Unblock Labs

An EatTheBlocks Company

Audit report

FireBot - FireVaultFBX

August 2022

Table of Contents

Summary

This report has been prepared by Unblock Labs for FireBot to discover issues and vulnerabilities in the source code of their FireVaultFBX smart contract as well as any contract dependencies used in the project. A comprehensive examination has been performed utilizing Static Analysis and Manual Code Review 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:

- 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.

Overview

Project summary

Contract name	FireVaultFBXV4
Platform	Polygon
Language	Solidity

Codebase	https://polygonscan.com/token/0x7ccfb7d2598421ea897ba94f5d3fd598f4067577
Revised codebase	https://polygonscan.com/token/0xa461b57d4794447bB53Ad584844C4A19C6CF132B

Audit summary

Delivery date	August 26, 2022
Methodology	Static Analysis, Manual Review

Vulnerability summary

Level	Total	Acknowledge	Mitigated	Resolved
Critical	1	0	0	1
High	2	2	0	0
Medium	1	0	0	1
Low	6	1	0	5
Information	3	0	0	3
Discussion	0	0	0	0

Audit scope

ID	Contract	Codebase
FBV	FireVaultFBXV4.sol	https://polygonscan.com/token/0x7ccfb7d2598421ea897ba94f5d3fd598f4067577

Findings

ID	Title	Category	Severity	Status
FBV-01	FBX tokens can be staked and unstaked on behalf of a user	Volatile Code	Critical	Resolved
FBV-02	Centralization related risks	Centralization / Privilege	High	Acknowledge
FBV-03	Staked funds should be stored on a smart contract	Centralization / Privilege	High	Acknowledge
FBV-04	Missing restrictions on fee and valuation settings	Logical issue	Medium	Resolved
FBV-05	Vault Address should be declared public	Coding style	Low	Resolved
FBV-06	No event emitted on state change	Language Specific	Low	Resolved
FBV-07	Unchecked ERC-20 transfer() / transferFrom() Call	Volatile Code	Low	Resolved
FBV-08	Missing visibility attribute for fields	Coding style	Low	Resolved
FBV-09	Immutable properties should be constant	Coding style	Low	Resolved
FBV-10	Fee basis point can be simplified	Coding style	Low	Acknowledge
FBV-11	Naming convention	Coding style	Information	Resolved
FBV-12	Incoherent interfaces naming	Coding style	Information	Resolved
FBV-13	Non informative variable name	Coding style	Information	Resolved

FBV-01 | FBX tokens can be staked and unstaked on behalf of a user

Category	Severity	Location	Status
Volatile Code	Critical	FireVaultFBXV4.sol: 114~121, 123~131	Resolved

Description

In the `stake()` and `unstake()` functions, the sender of the transaction (`msg.sender`) is not checked.

During staking, the process relies on the owner of the FBX tokens having approved the `FireVaultFBXV4` contract prior to the transaction. Once approved anyone can stake tokens on behalf of this user.

During unstaking, `msg.sender` is not checked, anyone can unstake all the FireVault tokens currently staked, up to the amount approved by `vault_address`.
An attacker can use this function to render the contract unusable by unstaking all the tokens.

Additionally an attacker can stake and unstake tokens for other users around his transactions to gain a price advantage. This is mitigated now by the small amount approved by vault_address, relative to the amount of tokens in the vault, but the larger the project grows, the more possible this attack will become.

Recommendation

The `stake()` and `unstake()` functions should verify that the sender of the transaction (`msg.sender`) is the owner of the tokens staked or unstaked.

Alleviation

`[UnblockLabs]`: The client opted to make the recommended changes

FBV-02 | Centralization related risks

Category	Severity	Location	Status
Centralization / Privilege	**High**	FireVaultFBXV4.sol: 40~42, 80~86, 88~94, 110 FireBotItemsV2.sol; 33, 37	Acknowledge

Description

In the contract `FireVaultFBX4`, the owner has authority over the following functions:

- `mint()`
- `set_pup_valuation_multiplier()`
- `set_box_threshold()`
- `set_exit_fee()`
- `set_daily_fee()`

Any compromise to the owner's private key account may allow an attacker to take advantage of this authority and mint new vault tokens, manipulate the price, or block the withdrawals of staked tokens.
If a hacker takes control of this account, they can withdraw the majority of the staked funds

In addition, the owner of the contract `FireBotItemsV2` has authority on a `mint()` functions and can create new NFTs freely. Since the supply of the different NFTs have a direct impact on the price of the `FireVault` token, the owner's private key can also be exploited to manipulate the price.

Recommendation

The risk describes the current project design and potentially makes iterations to improve in the security operation and level of decentralization, which in most cases cannot be resolved entirely at the present stage. We advise the client to carefully manage the privileged account's private key to avoid any potential risks of being hacked. In general, we strongly recommend centralized privileges or roles in the protocol be improved via a decentralized mechanism or smart-contract-based accounts with enhanced security practices, e.g., multi-signature wallets.

Indicatively, here are some feasible suggestions that would also mitigate the potential risk at a different level in terms of short-term, long-term and permanent:

Short Term:

Timelock and Multi sign combination mitigate by delaying the sensitive operation and avoiding a single point of key management failure.

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
  AND
- Assignment of privileged roles to multi-signature wallets to prevent a single point of failure due to the private key compromised;
  AND
- A medium/blog link for sharing the timelock contract and multi-signers addresses information with the public audience.

Long Term:

Timelock and DAO, the combination, mitigate by applying decentralization and transparency.

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
  AND
- Introduction of a DAO/governance/voting module to increase transparency and user involvement;
  AND
- A medium/blog link for sharing the timelock contract, multi-signers addresses, and DAO information with the public audience.

Permanent:

Renouncing the ownership or removing the function can be considered fully resolved.

- Renounce the ownership and never claim back the privileged roles;
  OR
- Remove the risky functionality.

Alleviation

`[UnblockLabs]`: The client will work to mitigate this on the next implementation

`[FireBot]`: “Once the revision of the contract will be deployed, we will start to work on the centralization risk and address it permanently”

FBV-03 | Staked funds should be stored on a smart contract

Category	Severity	Location	Status
Centralization / Privilege	**High**	FireVaultFBXV4.sol: 119	Acknowledge

Description

When a user stakes FBX tokens in `FireVaultFBXV4`, the tokens are transferred to `vault_address`. This account is not a smart contract and the private key is managed by the project owner.

This implies that fireFBX's withdrawal capacities in the contract are limited by the amount approved by the `vault_address`. The protocol is not entirely liquid in this way.

Any compromise to the vault_address's private key account could allow an attacker to access all the staked funds.

Additionally, since this account is not a smart contract, the transfer of tokens cannot be controlled and verified by the users.

Recommendation

Use a smart contract to store the FBX staked tokens or a multisign wallet as described above.

Implement a specific `pause()` and `unpause()` function if stopping the protocol is required in certain cases.

Alleviation

`[UnblockLabs]`: The client will work to mitigate this on the next implementation

`[FireBot]`: "Once the revision of the contract will be deployed, we will start to work on the centralization risk and address it permanently"

FBV-04 | Missing restrictions on fee and valuation settings

Category	Severity	Location	Status
Logical issue	**Medium**	FireVaultFBXV4.sol: 60, 68	Resolved

Description

In the current implementation of `FireVaultFBXV4` any value can be set to `set_exit_fee()` and `set_daily_fee()`, meaning the fees can be set to 100% of the deposit and withdrawal.

Recommendation

We recommend including a bound in the functions to minimize how high the fees can be set.

Alleviation

`[UnblockLabs]`: The client removed the `set()` methods and used constant fee values

FBV-05 | Vault Address should be declared public

Category	Severity	Location	Status
Coding style	🟡 **Low**	FireVaultFBXV4.sol: 30	Resolved

Description

The `vault_address` property is not publicly exposed from the contract which makes it more difficult to track related events off-chain.

Recommendation

We recommend exposing this property as `public`.

Alleviation

`[UnblockLabs]`: The client opted to make the recommended changes

FBV-06 | No event emitted on state change

Category	Severity	Location	Status
Language Specific	Low	FireVaultFBXV4.sol: 44~46, 52~54, 60~62, 68~70	Resolved

Description

The following functions do not emit events to pass the changes out of chain.

- `set_pup_valuation_multiplier()`
- `set_box_threshold()`
- `set_exit_fee()`
- `set_daily_fee()`

Recommendation

We recommend declaring and emitting corresponding events for all the essential state variables that are possible to be changed during runtime.

Alleviation

`[UnblockLabs]`: The client opted to make the recommended changes in `set_pup_valuation_multiplier()` and removed the other methods

FBV-07 | Unchecked ERC-20 transfer() / transferFrom() Call

Category	Severity	Location	Status
Volatile Code	🟡 **Low**	FireVaultFBXV4.sol: 119, 130	Resolved

Description

The return value of the transfer()/transferFrom() call is not checked.

Recommendation

Since some ERC-20 tokens return no values and others return a bool value, they should be handled with care. We advise using the OpenZeppelin's `SafeERC20.sol` implementation to interact with the `transfer()` and `transferFrom()` functions of external ERC-20 tokens. The OpenZeppelin implementation checks for the existence of a return value and reverts if `false` is returned, making it compatible with all ERC-20 token implementations.

Alleviation

`[UnblockLabs]`: The client opted to make the recommended changes

FBV-08 | Missing visibility attribute for fields

Category	Severity	Location	Status
Coding style	Low	FireVaultFBXV4.sol: 24, 25, 26, 27, 28, 30	Resolved

Description

The following properties do not have any visibility attribute specified and are considered internal by default.

- `uint256 pup_valuation_multiplier;`
- `uint256 box_threshold;`
- `uint256 exit_fee;`
- `uint256 daily_fee;`
- `uint256 last_fee_collection;`
- `address vault_address;`

Recommendation

Use an explicit visibility attribute

ie

```
uint256 private _pup_valuation_multiplier
```

Alleviation

`[UnblockLabs]`: The client opted to make the recommended changes

FBV-09 | Immutable properties should be constant

Category	Severity	Location	Status
Coding style	Low	FireVaultFBXV4.sol: 21, 22, 30	Resolved

Description

The following properties are not changed within the implementation of the contract and can be declared constant

- `FBX`
- `items`
- `vault_address`

Recommendation

Update the properties to be constant with name in uppercase.

```
IFireBotTokenV6 public constant FBX =
IFireBotTokenV6(0xD125443F38A69d776177c2B9c041f462936F8218);
IFireBotItemsV4 public constant ITEMS =
IFireBotItemsV4(0x2e14520C30370d114612552616964a3bCeD6176E);
address public constant VAULT_ADDRESS =
0xBd684239567341ed500224FfE21F5540930359A9;
```

Alleviation

`[UnblockLabs]`: The client opted to make the recommended changes

FBV-10 | Fee basis point can be simplified

Category	Severity	Location	Status
Coding style	🟡 **Low**	FireVaultFBXV4.sol: 35, 36, 30, 130, 135	Acknowledge

Description

The fees are declared as 18 decimals precision values. Usually, a basis point of `10000` is suitable for most projects as it allow a precision up to 0,001%

Recommendation

Declare a constant to represent the basis fees point.

```
uint256 public constant FEE_BASIS_POINT =100000;
```

and use it in the related functions

```
exit_fee = FEE_BASIS_POINT * 0.1 / 100;
```

Alleviation

`[UnblockLabs]`: The client opted to keep the `1e18` precision

FBV-11 | Naming convention

Category	Severity	Location	Status
Coding style	Information	FireVaultFBXV4.sol	Resolved

Description

To follow the naming conventions:
- properties and function names should use mixed casing
- constant should be uppercase
- do not use get_ / set_ as accessor

Recommendation

The following pattern

```
uint256 exit_fee;

function set_exit_fee(uint256 x) public onlyOwner {
  exit_fee = x;
}

function get_exit_fee() public view returns (uint256) {
  return exit_fee;
}
```

can be adapted using the following pattern:

```
uint256 public exitFee;
event ExitFeeChanged(uint256 fee);

function setExitFee(uint256 fee) external onlyOwner {
  exitFee = fee;
  emit ExitFeeChanged(fee);
}
```

Alleviation

`[UnblockLabs]`: The client opted to make the recommended changes

FBV-12 | Incoherent interfaces naming

Category	Severity	Location	Status
Coding style	● Information	FireVaultFBXV4.sol: 9, 14	Resolved

Description

The namings of the interfaces declared in FireVaultFBXV4 does not correspond to the contract names/versions
The interface `IFireBotTokenV6` refers to the contract `FireBotToken`
The interface `IFireBotItemsV4` refers to the contract `FireBotItemsV2`

Recommendation

Keep the namings similar to the contracts to enhance maintainability.

Alleviation

`[UnblockLabs]`: The client opted to make the recommended changes

FBV-13 | Non informative variable name

Category	Severity	Location	Status
Coding style	● Information	FireVaultFBXV4.sol: 44, 52, 60, 68	Resolved

Description

The following functions take a variable named `x` as argument which doesn't represent the meaning of the variable

- `set_pup_valuation_multiplier()`
- `set_box_threshold()`
- `set_exit_fee()`
- `set_daily_fee()`

Recommendation

A more explicit naming should be used;
ie: `set_pup_valuation_multiplier(uint256 multiplier)`

Alleviation

`[UnblockLabs]`: The client opted to make the recommended changes in `set_pup_valuation_multiplier()` and removed the other methods

Appendix

Finding Categories

Centralization / Privilege

Centralization / Privilege findings refer to either feature logic or implementation of components that act against the nature of decentralization, such as explicit ownership or specialized access roles in combination with a mechanism to relocate funds.

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.

Volatile Code

Volatile Code findings refer to segments of code that behave unexpectedly on certain edge cases that may result in a vulnerability.

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.

Disclaimer

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 Unblock Labs'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 Unblock Labs 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 intended 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. Unblock Labs's position is that each company and individual are responsible for their own due diligence and continuous security. Unblock Labs'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 Unblock Labs are 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, UNBLOCK LABS HEREBY DISCLAIMS ALL WARRANTIES, WHETHER EXPRESS, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO THE SERVICES, ASSESSMENT REPORT, OR OTHER MATERIALS. WITHOUT LIMITING THE FOREGOING, UNBLOCK LABS 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, UNBLOCK LABS 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, UNBLOCK LABS 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 UNBLOCK LABS NOR ANY OF UNBLOCK LABS’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. UNBLOCK LABS 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 UNBLOCK LABS’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 UNBLOCK LABS WITH RESPECT TO SUCH SERVICES, ASSESSMENT REPORT, AND ANY ACCOMPANYING MATERIALS.
THE REPRESENTATIONS AND WARRANTIES OF UNBLOCK LABS 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 UNBLOCK LABS 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.

Unblock Labs

An EatTheBlocks Company

www.unblock-labs.com