Everscale Bug Bounty Program

Introduction

Everscale is an asynchronous message passing system of stateful programmable agents called smart-contracts. Those contracts manage digital assets of users on their behalf. In this context, digital assets mean tokens, NFTs, native cryptocurrency, etc. Some smart-contracts lock value worth millions of dollars. Hence, to be trustworthy, the Everscale platform has to provide security, integrity and availability of user assets.

Program Objectives

The Everscale Bug Bounty Program aims to:

- Provide security researchers, white hackers, developers and other interested parties incentives to report found severe vulnerabilities to Everscale Platform developers
- Rapidly improve Everscale Platform security by addressing critical and major vulnerabilities in a timely, confidential manner
- Improve the overall public image of the platform as security-oriented and trustworthy

Definition of a Bug

Three pillars of the blockchain platform reliability are:

- *Security* - digital assets are allowed to be managed only by eligible users
- *Integrity* - digital assets are managed exactly in a way anticipated by users
- *Availability* - digital assets are constantly accessible to be managed for eligible users

Any Everscale Platform software defect possibly leading to partial or full disruption of one of the above is called ***a bug***. Throughout the document, we use the terms *bug, vulnerability* and *software defect* interchangeably.

Bug Bounty Scope

We are interested in finding bugs in the following software components. ***Only master branches*** of those repositories are in scope of this Bug Bounty.

- Everscale Node modules:
    - TVM virtual machine
    - Node protocol
    - Transaction Executor
    - Main data structures:
        - block structures
        - Low-level types

- TON Solidity Compiler
- TVM Linker
- Config smart-contract
- Safe Multisig Wallet
- Wrapped Ever
- Flat Qube
- Octus Bridge modules:
    - Relay
    - Contracts

Bugs Severity Ranking

To rank submitted bug reports it is suggested to use the DREAD[1] vulnerabilities rating system. The reported bug is evaluated by the following criteria:

- **D**amage potential - If the threat is exploited, how severe consequences are?
- **R**eproducibility - How easy is it to reproduce the attack?
- **E**xploitability - How easy is it to perform the attack?
- **A**ffected users - After successful attack, how many users would be affected and how important are they?
- **D**iscoverability - How easy is it to spot the vulnerability in the software?

For a specific bug, each criteria is assigned a number from 0 to 10. The overall bug rating is an arithmetic average of all those values. The higher the rating, the more valuable the finding, hence the higher reward.

Involved Parties

We distinguish the following parties involved in the Bug Bounty application and evaluation process:

- **Applicant** is a person that found software vulnerability and wants to submit it and receive the reward.
- **Reviewers** - a group of experts evaluating the submission.
- **Administrator** - a person responsible for receiving bug reports from Applicants and transferring them to the Review Committee, conducting their evaluation of the report. This person is the manager of the whole evaluation process, starting from receiving the initial report from the Applicant, and up to transferring the final reward, if any.

Application and Evaluation Process

1. Applicant conducts the vulnerability report that should reflect the following:
    a. Threat short title
    b. Affected components

[1] Developed by Microsoft.

    c. Description
    d. Result of attack
    e. Mitigation strategies
    f. DREAD vulnerability evaluation with rational for each item
2. Applicant sends the report to the Administrator. From that very moment, all details of the vulnerability shall stay undisclosed! This is of paramount importance for both Platform developers and the Applicant.
3. Administrator acknowledges the report and does the pre-evaluation of the report. If the report looks reasonable and within the scope of Bug Bounty Program, then the report is sent to the corresponding Reviewers for further in-depth evaluation. If the report is not good enough, or out of scope, the Administrator informs the Applicant about that.
4. Reviewers evaluate the vulnerability report and prepare the Report Evaluation Summary document, (RES). The RES document contains the experts opinion on how severe the presented vulnerability is, and if it deserves to be fixed.
5. Reviewers send the RES to the Administrator.
6. Administrator acknowledges the Applicant about the RES evaluation. If RES is positive, and the Applicant agrees on it, the Administrator initiates the reward transaction to the Application wallet according to the Reward Structure section.
7. After the bug is fixed, all involved parties are free to report the vulnerability to the public. Before that, it is strictly prohibited.

Report Evaluation Summary

Reviewers are responsible for in-depth technical evaluation of the report. In particular, they evaluate the submitted report from the perspective of DREAD:

- How severe are the consequences if the vulnerability gets exploited?
- How easy is it to reproduce the bug?
- How easy is it to exploit the bug on the real network?
- If the attack succeeds, what are the consequences for users and how important are they?
- How difficult is it to discover this bug having the software artifacts at hand?

For each item, RC assigns a number from 0 to 10. The final score is an arithmetic average of those. This score equipped with a short rationale is put into the Report Evaluation Summary document.

Reward Structure

The reward depends on the final score stated in the Report Evaluation Summary. The following severity levels and their respective rewards are suggested.

Bug Severity	**DREAD score**	**Reward (USD equivalent)**
Critical	[8 .. 10)	50000

High	[5 .. 7)	30000
Medium	[3 .. 5)	10000
Low	[1 .. 3)	1500
Information	[0 .. 1)	0

NOTE:
Rewards are paid in USDT tokens, LEVER[2] tokens or some of their combinations, depending on the current Bug Bounty budget restrictions. The specific reward structure is decided individually, after the bug report is accepted and approved. The current Bug Bounty budget restrictions are decided by DeFi Alliance members.

NOTE:
The Bug Bounty program may be stopped for some period of time without prior notice. In this case, if there are pending bug reports, they will be processed on a fair basis after the program is resumed.

Conflicts Resolution

In case the Applicant does not agree with the Reviewers decision, or Administrator decision regarding their bug report, the Administrator reserves the right to draw the final decision on the vulnerability severity evaluation and the reward. It is well understood by all parties that unfair or inadequate evaluation of Applicant's work will result in a significant reputation harm.

Non-Disclosure Agreement

After the application process is started, all involved parties shall not publish the vulnerability details within the public space. The vulnerability publishing is prohibited up until the moment the vulnerability is fixed in the software and corresponding updates are uploaded into the network.

Rules of Conduct

- Applicants shall not break the rules stated in this document. Otherwise, an Applicant may be banned from the program, temporarily or forever.
- Applicants shall not be affiliated with EverX Labs and Broxus companies.
- Applicants shall not use social engineering techniques to gain knowledge of bugs.
- Applicants shall not impersonate others work. It is prohibited to publish the report based on findings of another person.
- Applicants shall not test found vulnerabilities on the main network or public test networks. Only private networks shall be used for that purpose.

[2] The LEVER token is an EVER equivalent locked for 2 years after being minted. This token is not yet introduced into wide usage. A brief description may be found here.

- After the report is submitted, Applicant shall not disclose the report details to the public. This rule is canceled after the vulnerability is fixed.
- Both Administrator and Reviewers shall not disclose the report details to the public until the vulnerability has been fixed and reward is paid.
- Applicants shall not report a bug that was previously disclosed by others.
- Applicants shall not underreport or misinterpret found vulnerabilities
- Administrator shall provide feedback to the Applicant, and act on the report in a timely manner. If the process stalls, the Applicant has the right to ask for an action.

Contacts

Bug report shall be sent to bugreport@everx.dev

The applicant may also contact our support team on Telegram:
@Custler, @lotumba, @isheldon, @Rexagon, @UsernameBarsik, @prigolovko

Appendix A. Vulnerability Report Example

Here, we provide a brief example of how the Vulnerability Report should look like for some fictional vulnerability in the SafeMultisigWallet.

Threat title	Safe MultiSig allows unauthorized token transfers
Affected components	All deployed SafeMultiSig wallets in the network
Description	We consider the method SafeMultisigWallet: Transfer(param:uint, send_to: address) as vulnerable. If the value of param = 1, and the wallet has less than 3 owners, the method will not perform proper user authorization and transfer all the wallet funds to send_to address. This is due to incorrect check at line 123. The current check is: <incorrect check goes check>, the correct check would be: <correct check goes here>.
Result of attack	Complete loss of user funds in the worst case.
Mitigation strategies	To fix the code error: - Fix the check at line 123. - Inspect all probable places with similar checks in the source code

<table>
<tr><td></td><td>To overcome the bug on the deployed wallets:

- If you have less than 3 custodians, add extra wallet custodians so their total amount becomes at least 3.</td></tr>
<tr><td>DREAD score</td><td>
<table>
<tr><th>Criteria</th><th>Rational</th></tr>
<tr><td>Damage potential: 9</td><td>Wallet users lose their funds</td></tr>
<tr><td>Reproducibility: 9</td><td>The bug is reproducible on all wallets with a number of custodians less than 3.</td></tr>
<tr><td>Exploitability: 9</td><td>The bug can be exploited by a simple transaction with proper parameters</td></tr>
<tr><td>Affected users: 6</td><td>All users of the network with SafeMultiSig wallets with custodian number less than 3 . By our estimation, the total number of such user wallets is ~200 which is less than 5% of all the population, with total funds locked for 1$ mln worth of Ever.</td></tr>
<tr><td>Discoverability: 9</td><td>The incorrect check is easy to spot. You need no special knowledge to discover this bug.</td></tr>
</table>
Total: (9 + 9 + 9 + 6 + 9)/5 = 8.4 Critical
</td></tr>
</table>