EthStaker

Eth-docker Updates

Security Assessment Report

Version: 2.0

October, 2024

Contents

Introduction

Sigma Prime was commercially engaged to perform a time-boxed security review of the `eth-docker` updates in scope. The review focused solely on the security aspects of the Solidity implementation of the contract, though general recommendations and informational comments are also provided.

Disclaimer

Sigma Prime makes all effort but holds no responsibility for the findings of this security review. Sigma Prime does not provide any guarantees relating to the function of the smart contract. Sigma Prime makes no judgements on, or provides any security review, regarding the underlying business model or the individuals involved in the project.

Document Structure

The first section provides an overview of the functionality of the `eth-docker` updates contained within the scope of the security review. A summary followed by a detailed review of the discovered vulnerabilities is then given which assigns each vulnerability a severity rating (see Vulnerability Severity Classification), an *open/closed/resolved* status and a recommendation. Additionally, findings which do not have direct security implications (but are potentially of interest) are marked as *informational*.

The appendix provides additional documentation, including the severity matrix used to classify vulnerabilities within the `eth-docker` updates in scope.

Overview

`eth-docker` is an automation wrapper to run an Ethereum node. It uses the client's Docker images or source code and adds and entry point script to handle optional components such as MEV, checkpoint sync and additional parameters.

`eth-dockers` 's key management commands allows to import validator keys to the clients using the `keymanager` API.

Security Assessment Summary

Scope

The review was conducted on the files hosted on the eth-docker repository.

The scope of this time-boxed review was strictly limited to changes between 860acd1 and 4656f88.

Note: third party libraries and dependencies, such as OpenZeppelin, were excluded from the scope of this assessment.

Approach

The manual code review section of the report is focused on identifying any and all issues/vulnerabilities associated with the business logic implementation of the Bash scripts and Docker files. This includes their internal interactions, intended functionality and correct implementation of key management functionality, including its cryptographic functions used for key generation, key storage and network communication. Additionally, the manual review process focused on the hardening settings of the Docker configuration.

To support this review, the testing team used the following automated testing tools:

- ShellCheck: `https://github.com/koalaman/shellcheck`
- Snyk CLI: `https://github.com/snyk/cli`

Output for these automated tools is available upon request.

Coverage Limitations

Due to the time-boxed nature of this review, all documented vulnerabilities reflect best effort within the allotted, limited engagement time. As such, Sigma Prime recommends to further investigate areas of the code, and any related functionality, where majority of critical and high risk vulnerabilities were identified.

Findings Summary

The testing team identified a total of 1 issues during this assessment. Categorised by their severity:

- Informational: 1 issue.

Detailed Findings

This section provides a detailed description of the vulnerabilities identified within the `eth-docker` changes in scope. Each vulnerability has a severity classification which is determined from the likelihood and impact of each issue by the matrix given in the Appendix: Vulnerability Severity Classification.

A number of additional properties of the contracts, including gas optimisations, are also described in this section and are labelled as "informational".

Each vulnerability is also assigned a **status**:

- ***Open:*** the issue has not been addressed by the project team.
- ***Resolved:*** the issue was acknowledged by the project team and updates to the affected contract(s) have been made to mitigate the related risk.
- ***Closed:*** the issue was acknowledged by the project team but no further actions have been taken.

Summary of Findings

ID	Description	Severity	Status
ED2-01	Sensitive Data Can Be Handled By Secrets	Informational	**Closed**

ED2-01	Sensitive Data Can Be Handled By Secrets
Asset	`*.yml`
Status	**Closed:** See Resolution
Rating	Informational

Description

Sensitive data is managed through files, volumes or with environment variables, for example `/ee-secrets`.

Docker provides a way to handle sensitive data in a safer way through the usage of secrets. When a secret is added to a Docker swarm, Docker sends the secret to the swarm over a TLS connection which is then saved in an encrypted `Raft Log`. A Docker node has access to the log only if it is a swarm manager or if it is running a service that has access granted to that secret.

Secrets can be used in Docker Compose and in Container Build as well. The issue was rated as informational as for the current setup, sensitive data has reasonable access permissions at this time.

Recommendations

For easier access managment and safer storage the usage of secrets is advised over storing sensitive data in files or environment variables.

Resolution

The development team have opted not to use `Secrets` at this time due to the marginal benefit. The testing team have reviewed and agree with the following comments.

> *They are of marginal utility, if the host is breached on the user running Eth Docker, then by virtue of needing to be able to run Docker, that user can access the JWT secret no matter how it is stored. In addition, the engine port is kept within the docker bridge network by default, not mapped to host.*

Appendix A Vulnerability Severity Classification

This security review classifies vulnerabilities based on their potential impact and likelihood of occurance. The total severity of a vulnerability is derived from these two metrics based on the following matrix.

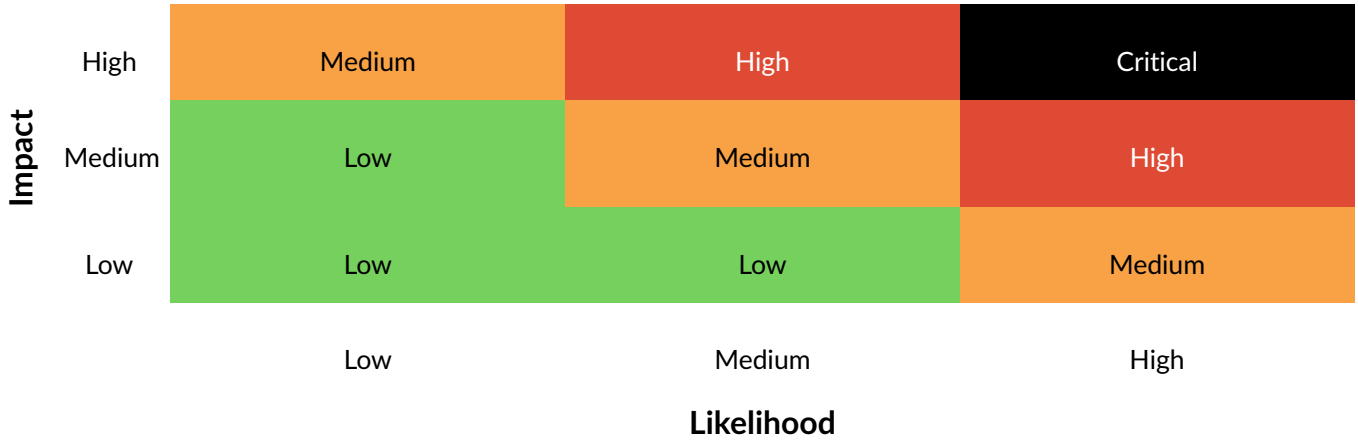

Table 1: Severity Matrix - How the severity of a vulnerability is given based on the *impact* and the *likelihood* of a vulnerability.