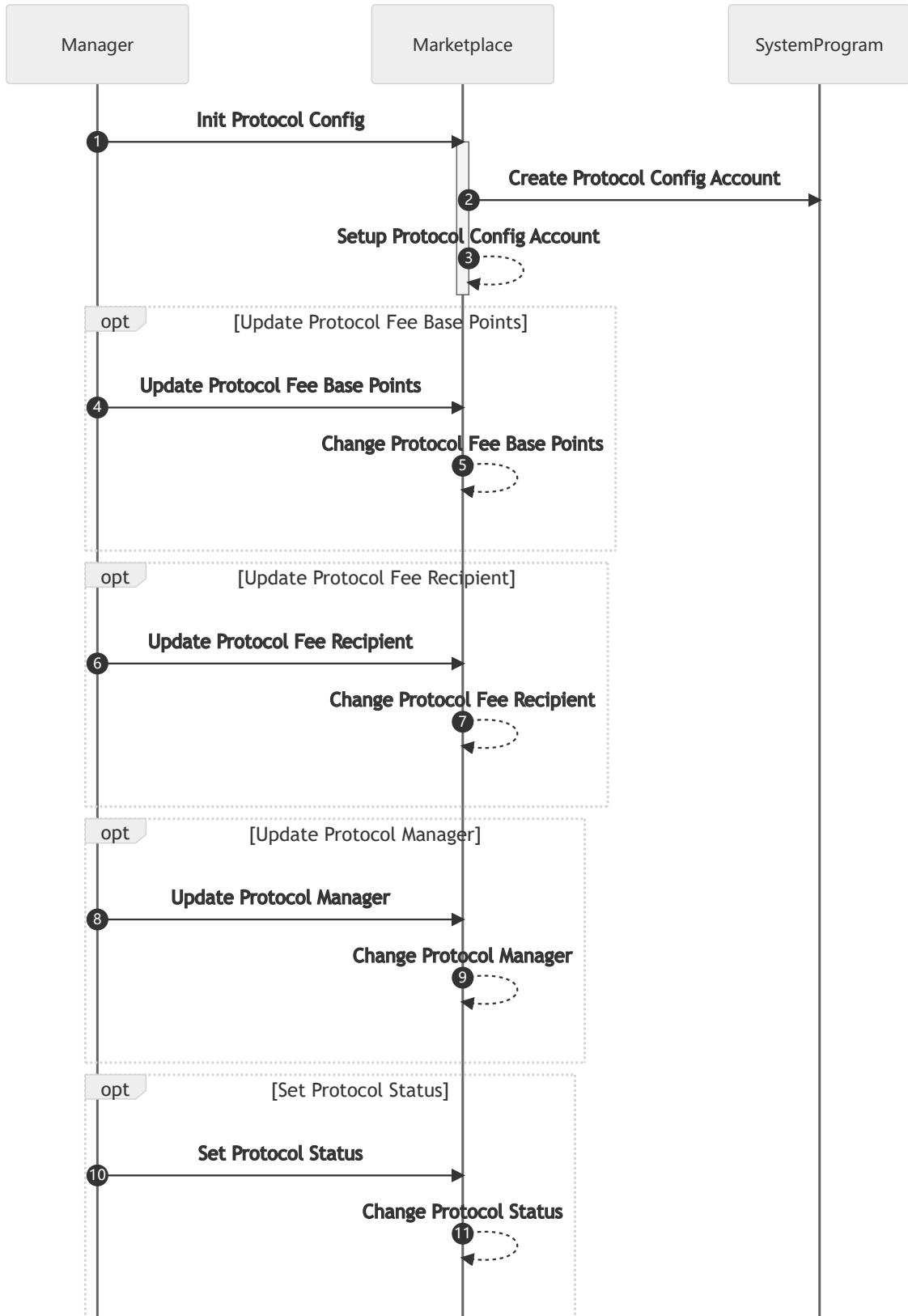
Marketplace

The first flow corresponds to the OKX NFT Marketplace that allows users (takers and makers) to sell and buy NFTs.



Protocol

The second graph correspond to the OKX NFT Marketplace protocol management that allows the program's manager to setup and modify global configurations of the program like the base fee of the executed orders, the fee recipient, the status (useful to freeze the program) and the protocol manager.



FINDINGS | OKX MARKETPLACE OF SOLANA



10

Total Findings

0

Critical

1

Major

0

Medium

5

Minor

4

Informational

This report has been prepared to discover issues and vulnerabilities for OKX Marketplace of Solana. Through this audit, we have uncovered 10 issues ranging from different severity levels. Utilizing the techniques of Manual Review & Static Analysis to complement rigorous manual code reviews, we discovered the following findings:

ID	Title	Category	Severity	Status
CCK-01	Outdated Packages	Language Specific	Minor	● Resolved
PRO-01	Anyone Can Update The Protocol Global Configuration	Logical Issue,	Major	● Resolved
PRO-02	Order Delegate Payment Is Not Updated During <code>update_price</code> Instruction	Logical Issue, Inconsistency	Minor	● Resolved
PRO-03	NFT Metadata Doesn't Have Sufficient Validations	Logical Issue	Minor	● Acknowledged
PRO-04	System Program Is Not Validated	Logical Issue	Minor	● Resolved
PRO-05	Native Mint Is Not Validated	Logical Issue	Minor	● Resolved
CKP-01	Typos	Coding Style	Informational	● Resolved
PRO-06	Missing <code>rent</code> Account Validation	Control Flow	Informational	● Resolved
PRO-07	Missing Rent-Exempt Validation When Update <code>protocol_fee_recipient</code>	Inconsistency, Logical Issue	Informational	● Acknowledged
PRO-08	Missing Metadata Validations On <code>make_order</code>	Logical Issue, Inconsistency	Informational	● Acknowledged

CCK-01 | OUTDATED PACKAGES

Category	Severity	Location	Status
Language Specific	● Minor	program/Cargo.toml: 18	● Resolved

Description

Some of the packages used on the project are outdated. For example, the `mpl-token-metadata` is on version `1.2.5` but the version `1.3.1` is available, that version improves the deserialization of the Metadata account what is used by this program.

The usage of yanked, outdated or vulnerable packages can lead to dangers not visualized on this audit because those packages could insert vulnerabilities on the codebase that were out of the scope, therefore not found.

Recommendation

We recommend analyzing the packages used by the project, checking that they don't contain vulnerabilities or missing features that can affect the program. Important to verify that the developers of the project were not using a different version of a package documentation or source code as a guide to develop the current code, this could lead to the assumption by some developers that some features are already implemented while they are not in the version of the package used the project.

If a package needs to hold an outdated version by any motived, justify and document the decision. We encourage the project to be always monitoring the state of the used packages, to prevent damage caused by zero-day vulnerabilities founded on the related packages.

Alleviation

[Certik]: The team heeded the advice and resolved the finding by updating the `mpl_token_metadata` to `1.3.2` in the commit [a376d4c662832c54043a3fec84b94ca84e695263](#)

PRO-01 | ANYONE CAN UPDATE THE PROTOCOL GLOBAL CONFIGURATION

Category	Severity	Location	Status
Logical Issue,	● Major	program/src/processor.rs: 918, 923	● Resolved

Description

All the instructions that update the protocol configurations forward their logic to the helper function `update_protocol_config_account`. This function modifies the data of the global protocol configuration (`ProtocolConfig` account). It can change the protocol's:

- Status: if the status is `false`, the program is frozen, this means that all the marketplace instructions are not callable
- Fee base points
- Fee recipient
- Manager

The mentioned helper function checks if the caller is signing the transaction but it doesn't check if the caller is the `protocol_manager` saved on the global protocol configuration (`ProtocolConfig` account). This means that the instructions using this helper function can be called by anyone, not only by the `protocol_manager`.

Recommendation

We recommend improving the validations of the helper function mentioned to assure that it follows the design and logic expected. Justify, document and test extensively any decision taken.

Alleviation

`[OKX]`: We found this bug while auditing and patches are applied.

PRO-02 | ORDER DELEGATE PAYMENT IS NOT UPDATED DURING `update_price` INSTRUCTION

Category	Severity	Location	Status
Logical Issue, Inconsistency	● Minor	program/src/processor.rs: 663	● Resolved

Description

In the `update_price` instruction, for buying orders (`order_info.is_buy = true`) the new payment stored on the wrapped SOL account is updated but the delegated amount is not updated. This means that the Vault account (used as the program's wallet) is allowed to manage only the first declared payment calculated using the price defined during the calling of the `make_order` instruction.

Recommendation

We recommend analyzing the side effects of the current flow of the `update_price` instruction to verify if it's needed to update the delegate amount and other values. The decisions taken should be justified, documented, and tested to improve the comprehension and robustness of the program.

Alleviation

[Certik]: The team heeded the advice and resolved the finding in the commit [a376d4c662832c54043a3fec84b94ca84e695263](#)

PRO-03 | NFT METADATA DOESN'T HAVE SUFFICIENT VALIDATIONS

Category	Severity	Location	Status
Logical Issue	● Minor	program/src/processor.rs: 492	● Acknowledged

I Description

The `metadata_account` is a PDA storing the metadata of an NFT. It is only used during the `take_order` instruction to get the `seller_fee_basis_points` (total royalty fee) and the `creators` (creators' address and share in percentage of the total royalty fee). This account is a PDA obtained using the `mpl_metadata_program` (Metaplex's metadata program) and the NFT's mint (`spl_token` mint). However, this account could be missing some validations, for example:

- The address of the official `mpl_metadata_program` is used to generate the PDA's address and to validate it with the passed address. However, the PDA could be owned by an unknown program replicating the `mpl_metadata_program` program.
- The `key` field could be unexpected between the following options:
 - Uninitialized,
 - EditionV1,
 - MasterEditionV1,
 - ReservationListV1,
 - MetadataV1,
 - ReservationListV2,
 - MasterEditionV2,
 - EditionMarker,
 - UseAuthorityRecord,
 - CollectionAuthorityRecord
- The `token_standard` field could be unexpected between the following options:
 - NonFungible
 - FungibleAsset
 - Fungible
 - NonFungibleEdition

I Recommendation

We recommend analyzing the different fields and options in the Metadata account to validate the only the expected NFTs are allowed to be exchanged in the marketplace. Justify, document, and test the decision taken.

I Alleviation

[OKX]: Our offchain component will check nft collections metadata and will warn users if the collection metadatas are malformed or fake.

PRO-04 | SYSTEM PROGRAM IS NOT VALIDATED

Category	Severity	Location	Status
Logical Issue	● Minor	program/src/processor.rs: 184, 233, 301, 412, 556, 785, 839	● Resolved

Description

The System Program account with the name `system_account` is not validated in multiple instructions by comparing with the official System Program ID.

```
system_account.key == &solana_program::system_program::id()
```

Recommendation

We recommend adding the account's validation to avoid unexpected behaviors. Document and test the implementation.

Alleviation

[Certik]: The team heeded the advice and resolved the finding in the commit [a376d4c662832c54043a3fec84b94ca84e695263](#)

PRO-05 | NATIVE MINT IS NOT VALIDATED

Category	Severity	Location	Status
Logical Issue	● Minor	program/src/processor.rs: 185, 322, 676, 759	● Resolved

Description

The Native Mint account with the name `native_mint_account` is not validated in multiple instructions by comparing with the official Native Mint address.

```
native_mint_account.key == &spl_token::native_mint::id()
```

Recommendation

We recommend adding the account's validation to avoid unexpected behaviors. Document and test the implementation.

Alleviation

[Certik]: The team heeded the advice and resolved the finding in the commit [a376d4c662832c54043a3fec84b94ca84e695263](#)

CKP-01 | TYPOS

Category	Severity	Location	Status
Coding Style	● Informational	program/src/error.rs: 45; program/src/instruction.rs: 62, 72, 115, 124, 133, 366; program/src/processor.rs: 92, 414, 509, 560, 635, 635, 796, 804, 933; program/src/utis.rs: 112, 127, 129	● Resolved

Description

There are typos in multiple files, they are ordered by file and location:

- On `instructions.rs`
 - `accout` should be `account`
 - `udpate` should be `update`
 - `expeected` should be `expected`
 - `Accouts` should be `Accounts`
 - `protocl` should be `protocol`
- On `processor.rs`
 - `Updata` should be `Update` or `Transfer` to be consistent with the instruction name
 - `remaining` should be `remaining`
 - `caclculate` should be `calculate`
 - `tranfer` should be `transfer`
 - `exclued` should be `excluded`
 - `transfered` should be `transferred`
 - `initilized` should be `initialized`
 - `Protocl` should be `Protocol` (2)
- On `utis.rs`
 - `Protocl` should be `Protocol`
 - `dervedid` should be `derived` (2)
- On `errors.rs`
 - `Protocl` should be `Protocol`

Recommendation

We advise the team to fix the typos.

Alleviation

[Certik] : The team heeded the advice and resolved the finding in the commit [a376d4c662832c54043a3fec84b94ca84e695263](#)

PRO-06 | MISSING `rent` ACCOUNT VALIDATION

Category	Severity	Location	Status
Control Flow	● Informational	program/src/processor.rs: 186, 323, 677, 760	● Resolved

Description

The account `rent` is not validated on the instructions `make_order` and `update_price` before calling the `initialize_mint` instruction of the `spl_token` program. This could lead to transactions that fail without a specific error if the `rent` account is incorrect.

The `initialize_mint` instruction on the used version `3.2.0` of the `spl_token` program, doesn't validate that the `rent` account address is correct, but it directly uses a hardcoded value of this address. Therefore, if the passed `rent` account is incorrect, the transaction will fail without a specific error due to a required account (the official `rent` account address) not being passed to the instruction.

Recommendation

We recommend analyzing possible scenarios where this could be a problem and reviewing the latest version (`3.3.0`) of the `spl_token` where passing the `rent` account is not necessary anymore. Justify, document, and test the decision taken.

Alleviation

[Certik]: The team heeded the advice and resolved the finding in the commit [a376d4c662832c54043a3fec84b94ca84e695263](https://github.com/okx/okx-marketplace-solana/commit/a376d4c662832c54043a3fec84b94ca84e695263)

PRO-07 | MISSING RENT-EXEMPT VALIDATION WHEN UPDATE `protocol_fee_recipient`

Category	Severity	Location	Status
Inconsistency, Logical Issue	● Informational	program/src/processor.rs: 807~814, 867	● Acknowledged

I Description

The account `protocol_fee_recipient` has a rent-exempt validation when the protocol configuration is set up using the instruction `init_protocol_config`. However, the instruction `update_protocol_fee_recipient` that allows to update this account doesn't have the same validation, if a no rent-exempt account is passed it will not throw any error.

I Recommendation

We recommend analyzing the situation while specifying the nature of the `protocol_fee_recipient` (what type of account it is) to verify if the check mentioned is necessary (this depends on the nature of the mentioned account). Document and test the use cases of the two mentioned functions used together.

I Alleviation

`[OKX]`: We can guarantee that `protocol_fee_recipient` is rent exempt offchain

PRO-08 | MISSING METADATA VALIDATIONS ON `make_order`

Category	Severity	Location	Status
Logical Issue, Inconsistency	● Informational	program/src/processor.rs: 167	● Acknowledged

I Description

The `make_order` instruction allows the maker to create an order in base to the **mint** of an expected NFT. However, the mint is missing validation to verify that it is the type of NFT that the program should accept. This is inconsistent with the validation made during the `take_order` instruction where the Metadata has some validations. This could lead to Orders that are impossible to take because during the `make_order` instruction they don't throw an error but during the `take_order` they do. Makers would not be informed about these problems in live time so they will not realize that their orders have problems.

I Recommendation

We suggest adding validation to the NFT's metadata during the execution of the `make_order` instruction. Justify, document, and test all the decisions taken.

I Alleviation

[OKX]: Our offchain component will check the validity of metadata

OPTIMIZATIONS | OKX MARKETPLACE OF SOLANA

ID	Title	Category	Severity	Status
CKP-02	Useless <code>token_program</code> Field On Order Account	Gas Optimization	Optimization	● Acknowledged
PRO-09	Inefficient Location Of <code>find_wsol_vault_address</code> Statement	Gas Optimization	Optimization	● Resolved
PRO-10	Inefficient Use Of Two <code>if</code> Statement Instead Of One <code>if</code> <code>else</code> Statement	Coding Style, Gas Optimization	Optimization	● Resolved

CKP-02 | USELESS `token_program` FIELD ON ORDER ACCOUNT

Category	Severity	Location	Status
Gas Optimization	● Optimization	program/src/processor.rs: 248; program/src/stat e.rs: 12~13	● Acknowledged

Description

The order account uses the field `token_program` to store the official token program account's address. This operation is made on the instruction `make_order` where the address is compared to the official address given by the library `sp1_token`. However, this field is never used by this instruction or other instruction. This means that the order account is storing `32` unused bytes that could be removed to reduce the rent costs. This can be translated to an extra of 0.0011136 SOL by order or 1 SOL every 898 orders.

Recommendation

We recommend analyzing if the mentioned field is necessary. The decision taken should be justified, documented and included in the tests.

Alleviation

`[OKX]`: The code has been deployed to production, will not apply this optimization.

PRO-09 | INEFFICIENT LOCATION OF `find_wsol_vault_address` STATEMENT

Category	Severity	Location	Status
Gas Optimization	● Optimization	program/src/processor.rs: 122~123	● Resolved

Description

The native function `find_program_address` allows to find the address of an account deterministically in base to input seeds (list of values like address, text, or numbers). This address is used by PDAs (Program Derivates Address) accounts. PDAs are managed by the program itself and nobody else. These addresses don't have a private key associated with their address (public key); therefore, no external actors (users or clients) can sign transactions by those addresses. This ensures that only the program owner of the PDA can sign transactions by this account. Finding an address in base to the input seeds that doesn't have a private key is not deterministically and requires a considerable amount of computing power by the on-chain program.

The pointed `find_wsol_vault_address` statement is executed always in the successful calls of the instruction `cancel_order`. However, it is useless in the case of a sale order (`order_info.is_buy = false`) because the seller doesn't have a `wsol_vault` account. This leads to unnecessary extra computing units used by the instruction when a sale order is cancelled.

Recommendation

We recommend analyzing where would be the best location for this statement. The decision should be justified, documented, and included in the test.

Alleviation

[Certik]: The team heeded the advice and resolved the finding in the commit [a376d4c662832c54043a3fec84b94ca84e695263](https://github.com/okx/okx-marketplace-solana/commit/a376d4c662832c54043a3fec84b94ca84e695263)

PRO-10 | INEFFICIENT USE OF TWO `if` STATEMENT INSTEAD OF ONE `if else` STATEMENT

Category	Severity	Location	Status
Coding Style, Gas Optimization	● Optimization	program/src/processor.rs: 149	● Resolved

Description

The instruction `close_order` contains two mutually exclusive consecutive `if` statements, this means the if the first one is activated the second one is not activated and the opposite situation. This is less efficient than using a `if else` logic.

Recommendation

We recommend analyzing why this decision was taken and whether it is more efficient to improve the quality of the code and reduce a small amount of computing power by using a `if else` logic for those two statements.

Alleviation

[Certik]: The team heeded the advice and resolved the finding in the commit [a376d4c662832c54043a3fec84b94ca84e695263](#)

APPENDIX | OKX MARKETPLACE OF SOLANA

Finding Categories

Categories	Description
Gas Optimization	Gas Optimization findings do not affect the functionality of the code but generate different, more optimal EVM opcodes resulting in a reduction on the total gas cost of a transaction.
Logical Issue	Logical Issue findings detail a fault in the logic of the linked code, such as an incorrect notion on how block.timestamp works.
Control Flow	Control Flow findings concern the access control imposed on functions, such as owner-only functions being invoke-able by anyone under certain circumstances.
Language Specific	Language Specific findings are issues that would only arise within Solidity, i.e. incorrect usage of private or delete.
Coding Style	Coding Style findings usually do not affect the generated byte-code but rather comment on how to make the codebase more legible and, as a result, easily maintainable.
Inconsistency	Inconsistency findings refer to functions that should seemingly behave similarly yet contain different code, such as a constructor assignment imposing different require statements on the input variables than a setter function.

Checksum Calculation Method

The "Checksum" field in the "Audit Scope" section is calculated as the SHA-256 (Secure Hash Algorithm 2 with digest size of 256 bits) digest of the content of each file hosted in the listed source repository under the specified commit.

The result is hexadecimal encoded and is the same as the output of the Linux "sha256sum" command against the target file.

DISCLAIMER | CERTIK

This report is subject to the terms and conditions (including without limitation, description of services, confidentiality, disclaimer and limitation of liability) set forth in the Services Agreement, or the scope of services, and terms and conditions provided to you ("Customer" or the "Company") in connection with the Agreement. This report provided in connection with the Services set forth in the Agreement shall be used by the Company only to the extent permitted under the terms and conditions set forth in the Agreement. This report may not be transmitted, disclosed, referred to or relied upon by any person for any purposes, nor may copies be delivered to any other person other than the Company, without CertiK' s prior written consent in each instance.

This report is not, nor should be considered, an "endorsement" or "disapproval" of any particular project or team. This report is not, nor should be considered, an indication of the economics or value of any "product" or "asset" created by any team or project that contracts CertiK to perform a security assessment. This report does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors, business, business model or legal compliance.

This report should not be used in any way to make decisions around investment or involvement with any particular project. This report in no way provides investment advice, nor should be leveraged as investment advice of any sort. This report represents an extensive assessing process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology.

Blockchain technology and cryptographic assets present a high level of ongoing risk. CertiK' s position is that each company and individual are responsible for their own due diligence and continuous security. CertiK' s goal is to help reduce the attack vectors and the high level of variance associated with utilizing new and consistently changing technologies, and in no way claims any guarantee of security or functionality of the technology we agree to analyze.

The assessment services provided by CertiK is subject to dependencies and under continuing development. You agree that your access and/or use, including but not limited to any services, reports, and materials, will be at your sole risk on an as-is, where-is, and as-available basis. Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. The assessment reports could include false positives, false negatives, and other unpredictable results. The services may access, and depend upon, multiple layers of third-parties.

ALL SERVICES, THE LABELS, THE ASSESSMENT REPORT, WORK PRODUCT, OR OTHER MATERIALS, OR ANY PRODUCTS OR RESULTS OF THE USE THEREOF ARE PROVIDED "AS IS" AND "AS AVAILABLE" AND WITH ALL FAULTS AND DEFECTS WITHOUT WARRANTY OF ANY KIND. TO THE MAXIMUM EXTENT PERMITTED UNDER APPLICABLE LAW, CERTIK HEREBY DISCLAIMS ALL WARRANTIES, WHETHER EXPRESS, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO THE SERVICES, ASSESSMENT REPORT, OR OTHER MATERIALS. WITHOUT LIMITING THE FOREGOING, CERTIK SPECIFICALLY DISCLAIMS ALL IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT, AND ALL WARRANTIES ARISING FROM COURSE OF DEALING, USAGE, OR TRADE PRACTICE. WITHOUT LIMITING THE FOREGOING, CERTIK MAKES NO WARRANTY OF ANY KIND THAT THE SERVICES, THE LABELS, THE ASSESSMENT REPORT, WORK PRODUCT, OR OTHER MATERIALS, OR ANY PRODUCTS OR RESULTS OF THE USE THEREOF, WILL MEET CUSTOMER' S OR ANY OTHER PERSON' S REQUIREMENTS, ACHIEVE ANY INTENDED RESULT, BE COMPATIBLE OR WORK WITH ANY SOFTWARE,

SYSTEM, OR OTHER SERVICES, OR BE SECURE, ACCURATE, COMPLETE, FREE OF HARMFUL CODE, OR ERROR-FREE. WITHOUT LIMITATION TO THE FOREGOING, CERTIK PROVIDES NO WARRANTY OR UNDERTAKING, AND MAKES NO REPRESENTATION OF ANY KIND THAT THE SERVICE WILL MEET CUSTOMER' S REQUIREMENTS, ACHIEVE ANY INTENDED RESULTS, BE COMPATIBLE OR WORK WITH ANY OTHER SOFTWARE, APPLICATIONS, SYSTEMS OR SERVICES, OPERATE WITHOUT INTERRUPTION, MEET ANY PERFORMANCE OR RELIABILITY STANDARDS OR BE ERROR FREE OR THAT ANY ERRORS OR DEFECTS CAN OR WILL BE CORRECTED.

WITHOUT LIMITING THE FOREGOING, NEITHER CERTIK NOR ANY OF CERTIK' S AGENTS MAKES ANY REPRESENTATION OR WARRANTY OF ANY KIND, EXPRESS OR IMPLIED AS TO THE ACCURACY, RELIABILITY, OR CURRENCY OF ANY INFORMATION OR CONTENT PROVIDED THROUGH THE SERVICE. CERTIK WILL ASSUME NO LIABILITY OR RESPONSIBILITY FOR (I) ANY ERRORS, MISTAKES, OR INACCURACIES OF CONTENT AND MATERIALS OR FOR ANY LOSS OR DAMAGE OF ANY KIND INCURRED AS A RESULT OF THE USE OF ANY CONTENT, OR (II) ANY PERSONAL INJURY OR PROPERTY DAMAGE, OF ANY NATURE WHATSOEVER, RESULTING FROM CUSTOMER' S ACCESS TO OR USE OF THE SERVICES, ASSESSMENT REPORT, OR OTHER MATERIALS.

ALL THIRD-PARTY MATERIALS ARE PROVIDED "AS IS" AND ANY REPRESENTATION OR WARRANTY OF OR CONCERNING ANY THIRD-PARTY MATERIALS IS STRICTLY BETWEEN CUSTOMER AND THE THIRD-PARTY OWNER OR DISTRIBUTOR OF THE THIRD-PARTY MATERIALS.

THE SERVICES, ASSESSMENT REPORT, AND ANY OTHER MATERIALS HEREUNDER ARE SOLELY PROVIDED TO CUSTOMER AND MAY NOT BE RELIED ON BY ANY OTHER PERSON OR FOR ANY PURPOSE NOT SPECIFICALLY IDENTIFIED IN THIS AGREEMENT, NOR MAY COPIES BE DELIVERED TO, ANY OTHER PERSON WITHOUT CERTIK' S PRIOR WRITTEN CONSENT IN EACH INSTANCE.

NO THIRD PARTY OR ANYONE ACTING ON BEHALF OF ANY THEREOF, SHALL BE A THIRD PARTY OR OTHER BENEFICIARY OF SUCH SERVICES, ASSESSMENT REPORT, AND ANY ACCOMPANYING MATERIALS AND NO SUCH THIRD PARTY SHALL HAVE ANY RIGHTS OF CONTRIBUTION AGAINST CERTIK WITH RESPECT TO SUCH SERVICES, ASSESSMENT REPORT, AND ANY ACCOMPANYING MATERIALS.

THE REPRESENTATIONS AND WARRANTIES OF CERTIK CONTAINED IN THIS AGREEMENT ARE SOLELY FOR THE BENEFIT OF CUSTOMER. ACCORDINGLY, NO THIRD PARTY OR ANYONE ACTING ON BEHALF OF ANY THEREOF, SHALL BE A THIRD PARTY OR OTHER BENEFICIARY OF SUCH REPRESENTATIONS AND WARRANTIES AND NO SUCH THIRD PARTY SHALL HAVE ANY RIGHTS OF CONTRIBUTION AGAINST CERTIK WITH RESPECT TO SUCH REPRESENTATIONS OR WARRANTIES OR ANY MATTER SUBJECT TO OR RESULTING IN INDEMNIFICATION UNDER THIS AGREEMENT OR OTHERWISE.

FOR AVOIDANCE OF DOUBT, THE SERVICES, INCLUDING ANY ASSOCIATED ASSESSMENT REPORTS OR MATERIALS, SHALL NOT BE CONSIDERED OR RELIED UPON AS ANY FORM OF FINANCIAL, TAX, LEGAL, REGULATORY, OR OTHER ADVICE.

CertiK | Securing the Web3 World

Founded in 2017 by leading academics in the field of Computer Science from both Yale and Columbia University, CertiK is a leading blockchain security company that serves to verify the security and correctness of smart contracts and blockchain-based protocols. Through the utilization of our world-class technical expertise, alongside our proprietary, innovative tech, we're able to support the success of our clients with best-in-class security, all whilst realizing our overarching vision; provable trust for all throughout all facets of blockchain.

