



Depfuzzer

Fuzzing confused dependencies

05/07/2024



Who are we



Pierre Martin

Pentester

@_Worty



Kévin Schouteeten

Pentester

@Scouty__



- French offensive security company
- 170 security experts
- 4 departments :
 - Pentest / Redteam
 - RE / VR
 - Development
 - IR
- Hexacon (Paris - october 2024)



Overview

- Software development registries landscape.
- Example of confused dependency and how to exploit it.
- A description of how Depfuzzer works.
- Demo.
- Conclusion.



Registries landscape



Registries landscape

- Open-source libraries and packages :
 - NodeJS -> <https://registry.npmjs.org/>
 - Python -> <https://pypi.org/>
 - Rust -> <https://crates.io/>
 - Golang -> <https://pkg.go.dev/>
 - ...



Registries landscape

- Package example in PyPI. :-)

```
$ pip3 install odd
Collecting odd
  Downloading odd-1.0.1-py3-none-any.whl (2.4 kB)
Installing collected packages: odd
Successfully installed odd-1.0.1

$ python3
Python 3.11.2 (main, Mar 13 2023, 12:18:29) [GCC 12.2.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> from odd import is_odd
>>> is_odd(9)
True
>>> is_odd(10)
False
```



- <https://medium.com/@alex.birsan/dependency-confusion-4a5d60fec610>

Dependency Confusion: How I Hacked Into Apple, Microsoft and Dozens of Other Companies

The Story of a Novel Supply Chain Attack



Alex Birsan · [Follow](#)
11 min read · Feb 9, 2021



Confused dependencies



Confused dependencies

Example with NPM

```
$ cat package.json
{
  "name": "dependency-confusion-demo",
  "version": "1.0.0",
  "description": "A demo to illustrate dependency confusion",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "dependencies": {
    "@pts2024-prez/private-package": "^1.0.0",
    "lodash": "^4.17.21"
  },
  "author": "Scouty",
  "license": "ISC"
}
```



Confused dependencies

Example with NPM

- Private registry

```
$ cat .npmrc  
@pts2024-prez:registry=http://localhost:4873
```

- *@pts2024-prez/private-package* installed from private registry
- *lodash* installed from default registry (*registry.npmjs.org*)



Confused dependencies

Example with NPM

- Package installation

```
$ npm install  
  
added 2 packages, and audited 2 packages in 2s  
  
found 0 vulnerabilities
```

- Package content

```
$ cat dependency-confusion-demo/node_modules/@pts2024-prez/private-package/index.js  
// index.js  
module.exports = function() {  
  console.log('This is a private package');  
};
```



Confused dependencies

Example with NPM

- *.npmrc* and *package-lock.json* alteration
 - Human error
 - Misconfiguration
 - Version control conflicts
 - Environment changes
 - Scripted automation

```
$ rm .npmrc && rm package-lock.json
```



Confused dependencies

Example with NPM

```
$ npm install
npm error code E404
npm error 404 Not Found - GET https://registry.npmjs.org/@pts2024-prez%2fprivate-package - Not found
npm error 404
npm error 404 '@pts2024-prez/private-package@^1.0.0' is not in this registry.
npm error 404
npm error 404 Note that you can also install from a
npm error 404 tarball, folder, http url, or git url.
npm error A complete log of this run can be found in: /home/user/.npm/_logs/2024-07-04T11_59_45_530Z-debug-0.log
```



Confused dependencies

Example with NPM

- Publish compromised package on *registry.npmjs.org*
- Same organization / Same name

```
$ npm publish --access public
npm notice
npm notice @pts2024-prez/private-package@1.0.0
[...]
npm notice integrity: sha512-lBrWveA9TtBjj[...]BttKY9k4LfuSw==
npm notice total files: 2
npm notice
npm notice Publishing to https://registry.npmjs.org/ with tag latest and public access
+ @pts2024-prez/private-package@1.0.0
```



Confused dependencies

Example with NPM

- Package installation

```
$ npm install  
  
added 2 packages, and audited 2 packages in 2s  
  
found 0 vulnerabilities
```

- Package content

```
$ cat dependency-confusion-demo/node_modules/@pts2024-prez/private-package/index.js  
// index.js  
module.exports = function() {  
  console.log('compromised');  
};
```



Context

Cool and
innovative mission



1400+
repos to scan



- Mission for one of our client
- Check if its possible to takeover dependencies
- 1400+ projects in the GitHub organisation
- Time for automation

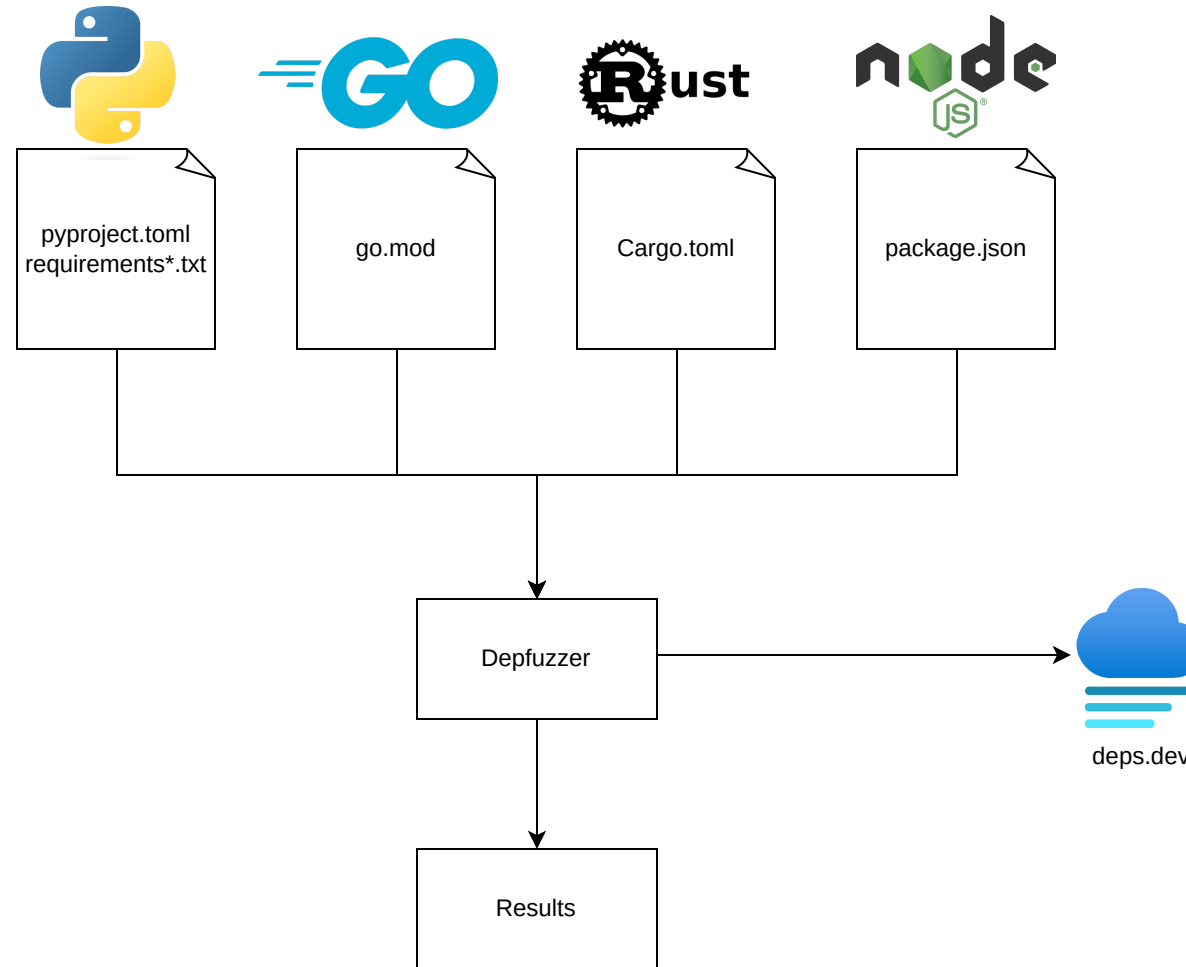


Depfuzzer



Depfuzzer

Overall diagram



Depfuzzer

<https://deps.dev>

The screenshot shows the homepage of deps.dev. The browser address bar displays "deps.dev". The main heading is "open / source / insights". Navigation links for "About", "Documentation", and "Blog" are visible. The primary message is "Understand your dependencies", followed by a paragraph explaining the importance of dependency graphs. A search bar at the bottom contains the text "Search for open source packages, advisories and projects" and a dropdown menu set to "All systems". A blue "Search" button is positioned to the right of the search bar. On the right side, a list of dependency systems and their package counts is displayed:

npm	PACKAGES	3.20M
Go	MODULES	1.20M
Maven	ARTIFACTS	625k
PyPI	PACKAGES	527k
NuGet	PACKAGES	399k
Cargo	CRATES	150k



Browser address bar: `deps.dev/pypi/requests/2.32.3/dependencies`

open/source/insights Search for open source packages, advisories and projects | PyPI 🔍

PyPI package **requests** ✔️ 2.32.3

Overview **Dependencies** Dependents Compare Versions

Filter dependencies by name, license, security advisory and more Table Graph

Package	Notes	Relation ↑	License	Dependencies
▶ certifi 2024.6.2		Direct	MPL-2.0	0
▶ charset-normalizer 3.3.2		Direct	MIT	0
▶ idna 3.7.0		Direct	UNKNOWN	0
▶ urllib3 2.2.2		Direct	UNKNOWN	0



Demo



Results

- We found some dependencies confusion
- We fuzz a lot of open-source projects and find few issues



Limitations

- A lot of false positives (parsing requirements files is complex)
- This tool is not meant to be 100% accurate (act as a PoC)
- Exploiting this might break some CI/CD, so be careful



Further work

- Scan for :
 - Nuget packages (.NET)
 - PHP Composer
- Amelioration for the requirements files parsing to reduce false positive



Conclusion



Conclusion

- Attacker can hide malicious code into legitimate one
- Always use an internal package registry in your company !



Conclusion

- Depfuzzer a tool to find confused dependencies
- PR are welcome
- <https://github.com/Synacktiv/DepFuzzer>



 **SYNACKTIV**



<https://www.linkedin.com/company/synacktiv>



<https://twitter.com/synacktiv>



<https://synacktiv.com>