Maker Dao - Nst

Security Review

Cantina Managed review by:
Christoph Michel, Lead Security Researcher
M4rio.eth, Security Researcher

November 24, 2023

Contents

1 Introduction

1.1 About Cantina

Cantina is a security services marketplace that connects top security researchers and solutions with clients. Learn more at cantina.xyz

1.2 Disclaimer

Cantina Managed provides a detailed evaluation of the security posture of the code at a particular moment based on the information available at the time of the review. While Cantina Managed endeavors to identify and disclose all potential security issues, it cannot guarantee that every vulnerability will be detected or that the code will be entirely secure against all possible attacks. The assessment is conducted based on the specific commit and version of the code provided. Any subsequent modifications to the code may introduce new vulnerabilities that were absent during the initial review. Therefore, any changes made to the code require a new security review to ensure that the code remains secure. Please be advised that the Cantina Managed security review is not a replacement for continuous security measures such as penetration testing, vulnerability scanning, and regular code reviews.

1.3 Risk assessment

Severity	Description
Critical	*Must* fix as soon as possible (if already deployed).
High	Leads to a loss of a significant portion (>10%) of assets in the protocol, or significant harm to a majority of users.
Medium	Global losses <10% or losses to only a subset of users, but still unacceptable.
Low	Losses will be annoying but bearable. Applies to things like griefing attacks that can be easily repaired or even gas inefficiencies.
Gas Optimization	Suggestions around gas saving practices.
Informational	Suggestions around best practices or readability.

1.3.1 Severity Classification

The severity of security issues found during the security review is categorized based on the above table. Critical findings have a high likelihood of being exploited and must be addressed immediately. High findings are almost certain to occur, easy to perform, or not easy but highly incentivized thus must be fixed as soon as possible.

Medium findings are conditionally possible or incentivized but are still relatively likely to occur and should be addressed. Low findings a rare combination of circumstances to exploit, or offer little to no incentive to exploit but are recommended to be addressed.

Lastly, some findings might represent objective improvements that should be addressed but do not impact the project's overall security (Gas and Informational findings).

2 Security Review Summary

The Maker Protocol, also known as the Multi-Collateral Dai (MCD) system, allows users to generate Dai (a decentralized, unbiased, collateral-backed cryptocurrency soft-pegged to the US Dollar) by leveraging collateral assets approved by the Maker Governance, which is the community organized and operated process of managing the various aspects of the Maker Protocol.

From Sep 20th to Sep 21st the Cantina team conducted a review of nst on commit hash 721f870a. The team identified a total of **5** issues in the following risk categories:

- Critical Risk: 0
- High Risk: 0
- Medium Risk: 0
- Low Risk: 0
- Gas Optimizations: 0
- Informational: 5

3 Findings

3.1 Informational

3.1.1 Deprecated non-standard ERC20 methods `increaseAllowance`/`decreaseAllowance`

Severity: Informational

Context: Nst.sol#L145, Nst.sol#L154

Description: The `increaseAllowance` and `decreaseAllowance` functions where adopted as being part of an ERC20 contract over the time as a measure to avoid front-running of `allowance`. As of our knowledge there is no demonstrated exploit involving the `allowance`, making these non-EIP functions obsolete in an ERC20 implementation. OpenZeppelin and Solady already voted and planned to remove them from the standard `ERC20` implementation and maybe move them into an extension, if desired.

Recommendation: Consider removing these non-standard ERC20 functions from the `Nst` token.

Maker: Removed in commit 8f621dcf.

Cantina: Fixed.

3.1.2 Dependence on `NstJoin` to be the only minter of `Nst`

Severity: Informational

Context: NstJoin.sol#L48

Description: The `NstJoin` contract has minting rights for the `NST` token contract. Exiting through the `NstJoin` mints NST; joining the `NstJoin` again does the reverse, it burns `NST` and increases the `vat.dai[usr]` DAI accounting variable in the Vat. A property one might expect from this contract is that everyone who exited can also join again. This is only true if `NstJoin` is the only minter of NST (and there are no other ways to mint NST that don't allow going back to Vat). The desired property can be encoded as `NST.totalSupply() = vat.dai[NstJoin]`.

Recommendation: Ensure that `NstJoin` is the only minter of `NST`. Alternatively, if new minters are added, ensure that there is a way for all NST holders to join the Vat again.

Maker: This is ensured in the deployment scripts. The sole ward of the token will be `nstJoin`.

Cantina: The issue is fixed in the deployment script provided.

3.1.3 Check that address contains code before calling it

Severity: Informational

Context: Nst.sol#L221

Description: The contract performs a low-level `isValidSignature` static-call to an EIP1271 contract. This call is always performed if verifying the signature failed and it does not check if the address is a contract. The call will still be done for EOAs and precompiles. Calling EOAs will revert as they won't return the expected 32 bytes but certain precompiles will return data and could in theory match the expected magic number. It should not be possible to verify signatures on behalf of precompiles and this could lead to issues if a protocol uses low-address as an indirect burn address.

Additional info: Potential precompiles that could validate are:

1) identity (`0x4`) but it returns more than 32 bytes (because of the calldata encoding `4 + 32 + X > 32`) and the return data size check would fail.
2) `sha256 / ripemd160` (`0x2, 0x3`) return 32 bytes but when decoding to `bytes4`, in the current solidity versions it checks that only the 4 most-significant bytes are set, so one would need to find an entire 32 bytes collision to match `isValidSignature.selector || 28-zero-bytes`, which is infeasible.

Recommendation: While the current checks make it impossible to perform a successful verification on the current precompiles, we still recommend checking if the contract contains code before performing the call. This also saves gas on the path when an EOA signature verification fails as it would now skip the call that is already known to fail.

Maker: Fixed in PR 5.

Cantina: Fixed.

3.1.4 Prefer `encodeCall` to `encodeWithSelector`

Severity: Informational

Context: Nst.sol#L222

Description: The contract performs a low-level static-call to an EIP1271 contract. It is using `abi.encodeWithSelector` to encode the parameters.

Recommendation: Consider using `abi.encodeCall` as it type-checks that `digest` and `signature` are indeed the correct types defined as the arguments for `isValidSignature`. Wrong or missing argument types can be caught already during compile time, whereas `encodeWithSelector` does not give this guarantee.

```
staticcall(abi.encodeCall(IERC1271.isValidSignature, (digest, signature)))
```

Maker: Fixed in PR 5.

Cantina: Fixed.

3.1.5 Document why `unchecked` usage is safe

Severity: Informational

Context: Nst.sol#L103, Nst.sol#L129

Description: The contract uses `unchecked` arithmetic to save on gas. There are comments specifying why it's safe to perform an unchecked math operation for certain unchecked blocks, but not for all of them:

- Nst.sol#L103, Nst.sol#L129: Unchecked addition here is safe as the sum of all balances equals the `totalSupply`, and any overflow would have occurred already when increasing the `totalSupply`.

Recommendation: Consider commenting on all non-obvious unchecked code usage.

Maker: Fixed in PR 5.

Cantina: Fixed.

4 Additional Comments

The Cantina team reviewed MakerDao's nst changes holistically on commit hash 93abbc714ffa5662cdb264865829752e2ea63df9 and determined that all issues were resolved and no new issues were identified.