

Twitter Thread by [Alex Birsan](#)



Alex Birsan

[@alxbrsn](#)



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

■Check the thread after reading for a few bonus

Scope:

Other than the mentioned .NET bug, only one other team said the finding was out of scope, using this as a reason to reduce the bounty from P1 to P2.

We've also purposely avoided reporting to at least one program that had "internal or development services" listed as OOS

Pytosquatting:

In Python, the 'install' name of a package can be different from the 'import' name. This allows for a unique type of typosquatting attack by uploading under unclaimed import names on PyPI, where it can be downloaded by developers, or even automated tools.

PyYAML Documentation

PyYAML is a YAML parser and emitter for Python.

Installation

Simple install:

```
pip install pyyaml
```

To install from source, download the source package *PyYAML-5.1.tar.gz* and unpack it. Go to the directory *PyYAML-5.1* and run:

```
$ python setup.py install
```

If you want to use LibYAML bindings, which are much faster than the pure Python version, you need to download and install [LibYAML](#). Then you may build and install the bindings by executing

```
$ python setup.py --with-libyaml install
```

In order to use [LibYAML](#) based parser and emitter, use the classes `CParser` and `CEmitter`. For instance,

```
from yaml import load, dump
try:
    from yaml import CLoader as Loader, CDumper as Dumper
except ImportError:
    from yaml import Loader, Dumper

# ...

data = load(stream, Loader=Loader)

# ...
```

I tried this out against some Google open source projects and it actually got me inside two *.corp.google.com machines.

However, just like typosquatting, this mostly relies on one-off mistakes. Google did not accept it as a valid vuln, and I fully agree with their assessment.

Fun fact: those packages are long gone from PyPI but for some reason they are still up on some sort of Chinese mirror, and I still get various callbacks from China to this day.

FRI DEC 04 2020 23:14:30 GMT	CHINANET-BACKBONE No.31,Jin-rong Street	61.132.109.6	tensorflow_serving	object-detection-6454b7fcb5-ct4wk
SUN DEC 06 2020 06:46:10 GMT	CT-FOSHAN-IDC CHINANET Guangdong province network	14.152.40.70	tensorflow_serving	object-detection-f896cc99-jjjb2
SUN DEC 06 2020 17:57:20 GMT	CHINANET-BACKBONE No.31,Jin-rong Street	61.132.109.6	tensorflow_serving	object-detection-6454b7fcb5-ct4wk
SUN DEC 06 2020 19:09:31 GMT	CHINANET-BACKBONE No.31,Jin-rong Street	61.132.109.6	tensorflow_serving	object-detection-6454b7fcb5-m8xk7
SUN DEC 06 2020 19:56:15 GMT	CHINANET-BACKBONE No.31,Jin-rong Street	61.132.109.6	tensorflow_serving	object-detection-6454b7fcb5-m8xk7
MON DEC 07 2020 06:29:37 GMT	CHINANET-BACKBONE No.31,Jin-rong Street	221.228.15.78	tensorflow_serving	face-recognition-7b46b654d9-9s564
MON DEC 07 2020 06:30:19 GMT	CHINANET-BACKBONE No.31,Jin-rong Street	221.228.15.182	tensorflow_serving	object-detection-7db698448c-zz4t1
MON DEC 07 2020 06:42:17 GMT	CHINANET-BACKBONE No.31,Jin-rong Street	221.228.15.78	tensorflow_serving	face-recognition-8b9c7ff78-5hwlp
MON DEC 07 2020 11:00:57 GMT	CHINANET-BACKBONE No.31,Jin-rong Street	61.132.109.6	tensorflow_serving	object-detection-6454b7fcb5-275p2

Finally, a brief look at how each package hosting service responded after seeing the test packages:

- * npm - initially deleted a few, stopped after my intentions were clarified
- * PyPI - deleted all packages, made it clear that testing this is not allowed
- * RubyGems - no reaction