# Fast, Broad, and Elusive: How Vidar Stealer 2.0 Upgrades Infostealer Capabilities

By By: Junestherry Dela Cruz Oct 21, 2025 Read time: 7 min (1987 words)

Published: 2025-10-21 · Archived: 2026-04-05 13:39:25 UTC

Malware

Trend™ Research examines the latest version of the Vidar stealer, which features a full rewrite in C, a multithreaded architecture, and several enhancements that warrant attention. Its timely evolution suggests that Vidar is positioning itself to occupy the space left after Lumma Stealer's decline.

---

Key Takeaways:

- Vidar 2.0's release coincides with a decline in Lumma Stealer activity, resulting in a spike in threat actor adoption and heightened campaign activity.
- The new version is completely rewritten in C, introducing multithreaded architecture for faster, more efficient data exfiltration and improved evasion capabilities.
- Enhanced credential extraction methods allowed Vidar 2.0 to bypass advanced browser security features, such as Chrome's AppBound encryption, through direct memory injection.
- Vidar 2.0 systematically targets a broad scope of data, including credentials from browsers, cloud services, cryptocurrency wallets, gaming platforms, and various communication apps such as Discord and Telegram.
- Trend Vision One™ detects and blocks the specific IoCs referenced in this article, while providing customers with access to hunting queries, actionable threat insights, and intelligence reports related to Vidar Stealer.

On October 6, 2025, the developer known as "Loadbaks" announced the release of Vidar Stealer v2.0 on underground forums. This new version features a complete transition from C++ to a pure C implementation, allegedly enhancing performance and efficiency. Its release coincides with a decline in activity surrounding the Lumma Stealer, suggesting cybercriminals under its operation are exploring alternatives like Vidar and StealC.

Vidar 2.0 is said to introduce a range of concerning features, including advanced anti-analysis measures, multithreaded data theft capabilities, and sophisticated methods for extracting browser credentials. With a consistent price point of US$300, it offers attackers powerful tools that are both cost-effective and efficient.

Overview of Vidar

Vidar originated in 2018 as an information stealer on Russian-language underground forums, initially leveraging the Arkei stealer source code. It quickly gained traction due to its comprehensive ability to steal browser credentials and cryptocurrency wallets, coupled with a stable, well-supported operation, and a competitive US$300 lifetime price. Over the years, Vidar set itself apart from competitors like Raccoon and RedLine by

consistently adding support for new browsers, wallets, and two-factor authentication applications, maintaining a loyal user base through ongoing updates and reliable developer support.

According to the October 2025 announcement, Vidar 2.0 features a complete architectural rework, with its developers emphasizing improvements in performance, evasion techniques, and overall capabilities. The update is described as a significant technical evolution, aiming to address previous limitations and maintaining its effectivity in a shifting threat landscape.

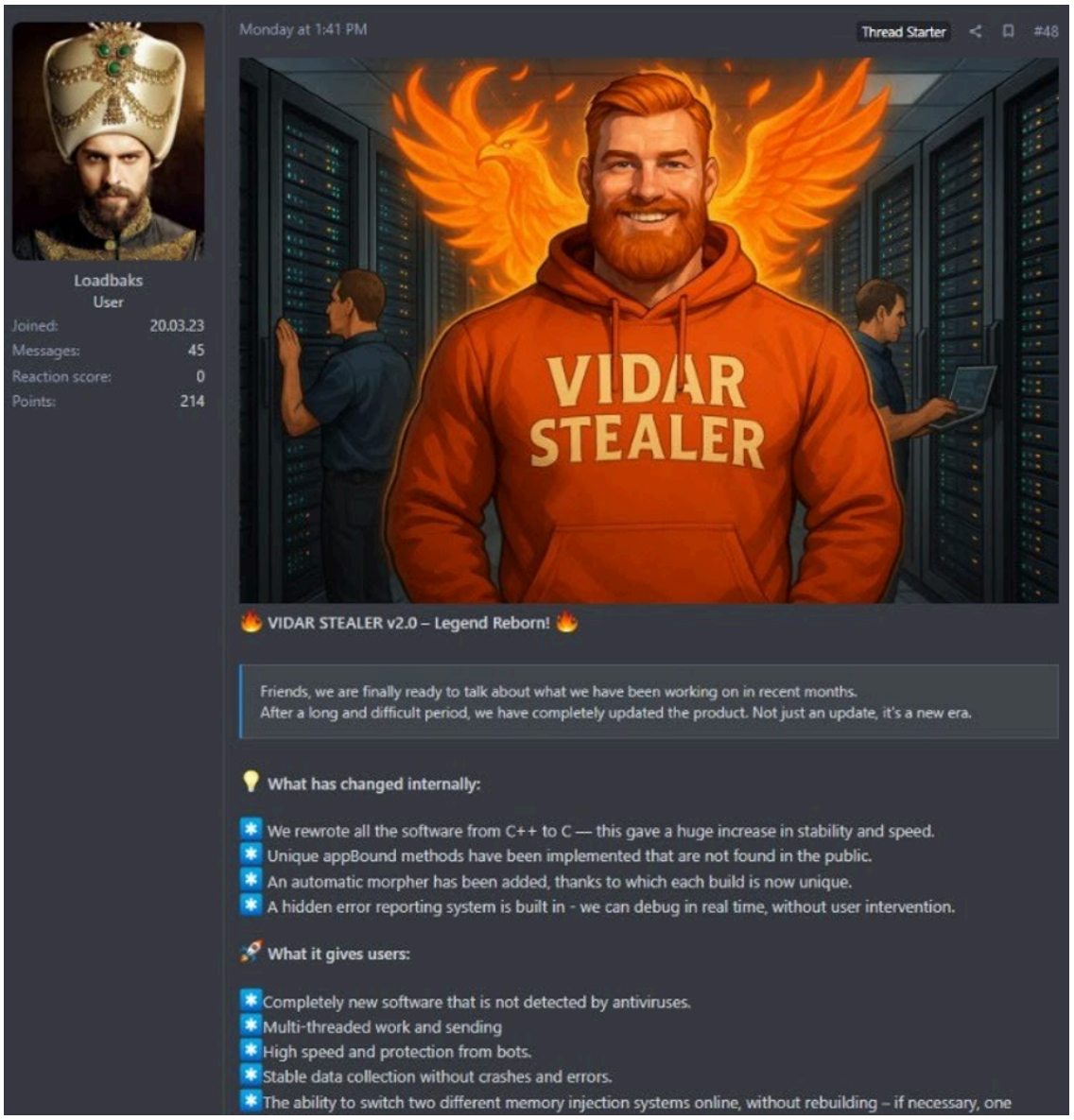


Figure 1. Vidar developer announcing the release of version 2.0

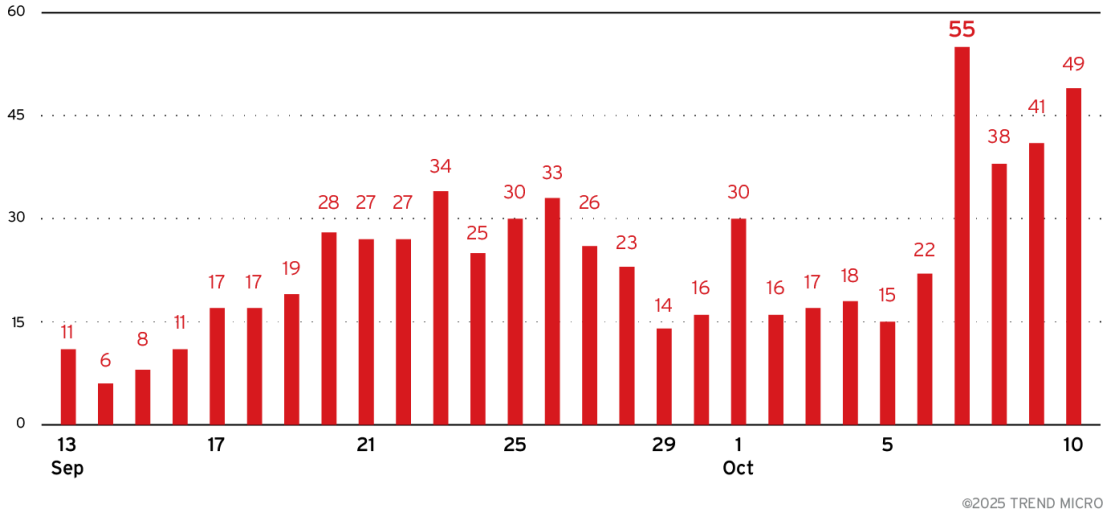


Figure 2. A major spike in Vidar activity after the release of version 2 monitored from Sept. to Oct. 10, 2025

What’s new in Vidar 2.0

Four significant changes have been introduced in this new iteration of the Vidar stealer, chief among them being several core architectural and functional changes. In this section, we examine each one to better understand what has changed and the implications of these changes.

**Complete C language rewrite**

According to the Vidar author "Loadbaks," the development team "rewrote the entire software from C++ to C — this gave a huge increase in stability and speed." This fundamental architectural change represents a complete departure from the previous codebase, with the developers claiming significant performance improvements and enhanced stability through the elimination of C++ dependencies and runtime overhead.

**Multithreaded architecture**

The Vidar author claims that "the unique multithreading system allows extremely efficient use of multi-core processors. It performs data-collection tasks in parallel threads, greatly speeding up the process." This represents a significant enhancement to the malware's operational efficiency, promising faster data collection and exfiltration through parallel processing capabilities that can leverage modern multi-core processor architectures.

Based on our analysis, the malware uses an advanced multi-threading system that automatically adjusts its performance based on the victim's computer specifications. It scales its operations by creating more worker threads on powerful systems and fewer threads on weaker machines, ensuring optimal performance without overwhelming the target system. This approach allows the malware to steal data from multiple sources simultaneously - such as browsers, cryptocurrency wallets, and files - rather than processing them one at a time. The parallel processing significantly reduces the time the malware needs to remain active on the system, making it harder for security software to detect and stop the theft operation.

```
if ( v8 != 6177397 )
{
  GetSystemInfo(&SystemInfo);
  dwNumberOfProcessors = SystemInfo.dwNumberOfProcessors;
  Buffer.dwLength = 64;
  GlobalMemoryStatusEx(&Buffer);
  v13 = 2117496258;
  if ( Buffer.ullAvailPhys >> 20 >= 0x801 )
    v13 = -1678067288;
  v14 = -1069532366;
  if ( Buffer.ullAvailPhys < 0x20000000 )
    v14 = -1484655675;
  v15 = 69325989;
```

Figure 3. Thread count is dynamically calculated based on CPU Core count and available physical memory

Browser credential extraction and AppBound bypass techniques

Vidar 2.0 has "implemented unique appBound methods that aren't found in the public domain" according to its developer. This capability specifically targets Chrome's enhanced security measures introduced in recent versions, claiming to bypass application-bound encryption that was designed to prevent unauthorized credential extraction by binding encryption keys to specific applications.

Binary analysis reveals that Vidar 2.0 implements comprehensive browser credential extraction capabilities targeting both traditional browser storage methods and Chrome's latest security protections across multiple browser platforms including Chrome, Firefox, Edge, and other Chromium-based browsers. Among its traditional credential extraction techniques, the malware employs a tiered approach that includes systematic enumeration of browser profiles and attempting to extract encryption keys from Local State files using standard DPAPI decryption.


Figure 4. Vidar initially attempts traditional credential access methods such as extracting and decryption of keys from Browser Local State files

The malware also employs an advanced technique that launches browsers with debugging enabled and injects malicious code directly into running browser processes using either shellcode or reflective DLL injection. The injected payload extracts encryption keys directly from browser memory, then communicates the stolen keys back

to the main malware process via named pipes to avoid disk artifacts. This approach can bypass Chrome's AppBound encryption protections by stealing keys from active memory rather than attempting to decrypt them from storage.


Figure 5. Encryption keys stolen from browser memory are sent back to malware process via named pipes

Automatic polymorphic builder

Lastly, Vidar's author also boasts an "added an automatic morpher, so every build is now unique." This feature is designed to generate samples with distinct binary signatures, making static detection methods more difficult.

Binary analysis reveals that the new version of Vidar employs heavy use of control flow flattening, implementing complex switch-case structures with numeric state machines that can make reverse engineering more difficult. This obfuscation method transforms the natural program flow into a series of state transitions controlled by switch statements, effectively obscuring the original program logic. This same control flow flattening technique has been observed in Lummastealer samples, suggesting the adoption of similar obfuscation frameworks within the information stealer ecosystem.

```
__int64 __fastcall mal_stealcloudcreds(__int64 a1)
{
  unsigned int v1; // edi
  int v3; // eax
  unsigned int v4; // ebx
  int v5; // ebp
  int v6; // r14d
  int v7; // ebp
  int v8; // r14d
  int v9; // eax
  bool v11; // [rsp+2Fh] [rbp-49h]

  v3 = -964403949;
  v4 = 0;                                       // Control Flow Flattening
  while ( 1 )
  {
    while ( 1 )
    {
      while ( 1 )
      {
        while ( v3 > 894956600 )
        {
          if ( v3 <= 1097256462 )
          {
            if ( v3 == 894956601 )
            {
              v7 = v4 + mal_findazure(a1);
              v8 = v7 + mal_findaws(a1);
              v9 = mal_msalidentity(a1);
              v4 = v8 + v9;
              v11 = !((v8 + v9 < 0) ^ __OFADD__(v8, v9) | (v8 + v9 == 0));
              v3 = 2112397915;
              if ( (((_BYTE)dword_14005C034 * ((_BYTE)dword_14005C034 + 1)) & 1) == 0 )
                v3 = -2139591569;
              if ( dword_14005C038 < 10 )
                v3 = -2139591569;
            }
            else
```

Figure 6. Control flow flattening obfuscation in Vidar 2.0

Vidar 2.0 technical analysis and execution flow

Vidar 2.0's execution flow reveals a carefully orchestrated sequence of operations designed to maximize data collection while evading detection through advanced anti-analysis techniques, multithreaded processing, and adaptive evasion mechanisms.

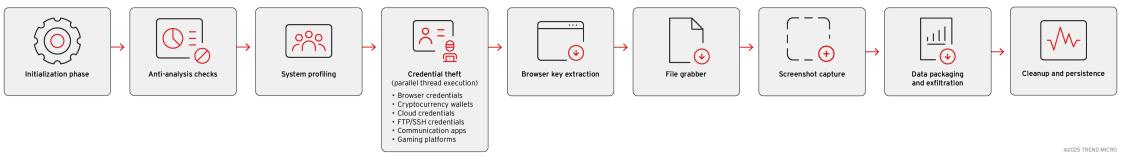


Figure 7. Vidar 2.0’s execution flow

**Initialization and Evasion (Phases 1-2):** Vidar 2.0 begins execution with a comprehensive initialization phase that establishes its multithreaded architecture and implements control flow obfuscation through complex state machines. The malware then performs extensive anti-analysis checks including debugger detection, timing verification, system uptime validation, and hardware profiling to ensure execution only occurs on genuine victim systems rather than analysis environments. These checks must all pass for execution to continue, with any failure resulting in immediate termination to evade sandbox detection.

**Intelligence Gathering and Data Theft (Phases 3-6):** Following successful evasion, the malware conducts thorough system profiling to collect victim information before launching parallel credential theft operations across multiple categories. The sophisticated browser credential extraction employs both standard DPAPI decryption and advanced memory injection techniques to bypass Chrome's v20 AppBound encryption, while simultaneously targeting cryptocurrency wallets, cloud credentials, communication applications, and gaming platforms. The file grabber component systematically searches for valuable files across user directories and removable drives, focusing on cryptocurrency keys and potential credential files.

Data theft capabilities

| | |
|---|---|
| Browser credentials | • Login Data - Chrome/Edge passwords<br>• Web Data - Chrome/Edge autofill/credit cards<br>• logins.json - Firefox passwords<br>• formhistory.sqlite - Firefox form history<br>• cookies.sqlite - Firefox cookies<br>• places.sqlite - Firefox browsing history<br>• key4.db - Firefox master encryption key<br>• passwords.txt - Exported passwords<br>• \Network\Cookies - Network cookie files |
| Cryptocurrency wallets | • Local Extension Settings - Browser wallet extensions<br>• Sync Extension Settings - Synced wallet data<br>• \IndexedDB\chrome-extension_ - Extension databases<br>• 0.indexeddb.leveldb - LevelDB wallet storage<br>• \Monero - Monero wallet directory |
| Cloud credentials | • \.aws - AWS CLI credentials<br>• \.azure - Azure CLI credentials<br>• \.IdentityService - Azure identity tokens<br>• msal.cache - Microsoft Authentication Library cache (Office 365, Azure AD tokens) |
| FTP/SSH credentials | • \AppData\Roaming\FileZilla\recentservers.xml<br>• Software\Martin Prikryl\WinSCP 2\Sessions<br>• Software\Martin Prikryl\WinSCP 2\Configuration |
| Gaming/social platforms | • loginusers.vdf - Steam login sessions<br>• libraryfolders.vdf - Steam library info<br>• config.vdf - Steam configuration |

| | |
|---|---|
| | • DialogConfig.vdf - Steam dialog settings<br>• ssfn* - Steam Guard files<br>• \Telegram Desktop* - Telegram data<br>• Discord token files<br>• Communication app session files |
| Targeted browsers | Chrome, Microsoft Edge, Opera, Opera GX, Vivaldi, Firefox, Waterfox, Palemoon |

Table 1. Summary of Vidar 2.0's data theft capabilities

**Exfiltration and Cleanup (Phases 7-9):** The final phases involve screenshot capture for additional intelligence value, followed by comprehensive data packaging and exfiltration through HTTP multipart form submissions to a round-robin command-and-control (C&C) infrastructure that includes Telegram bots and Steam profiles as communication channels. The malware employs different operation modes to categorize stolen data and uses specific authentication tokens and build identifiers for tracking and victim management. Execution concludes with systematic cleanup of temporary artifacts and proper thread pool shutdown, demonstrating the malware's attention to operational security and forensic evasion.

```
if ( (_DWORD)result == -1558523533 )
{
  sub_1400431B0((__int64)&unk_14005C0E0, 32LL, (__int64)"16");// Build Version ID
  sub_1400431B0(
    (__int64)&unk_14005C100,
    64LL,
    (__int64)"0092e8e3725186dcd624d541a4875709                              ");// Bot version ID
  byte_14005C13F = 0;
  result = 2272844187LL;
  if ( !byte_14005C120 )
    result = 1677944871LL;
}
else
{
  mal_build_array((__int64)&unk_14005C1F0, (__int64)"https://telegram.me/ahnadar", 0x100uLL);
  mal_build_array((__int64)&unk_14005C2F0, (__int64)"ww_q1", 0x100uLL);
  mal_build_array(
    (__int64)&unk_14005C3F0,
    (__int64)"Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:145.0) Gecko/20100101 Firefox/145.0",
    0x100uLL);
  mal_build_array(
    (__int64)&unk_14005C4F0,
    (__int64)"https://steamcommunity.com/profiles/76561198780411257",
    0x100uLL);
  mal_build_array((__int64)&unk_14005C5F0, (__int64)"ww_q1", 0x100uLL);
  mal_build_array(
    (__int64)&unk_14005C6F0,
    (__int64)"Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:145.0) Gecko/20100101 Firefox/145.0",
    0x100uLL);
  mal_build_array((__int64)&unk_14005C7F0, (__int64)"https://telegram.me/ahnadar", 0x100uLL);
  mal_build_array((__int64)&unk_14005C8F0, (__int64)"ww_q1", 0x100uLL);
  mal_build_array(
    (__int64)&unk_14005C9F0,
    (__int64)"Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:145.0) Gecko/20100101 Firefox/145.0",
    0x100uLL);
  mal_build_array(
    (__int64)&unk_14005CAF0,
    (__int64)"https://steamcommunity.com/profiles/76561198780411257",
    0x100uLL);
  mal_build_array((__int64)&unk_14005CBF0, (__int64)"ww_q1", 0x100uLL);
  mal_build_array(
    (__int64)&unk_14005CCF0,
    (__int64)"Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:145.0) Gecko/20100101 Firefox/145.0",
```

Figure 8. Initialization and configuration of C&C server communication

MITRE ATT&CK Matrix

| Tactic | Technique ID | Technique Name |
|---|---|---|
| Defense Evasion | T1622 | Debugger Evasion |
| | T1497.001 | Virtualization/Sandbox Evasion |
| | T1027 | Obfuscated Files or Information |
| | T1055.001 | Process Injection: Dynamic-link Library Injection |
| | T1055.002 | Process Injection: Portable Executable Injection |
| Discovery | T1082 | System Information Discovery |
| | T1083 | File and Directory Discovery |
| | T1087.001 | Account Discovery: Local Account |
| | T1518.001 | Software Discovery: Security Software Discovery |
| Credential Access | T1555.003 | Credentials from Password Stores: Credentials from Web Browsers |
| | T1555 | Credentials from Password Stores |
| | T1552.001 | Unsecured Credentials: Credentials In Files |
| | T1528 | Steal Application Access Token |
| Collection | T1005 | Data from Local System |
| | T1113 | Screen Capture |
| Command and control | T1071.001 | Application Layer Protocol: Web Protocols |
| | T1102.001 | Web Service: Dead Drop Resolver |
| | T1573 | Encrypted Channel |
| Exfiltration | T1041 | Exfiltration Over C2 Channel |
| | T1020 | Automated Exfiltration |

Table 2. Observed MITRE ATT&CK tactics and techniques of Vidar Stealer 2.0

Conclusion

As Lumma Stealer activity continues to decline and underground actors migrate to Vidar and StealC alternatives, security teams should anticipate increased Vidar 2.0 prevalence in campaigns through Q4 2025. The malware's

technical capabilities, proven developer track record since 2018, and competitive pricing position it as a likely successor to Lumma Stealer's dominant market position.

Vidar 2.0's streamlined exfiltration routines, broader data stealing ability, and increased resistance to takedown measures, all aim toward a higher success rate for attacks and data breaches. Its enhanced anti-analysis features and rapid self-deletion also present additional challenges for detection and investigation.

Vidar's evolution comes at an opportune time. Whether this is by design or coincidence, proactive defense and continuous monitoring in combating infostealers remain as critical as ever. Organizations must ensure endpoint solutions are fully utilized and updated, while maintaining strong policies for credential management and user education, to protect against evolving threats like Vidar.

Proactive security with Trend Vision One™

Trend Vision One™one-platform is the only AI-powered enterprise cybersecurity platform that centralizes cyber risk exposure management and security operations, delivering robust layered protection across on-premises, hybrid, and multi-cloud environments.

**Trend Vision One™ Threat Intelligence**

To stay ahead of evolving threats, Trend Micro customers can access Trend Vision One™ Threat Insightsproducts, which provides the latest insights from Trend™ Research on emerging threats and threat actors.

**Trend Vision One Threat Insights**

- Emerging Threats: Vidar Stealer v2.0: Emergence and Technical Analysis

**Trend Vision One Intelligence Reports (IOC Sweeping)**

- Vidar Stealer v2.0: Emergence and Technical Analysis

**Hunting Queries**

**Trend Vision One Search App**

Trend Vision One customers can use the Search App to match or hunt the malicious indicators mentioned in this blog post with data in their environment.

malName:*VIDAR* AND eventName:MALWARE_DETECTION AND LogType: detection AND LogType: detection

More hunting queries are available for Trend Vision One customers with Threat Insights entitlement enabledproducts.

Indicators of Compromise (IoCs)

Indicators of Compromise can be found here.

Tags