PyCon APAC 2023 in Tokyo, Japan

Dodging Dependency Confusion In A Nutshell

Lemtaffah Khalil Abdellah

Outline of the talk

- Whoami?
- Introductory example
- PyPI and packages overview
- What is a dependency?
- Typosquatting
- Dependency confusion
- Real-world examples
- Dodging techniques

Whoami?

Whoami?

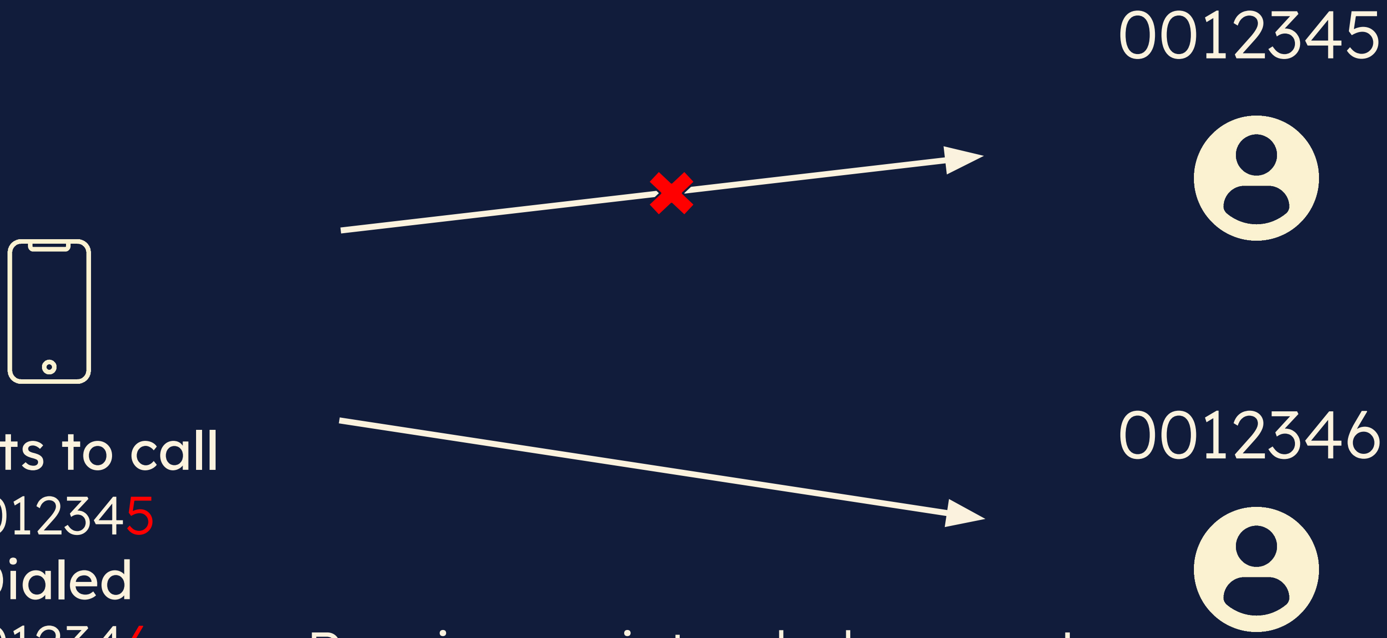
- Cybersecurity MSc Student
- Vulnerability Management Specialist at ERICSSON
- Web security enthusiast
- Free time bug bounty hunter
- Loves python, nature and cats

Quick disclaimer

The content discussed on this talk is for educational and awareness purposes only.

Wrong phone number example

Wrong phone number example

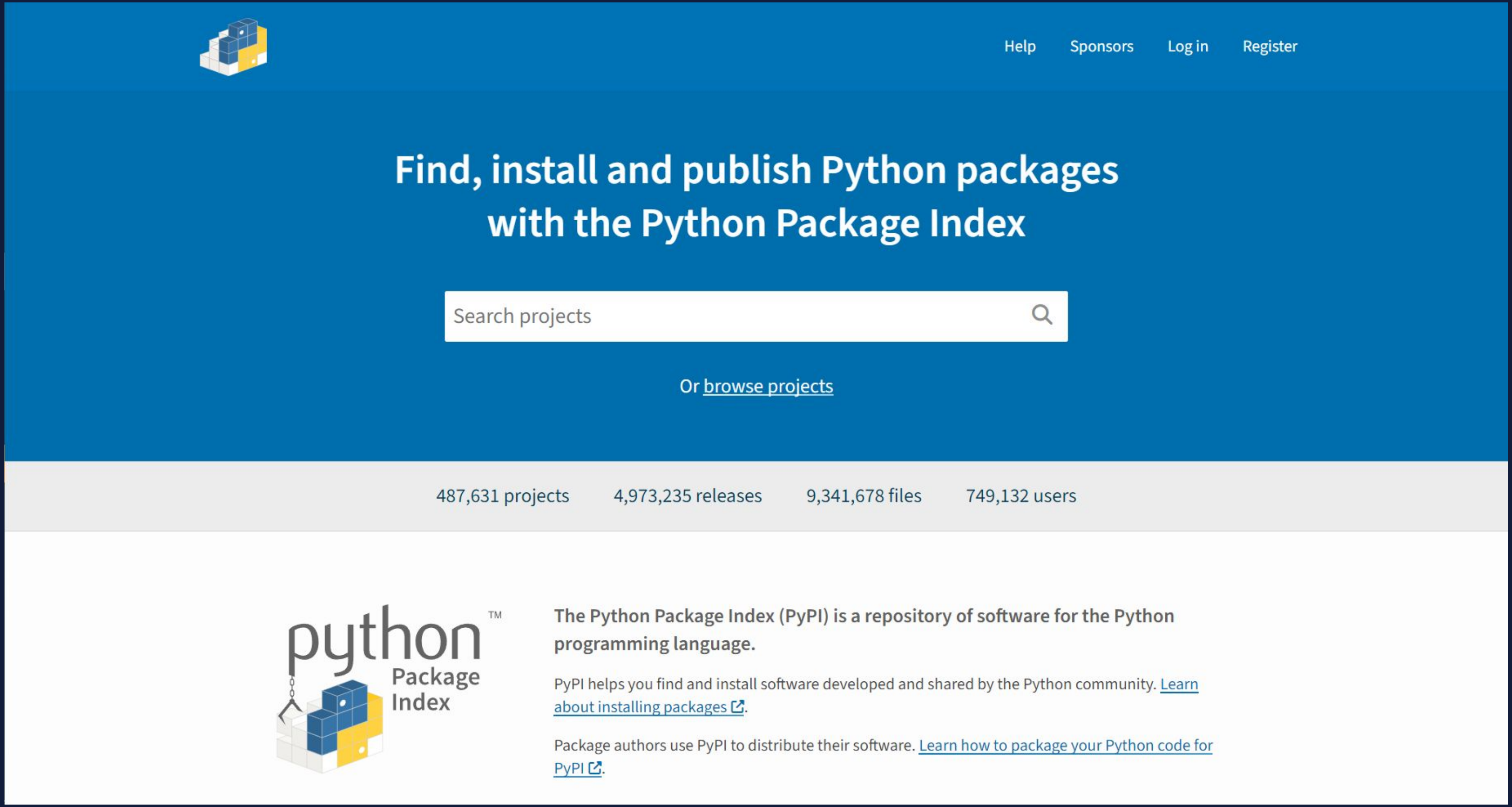


Wants to call
0012345
Dialed
0012346

Receives unintended answer!

PyPI overview

PyPI overview



Find, install and publish Python packages with the Python Package Index

Or browse projects

487,631 projects 4,973,235 releases 9,341,678 files 749,132 users

The Python Package Index (PyPI) is a repository of software for the Python programming language.

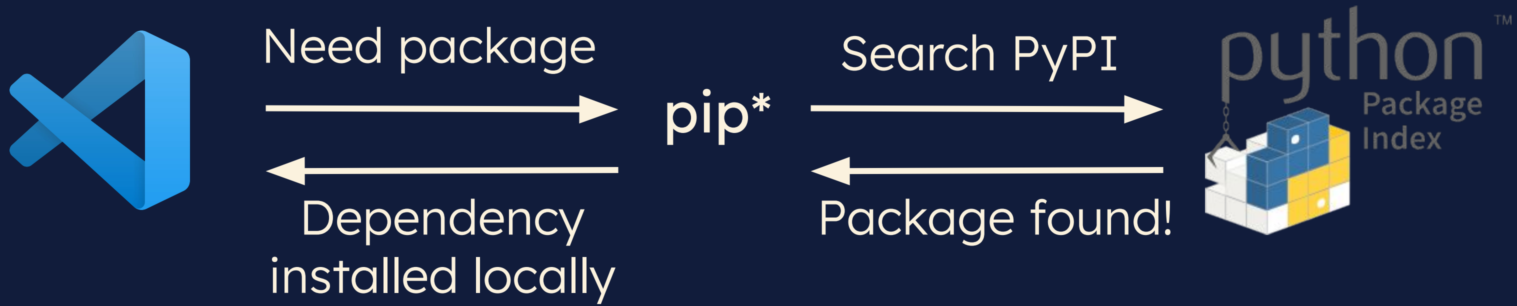
PyPI helps you find and install software developed and shared by the Python community. Learn about installing packages.

Package authors use PyPI to distribute their software. Learn how to package your Python code for PyPI.

PyPI overview

Python Package Index - A centralized and big repository containing hundreds of thousands of python projects “packages”.

PyPI overview



*pip needs to be installed from PyPI

PyPI overview

A lot of package managers:

npm python™ Package Index CONDA

PyPI overview

When you download and install a package from those sources, then you trust the code base to run on your machine.

What is a dependency?

What is a dependency?

A set of code which helps your project perform a certain action, such as:

- Requests - for making HTTP requests
- Pandas - for data manipulation
- Beautifulsoup4 - for web scraping
- …

What is a dependency?

```
$ pip install package_name
```

What is a dependency?

```
$ cat requirements.txt
requests==2.26.0
flask==2.1.1
numpy==1.21.3
pandas==1.3.3
matplotlib==3.4.3
```

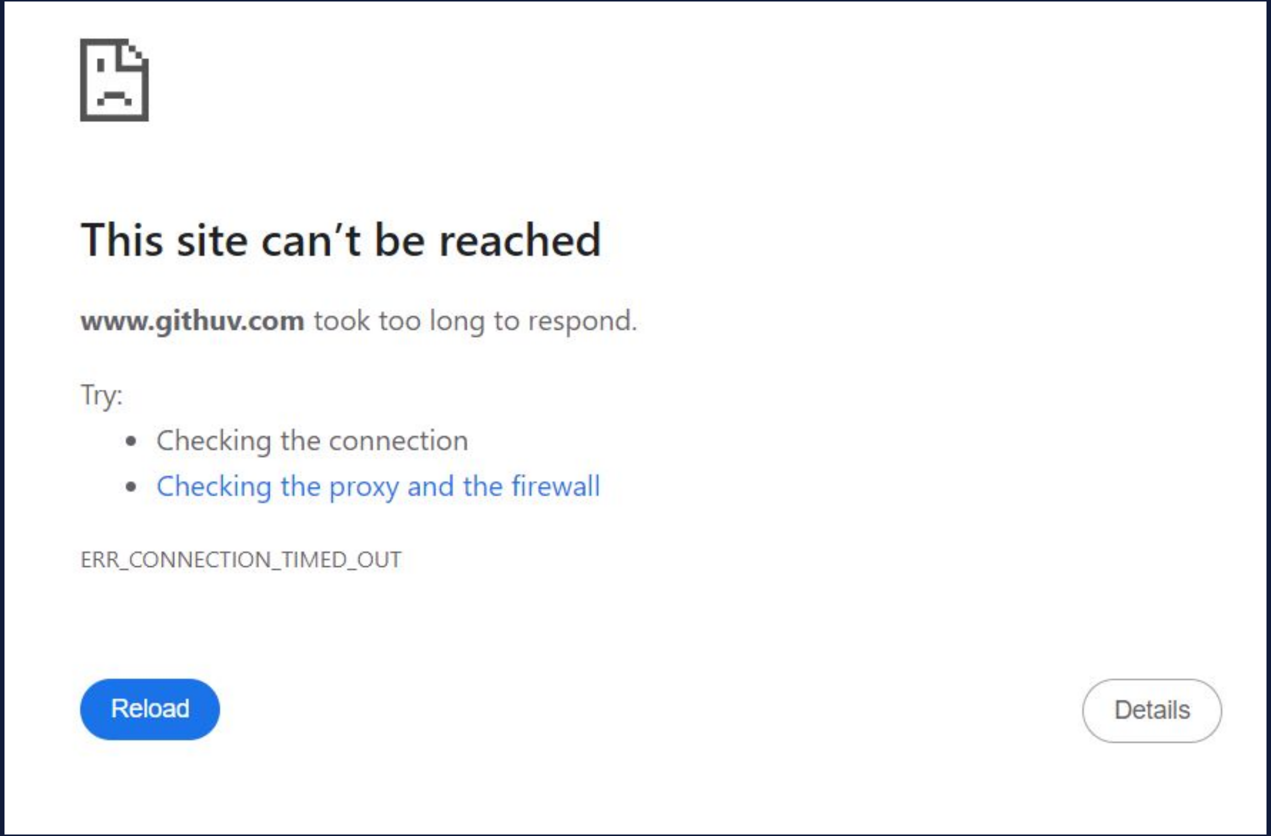
Typosquatting

Typosquatting

A method that exploits human typos, by registering false domain names.

E.g.: https://githuv.com/

Typosquatting



This site can't be reached

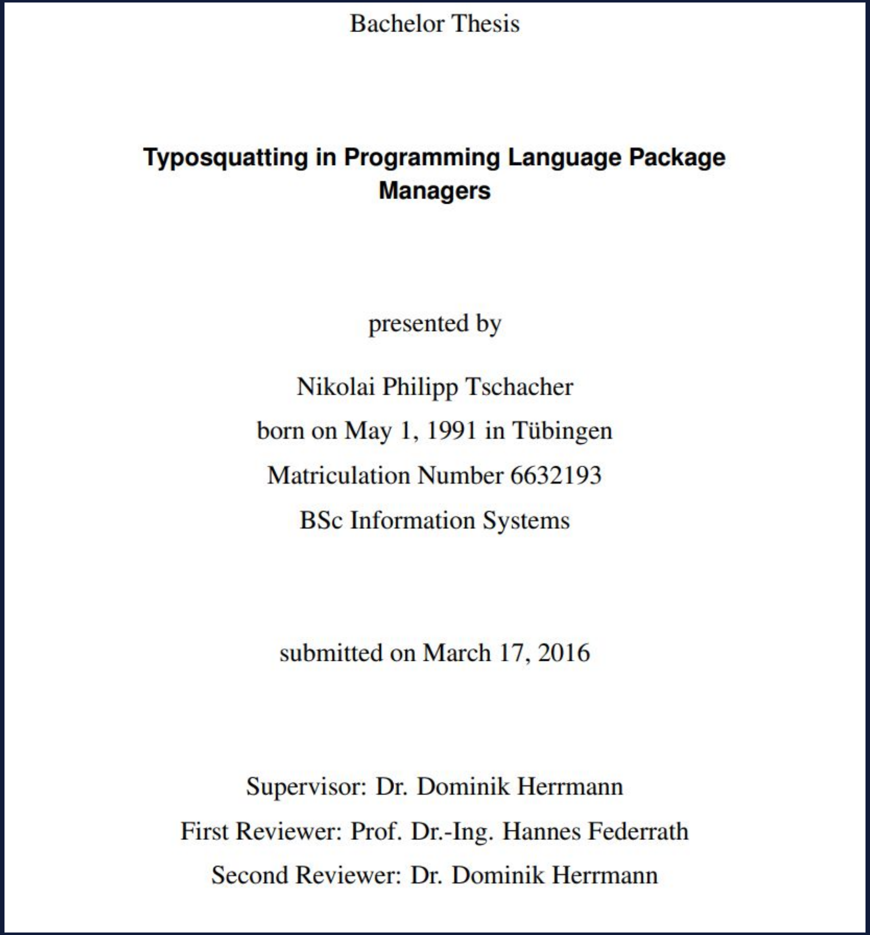
www.githuv.com took too long to respond.

Try:

- Checking the connection
- Checking the proxy and the firewall

ERR_CONNECTION_TIMED_OUT

Typosquatting



Bachelor Thesis

Typosquatting in Programming Language Package Managers

presented by

Nikolai Philipp Tschacher

born on May 1, 1991 in Tübingen

Matriculation Number 6632193

BSc Information Systems

submitted on March 17, 2016

Supervisor: Dr. Dominik Herrmann

First Reviewer: Prof. Dr.-Ing. Hannes Federrath

Second Reviewer: Dr. Dominik Herrmann

Typosquatting

Results:

- 17 289 unique hosts installed the typosquatted packages
- 50% of the installations were done with administrative rights
- Highly secured institutions were victims

Dependency confusion

Dependency confusion

It occurs when pip is confused about the dependency, whether:

- The package doesn’t exist anymore
- No specified version
- The package name is misspelled

Dependency confusion

- The package doesn’t exist anymore:

```
requests==2.26.0
flask==2.1.1
numpy==1.21.3
internal-lib-x==1.0.0
internal-lib-y==0.0.9
pandas==1.3.3
matplotlib==3.4.3
```

Dependency confusion

- The package doesn’t exist anymore:

```
requests==2.26.0
flask==2.1.1
numpy==1.21.3
internal-lib-x==1.0.0
internal-lib-y==0.0.9
pandas==1.3.3
matplotlib==3.4.3
```

Dependency confusion

- The package doesn’t exist anymore:

```
requests==2.26.0
flask==2.1.1
numpy==1.21.3
internal-lib-x==1.0.0
internal-lib-y==0.0.9
pandas==1.3.3
matplotlib==3.4.3
```

```
$ pip install -r requirements.txt
--index-url http://company.x/
```

✓

Dependency confusion

- The package doesn’t exist anymore:

```
requests==2.26.0
flask==2.1.1
numpy==1.21.3
internal-lib-x==1.0.0
internal-lib-y==0.0.9
pandas==1.3.3
matplotlib==3.4.3
```

```
$ pip install -r requirements.txt
--extra-index-url http://company.x/
```

✗

Dependency confusion

- The package doesn’t exist anymore:

`--index-url`	`--extra-index-url`
Looks for the packages in the specified repository only.	Looks for the packages in the specified repository AND PyPI.

Dependency confusion

- No specified version

```
requests
flask
numpy
internal-lib-x
internal-lib-y
pandas
matplotlib
```

Dependency confusion

- No specified version

```
requests
flask
numpy
internal-lib-x
internal-lib-y
pandas
matplotlib
```

Some conditions are needed:

- The library should be internal
- The `--extra-index-url` should be specified
- The attacker should somehow know the package names

Dependency confusion

- No specified version

```
requests
flask
numpy
internal-lib-x
internal-lib-y
pandas
matplotlib
```

$\longrightarrow$

```
internal-lib-x==99.99.99
```

pip will automatically download and install the higher version in this case from PyPI

Dependency confusion

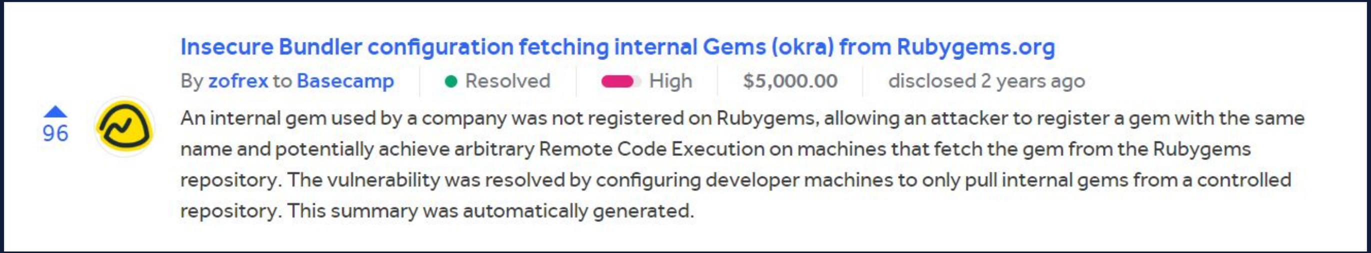
- The package name is misspelled

`reqests` $\longrightarrow$ `requests`

pip will look for `reqests` in PyPI,
if it's found, then download!

Real-world examples

Real-world examples



Insecure Bundler configuration fetching internal Gems (okra) from Rubygems.org

By zofrex to Basecamp · ● Resolved · High · $5,000.00 · disclosed 2 years ago

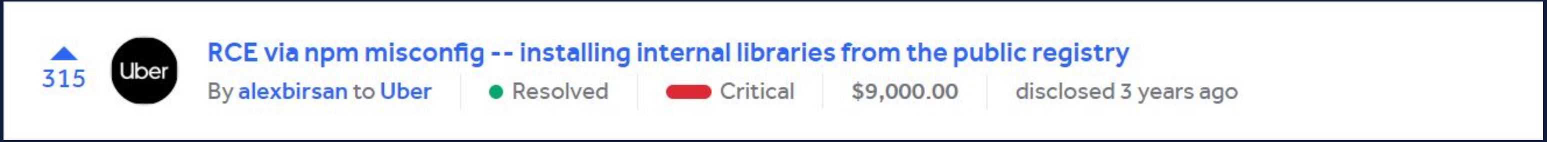
96

An internal gem used by a company was not registered on Rubygems, allowing an attacker to register a gem with the same name and potentially achieve arbitrary Remote Code Execution on machines that fetch the gem from the Rubygems repository. The vulnerability was resolved by configuring developer machines to only pull internal gems from a controlled repository. This summary was automatically generated.

Bounty: $5,000

Ruby Gems issue

Real-world examples


315

Uber

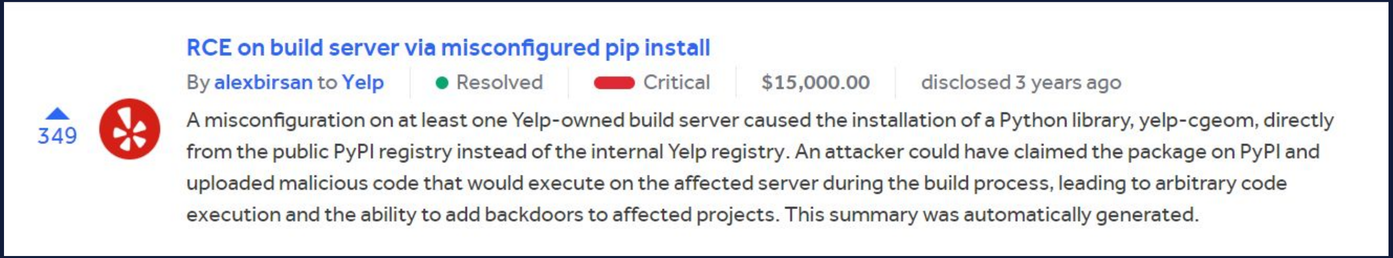
RCE via npm misconfig -- installing internal libraries from the public registry

By alexbirsan to Uber | ● Resolved | 🟥 Critical | $9,000.00 | disclosed 3 years ago

Bounty: $9,000

Npm issue

Real-world examples



RCE on build server via misconfigured pip install

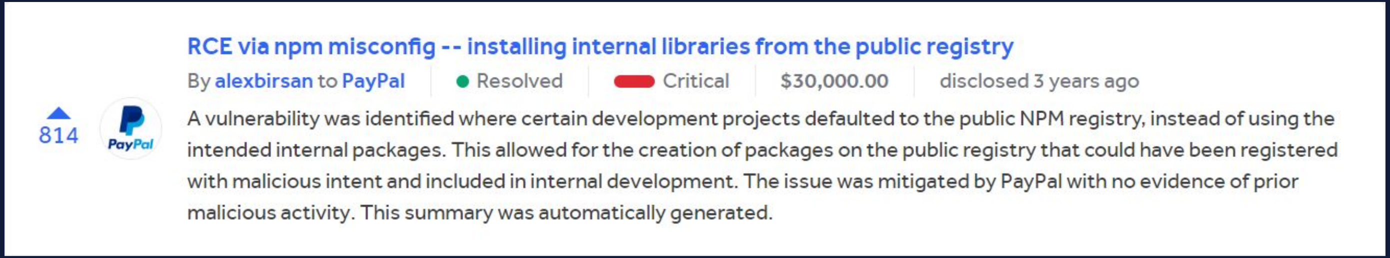
349 By alexbirsan to Yelp | ● Resolved | 🟥 Critical | $15,000.00 | disclosed 3 years ago

A misconfiguration on at least one Yelp-owned build server caused the installation of a Python library, yelp-cgeom, directly from the public PyPI registry instead of the internal Yelp registry. An attacker could have claimed the package on PyPI and uploaded malicious code that would execute on the affected server during the build process, leading to arbitrary code execution and the ability to add backdoors to affected projects. This summary was automatically generated.

Bounty: $15,000

Pip issue

Real-world examples



RCE via npm misconfig -- installing internal libraries from the public registry

By alexbirsan to PayPal | ● Resolved | 🟥 Critical | $30,000.00 | disclosed 3 years ago

814

A vulnerability was identified where certain development projects defaulted to the public NPM registry, instead of using the intended internal packages. This allowed for the creation of packages on the public registry that could have been registered with malicious intent and included in internal development. The issue was mitigated by PayPal with no evidence of prior malicious activity. This summary was automatically generated.

Bounty: $30,000

Npm issue

Dodging techniques

Dodging techniques

Dependency hashing:

- Add this argument to pip command `--require-hashes`
- Add the hash values to the requirements file

```
FooProject == 1.2 \
--hash=sha256:2cf24dba5fb0a30e26e83b2ac5b9e29e1b161e5c1fa
7425e73043362938b9824 \
--hash=sha256:486ea46224d1bb4fb680f34f7c9ad96a8f24ec88be7
3ea8e5a6c65260e9cb8a7
```

Pip documentation -> secure installs

Dodging techniques

Dependency pinning:

- Specify the exact versions you need, keep them updated

```
company-lib-x==13.3.7
```

Dodging techniques

Prevention:

- Prevent installing packages without checking
- Prevent public registry package name freedom
- Perform regular security scans
- …

Dodging techniques

Security awareness!
Raise the importance of security inside your project and take it seriously.

Thanks for your attention!

@splint3rsec

PyCon APAC 2023 in Tokyo, Japan