Introduction

A time-boxed security review of the **Ethena** protocol was done by **pashov**, with a focus on the security aspects of the application's smart contracts implementation.

Disclaimer

A smart contract security review can never verify the complete absence of vulnerabilities. This is a time, resource and expertise bound effort where I try to find as many vulnerabilities as possible. I can not guarantee 100% security after the review or even if the review will find any problems with your smart contracts. Subsequent security reviews, bug bounty programs and on-chain monitoring are strongly recommended.

About **pashov**

Krum Pashov, or **pashov**, is an independent smart contract security researcher. Having found numerous security vulnerabilities in various protocols, he does his best to contribute to the blockchain ecosystem and its protocols by putting time and effort into security research & reviews. Check his previous work here or reach out on Twitter @pashovkrum.

About **Ethena**

The Ethena protocol is building `USDe` which will be a synthetic dollar with yield bearing properties, deployed on Ethereum. The stablecoin will be 100% collateralized with no collateral within the banking system, using as collateral USDC, stETH and other LSDs. The yield is expected to come from `stETH` and arbitrage. The `USDe` smart contract's minting and redeeming will be handled in a trusted manner by the Ethena team.

More docs

Observations

The protocol has a high degree of centralization:

- The Ethena team can freely do minting & redeeming of `USDe` tokens
- The owner of the `StakedUSDe` token contract can manipulate anyone's balance by blacklisting an address and then calling `redistributeLockedAmount` to move his balance to a destination address
- Users can't mint/redeem `USDe` as they only sign transactions which are later executed by Ethena. Ethena does a good amount & variety of off-chain checks on users based on KYC, AML whitelisting and others.

A delegated signer can call `redeem` for a user (if he has approved the `EthenaMinting` contract to burn his `USDe` tokens) and can control the `beneficiary` address who will receive the collateral assets.

Privileged Roles & Actors

- `USDe` minter - can mint any amount of `USDe` tokens to any address. Expected to be the `EthenaMinting` contract
- `USDe` owner - can set token `minter` and transfer ownership to another address
- `USDe` token holder - can not just transfer tokens but burn them and sign permits for others to spend their balance
- `StakedUSDe` admin - can rescue tokens from the contract and also to redistribute a fully restricted staker's `stUSDe` balance, as well as give roles to other addresses (for example the `FULL_RESTRICTED_STAKER_ROLE` role)
- `StakedUSDeV2` admin - has all power of "`StakedUSDe` admin" and can also call the `setCooldownDuration` method
- `REWARDER_ROLE` - can transfer rewards into the `StakedUSDe` contract that will be vested over the next 8 hours
- `BLACKLIST_MANAGER_ROLE` - can do/undo full or soft restriction on a holder of `stUSDe`
- `SOFT_RESTRICTED_STAKER_ROLE` - address with this role can't stake his `USDe` tokens or get `stUSDe` tokens minted to him
- `FULL_RESTRICTED_STAKER_ROLE` - address with this role can't burn his `stUSDe` tokens to unstake his `USDe` tokens, neither to transfer `stUSDe` tokens. His balance can be manipulated by the admin of `StakedUSDe`
- `MINTER_ROLE` - can actually mint `USDe` tokens and also transfer `EthenaMinting`'s token or ETH balance to a custodian address
- `REDEEMER_ROLE` - can redeem collateral assets for burning `USDe`
- `EthenaMinting` admin - can set the maxMint/maxRedeem amounts per block and add or remove supported collateral assets and custodian addresses
- `GATEKEEPER_ROLE` - can disable minting/redeeming of `USDe` and remove `MINTER_ROLE` and `REDEEMER_ROLE` roles from authorized accounts

Severity classification

Severity	Impact: High	Impact: Medium	Impact: Low
Likelihood: High	Critical	High	Medium
Likelihood: Medium	High	Medium	Low
Likelihood: Low	Medium	Low	Low

Impact - the technical, economic and reputation damage of a successful attack

Likelihood - the chance that a particular vulnerability gets discovered and exploited

Severity - the overall criticality of the risk

Security Assessment Summary

***review commit hash* - 1d0f746d5ebbc5b2254d5d311ac7399aab66ec17**

***fixes review commit hash* - deaa94b3df961d995e93e408b70b0d085ca1866c**

Scope

The following smart contracts were in scope of the audit:

- `EthenaMinting`
- `USDe`
- `StakedUSDe`
- `StakedUSDeV2`
- `SingleAdminAccessControl`

Findings Summary

ID	Title	Severity	Status
[L-01]	Malicious actor can evade the `FULL_RESTRICTED_STAKER_ROLE` role	Low	Fixed
[L-02]	Unchecked method return values can lead to errors	Low	Acknowledged
[L-03]	The `SOFT_RESTRICTED_STAKER_ROLE` gives a false sense of security	Low	Acknowledged
[L-04]	EIP712 is not correctly implemented for the `Route` struct	Low	Fixed

Detailed Findings

[L-01] Malicious actor can evade the `FULL_RESTRICTED_STAKER_ROLE` role

The protocol has implemented the `FULL_RESTRICTED_STAKER_ROLE` role so that the `StakedUSDe` owner has authority over blacklisting addresses and then manipulating their balances. This mechanism is flawed and an attacker can bypass it - let's look at an example:

1. A user does a malicious action or is understood to be a bad actor. The `StakedUSDe` owner decides to blacklist his address
2. The `StakedUSDe` owner sends a transaction with a call to `addToBlacklist` for the address of the malicious user
3. The user is monitoring Ethereum's mempool and front-runs this transaction, sending all of his `stUSDe` balance to another address he controls
4. Now his old address is blacklisted as he has the `FULL_RESTRICTED_STAKER_ROLE` role, but all of his tokens are in his new address and can be used as normal

The current best solution is enforcing the usage of a private mempool service for all admin actions.

Discussion

Ethena: Fixed. Transactions will be always submitted only through Flashbots.

[L-02] Unchecked method return values can lead to errors

In the `StakedUSDe` contract we have the `addToBlacklist` and `removeFromBlacklist` methods that call `_grantRole` and `_revokeRole` respectively. The return values of the latter should be checked in both cases, so that when granting a role it is certain that the `target` didn't already have this role, and when revoking a role it is certain that the `target` did actually have this role.

Discussion

Ethena: Acknowledged. Won't fix since it would add gas cost overhead.

[L-03] The `SOFT_RESTRICTED_STAKER_ROLE` gives a false sense of security

The role forbids its holders to deposit tokens into `StakedUSDe` or receive minted shares from it. This protection can be bypassed by an account transferring his tokens to another address that doesn't have the role, but the initial account controls it, and can deposit tokens or mint share from `StakedUSDe` successfully. Consider removing the role and sticking to using `FULL_RESTRICTED_STAKER_ROLE` only.

Discussion

Ethena: Acknowledged. This is a part of the design. Ethena is legally required to not pay yield to users from certain countries, and we will show best effort, including having on chain restrictions. This covers us legally, but sophisticated users will be able to bypass this restriction in a variety of ways, such as buying/selling stUSDe on open market or transferring funds to different addresses to stake/unstake.

[L-04] EIP712 is not correctly implemented for the `Route` struct

As stated in the EIP712 standard - "The array values are encoded as the keccak256 hash of the concatenated encodeData of their contents". This is not correctly followed in `EthenaMinting::encodeRoute` as it does not do this for the `Route` struct array fields. While this is usually a more serious problem, the method is not called in the protocol and can just be removed.

Discussion

Ethena: Fixed. The `encodeRoute` method has been removed.