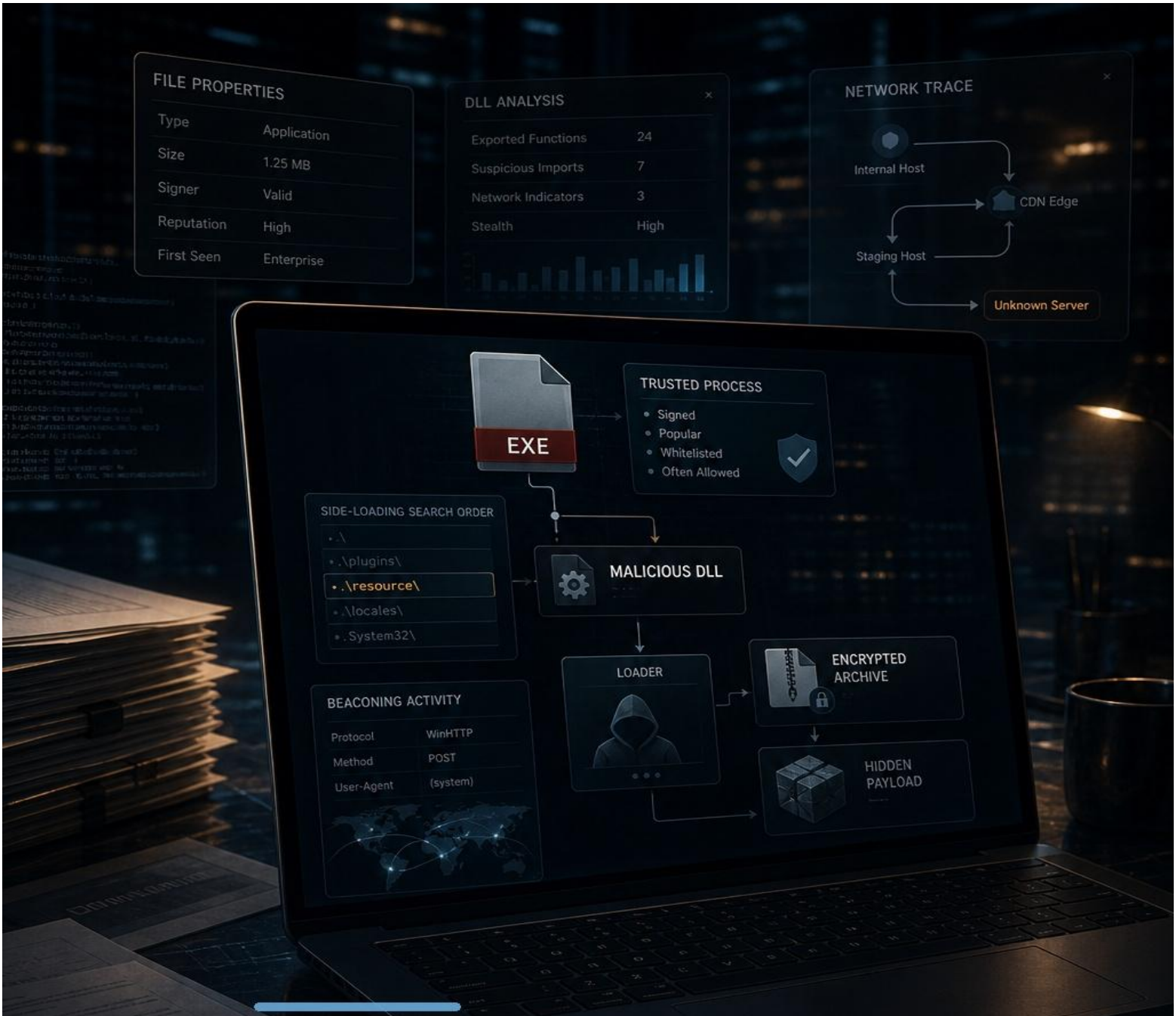


# CTI-Adobe Reader Side-Loading Malware Delivery Chain

Erik Westhovens

May 13, 2026

REPORT HEADER

# Cyber Threat Intelligence Report

**Subject:** Static Triage of a Socially Engineered Loader Chain Using a Signed Adobe Binary, WinHTTP Abuse, and a Concealed Second Stage

**Audience:** SOC, Incident Response, Threat Hunting, Malware Analysis, Security Leadership

**Date:** May 13, 2026

**Author:** Erik Westhovens

# Management Summary

The submitted sample set is assessed with high confidence to represent a malicious delivery chain that abuses a legitimate, validly signed Adobe Reader binary as a loader. The file is disguised as copyright-infringement evidence and is clearly designed to trigger urgency, trust, and user execution. The broader chain strongly indicates DLL side-loading, with a legitimate executable exposing WinHTTP functionality while the actual malicious stage appears to be concealed in companion files.

The risk is materially high because the visible executable benefits from the trust of a signed Adobe binary while adjacent artifacts point to intentional staging and defense evasion. These artifacts include a password-protected RAR container, a disguised WinRAR command-line helper renamed with a `.png` extension, and a hidden directory filled with decoy business documents. Taken together, the structure is consistent with a deliberately assembled malware installation chain rather than an isolated suspicious file.

Additional investigative context materially increases the severity assessment: the malware is now understood to hijack Microsoft 365 browser credentials or session material, authenticate into Microsoft 365, enumerate the data available to the compromised identity, and attempt exfiltration through Telegram or Discord. This changes the case from a probable staged loader alone into a clear cloud-account compromise and data-theft risk. Any host that executed this chain should be treated as potentially fully compromised, and any Microsoft 365 identity used on that host should be treated as potentially exposed.

### Key Takeaways

- This is best assessed as a malicious delivery chain and Trojan-style loader using DLL side-loading rather than as a benign signed-file anomaly.
- The trusted Adobe binary appears to function as a proxy loader while hidden artifacts support staged unpacking and likely second-stage deployment.
- The Microsoft 365 theft objective and likely Telegram or Discord exfiltration path make cloud-side investigation as important as endpoint triage.

Chapter 1

# 1. Executive Assessment and Verdict

This sample set should be treated as a malicious delivery chain built around signed binary abuse and staged payload concealment.

The strongest current verdict is malicious delivery chain / Trojan loader using DLL side-loading. The visible executable is not best understood as a custom commodity dropper, but as a trusted signed Adobe Reader binary used to load or support hidden malicious components. In operational terms, that makes the case more dangerous, not less, because trust is being exploited as part of the execution path.

Risk should be classified as critical. The chain supports initial access through social engineering, defense evasion through signed binary proxy execution, DLL side-loading, payload concealment inside an encrypted archive, likely outbound communication through WinHTTP, browser-based Microsoft 365 credential or session hijacking, cloud data enumeration, and attempted exfiltration through Telegram or Discord. These are not isolated indicators; they reinforce one another across the full file set and across both endpoint and cloud activity.

Confidence is high for the malicious delivery-chain conclusion and for the Microsoft 365 data-theft objective, medium for the exact payload family, and low for the final hard-confirmed command-and-control endpoint at the current stage. That confidence split is analytically normal for a case where core behavior is understood but some hidden payload layers remain encrypted or otherwise unavailable in plaintext.

### Assessment

- Verdict: malicious delivery chain / Trojan loader using DLL side-loading.
- Risk level should be treated as critical even without a final C2 IP.
- The trusted signed binary is part of the threat mechanism, not evidence against compromise.
- Cloud-account compromise materially expands the blast radius beyond the endpoint.

Chapter 2

# 2. Infection Theme and Social Engineering Lure

The execution trigger is built around legal pressure and urgency rather than technical deception alone.

The primary lure filename is `Bewijs_van_Auteursrechtinbreuk_ID5V35CX1C.exe`, which translates to evidence of copyright infringement. That theme is highly effective because it combines legal pressure, urgency, and curiosity. The victim is pushed toward immediate execution out of concern rather than technical trust alone.

This kind of lure is operationally significant because it is designed to bypass ordinary hesitation. A user may expect that a legal or rights-related complaint requires urgent review, and that urgency can suppress normal caution about attachment handling or file provenance.

The likely objective of the lure is straightforward: induce the victim to launch what appears to be a trusted executable, which then loads or supports hidden components and advances the next stage of the infection chain.

### Lure Analysis

- The lure uses legal pressure to increase execution likelihood.
- Urgency and fear are central to the user-execution mechanism.
- The visible executable is only the beginning of the attack chain.

Chapter 3

# 3. Loader Chain and Signed Binary Abuse

The main executable is a legitimate Adobe Reader binary whose trust characteristics appear to be deliberately abused.

The principal executable is a validly signed Adobe Reader binary rather than an obviously modified bespoke dropper. That matters because the file inherits trust from a real software publisher, making it more likely to evade suspicion by users and potentially by security controls that rely on trust or signing context.

The digital signature details support this interpretation. The file is signed by Adobe Inc. and chains to DigiCert Trusted G4 Code Signing infrastructure. In isolation, that signature would support legitimacy. In this case, however, the signed status appears to be part of the abuse path rather than evidence of safety.

The final analyst judgment is that the binary is very likely being used as a trusted loader for hidden components. Operationally, this should be treated as signed-binary proxy execution and abuse of a legitimate executable rather than as a harmless signed application present beside suspicious files.

### Signed Binary Abuse

- The loader EXE is legitimate and signed, but likely abused in context.
- Trust abuse is central to the malware-delivery model.
- A valid signature should not override chain-based malicious context.

Chapter 4

# 4. DLL Side-Loading and WinHTTP Exposure

The strongest technical pattern in the sample set is DLL side-loading supported by exposed WinHTTP functionality.

The executable imports `WINHTTP.dll` and exposes a set of WinHTTP-related functions, including `WinHttpOpen`, `WinHttpConnect`, `WinHttpOpenRequest`, `WinHttpSendRequest`, `WinHttpReceiveResponse`, `WinHttpQueryHeaders`, `WinHttpReadData`, and `WinHttpSetStatusCallback`. On its own, WinHTTP usage is not inherently malicious. In the full context of this chain, however, it becomes a major indicator of abuse.

The wider artifact set strongly supports a DLL side-loading scenario. A legitimate executable with network-capable imports is placed next to concealed or staged components, while the likely malicious stage remains hidden in companion files or an encrypted archive. The overall structure suggests that a local `winhttp.dll` or related side-loaded component may provide the real malicious functionality.

This interpretation is further supported by the absence of a clean plaintext second stage. The delivery chain appears designed so that the visible signed binary looks legitimate while the actual malicious capability is concealed until runtime or post-unpacking. Based on follow-on investigation, that concealed capability now appears to include browser credential or session hijacking against Microsoft 365, followed by access-token abuse or session reuse for cloud-side data discovery.

### Technical Finding

- WinHTTP imports materially strengthen the side-loading hypothesis.
- The likely malicious logic is displaced into companion components rather than the signed EXE itself.
- Context makes the difference between benign import behavior and malicious execution flow.

Chapter 5

# 5. Concealed Payload Ecosystem and File Inventory

The surrounding directory structure shows deliberate staging, concealment, and unpacking support.

Beyond the primary executable, the sample set contains a hidden or system-marked directory named `x` that includes hundreds of lure-style files and several highly suspicious objects. This is not normal software packaging. It is consistent with camouflage and staged malware delivery.

The most important suspicious companion files are the password-protected archive

`x\BIZ_2022 년_멘토링계획서_내부용.dat` and the disguised helper binary

`x\MEM_KB_구매요청서_2022.png`. The latter is not an image but a PE binary identified as WinRAR command-line tooling, likely intended to support unpacking or handling of concealed payload content.

The large volume of decoy `.docx`, `.xlsx`, `.pptx`, and `.pdf` filenames likely serves two functions: camouflage for the true malicious artifacts and credibility reinforcement for the broader lure theme. The sample is therefore best understood as a miniature malicious ecosystem rather than a single-file case.

### Observed Inventory

- The hidden `x` directory is a major structural indicator of staged delivery.
- The disguised `.png` helper is actually a WinRAR-related PE binary.
- Decoy business documents support camouflage and social credibility.

Chapter 6

# 6. Encrypted Archive, Public Correlation, and Stage-Two Behavior

The encrypted archive most likely contains the operational second stage, and current investigation now clarifies the likely cloud-theft objective.

The file `BIZ_2022 년_멘토링계획서_내부용.dat` is a password-protected RAR archive and is highly likely to contain the next stage of the chain. Plausible contents include a malicious `winhttp.dll`, configuration data, a second-stage payload, loader scripts, or additional support files. Given the structure of the surrounding chain, this archive is more likely to be the core payload container than incidental content.

Public threat-intelligence correlation strengthens this interpretation. The SHA256 of the loader EXE has been publicly associated with an Adobe Reader-themed sample, while a related public `winhttp.dll` sample has been identified as malicious. This does not prove that the exact same companion DLL is present here, but it supports the scenario in which the visible Adobe binary is legitimate while malicious behavior is delivered through a companion DLL or side-loaded component.

Follow-on investigative findings now indicate that the second-stage objective includes hijacking Microsoft 365 browser credentials or session artifacts, authenticating into Microsoft 365, enumerating the data available to the compromised user, and attempting exfiltration through Telegram or Discord. The current analytical limitation is that the archive remains encrypted with an unknown password and the likely malicious `winhttp.dll` is not yet available in plaintext. That prevents full family-level confirmation and deeper implementation detail, but it does not materially weaken the operational understanding of the threat.

### Stage-Two Assessment

- The encrypted archive is the most likely container for the real malicious stage.
- Public hash correlation supports a legitimate-loader-plus-malicious-companion scenario.
- Encryption limits family-level attribution but not the Microsoft 365 theft and exfiltration assessment.

Chapter 7

# 7. Microsoft 365 Account Abuse, Data Discovery, and Exfiltration

The most important follow-on behavior is cloud-account abuse and attempted data exfiltration rather than only local execution.

The malware is now understood to hijack Microsoft 365 browser credentials or session material and use that access to authenticate into Microsoft 365. Once inside the tenant context, the actor appears to inspect what data is accessible to the compromised identity and then attempt exfiltration through consumer or collaboration channels such as Telegram or Discord.

This behavior materially changes the investigative scope. The incident is no longer limited to malware containment on the endpoint; it becomes a cloud data-access case requiring review of sign-in activity, mailbox and SharePoint access, OneDrive usage, Teams-related data visibility, application-consent history, and any unusual export or download behavior tied to the affected identity.

From a containment perspective, the compromised endpoint and the Microsoft 365 identity must be treated as a shared incident surface. Token theft, browser-session hijacking, replay of web sessions, or captured credentials can all allow the actor to continue cloud-side activity even after the original file chain is removed locally unless sessions are revoked and the account is remediated.

### Cloud Impact

- Cloud investigation is mandatory because the threat objective includes Microsoft 365 data access.
- Endpoint cleanup alone is insufficient if tokens, sessions, or credentials were stolen.
- Telegram and Discord should be treated as plausible exfiltration channels during scope review.

Chapter 8

# 8. Detection Opportunities, ATT&CK Mapping, and Exfiltration Analysis Guidance

Detection and response now need to cover both host artifacts and Microsoft 365 data-exfiltration analysis.

Host-based detection opportunities still include execution of Adobe Reader-like binaries outside standard Adobe install paths, loading of `winhttp.dll` from non-system locations, presence of password-protected archives adjacent to signed loaders, PE files disguised as images, WinRAR CLI tooling in user-content directories, and hidden directories containing large sets of lure documents. These remain strong chain-level artifacts for hunting and detection engineering.

The sample also aligns well to ATT&CK techniques including `T1566.001` Spearphishing Attachment, `T1036` Masquerading, `T1218` System Binary Proxy Execution, `T1574.002` DLL Side-Loading, `T1027` Obfuscated or Compressed Files and Information, `T1140` Deobfuscate or Decode Files or Information, `T1105` Ingress Tool Transfer, `T1071.001` Web Protocols, `T1204.002` User Execution, and cloud-account abuse patterns consistent with stolen credentials or session tokens and data exfiltration to external services.

For exfiltration analysis, priority actions should include reviewing Microsoft Entra ID sign-in logs for unusual browser or location patterns, checking session revocation timing, examining Exchange Online, SharePoint, OneDrive, and Teams access tied to the affected user, identifying unusual downloads, exports, or API-driven bulk access, and reviewing proxy, DNS, firewall, and EDR telemetry for outbound traffic or browser activity involving Telegram and Discord domains, APIs, desktop clients, or web sessions. Investigators should also compare the data touched by the user during the suspected compromise window against the user's normal business role to identify over-broad access or targeted collection.

### Detection and Analysis

- Correlate endpoint evidence with Entra ID and Microsoft 365 audit telemetry.
- Review Telegram- and Discord-related outbound activity as part of exfiltration scoping.
- Look for unusual downloads, exports, and access to data beyond the user's normal role.

Chapter 9

# 9. Immediate Actions, Residual Risk, and Final Analyst Note

Containment must now include endpoint, browser, identity, and cloud-data scope review.

Recommended immediate actions are to isolate the affected endpoint, quarantine the entire sample directory rather than only the EXE, inspect non-system `winhttp.dll` module loads, search for similar lure files across mail stores and user directories, revoke all active Microsoft 365 sessions for the affected user, force credential reset and MFA re-registration where appropriate, review browser-stored credential and cookie exposure, and begin tenant-side scoping for data access and export activity.

Residual risk remains high because the final payload layer has not yet been decrypted or fully recovered. Even if the endpoint artifacts are removed, there remains a meaningful possibility of continued access through stolen sessions, replayed tokens, harvested credentials, or additional hidden tooling. Data-loss risk should therefore be treated as active until cloud-side scoping is complete.

Final analyst note: this sample is operationally dangerous not because the visible EXE is obviously custom malware, but because a legitimate and validly signed Adobe binary appears to be used as a trusted loader for hidden components. The combination of WinHTTP exposure, an encrypted RAR payload, a disguised WinRAR helper, a probable side-loaded companion DLL, and now the confirmed Microsoft 365 credential-hijacking and cloud-exfiltration objective makes this chain suitable for evasive delivery, staged compromise, and business-impacting data theft. Any host that executed this chain and any Microsoft 365 identity used on that host should be treated as compromised until proven otherwise.

### Containment Guidance

- Containment must include both endpoint remediation and Microsoft 365 identity response.
- Residual risk stays high until cloud-side scoping is complete.
- Treat executed hosts and affected identities as compromised until disproven.

Chapter 10

# 10. Sources

This report is based on local static analysis, follow-on investigative findings, and limited public hash correlation.

Primary source 1: local static analysis of the submitted sample set.

Primary source 2: follow-on investigative finding that the malware hijacks Microsoft 365 browser credentials or session material, authenticates into Microsoft 365, inspects accessible data, and attempts exfiltration through Telegram or Discord.

Primary source 3: public hash-correlation context from Hybrid Analysis: https://hybrid-analysis.com/submissions/sandbox/files?lang=%2Fsubmissions%2Fsandbox%2Ffiles%253Flang%253D%2Fsubmissions%2Fsandbox%2Ffiles%25253Flang%25253D%2Fsubmissions%2Fsandbox%2Ffiles%2525253Flang%2525253D%2Fsubmissions%2Fsandbox%2Ffiles%252525253Flang%252525253D%2Fsubmissions%2Fsandbox%2Ffiles%25252525253Flang%25252525253D%2Fsubmissions%2Fsandbox%2Ffiles%2525252525253Flang%2525252525253D%2Fsubmissions%2Fsandbox%2Ffiles%252525252525253Flang%252525252525253D%2Fsubmissions%2Fsandbox%2Ffiles%25252525252525253Fsort%252525252525253D%2525252525252525255Etimestamp%252525252525252526page%25252525252525253D10%252525252525252526lang%25252525252525253Dzh%2525252525252526lang%252525252525253Dar%25252525252526lang%2525252525253Dtr%252525252526lang%25252525253Did%2525252526lang%252525253Dvi%25252526lang%2525253Dpl%252526lang%25253Dzh%2526lang%253Dfr%26lang%3Dru