Coverity Scan Static Analysis Report

Hardhard Enterprises

**T1** 2023

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** |
|  | V0.1 |  |  |
|  |  |  |  |
|  |  |  |  |

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 ***Untrsuted loop bound*** type defect identified in the following CIDs:  
***CID1520706***

# 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

## Outcomes

The report looks at the untrusted loop bounds error from CID1520706.

## Observations

## 

This issue is highlighting a potential security vulnerability or bug in the code related to using a tainted loop boundary.

The *TAINTED\_SCALAR* loop has a loop bound that is consisted untrusted. This means that the value may come from an untrusted source or is not properly validated or sanitissed before being used as a loop boundary.

The *tainted\_data* part of the message points out that there is some data, potentially a variable or expression, that is considered tainted. This typically means that it originates from an untrusted or external source that hasn’t been validated or sanitized to ensure it’s safe to use.

The tainted data, in this case, the expression *aduLen*, is being passed as an argument to a function call *run\_ltpdriver*.

The function *run\_ltpdriver* is using the tained expression *aduLen* as a loop boundary, meaning it determines how many times a loop inside *run\_ltpdriver* will run.

The issue is warning that there is a potential security problem in the case where the trained data *aduLen* is being used to control a loop. Using tainted data in loop boundaries can be a source of vulnerabilities like buffer overflows, infinite loops, or other unexpected behavoiur if the tainted data isn’t properly validated or sanitized before being used in this context.

# Conclusions and Recommendations

To address this issue, the code needs to be reviewed to ensure any data coming from untrusted sources is properly validated and sanitised before being used as a loop boundary, and consider using safer constructs or methods to control your loops to prevent security vulnerabilities.

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.