Cyberscope

Audit Report

Venus Protocol

February 2025

Network	ETH
Address	0xd3CC9d8f3689B83c91b7B59cAB4946B063EB894A
Audited by	© cyberscope

Table of Contents

Table of Contents 1
Risk Classification 3
Review 4
Audit Updates 4
Source Files 4
Findings Breakdown 5
Diagnostics 6
BC - Blacklists Addresses 7
Description 7
Recommendation 7
BT - Burns Tokens 8
Description 8
Recommendation 8
CCR - Contract Centralization Risk 9
Description 9
Recommendation 10
MT - Mints Tokens 11
Description 11
Recommendation 11
ST - Stops Transactions 12
Description 12
Recommendation 12
UTPD - Unverified Third Party Dependencies 14
Description 14
Recommendation 14
L04 - Conformance to Solidity Naming Conventions 15
Description 15
Recommendation 15
L09 - Dead Code Elimination 16
Description 16
Recommendation 17
L16 - Validate Variable Setters 18
Description 18
Recommendation 18
Functions Analysis 19
Inheritance Graph 20
Flow Graph 21
Summary 22
Disclaimer 23

About Cyberscope **24**

Risk Classification

The criticality of findings in Cyberscope's smart contract audits is determined by evaluating multiple variables. The two primary variables are:

1. **Likelihood of Exploitation**: This considers how easily an attack can be executed, including the economic feasibility for an attacker.
2. **Impact of Exploitation**: This assesses the potential consequences of an attack, particularly in terms of the loss of funds or disruption to the contract's functionality.

Based on these variables, findings are categorized into the following severity levels:

1. **Critical**: Indicates a vulnerability that is both highly likely to be exploited and can result in significant fund loss or severe disruption. Immediate action is required to address these issues.
2. **Medium**: Refers to vulnerabilities that are either less likely to be exploited or would have a moderate impact if exploited. These issues should be addressed in due course to ensure overall contract security.
3. **Minor**: Involves vulnerabilities that are unlikely to be exploited and would have a minor impact. These findings should still be considered for resolution to maintain best practices in security.
4. **Informative**: Points out potential improvements or informational notes that do not pose an immediate risk. Addressing these can enhance the overall quality and robustness of the contract.

Severity	Likelihood / Impact of Exploitation
Critical	Highly Likely / High Impact
Medium	Less Likely / High Impact or Highly Likely/ Lower Impact
Minor / Informative	Unlikely / Low to no Impact

Review

Contract Name	XVS
Compiler Version	v0.8.13+commit.abaa5c0e
Optimization	200 runs
Explorer	https://etherscan.io/address/0xd3cc9d8f3689b83c91b7b59cab4946b063eb894a
Address	0xd3cc9d8f3689b83c91b7b59cab4946b063eb894a
Network	ETH
Symbol	XVS
Decimals	18
Total Supply	757,828.359

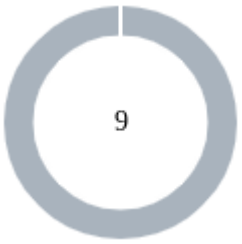
Audit Updates

Initial Audit	03 Feb 2025

Source Files

Filename	SHA256
XVS.sol	7ee4213317c0be104f80ed0414b61b2c25098e7ed15d3fbf1e70cf67bf80c961
TokenController.sol	63f9cb2342c0f5ce877009aab2136e80da39cac5d0baf75acb96e105844035ad

Findings Breakdown



Critical	0
Medium	0
Minor / Informative	9

Severity	Unresolved	Acknowledged	Resolved	Other
Critical	0	0	0	0
Medium	0	0	0	0
Minor / Informative	9	0	0	0

Diagnostics

Critical Medium Minor / Informative

Severity	Code	Description	Status
●	BC	Blacklists Addresses	Unresolved
●	BT	Burns Tokens	Unresolved
●	CCR	Contract Centralization Risk	Unresolved
●	MT	Mints Tokens	Unresolved
●	ST	Stops Transactions	Unresolved
●	UTPD	Unverified Third Party Dependencies	Unresolved
●	L04	Conformance to Solidity Naming Conventions	Unresolved
●	L09	Dead Code Elimination	Unresolved
●	L16	Validate Variable Setters	Unresolved

BC - Blacklists Addresses

Criticality	Minor / Informative
Location	TokenController.sol#L118
Status	Unresolved

Description

The authorized addresses have the authority to stop addresses from transactions. Those addresses may take advantage of it by calling the `updateBlacklist` function.

```
function updateBlacklist(address user_, bool value_) external {
        _ensureAllowed("updateBlacklist(address,bool)");
        _blacklist[user_] = value_;
        emit BlacklistUpdated(user_, value_);
    }
```

Recommendation

The team should carefully manage the private keys of the authorized addresses. We strongly recommend a powerful security mechanism that will prevent a single user from accessing the contract admin functions.

Temporary Solutions:

These measurements do not decrease the severity of the finding

- Introduce a time-locker mechanism with a reasonable delay.
- Introduce a multi-signature wallet so that many addresses will confirm the action.
- Introduce a governance model where users will vote about the actions.

Permanent Solution:

- Renouncing the ownership, which will eliminate the threats but it is non-reversible.

BT - Burns Tokens

Criticality	Minor / Informative
Location	XVS.sol#L41
Status	Unresolved

Description

The authorized addresses have the authority to burn tokens from a specific address. Those addresses may take advantage of it by calling the `burn` function. As a result, the targeted address will lose the corresponding tokens, and the authorized address will increase its minting limit.

```
   function burn(address account_, uint256 amount_) external
whenNotPaused {
        _ensureAllowed("burn(address,uint256)");
        _burn(account_, amount_);
        _increaseMintLimit(msg.sender, amount_);
    }
```

Recommendation

The team should carefully manage the private keys of the authorized addresses. We strongly recommend a powerful security mechanism that will prevent a single user from accessing the contract admin functions.

Temporary Solutions:

These measurements do not decrease the severity of the finding

- Introduce a time-locker mechanism with a reasonable delay.
- Introduce a multi-signature wallet so that many addresses will confirm the action.
- Introduce a governance model where users will vote about the actions.

Permanent Solution:

- Renouncing the ownership, which will eliminate the threats but it is non-reversible.

CCR - Contract Centralization Risk

Criticality	Minor / Informative
Location	TokenController.sol#L131,150,166
Status	Unresolved

Description

The contract's functionality and behavior are heavily dependent on external parameters or configurations. While external configuration can offer flexibility, it also poses several centralization risks that warrant attention. Centralization risks arising from the dependence on external configuration include Single Point of Control, Vulnerability to Attacks, Operational Delays, Trust Dependencies, and Decentralization Erosion.

```
    function setAccessControlManager(address newAccessControlAddress_)
external onlyOwner {
        ensureNonzeroAddress(newAccessControlAddress_);
        emit NewAccessControlManager(accessControlManager,
newAccessControlAddress_);
        accessControlManager = newAccessControlAddress_;
    }
```

```
    function setMintCap(address minter_, uint256 amount_) external {
        _ensureAllowed("setMintCap(address,uint256)");
        ...
        minterToCap[minter_] = amount_;
        emit MintCapChanged(minter_, amount_);
    }

    function migrateMinterTokens(address source_, address destination_)
external {
        _ensureAllowed("migrateMinterTokens(address,address)");
    ...
    }
```

Recommendation

To address this finding and mitigate centralization risks, it is recommended to evaluate the feasibility of migrating critical configurations and functionality into the contract's codebase itself. This approach would reduce external dependencies and enhance the contract's self-sufficiency. It is essential to carefully weigh the trade-offs between external configuration flexibility and the risks associated with centralization.

MT - Mints Tokens

Criticality	Minor / Informative
Location	XVS.sol#L27
Status	Unresolved

Description

The authorized addresses have the authority to mint tokens. These addresses may take advantage of it by calling the `mint` function. As a result, the contract tokens will be highly inflated.

```
   function mint(address account_, uint256 amount_) external
whenNotPaused {
        _ensureAllowed("mint(address,uint256)");
        _isEligibleToMint(msg.sender, account_, amount_);
        _mint(account_, amount_);
    }
```

Recommendation

The team should carefully manage the private keys of the authorized addresses' accounts. We strongly recommend a powerful security mechanism that will prevent a single user from accessing the contract admin functions.

Temporary Solutions:

These measurements do not decrease the severity of the finding

- Introduce a time-locker mechanism with a reasonable delay.
- Introduce a multi-signature wallet so that many addresses will confirm the action.
- Introduce a governance model where users will vote about the actions.

Permanent Solution:

- Renouncing the ownership, which will eliminate the threats but it is non-reversible.

ST - Stops Transactions

Criticality	Minor / Informative
Location	XVS.sol#L55
Status	Unresolved

Description

The authorized addresses have the authority to stop the transactions for all users including the owner. Those addresses take advantage of it by calling the `pause` function. As a result, the contract will revert all transactions, including buys, sells and trasnfers.

```
function pause() external {
        _ensureAllowed("pause()");
        _pause();
    }

    function unpause() external {
        _ensureAllowed("unpause()");
        _unpause();
    }

    function _beforeTokenTransfer(address from_, address to_, uint256
amount_) internal override whenNotPaused {
    ...
    }
```

Recommendation

The team should carefully manage the private keys of the authorized addresses. We strongly recommend a powerful security mechanism that will prevent a single user from accessing the contract admin functions.

Temporary Solutions:

These measurements do not decrease the severity of the finding

- Introduce a time-locker mechanism with a reasonable delay.
- Introduce a multi-signature wallet so that many addresses will confirm the action.

- Introduce a governance model where users will vote about the actions.

Permanent Solution:

- Renouncing the ownership, which will eliminate the threats but it is non-reversible.

UTPD - Unverified Third Party Dependencies

Criticality	Minor / Informative
Location	TokenController.sol#L258
Status	Unresolved

Description

The contract uses an external contract in order to determine the transaction's flow. The external contract is untrusted. As a result, it may produce security issues and harm the transactions.

```
    function _ensureAllowed(string memory functionSig_) internal view {
        if
(!IAccessControlManagerV8(accessControlManager).isAllowedToCall(msg.send
er, functionSig_)) {
            revert Unauthorized();
        }
    }
```

Recommendation

The contract should use a trusted external source. A trusted source could be either a commonly recognized or an audited contract. The pointing addresses should not be able to change after the initialization.

L04 - Conformance to Solidity Naming Conventions

Criticality	Minor / Informative
Location	TokenController.sol#L24
Status	Unresolved

Description

The Solidity style guide is a set of guidelines for writing clean and consistent Solidity code. Adhering to a style guide can help improve the readability and maintainability of the Solidity code, making it easier for others to understand and work with.

The followings are a few key points from the Solidity style guide:

1. Use camelCase for function and variable names, with the first letter in lowercase (e.g., myVariable, updateCounter).
2. Use PascalCase for contract, struct, and enum names, with the first letter in uppercase (e.g., MyContract, UserStruct, ErrorEnum).
3. Use uppercase for constant variables and enums (e.g., MAX_VALUE, ERROR_CODE).
4. Use indentation to improve readability and structure.
5. Use spaces between operators and after commas.
6. Use comments to explain the purpose and behavior of the code.
7. Keep lines short (around 120 characters) to improve readability.

```
mapping(address => bool) internal _blacklist
```

Recommendation

By following the Solidity naming convention guidelines, the codebase increased the readability, maintainability, and makes it easier to work with.
Find more information on the Solidity documentation
https://docs.soliditylang.org/en/stable/style-guide.html#naming-conventions.

L09 - Dead Code Elimination

Criticality	Minor / Informative
Location	TokenController.sol#L213,236
Status	Unresolved

Description

In Solidity, dead code is code that is written in the contract, but is never executed or reached during normal contract execution. Dead code can occur for a variety of reasons, such as:

- Conditional statements that are always false.
- Functions that are never called.
- Unreachable code (e.g., code that follows a return statement).

Dead code can make a contract more difficult to understand and maintain, and can also increase the size of the contract and the cost of deploying and interacting with it.

```
function _isEligibleToMint(address from_, address to_, uint256 amount_)
internal {
        uint256 mintingCap = minterToCap[from_];
        uint256 totalMintedOld = minterToMintedAmount[from_];
        uint256 totalMintedNew = totalMintedOld + amount_;

        if (totalMintedNew > mintingCap) {
...
        minterToMintedAmount[from_] = totalMintedNew;
        uint256 availableLimit;
        unchecked {
            availableLimit = mintingCap - totalMintedNew;
        }
        emit MintLimitDecreased(from_, availableLimit);
    }

...
```

Recommendation

To avoid creating dead code, it's important to carefully consider the logic and flow of the contract and to remove any code that is not needed or that is never executed. This can help improve the clarity and efficiency of the contract.

L16 - Validate Variable Setters

Criticality	Minor / Informative
Location	TokenController.sol#L90,153
Status	Unresolved

Description

The contract performs operations on variables that have been configured on user-supplied input. These variables are missing of proper check for the case where a value is zero. This can lead to problems when the contract is executed, as certain actions may not be properly handled when the value is zero.

```
accessControlManager = accessControlManager_
accessControlManager = newAccessControlAddress_
```

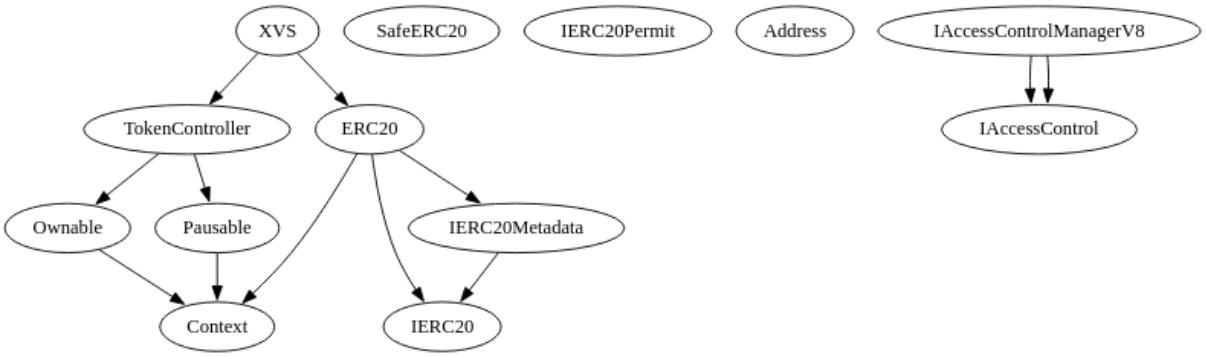
Recommendation

By adding the proper check, the contract will not allow the variables to be configured with zero value. This will ensure that the contract can handle all possible input values and avoid unexpected behavior or errors. Hence, it can help to prevent the contract from being exploited or operating unexpectedly.

Functions Analysis

Contract	Type	Bases		
	Function Name	**Visibility**	**Mutability**	**Modifiers**
XVS	Implementation	ERC20, TokenController		
		Public	✓	ERC20 TokenController
	mint	External	✓	whenNotPaused
	burn	External	✓	whenNotPaused
	_beforeTokenTransfer	Internal	✓	whenNotPaused
TokenController	Implementation	Ownable, Pausable		
		Public	✓	-
	pause	External	✓	-
	unpause	External	✓	-
	updateBlacklist	External	✓	-
	setMintCap	External	✓	-
	setAccessControlManager	External	✓	onlyOwner
	migrateMinterTokens	External	✓	-
	isBlackListed	External		-
	_isEligibleToMint	Internal	✓	
	_increaseMintLimit	Internal	✓	
	_ensureAllowed	Internal		

Inheritance Graph



Flow Graph


Legend
Internal Call
External Call
Defined Contract
Undefined Contract
Pausable
Ownable
IAccessControl
IAccessControlManagerV8
Context
IERC20Metadata
ERC20
XVS
IERC20
SafeERC20
IERC20Permit
TokenController
Address

Summary

Venus Protocol is an interesting project that has a friendly and growing community. This audit investigates security issues, business logic concerns, and potential improvements.

Disclaimer

The information provided in this report does not constitute investment, financial or trading advice and you should not treat any of the document's content as such. 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 Cyberscope's prior written consent. 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 regarded as an indication of the economics or value of any "product" or "asset" created by any team or project that contracts Cyberscope to perform a security assessment. This document 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 represents an extensive assessment 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 Cyberscope's position is that each company and individual are responsible for their own due diligence and continuous security Cyberscope'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 Cyberscope are subject to dependencies and are 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.

About Cyberscope

Cyberscope is a blockchain cybersecurity company that was founded with the vision to make web3.0 a safer place for investors and developers. Since its launch, it has worked with thousands of projects and is estimated to have secured tens of millions of investors' funds.

Cyberscope is one of the leading smart contract audit firms in the crypto space and has built a high-profile network of clients and partners.

The Cyberscope team

cyberscope.io