**T1** 2023

Coverity Scan Static Analysis Report

Hardhard Enterprises

Statement of Intent

Overview

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

The primary purpose of this document is to validate the issue identified via the automated detection process to eliminate false positives.

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

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. This will 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 words 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; SAR\_{CID}. For example, when investigating issue 123456 the file name would be SAR\_123456.docx

Document History

|  |  |  |  |
| --- | --- | --- | --- |
| **Dates** | **Version** | **Author** | **Comments** |
| 6/05/2023 | V0.1 | Moe Khant Kyaw | Initial Document and analysis |
| 12/05/2023 | V0.2 | Moe Khant Kyaw | Finalizing Investigation and documentation |

Table of Content

Contents

[Introduction 3](#_Toc119848724)

[Objective 3](#_Toc119848725)

[Scope 3](#_Toc119848726)

[Acronyms and Abbreviations 3](#_Toc119848727)

[Code Review and Analysis 4](#_Toc119848728)

[Outcomes 4](#_Toc119848729)

[Observations 4](#_Toc119848730)

[Supporting Evidence 4](#_Toc119848731)

[Conclusions and Recommendations 4](#_Toc119848732)

[References 5](#_Toc119848733)

[Appendix 6](#_Toc119848734)

# Introduction

## Objective

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

* Indeed, defects.
* Potentially exploitable.

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

* Recommendation(s) to fix.
* Any exploit for consideration.

## Scope

This static code analysis is limited to the ***High Impact Quality*** type defect identified in the following CIDs:  
***1520670***

# Acronyms and Abbreviations

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

|  |  |
| --- | --- |
| **Acronym** | **Meaning** |
| DTN | Delay/Disruption Tolerant Network |
| ION | Interplanetary Overlay Network |

# Code Review and Analysis

## Overview

CID 1520670 has been marked as high impact issue, Out-of-bounds access type under Memory-Corruptions category by Coverity. This issue is marked as CWE-117: Improper restriction of Operations within the Bounds of a Memory Buffer, also known as buffer overflow, which happens when the program is trying to read or write outside of the limit of intended buffer.

## 

## Observations

CID 1520670 is found in “*loadCriticalBestRoutesList*” function under “/bpv7/cgr/libcgr.c” filepath. This function picks out the best route to specific end node by utilizing the list of proximate nodes, (routingObj->proximateNodes). If successful, the route is added it to the bestRoutes list and clean up the routes list.

This issue happens in line 2234, when it is iterating over a list of proximateNodes. elt2 variable, of PsmAddress data type, has been assigned 18446744073709551615 in the for loop on line 2232, in a call to ‘sm\_list\_first’ function. The loop begins by taking elt2 as the first element using ‘sm\_list\_first’ function and continuing until elt2 becomes invalid. elt2 is passed as parameter of sm\_list\_next function (See Figure 1). The value of elt2 is exceptionally large for sm\_list\_next function to handle and it is invalid for psmAddress data type. This raises as out-of-bounds access error in Coverity scan.

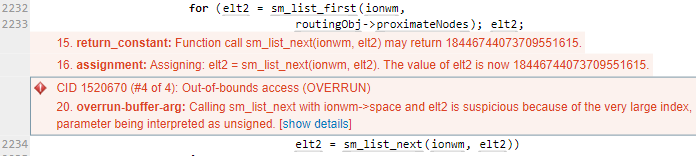


Fig 1: Out-of-bounds access error

# 

# Conclusions and Recommendations

This issue can expose to risks such as memory corruption, data integrity issues, program crashes and security vulnerabilities. Attackers can exploit this to gain unauthorized access and perform malicious activities.

To prevent this issue from happening, we can do the followings:

* Verify the input value of parameters for the functions
* Validation for the value of elt2 to ensure it is within the limit for psmAddress data type.
* Ensure the list ‘routingObj->proximateNodes’ is valid and can handle enough elements to support the loop iterations.

It is important to validat input values to mitigate out-of-bound access error.

References

*Common weakness enumeration* (no date) *CWE*. Available at: https://cwe.mitre.org/data/definitions/119.html (Accessed: May 6, 2023).

Appendix

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