Coverity Scan Vulnerability Assessment Report

Hardhard Enterprises

**T3** 2022

Statement of Intent

Overview

This document aims to provide a record of the vulnerability assessment performed on a specific issue from the Coverity SAST scan for the NASA ION Open-Source code 4.1.1 project and is a follow-up report from the static analysis report.

The primary purpose of this document is to provide an in-depth assessment of the identified vulnerability. Assessment includes but is not limited to; Recreating the issue, investigating the potential for exploitation, deploying various exploits and reporting on the researcher's findings.

Depending on findings, secondary purposes can include but are not limited to listing/providing recommended fixes alongside a list of potential attack vectors/scenarios.

Reporting Best Practices

Please ensure best practices are kept when completing the document via regularly updating the Acronyms and Abbreviations table alongside any iterations made to the Document History table. Adhering to these practices will better inform readers and allow other members to identify any updates and progress made across trimesters easily.

When using code snippets, please use screenshots that are clear and easy to read. Alternatively, use the Microsoft Word built-in code formatter found [here](https://appsource.microsoft.com/en-us/product/office/WA104382008?tab=Overview).

Document Naming Conventions

Naming conventions for this file are as follow; VAR\_{CID}. For example, when investigating issue 123456, the file name would be VAR\_123456.docx

Document History

|  |  |  |  |
| --- | --- | --- | --- |
| **Dates** | **Version** | **Author** | **Comments** |
| 5/12/2022 | V1.0 | Tristen Clifton | Initial document |
|  |  |  |  |
|  |  |  |  |

Table of Content

Contents

[Introduction 3](#_Toc119932107)

[Objective 3](#_Toc119932108)

[Scope 3](#_Toc119932109)

[Acronyms and Abbreviations 3](#_Toc119932110)

[Vulnerability Assessment Report 4](#_Toc119932111)

[Recreating the Issue 4](#_Toc119932112)

[Overview 4](#_Toc119932113)

[Finding and Observations 4](#_Toc119932114)

[Exploiting the Issue 4](#_Toc119932115)

[Overview 4](#_Toc119932116)

[Finding and Observations 4](#_Toc119932117)

[Supporting Evidence 4](#_Toc119932118)

[Recommendation(s) to fix. 4](#_Toc119932119)

[Conclusions and Recommendations 4](#_Toc119932120)

[References 5](#_Toc119932121)

[Appendix 6](#_Toc119932122)

# Introduction

## Objective

The primary objective of this assessment is to determine whether the defects identified in the Coverity Report for the ION Open Source 4.1.1 project are:

* Indeed, recreatable.
* Indeed, exploitable.

The secondary objective of this analysis, where applicable, is to provide the following:

* Comprehensive details on the exploit itself.
* Any additional exploitable scenarios.
* A detailed list of attack vectors.
* Recommendation(s) to fix.

## Scope

This vulnerability assessment is limited to the **Out-of-bounds read** type defect identified in the following CIDs: **1520868**

# Acronyms and Abbreviations

Please keep an updated list of acronyms and abbreviations used throughout the report.

|  |  |
| --- | --- |
| **Acronym / Abbr.** | **Meaning** |
| DTN | Delay/Disruption Tolerant Network |
| ION | Interplanetary Overlay Network |
| CID | Coverity Issue Identification Number |

# Vulnerability Assessment Report

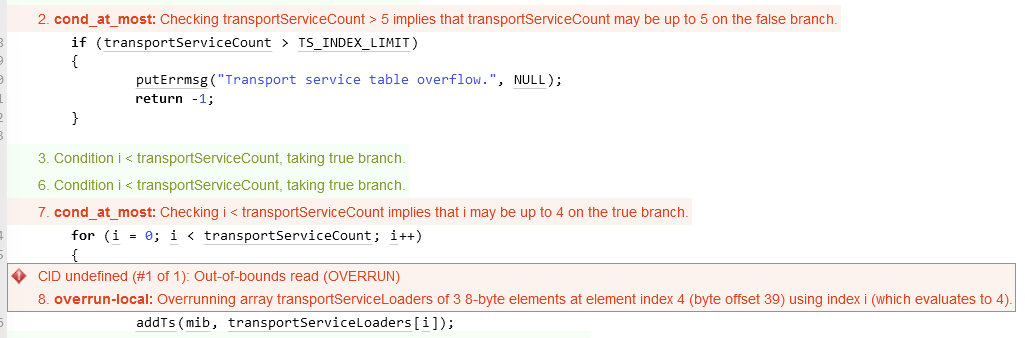
## Recreating the Issue

### Overview

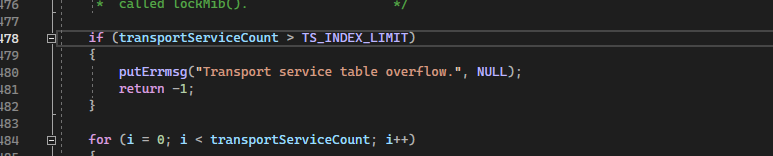
When performing static code analysis using the ION Open Source 4.1.1 dashboard for CID 1520868, there is a high impact problem that involves a reported out of bounds read. An issue of this category could potentially cause the program to crash or cause other unwanted side effects.

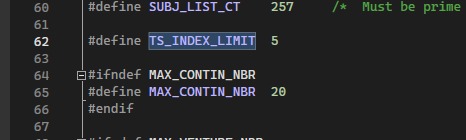
### Finding and Observations

The static analysis tool reported that there was for loop that iterated through an array that could potentially repeat to a value that was higher than what could be indexed by the array. This is because the value was initially checked for a limit of 5, however it seemd possible that a value of 5 could be generated resulting in attempt to index an array outside the bounds of the array, in the test case the array was size 4 but the loop could potentially go up to 5.



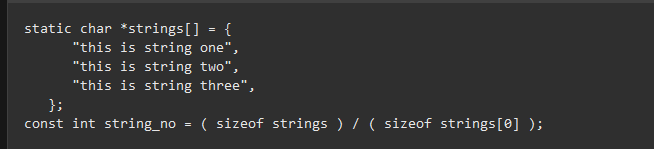
As you can see, a physical limit of 5 is defined as a static value.



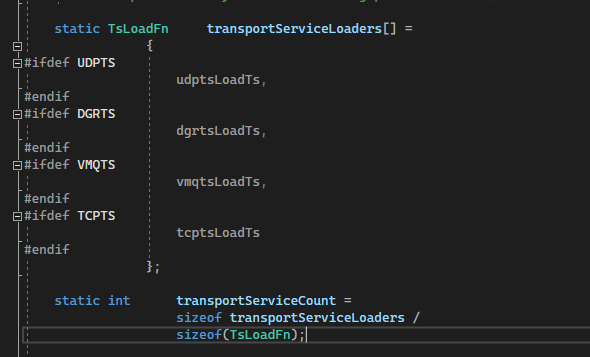


However with further digging it was found that transportServiceCount could never exceed the size of the array, this is because it is generated by calucating the size of the array.

Traditional methods of calcauting the size of an array in the c language use a method like this.



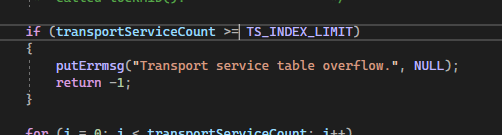
Within the source code a siliar method is used to set transportServiceCount to a value that represents the size of the array.

  
therefore a value cannot be generated that exeeds the bounds of the array

# Conclusions and Recommendations

In conclusion, the reported generated a false positive, this is because the standard process would be to create a limit directly for the loop, rather than for the value.

One change that could be implemented that would not effect the function of the code but prevent the issue from being detected in future scans would be to change the value from > to >= as it would prevent the system from thinking that a value of 5 could pass through to the loop.



This change as described in this image would prevent the issue from being detected by the scan, as well as a failsafe since the loop is already bound to 0-3.

References  
Please keep an updated references list in APA7; The Deakin referencing guide can be found [here](https://www.deakin.edu.au/__data/assets/pdf_file/0009/2236752/Deakin-guide-to-APA7.pdf).

Appendix

Include additional information/documentation here to help the readers understand complex information.