# **snyk** Vulnerability DB

Snyk Vulnerability Database > npm > rollup-plugin-dev-server

## **Directory Traversal**

Affecting rollup-plugin-dev-server package, versions \*



### How to fix?

There is no fixed version for rollup-plugin-dev-server .

#### Overview

rollup-plugin-dev-server is a server for serving rolled up bundles like webpack-dev-server.

Affected versions of this package are vulnerable to Directory Traversal. There is no path sanitization in readFile operation inside the readFileFromContentBase function.

### PoC by JHU System Security Lab

1. Create and run a server:

var server = require("rollup-plugin-dev-server"); server({ host: 'localhost', port: 10001 })

- 2. Create a file sensitive-file in the server's parent directory
- 3. curl -s --path-as-is http://127.0.0.1:10001/../sensitive-file

#### Detail

A Directory Traversal attack (also known as path traversal) aims to access files and directories that are stored outside the intended folder. By manipulating files with "dot-dot-slash (.../)" sequences and its variations, or by using absolute file paths, it may be possible to access arbitrary files and directories stored on file system, including application source code, configuration, and other critical system files.

Directory Traversal vulnerabilities can be generally divided into two types:

Information Disclosure: Allows the attacker to gain information about the folder structure or read the contents of sensitive files on the system.

st is a module for serving static files on web pages, and contains a vulnerability of this type. In our example, we will serve files from the public route.

If an attacker requests the following URL from our server, it will in turn leak the sensitive private key of the root user.

curl http://localhost:8080/public/%2e%2e/%2e%2e/%2e%2e/%2e%2e/%2e%2e/root/.ssh/id\_rsa

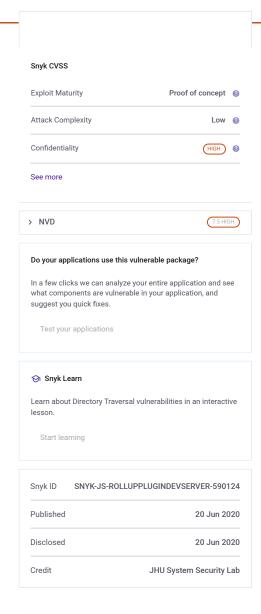
Note %2e is the URL encoded version of . (dot).

Writing arbitrary files: Allows the attacker to create or replace existing files. This type of vulnerability is also known as Zip-Slip.

One way to achieve this is by using a malicious zip archive that holds path traversal filenames. When each filename in the zip archive gets concatenated to the target extraction folder, without validation, the final path ends up outside of the target folder. If an executable or a configuration file is overwritten with a file containing malicious code, the problem can turn into an arbitrary code execution issue quite easily.

The following is an example of a zip archive with one benign file and one malicious file. Extracting the malicious file will result in traversing out of the target folder, ending up in /root/.ssh/ overwriting the authorized\_keys file:

2018-04-15 22:04:29 ..... 19 19 good.txt 2018-04-15 22:04:42 ..... 20 20 ../../../../root/.ssh/authorized\_keys



Q Search by package name or CVE

Report a new vulnerability Found a mistake?

Snyk Open Source
Snyk Code
Snyk Container
Snyk Infrastructure as Code
Test with Github
Test with CLI
RESOURCES
Vulnerability DB
Documentation

PRODUCT

Blog FAQs

COMPANY

About

Jobs

. .

Policies

Do Not Sell My Personal Information

CONTACT US

Support

Report a new vuln

Press Kit

Events

FIND US ONLINE

TRACK OUR DEVELOPMENT





© 2022 Snyk Limited

Registered in England and Wales. Company number: 09677925

Registered address: Highlands House, Basingstoke Road, Spencers Wood, Reading, Berkshire, RG7 1NT.