Malicious VS Code Extension Impersonating “Material Icon Theme” Found in Marketplace

nextron-systems.com/2025/11/28/malicious-vs-code-extension-impersonating-material-icon-theme-found-in-marketplace

Marius Benthin November 28, 2025

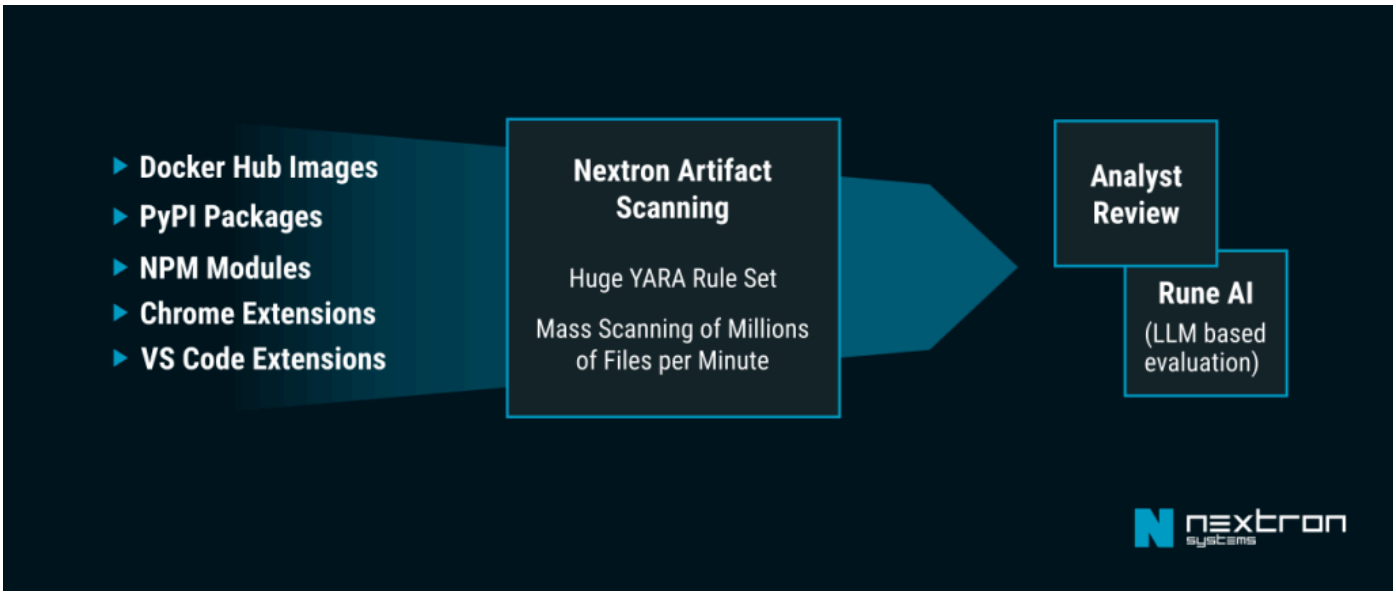
Over the last weeks we’ve been running a new internal artifact-scanning service across several large ecosystems. It’s still growing feature-wise, LLM scoring and a few other bits are being added, but the core pipeline is already pulling huge amounts of stuff every week – Docker Hub images, PyPI packages, NPM modules, Chrome extensions, VS Code extensions. Everything gets thrown through our signature set that’s built to flag obfuscated JavaScript, encoded payloads, suspicious command stubs, reverse shells, and the usual “why is this here” indicators.

The only reason this works at the scale we need is THOR Thunderstorm running in Docker. That backend handles the heavy lifting for millions of files, so the pipeline just feeds artifacts into it at a steady rate. Same component is available to customers; if someone wants to plug this kind of scanning into their own CI or ingestion workflow, Thunderstorm can be used exactly the way we use it internally.

We review millions of files; most of the noise is the classic JS-obfuscation stuff that maintainers use to “protect” code; ok… but buried in the noise you find the things that shouldn’t be there at all. And one of those popped up this week.

Our artifact scanning approach

We published an article this year about blind spots in security tooling and why malicious artifacts keep slipping through the standard AV checks. That's the gap this whole setup is meant to cover. AV engines choke on obfuscated scripts, and LLMs fall over as soon as you throw them industrial-scale volume. Thunderstorm sits in the middle – signature coverage that hits encoded payloads, weird script constructs, stagers, reverse shells, etc., plus the ability to scale horizontally in containers.



The workflow is simple:

- pull artifacts from Docker Hub, PyPI, NPM, the VS Code Marketplace, Chrome Web Store;
- unpack them into individual files;
- feed them into Thunderstorm;
- store all hits;
- manually review anything above a certain score.

We run these scans continuously. The goal is to surface the obviously malicious uploads quickly and not get buried in the endless "maybe suspicious" noise.

The finding: malicious VS Code extension with Rust implants

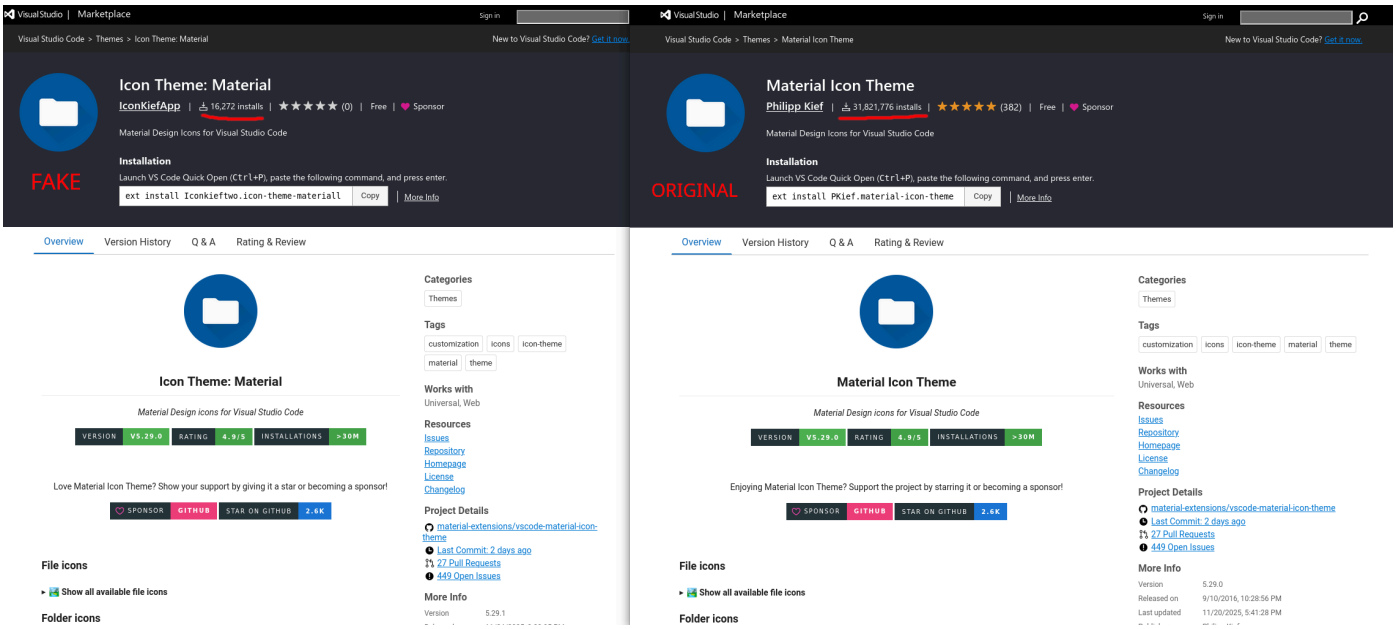
While reviewing flagged VS Code extensions, Marius stumbled over an extension named "Icon Theme: Material", published under the account "IconKiefApp". It mimics the legitimate and extremely popular Material Icon Theme extension by Philipp Kief. Same name pattern, same visuals, but not the same author.

The fake extension had more than 16,000 installs already.

Inside the package we found two Rust implants: one Mach-O, one Windows PE. The paths looked like this:

icon-theme-materiall.5.29.1/extension/dist/extension/desktop/

The Mach-O binary contains a user-path string identical in style to the GlassWorm samples reported recently by Koi (VT sample link below). The PE implant shows the same structure. Both binaries are definitely not part of any real icon-theme extension.



The malicious and the legitimate extension

```
.rdata:00000001800F0A0B                 db    0
.rdata:00000001800F0A0C                 db  25h ; %
.rdata:00000001800F0A0D                 db    0
.rdata:00000001800F0A0E                 db    0
.rdata:00000001800F0A0F                 db    0
.rdata:00000001800F0A10 aIndexJs        db 'index.js'           ; DATA XREF: tr_unknown__sub_1800026DF+5D0↑o
.rdata:00000001800F0A18 aVarHttpsRequir db 'var https = require("https");',0Ah
.rdata:00000001800F0A18                                         ; DATA XREF: .rdata:off_1800F0AE0↓o
.rdata:00000001800F0A36                 db 'const secretKey = ',27h
.rdata:00000001800F0A49 aConstIvBufferF_0 db 27h,';',0Ah        ; DATA XREF: .rdata:00000001800F0AF0↓o
.rdata:00000001800F0A4C                 db 'const _iv = Buffer.from(',27h
.rdata:00000001800F0A65 aBase64GlobalAt db 27h,', "base64");',0Ah
.rdata:00000001800F0A65                                         ; DATA XREF: .rdata:00000001800F0B00↓o
.rdata:00000001800F0A73                 db 'global.atob = function(str) { return Buffer.from(str, ',27h,'base'
.rdata:00000001800F0AAE                 db '64',27h,').toString(',27h,'binary',27h,'); };',0Ah
.rdata:00000001800F0ACA                 db 'eval(atob(',27h
.rdata:00000001800F0AD5 asc_1800F0AD5   db 27h,'));'            ; DATA XREF: .rdata:00000001800F0B10↓o
.rdata:00000001800F0AD9                 db    0
.rdata:00000001800F0ADA                 align 20h
.rdata:00000001800F0AE0 off_1800F0AE0   dq offset aVarHttpsRequir
.rdata:00000001800F0AE0                                         ; DATA XREF: tr_unknown__sub_1800026DF+640↑o
.rdata:00000001800F0AE0                                         ; "var https = require(\"https\");\nconst "...
.rdata:00000001800F0AE8                 db  31h ; 1
```

PE Strings

```
HEADER:0000000000000780 ; LC_ID_DYLIB - dynamically linked shared lib ident
HEADER:0000000000000780                 dylib_command <0xD, 0x70, <<0x18>, 1, 0, 0>>
HEADER:0000000000000798 aUsersDavidioas DCB "/Users/davidioasd/Downloads/rust_implant/target/release/deps/lib" ; library's path name
HEADER:00000000000007D8                 DCB "rust_implant.dylib",0
HEADER:00000000000007EB                 ALIGN 0x10
HEADER:00000000000007F0 ; LC_DYLD_INFO - compressed dyld information
HEADER:00000000000007F0                 dyld_info_command <0x80000022, 0x30, 0x180000, 0x780, 0x180780, 0x1B8,\
```

Mach-O path: /Users/davidioasd/Downloads/rust_implant/…

The malicious extension:

https://marketplace.visualstudio.com/items?itemName=Iconkieftwo.icon-theme-materiall

The legitimate one:

https://marketplace.visualstudio.com/items?itemName=PKief.material-icon-theme

Related GlassWorm sample:

https://www.virustotal.com/gui/file/eafeccc6925130db1ebc5150b8922bf3371ab94dbbc2d600d9cf7cd6849b056e

IOCs

VS Code Extension

0878f3c59755ffaf0b639c1b2f6e8fed552724a50eb2878c3ba21cf8eb4e2ab6
icon-theme-materiall.5.29.1.zip

Rust Implants

6ebeb188f3cc3b647c4460c0b8e41b75d057747c662f4cd7912d77deaccfd2f2
(os.node) PE
fb07743d139f72fca4616b01308f1f705f02fda72988027bc68e9316655eadda
(darwin.node) MACHO

Signatures

YARA rules that triggered on the samples:
SUSP_Implant_Indicators_Jul24_1
SUSP_HKTL_Gen_Pattern_Feb25_2

Status

We already reported the malicious extension to Microsoft. The previous version, 5.29.0, didn't contain any implants. The publisher then pushed a new update, version 5.29.1, on 28 November 2025 at 11:34, and that one *does* include the two Rust implants.

As of now (28 November, 14:00 CET), the malicious 5.29.1 release is still online. We expect Microsoft to remove the extension from the Marketplace. We'll share more details once we've fully unpacked both binaries and mapped the overlaps with the GlassWorm activity.

Closing

This is exactly the kind of thing the artifact-scanner was built for. Package ecosystems attract opportunistic uploads; VS Code extensions are no different. We'll keep scanning the big ecosystems and publish findings when they're clearly malicious. If you maintain an extension or a package registry and want to compare detections with us, feel free to reach out; we're adding more sources week by week.

Update 29.11.2025

Since we published the initial post, a full technical analysis of the Rust implants contained in the malicious extension has been completed. The detailed breakdown is now available in our follow-up article: "Analysis of the Rust implants found in the malicious VS Code extension".

That post describes how the implants operate on Windows and macOS, their command-and-control mechanism via a Solana-based wallet, the encrypted-payload delivery, and fallback techniques including a hidden Google Calendar-based channel.

Readers who want full technical context, IOCs and deeper insight are encouraged to review the new analysis.