Learn more: github.com/davcortez/unpacking-the-threat

Unpacking the Threat: Malicious Packages in Pypi

@davcortez

Whoami

- David Cortez
- Desarrollador Python.
- Entusiasta de la seguridad informática.

@davcortez

Agenda

- Conceptos principales
- PyPi
- Ataques a cadena de suministro (Supply chain attacks)
- Typosquatting
- ¿Cómo podemos protegernos?

@davcortez

Conceptos fundamentales

- Malware
- Arbitrary Code Execution
- Package

PyPI · The Python Package Inde

pypi.org

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

Search projects

Or browse projects

414,482 projects 3,929,715 releases 7,041,049 files 638,062 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.

Trending projects

Trending projects as downloaded by the community

New releases

Hot off the press: the newest project releases

Pypi numbers

380k projects
600k users
server >900 terabytes/day
servers >2 billion requests/day

Fuente: https://di.dev/powering-pypi

PyPI new user and new project registrations temporarily suspended.

Incident Report for Python Infrastructure

Resolved

Suspension has been lifted.

Posted 4 months ago. May 21, 2023 - 21:57 UTC

Identified

New user and new project name registration on PyPI is temporarily suspended. The volume of malicious users and malicious projects being created on the index in the past week has outpaced our ability to respond to it in a timely fashion, especially with multiple PyPI administrators on leave.

While we re-group over the weekend, new user and new project registration is temporarily suspended.

Posted 4 months ago. May 20, 2023 - 16:02 UTC

This incident affected: PyPI (pypi.org - General).

Six Malicious Python Packages in the PyPI Targeting Windows Users

4,982 people reacted | 12 | 11 min. read

Threat actors published more than 451 unique malware-laced Python packages on the official Python Package Index (PyPI) repository.

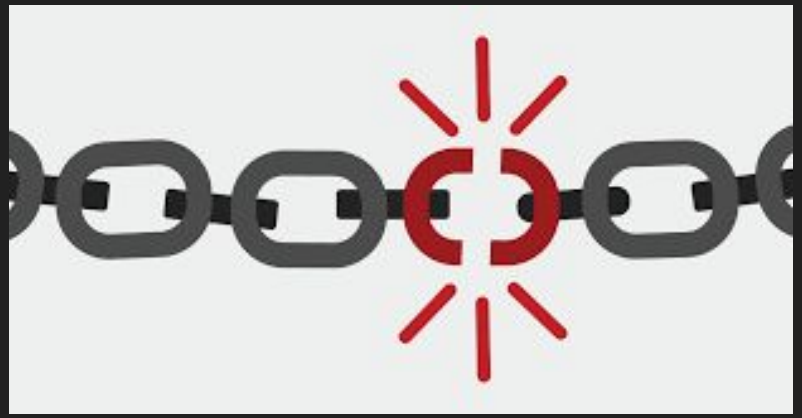
Phylum researchers spotted more than 451 unique Python packages on the official Python Package Index (PyPI) repository in an attempt to deliver clipper malware on the developer systems.

According to the experts, the activity is still ongoing and is part of a **malicious campaign** that they discovered on November 2022.

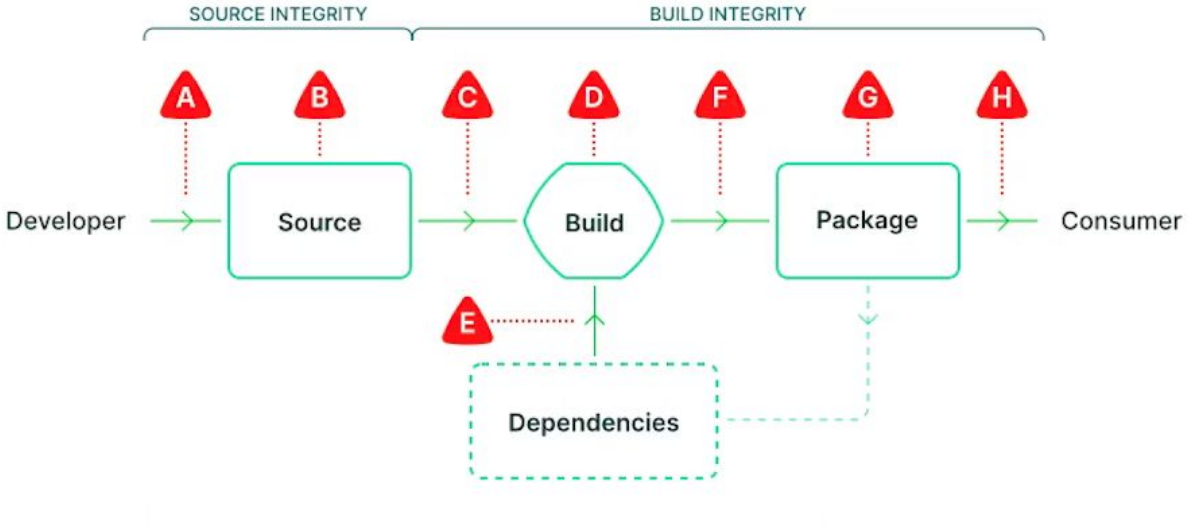
July 21, 2023 • 5 min read

Divide and Hide: How malicious code lived on PyPI for 3 months

The Station 9 research team discovered malicious code that was divided and distributed across different packages, remaining obfuscated for months while getting nearly 2000 downloads.

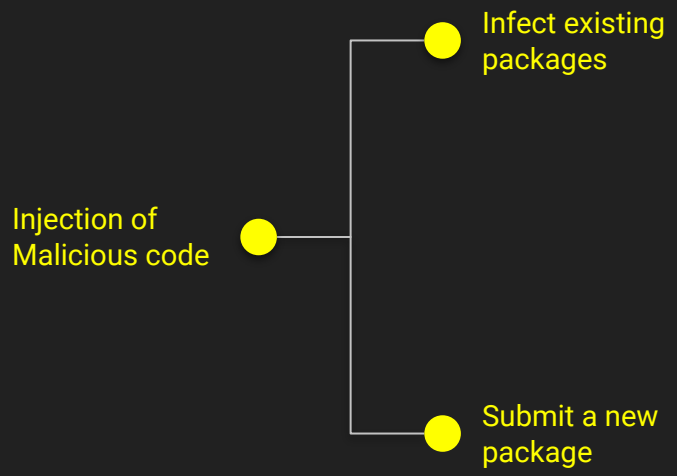


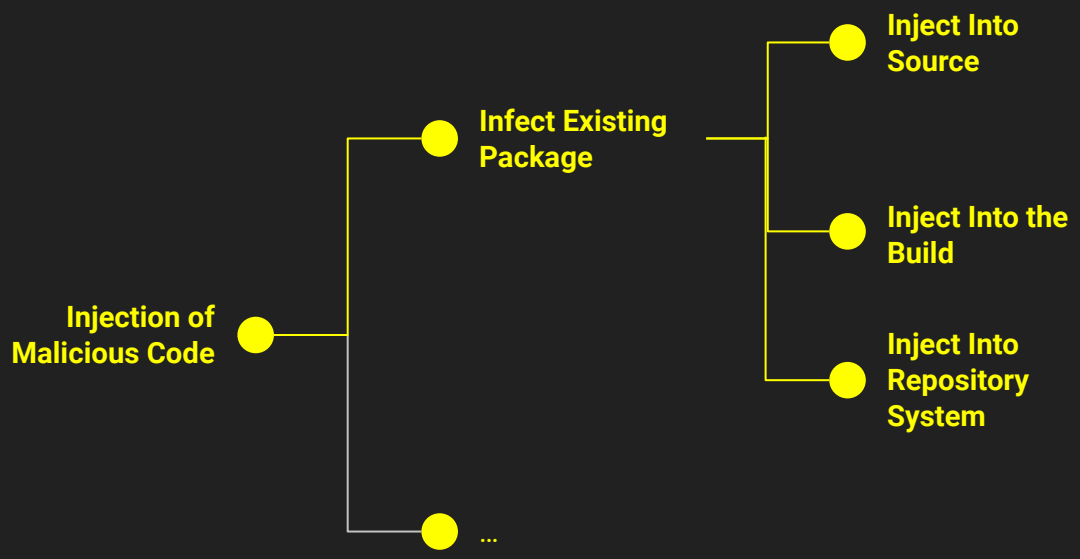
Supply Chain Attacks

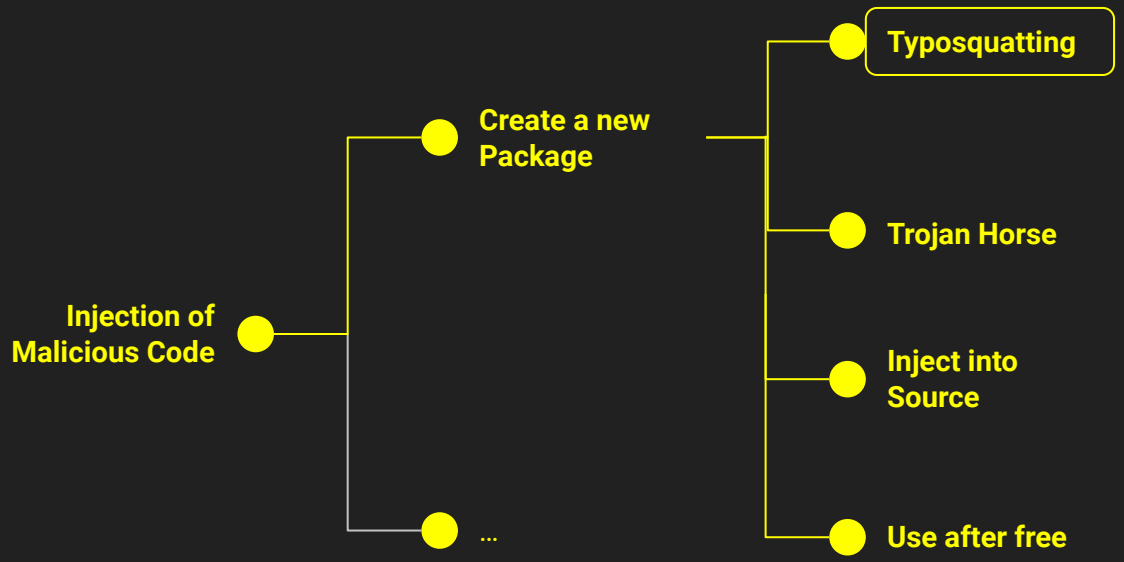


A Submit unauthorized change
B Compromise source repo
C Build from modified source
D Compromise build process
E Use compromised dependency
F Upload modified package
G Compromise package repo
H Use compromised package

Software supply-chain threats (source https://slsa.dev/spec/v0.1/index)


Injection of
Malicious code
Infect existing
packages
Submit a new
package


Injection of Malicious Code
Infect Existing Package
...
Inject Into Source
Inject Into the Build
Inject Into Repository System


Injection of Malicious Code
Create a new Package
...
Typosquatting
Trojan Horse
Inject into Source
Use after free

Typosquatting

Cuál es el porcentaje de paquetes estimados que contengan typosquatting?

De acuerdo a Taylor y otros, en su paper “Defending Against Package Typosquatting”, es el 3%.

requesys

o

requests

python-dotenv
o
dotenv-python


Mon
Tue-Thur
Fri -Sat
Sunday

libcurl

o

pycurl

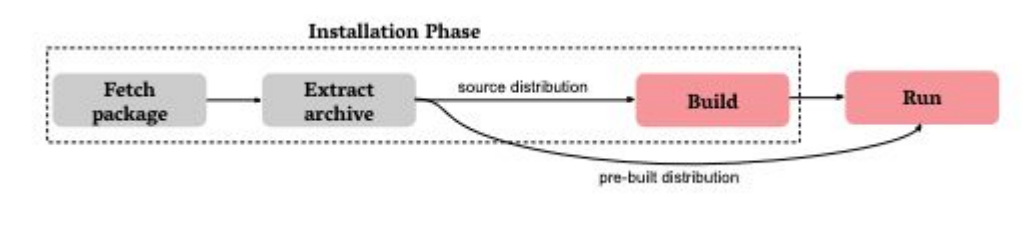
jellyfish

o

jellyfish

Arbitrary Code execution Strategies

- Install-Time Execution
- Runtime Execution



The Hitchhiker's Guide to Malicious Third-Party Dependencies. (2023). Ladisa y otros.

Conditional Execution

- Application State
- Operating System

Evasion techniques

- String obfuscation
- Code obfuscation
- Suppress warnings

String obfuscation

- Encoding (e.g. Base64, hex)
- Binary Arrays
- Compression (e.g. gzip)
- Encryption (e.g. AES)
- String Concatenation

Code obfuscation

- Encoding, Compression and Encryption
- Dead/Useless Code Insertion
- Split Code into Multiple files
- Hide code into dependency tree
- Split code into multiple dependencies
- Steganography

Code obfuscation

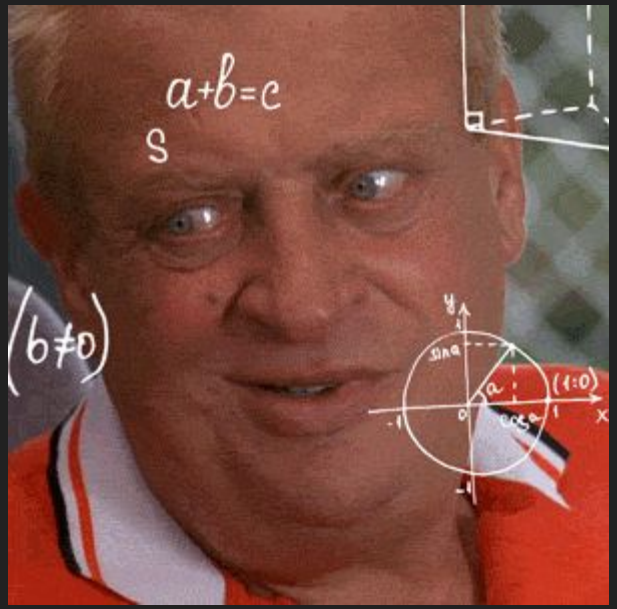
- Visual Deception
- Polyglot Malwares in In-Line Assembly

ALGUNAS IDEAS

¿Cuál es el número de empresas que usan Odoo?

¿Cuántas de ellas usan addons disponibles en Pypi?

¿Cuál sería el impacto si su seguridad es comprometida?



DISCLAIMER!

THIS IS USED FOR EDUCATIONAL PURPOSES ONLY

```
├── .editorconfig
├── .eslintrc.yml
├── .flake8
├── .isort.cfg
├── .pre-commit-config.yaml
├── LICENSE
├── README.md
├── doc
├── l10n_ec_account_edi
├── l10n_ec_base
├── l10n_ec_ote
├── requirements.txt
└── setup.py
```

Crear el paquete malicioso

odoo-addon-l10n-ec-ote

https://github.com/OCA/pos

Statistics

GitHub statistics:

- **Stars:** 225
- **Forks:** 560
- **Open issues:** 19
- **Open PRs:** 63

View statistics for this project via Libraries.io, or by using our public dataset on Google BigQuery

Carga de paquete malicioso a Pypi

```
pip install odoo-addon-l10n-ec-ote
```

Víctima instala el paquete

Exfil Data (Ssh keys, variables de entorno, etc..)


```
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAAABG5vbmUAAAAEbm9uZQAAAAAAAAAABAAABFwAAAAdzc2gtcn
NhAAAAAwEAAQAAAQEAyFIMJHkZiTCaDBkarbU0mcn042TyKZTrFrpMJdZ/XFzpGwHxi5gx
gsd0aFlUhDP74kvTtWKHSSJaAidUV8dhnYTHg0VZo+NJZVtL56dVK7uXbMVbjFnVOxln0j
6+3PHSCDYvRny9dpgG+Y3kLvfQNTE2ZLIZKezMlbylVGOcKsgzKOIPhBEVO8FSGWsbC75A
oh2T2diIw8OI6TyT+bmAmbCFfHHzW3Q2ZOhjyfmVP8jmxilTcsU6O1+fjoRJxzytIWs5Kr
uC1FHiH7qikrnbUtic6oBuyNji4H/Hgw0H2EwEPgC+H5dhJqvCSarnwnaGsyUetJIAWNSI
o3kK8hTTDQAAA9A2T8OrNk/DqwAAAAdzc2gtcnNhAAABAQDIUgwkeRmJMJcMGRqttTSZyf
```


HUH!

python

Sign Up | Log In

What methods should we implement to detect malicious content?

PSF · PyPI Q4 RFI

Sumana Harihareswara sumanah — Sep '19

Let's discuss here: what methods should Warehouse use to detect malicious content?

Some relevant GitHub issues: #5117 and (on typosquatting specifically) #4998 .

1

Improving risks and consequences against typosquatting on pypi

created	last reply	23	13.8k	13	11	16
Sep '19	Sep '19	replies	views	users	likes	links

Frequent Posters

Popular Links

Detect malicious packages, for later removal · Issue #5117 · pypa/warehouse · GitHub github.com

Detect packages being published with typo'ish names · Issue #4998 · pypa/warehouse ·... github.com

GitHub - crev-dev/crev: Scalable, social, Code REView and recommendation system that... github.com

https://forms.gle/redWdNhwMqzRG1jC8

rfi.md · GitHub github.com

Sep 2019

1 / 24

Sep 2019

Sep '19

Labels | Milestones

New issue

Package signing & detection/verification

Past due by over 3 years 86% complete

Security work funded by a gift from Facebook https://pyfound.blogspot.com/2018/12/upcoming-pypi-improvements-for-2019.html ....

(1) Cryptographic signing and verification of artifacts (PEP 458/TUF or similar) (2) Automated detection of malicious uploads (3) Further work on API tokens + multi-factor authentication, should the need arise (4) UI design around new features mentioned above (5) User adoption planning/design (6) Documentation.

PSF plans to do this work in the second half of 2019.

Show less

3 Open 20 Closed

- **Implement soft deletes for projects, releases and files** feature request needs discussion UX/UI — 16
  #4440 opened on Aug 2, 2018 by di
- **Detect packages being published with typo'ish names** feature request — 9
  #4998 opened on Nov 2, 2018 by aaronlelevier
- **[WIP] warehouse: TUF initialization** — 9
  #7488 opened on Mar 3, 2020 by woodruffw • Draft

Announcing the launch of PyPI Malware Reporting and Response project

by: **Shamika Mohanan** · 2023-06-22

`#security`

We are pleased to announce that the PSF has received funding from the Center for Security and Emerging Technology (CSET) to develop and improve the infrastructure for malware reporting and response on PyPI. This project will be executed over the coming year.

Currently, malware reports are submitted to PyPI admins by email before being manually triaged and responded to. There is an opportunity for improvement in streamlining the report submission process and the tools used to triage and respond to them. The current process cannot scale easily or handle duplication of reports. It is not easy to measure time to remediation and is currently impossible to implement automated takedown of threats.

PEP 458
PEP 480

Caso de estudio
fshec2

Name	Size	Packed Size	Modified
full.pyc	5 299	5 632	2023-04-14 17:01
main.py	544	1 024	2023-04-15 12:36
__init__.py	30	512	2023-04-15 12:45

Page not found (404)

Request Method: GET
Request URL: http://13.51.44.246/what_do_we_have_here

Using the URLconf defined in `ec2_django_project.urls`, Django tried these URL patterns, in this order:

```
1. admin/
2.
3. commands/ [name='commands']
4. cron_script/ [name='cron_script']
5. upload/ [name='upload_file']
6. uploaded_files/ [name='uploaded_files']
7. download/<int:file_id>/ [name='download_file']
```

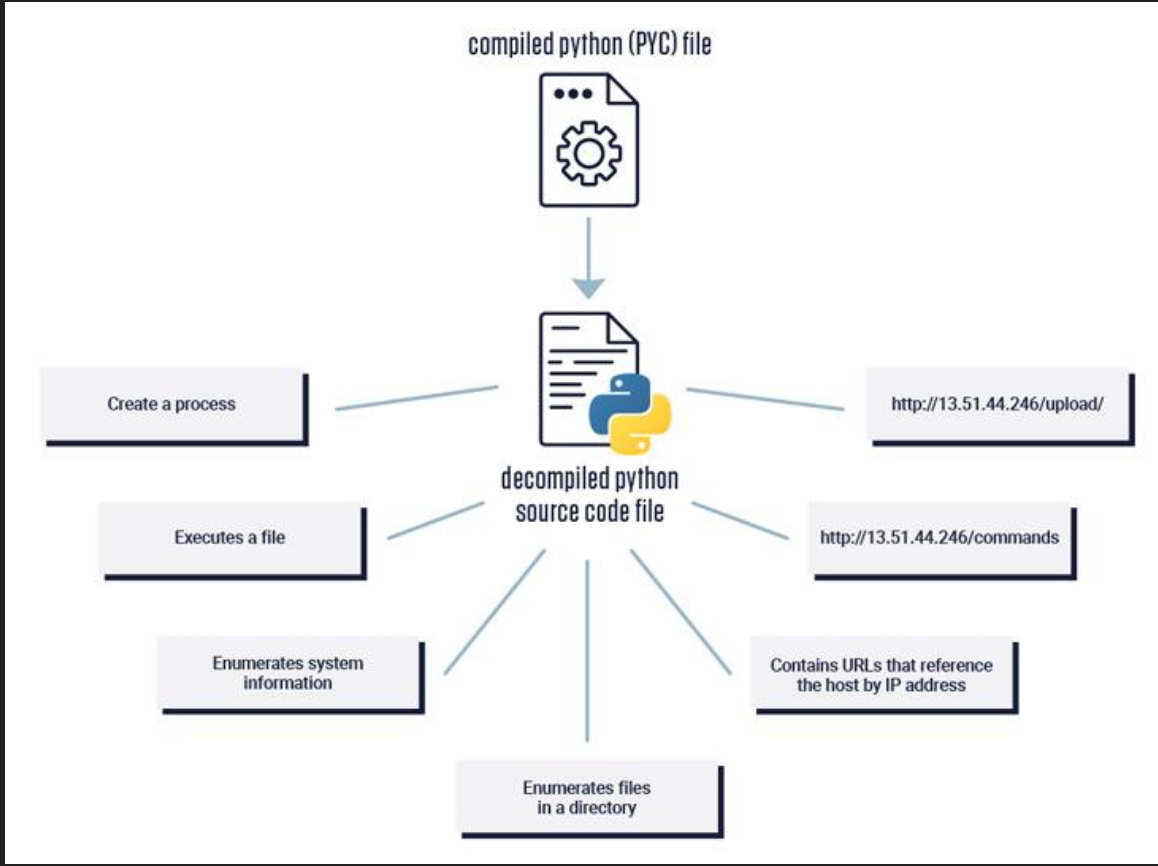
The current path, `what_do_we_have_here`, didn't match any of these.

You're seeing this error because you have `DEBUG = True` in your Django settings file. Change that to `False`, and Django will display a standard 404 page.

When python bytecode bites. (Reversing Labs)

```
uploads/dcde.ods
uploads/dcde_ETovPoN.ods
uploads/dcde_uT8U4b1.ods
uploads/keylogs.txt
uploads/keylogs_LodzfXB.txt
uploads/keylogs_9KTElhg.txt
uploads/keylogs_cE3JlSp.txt
uploads/keylogs_ieH8Fpl.txt
uploads/keylogs_dDAlmIR.txt
uploads/mycron
uploads/keylogs.txt
uploads/desktop__desktop-7G-Series__crontab_default_WH6EqAR.txt
uploads/desktop__desktop-7G-Series__user_HGA6F92.txt
uploads/desktop__desktop-7G-Series__all_tg20BOx.txt
uploads/desktop__desktop-7G-Series__crontab_default_HFEvopm.txt
uploads/desktop__desktop-7G-Series__user_ejG4BT4.txt
uploads/desktop__desktop-7G-Series__all_QJ6khyO.txt
uploads/T[redacted]_A[redacted]__LAPTOP-UH9S5HF2__user.txt
```

When python bytecode bites. (Reversing Labs)



When python bytecode bites. (Reversing Labs)

Entonces, qué podemos hacer?

- Hacer la debida diligencia antes de descargar y ejecutar paquetes de fuentes 3ras).
  - Nombre similar a otros existentes
  - Fecha que fue agregado a Pypi
  - Descripción
  - Autor
  - Enlace a repositorio

Entonces, qué podemos hacer?

- O usar un Artifact registry (jfrog, google)
- No ejecutar pip como usuario root
- Usar pip con el modo hash-check

Entonces, qué podemos hacer?

- Priorizar la seguridad en el proceso de desarrollo. Implementado medidas de seguridad robustas como: code reviews, pruebas automatizadas, entre otras, para identificar y remediar vulnerabilidades antes del desarrollo.
- Estar al día con las actualizaciones y parches de seguridad.
- Educación en temas de seguridad informática.

Algunas ideas para identificar Typosquatting packages.

- Levenshtein distance
- Damerau-Levenshtein distance

Tools

lyvd/**bandit4mal**

A fork of Bandit tool with patterns to identifying malicious python code.

0 Contributors

0 Issues

9 Stars

1 Fork

Tools

“Ensure that you know what software is being used and establish the criticality of each tool.” -by Dustin S. Sachs

@davcortez

Recursos

Blogs/Websites

https://github.com/pypi/warehouse/issues/5117.

Papers

The Hitchhiker’s Guide to Malicious Third-Party Dependencies. (2023). Piergiorgio Ladisa, Merve Sahin, Serena Elisa Ponta, Marco Rosa, Matias Martinez, Olivier Barais.

Backstabber's Knife Collection: A Review of Open Source Software Supply Chain Attacks. (2020). Marc Ohm, Henrik Plate, Arnold Sykosch, Michael Meier.

Malware packages

https://github.com/rsc-dev/pypi_malware.

@davcortez

Recursos

Blogs/Websites

https://www.reversinglabs.com/blog/when-python-bytecode-bites-back-who-checks-the-contents-of-compiled-python-files.

https://peps.python.org/pep-0458/.

https://peps.python.org/pep-0480/.

Papers

Defending Against Package Typosquatting. (2022). Duc-Ly Vu, Zhary Newman, John Speed Meyers.

A Benchmark Comparison of Python Malware Detection Approaches. (2022). Duc-Ly Vu, Zachary Newman, John Speed Meyers.

@davcortez