PyPI Supply Chain Attack Uncovered: Colorama and Colorizr Name Confusion - Checkmarx

Archived: 2026-04-05 17:30:57 UTC

Checkmarx Zero researcher Ariel Harush has discovered evidence of a malicious package campaign that is consistent with live adversarial activity and adversarial research and testing. This campaign targets Python and NPM users on Windows and Linux via typo-squatting and name-confusion attacks against `colorama` (a widely-used Python package for colorizing terminal output) on PyPI and the similar `colorizr` JavaScript package on NPM. These malicious packages were uploaded to PyPI.

- Multiple packages uploaded to PyPI with significantly risky payloads were uploaded with names similar to legitimate packages in both PyPI and NPM.
- The tactic of using the name from one ecosystem (NPM) to attack users of a different ecosystem (PyPI) is unusual.
- Payloads allow persistent remote access to and remote control of desktops and servers, as well as harvesting and exfiltrating sensitive data.
- Windows payloads attempt to bypass antivirus/endpoint protection controls to avoid detection.
- Packages have been removed from public repositories, limiting immediate potential for damage.

These behaviors are consistent with targeted adversarial activity and coordinated campaigns. It is likely, based on this pattern, that these were created either to attack a particular target or set of targets. No clear attribution data is currently available, so we do not know whether this campaign is connected to a well-known adversary.

Cross-Platform Supply Chain Attacks Targeting Users of Colorama and Colorizr

In the ever-escalating game of cat and mouse in open-source security, threat actors continue to evolve. This is expected. But this supply chain attack campaign targeting Colorama users stood out not just for its creativity, but for its scope. By combining typo-squatting and related name confusion attacks, cross-ecosystem baiting, and multi-platform payloads, this attack serves as a reminder of how opportunistic and sophisticated open-source supply chain threats have become.

Typos That Hurt: Colorama Copycats

When we uncovered a wave of malicious packages uploaded to PyPI, seemingly designed to trick developers into installing them by mistake, we were immediately concerned. These packages closely mimic the names of two popular libraries:

- `colorama` (a widely used Python package for terminal color control)
- `colorizr` (an NPM package used for similar functionality in JavaScript)

One especially unusual facet of this campaign is the cross-ecosystem name confusion tactic. Several of the fake PyPI packages mimic the naming conventions of the NPM package `colorizr`. This suggests either a deliberate effort to sow confusion, or the possibility of future attacks branching into the NPM ecosystem.

The payloads have Windows and Linux variants, with common features including:

- Accessing and exfiltrating sensitive configuration information
- Establishing remote control / remote access for the attacker
- Establishing persistence and “command and control” (C2) mechanisms consistent with expectation of establishing a long-term foothold
- Attempts to hide from detection and evade endpoint security controls

Never Miss Our Checkmarx Research Updates!

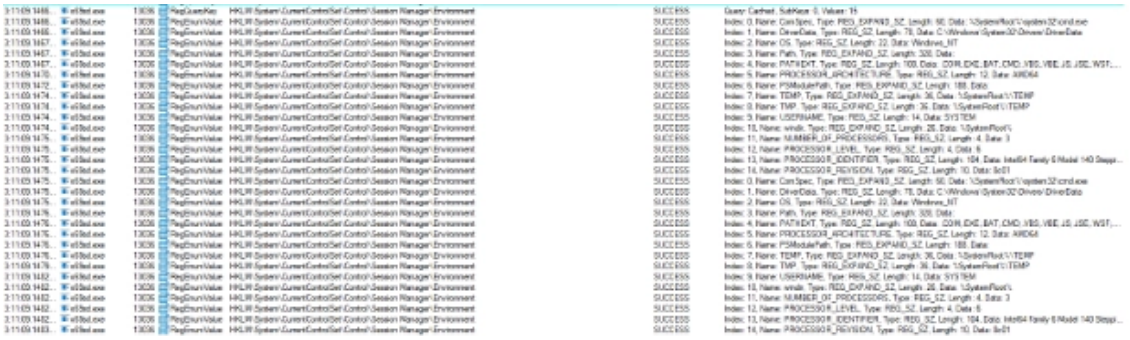
Windows Payloads: Malware With Persistence and Evasion

Closely examining the package variants that target Windows, we were able to link some payloads back to a GitHub account: `github[.]com/s7bhme`. This repo hosted various suspicious executables, and a Python project template named `branchtemplaterepo`.

Key Behaviors Identified:

Environment Variable Harvesting

The malware grabs environment variables from the Windows registry — a tactic that may expose sensitive information such as credentials or configuration secrets.



Indications of Environment Variable access; sample screenshot

Persistence via Task Scheduler

The payload delivery process creates scheduled tasks pointing to different file paths, each running a separate payload. This suggests a modular setup where multiple components are deployed together.

Example screenshot of Task Scheduler showing multiple payload configurations

Antivirus Awareness

The malware checks for installed security software and alters its behavior accordingly to avoid detection.

Indication of process list access checking for common anti-malware programs

We also observed a payload installing and running checks for anti-malware tools on the infected host. We observed several specific anti-detection behaviors, including running the following commands:

```
"C:\Program Files\Windows Defender\MpCmdRun.exe" -RemoveDefinitions -All
```

This command aims to remove all **malware definitions** from windows defender

```
Set-MpPreference -DisableIOAVProtection $true
```

Powershell snippet which Disables **IOAV (Input/Output Antivirus)** scanning, preventing checking files downloaded from the internet for safety

These behaviors show clear adversarial intent: establish a foothold, stay hidden, and collect sensitive data.

Linux Payloads: Advanced Backdoors with Remote Control

On the Linux side, we analyzed two packages: Colorizator and coloraiz. These contain base64-encoded payloads buried in `src/colorizator/__init__.py`. Once decoded and executed, the payloads initiate a sophisticated infection chain:

The Attack Path:

1. **RSA Key Drop**: A public key is written to `/tmp/pub.pem`. This key is later used to encrypt the output of a later `gs-netcat` command before that output is exfiltrated.
2. **Remote Bash Download**: A script is fetched from `[https:]//gsocket[.]io/y`, likely used to install gs-netcat, a tool for establishing encrypted reverse shells.
3. **Encrypted Output Exfiltration**: The gs-netcat output is encrypted using the RSA key, base64-encoded, and silently uploaded to Pastebin via its API using valid developer and user keys.
4. **Cleanup**: Temporary files are deleted to remove traces of the activity.

The Remote Access Script — A Full-Fledged Swiss Army Knife

The downloaded bash script is portable, stealthy, and feature-rich:

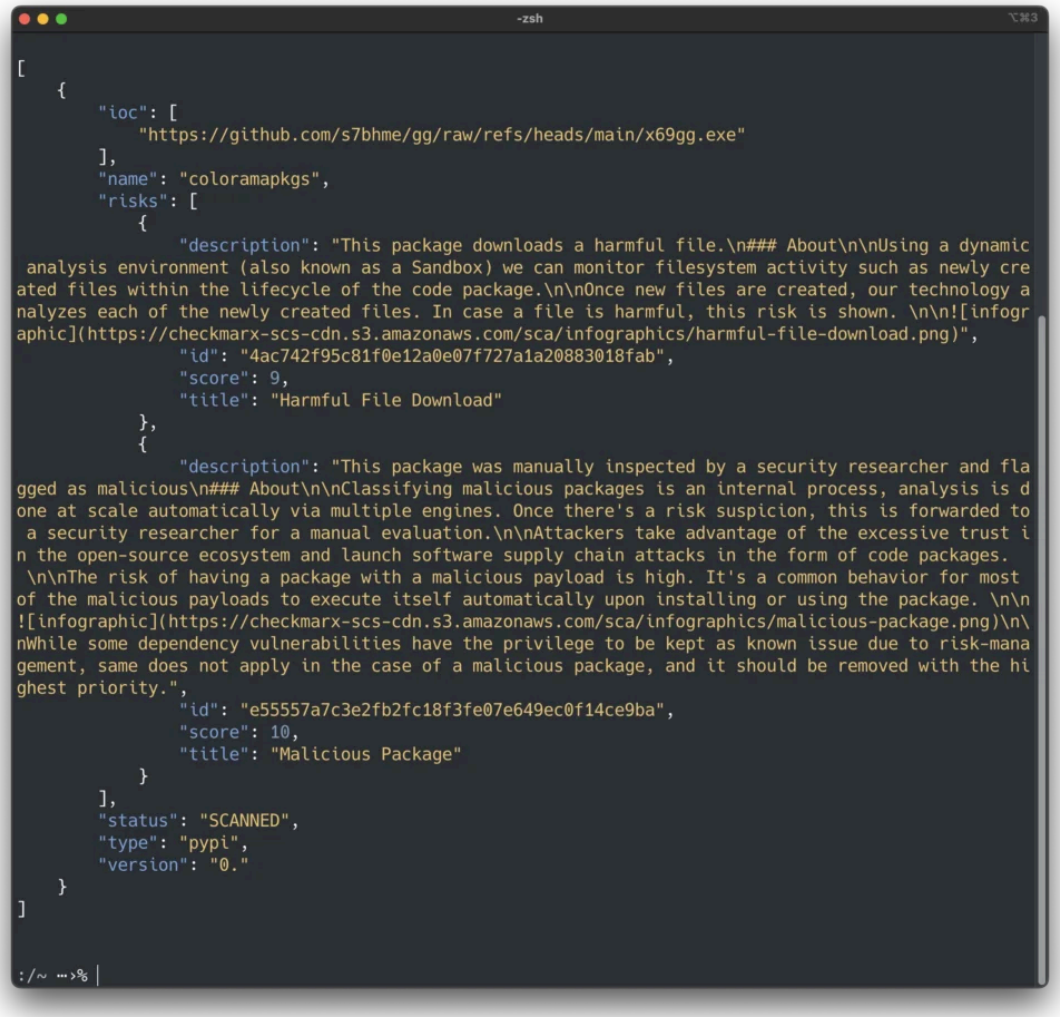
- **Persistence** through systemd, shell profile injection, crontabs (scheduled task configuration files), and `rc.local` (startup script) edits.
- **Stealth** by masquerading as kernel processes and preserving timestamps.
- **Remote Control** via environment-based configuration.
- **Exfiltration and C2 (Command and Control)** using gs-netcat and encrypted communication.
- **Webhook notifications** to platforms like Discord, Telegram, and custom URLs.

This isn't your average script kiddie toolkit — it's a highly capable backdoor designed to remain hidden and maintain long-term access.

Key Indicators of Compromise (IoC)

Type	Value	Description
GitHub Repo	`[https]://github[.]com/s7bhme`	Repository hosting malicious payloads and templates
Webhook URL	`[https]://webhook[.]site/dc3c1af9-ea3d-4401-9158-eb6dda735276`	Endpoint used by malware to exfiltrate data or notify
Package Owner	rick_grimes	Uploaded `Colorizator`(1.2.3, 2.1.2) (Linux)
Package Owner	morty_smith	Uploaded `coloraiz`(1.0.1, 1.0.2, 1.0.3) (Linux)
Package Owner	reven	Uploaded `coloramapkgsw` (0.1.0), `coloramapkgsdow` (0.1.0) (Windows)
Package Owner	m5tl	Uploaded `coloramashowtemp` (0.1.0) (Windows)
Package Owner	dsss	Uploaded `coloramapkgs`(0.1.0), `readmecolorama` (0.1.0) (Windows)
File Hash (SHA256)	`d30c78c64985a42c34ef142fd8754a776c8db81228bafc385c5bd429252e4612`	Malicious Linux bash script (downloaded by payload)
File Hash (SHA256)	`daef5255eac4a4d16940e424c97492c6bad8fdafd2420632c371b9d18df3b47f`	Windows payload (`x69gg.exe`) executed by Python script

These IOCs are represented in the Checkmarx Malicious Package Protection component, including the Threat Intelligence API, for inclusion into customer programs.



Output for `coloramapkgs` query with the Checkmarx Threat Intel API

Attribution Is a Challenge

Initially, the similarities in naming and upload timing led us to believe that both the Linux and Windows payloads were deployed by the same actor. But as our investigation progressed, differences in tooling, tactics, and infrastructure suggest otherwise.

At this time, we can't definitively attribute both payload sets to a single source. They **may be separate campaigns** exploiting a similar typo-squatting tactic — a reminder of how quickly malicious techniques spread in cybercrime ecosystems.

Recommended Response

While this particular set of packages is no longer available from public sources, defenders should be prepared to detect and respond rapidly to this and similar attack patterns. While individual organization threat models may indicate additional controls and behaviors, we recommend, at minimum:

- Examine deployed and deployable application code for malicious package names and indicators of compromise

- Examine private package repositories and proxies (such as Artifactory); remove any instances of malicious packages and add them to a block list
- Ensure installation of these packages is blocked on developer desktops, test environments, etc.

Checkmarx customers can use Malicious Package Protection features, including our Threat Intel API, to automate many aspects of these tasks

Tags:

AppSec

Checkmarx Security Research Team

Malicious Packages

PyPi

Python

Supply Chain Security
