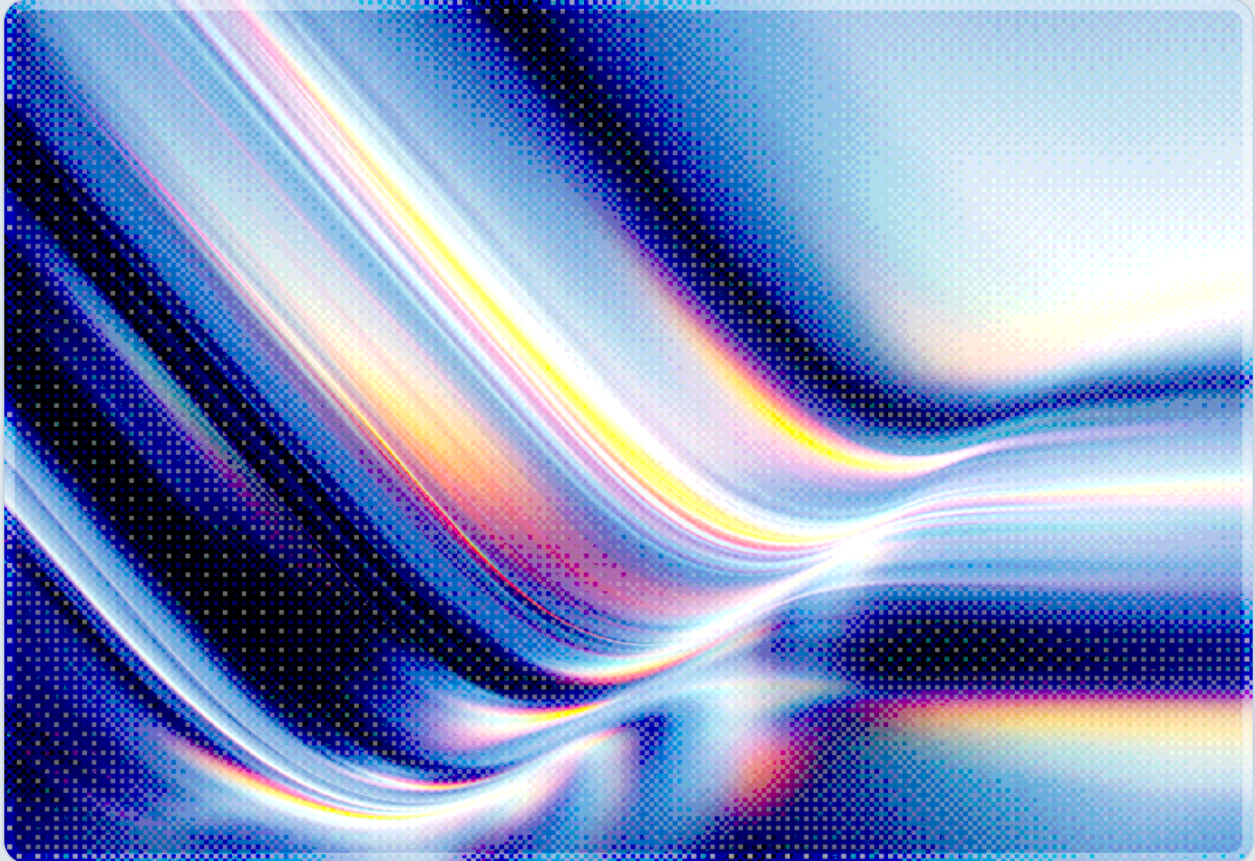


# AOMEI Backupper Update Tampering: Minute-Long Windows, Persistent Backdoor

Security Implications and Guidance for Unsigned Update  
Channels

2026-10-17

 AI-assisted Rapid Research



CSAI

cloud  
**CSA** security  
alliance®

**© 2026 Cloud Security Alliance. Some rights reserved.**

You may download, store, display, view, print, redistribute, and link to this document in its original, unmodified form, provided that attribution to the Cloud Security Alliance is maintained and all trademark and copyright notices remain intact.

This document may not be modified or altered. You may quote portions of the document as permitted by the Fair Use provisions of the United States Copyright Act, provided that attribution is given to the Cloud Security Alliance.

This document may be shared on professional and social media platforms in its original form with attribution.

*This document was generated with AI assistance and has not undergone official CSA review and approval processes.*

---

## Key Takeaways

Between May and July 2026, an actor tracked by Google Threat Intelligence Group (GTIG) as UNC7017 modified two configuration files on the vendor-operated servers that AOMEI Backupper and AnyViewer poll for updates. Each modification lasted five to ten minutes before the files reverted to legitimate content, which is why point-in-time checks of the vendor's URLs would have shown nothing wrong [1][2]. Any installed client that checked for an update inside one of the three confirmed windows could download and run a downloader called CROSSROAD, which then staged a backdoor named SHINYROAD inside a trojanized copy of the commercial Radmin remote administration tool [1].

The weakness lay in the update protocol, not in a software vulnerability. According to GTIG, the client trusted two unsigned files, and the checksum meant to validate the download was stored in the same unsigned file that named the package [1]. Transport-layer HTTPS protected the connection to the server but could not tell the client that the server's content had been altered.

Server-side remediation does not clean machines that were already infected. AOMEI told Mandiant on July 29, 2026 that it had taken corrective action to secure its servers, but the backdoor persists on endpoints that installed it earlier [1][3]. Organizations that run AOMEI Backupper or AnyViewer should hunt for the artifacts listed below rather than assuming a vendor fix resolved their exposure. Because backup software typically runs with elevated privilege and can read most disk contents, a compromise of it may expose more data than a compromise of a typical utility.

## Background

AOMEI Backupper is a Windows backup and disaster recovery product, and AnyViewer is a remote access product from the same vendor. Both use a periodic HTTPS poll to retrieve a small configuration file, `down_html.ini`, from the vendor's update path. That file carries a version number, an MD5 value, and a download URL for an upgrade package. A separate notification service reads a second file, `Notifycfg.ini`, which defines pop-up windows the product displays and can launch local programs from the software's directory [1].

The discovery path shows how little evidence survived. The ManageEngine Threat Analysis Lab (METAL) first saw the behavior on May 28, 2026, when endpoint detection telemetry showed AOMEI Backupper installing Radmin on two unrelated endpoints in different countries [2]. By the time one affected organization escalated the activity on June 7, the dropper had deleted itself and its hash was not present in VirusTotal [2]. METAL then monitored AOMEI's delivery URLs and captured a tampering event on July 16, and GTIG and Mandiant report observing the pattern recurring on July 20 [1][2]. METAL reports that its behavioral telemetry contradicted an initial vendor explanation, given when the activity was reported on June 7, that the traffic had been intercepted in transit [2]. The sources reviewed do not include a published AOMEI statement describing that explanation, so it is characterized here only through METAL's account.

GTIG tracks the operator as UNC7017 and describes its attribution as based on consistent tooling and infrastructure patterns across the waves [1]. The researchers did not name a nation-state or criminal group, and they state that the means by which the actor obtained write access to AOMEI's servers has not been disclosed [1]. GTIG's observations are consistent with indiscriminate delivery to any polling client, followed by selective follow-on activity. Confirmed victims were identified in IT services, telecommunications, and higher education across 11 countries: Australia, Brazil, India, Pakistan, Portugal, Singapore, South Africa, Spain, Switzerland, Türkiye, and the United States [1][3]. Because the windows were short and some may fall outside the researchers' telemetry, the total number of affected systems is unknown.

Update-channel compromise has precedent in 2025–2026. Reporting on the Notepad++ incident describes attackers who controlled the hosting that served the updater's download-URL script and selectively redirected requests to trojanized installers, an operation researchers assessed as consistent with a state-sponsored actor [4]. The AOMEI case differs in operating style: the actor modified files for minutes at a time and delivered to any client that polled, whereas the Notepad++ activity was reported as selective and long-lived [3][4]. Both cases show that an update server can serve as a distribution point for any actor who controls it.

## Security Analysis

### The tampering windows

GTIG and METAL document three confirmed windows, summarized below. The times come from GTIG's published analysis [1].

| Date (2026) | Window (UTC) | Duration   | Notable change                                                                    |
|-------------|--------------|------------|-----------------------------------------------------------------------------------|
| May 28      | 05:40–05:50  | 10 minutes | First observed delivery of CROSSROAD; discovered through endpoint telemetry       |
| July 16     | 08:33–08:39  | 6 minutes  | Tampering captured by METAL's URL monitoring                                      |
| July 20     | 10:35–10:40  | 5 minutes  | Download moved to Google Cloud Storage; pop-up used to coerce NTLM authentication |

The short duration likely benefits the attacker in two ways, though this note treats that as an inference from the documented durations. It limits the population that downloads the payload to systems that poll during the window, which reduces the chance of broad detection. It also defeats the most common triage step, which is to fetch the vendor's URL after an alert and compare the result with known-good content. GTIG advises responders to preserve cached configuration files from the endpoint instead, and to correlate alerts with exact timestamps in proxy and firewall logs [1].

## Infection chain

When a client received the altered `down_html.ini`, it downloaded the package it named and executed `upgrade-now.exe`, which GTIG calls CROSSROAD [1]. CROSSROAD performs several checks before acting. It queries IP-reflection services and aborts if the reported country is China, it uses a mutex to prevent concurrent instances, and it restores `down_html.ini` to its legitimate contents to conceal the change [1]. It then downloads a component archive from attacker-controlled infrastructure, unpacks it into `%LOCALAPPDATA%\Radmin\`, and registers a scheduled task named `RadminComponentRuntime` that runs every three hours [1]. Finally, it deletes itself using a child process and NTFS alternate-data-stream renaming, a technique that explains why METAL found no dropper to analyze [1][2].

The staged content includes unmodified, signed Radmin binaries from Famatech together with an unsigned DLL, `BlkScrDl.dll`, which the signed executable loads through DLL search-order hijacking [1]. That DLL registers a COM server, `RuntimeComponent.dll`, which embeds a PocketPy interpreter and executes encrypted Python bytecode in memory [1]. GTIG calls this stage SHINYROAD. It provides an interactive remote shell, adds randomized junk traffic to its communications, and communicates with a single hardcoded command-and-control address over a custom protocol [1].

SHINYROAD has no persistence of its own and relies on the scheduled task CROSSROAD created [1]. Because the signed Radmin executable loads the unsigned DLL, publisher-based allowlisting that evaluates only the executable may not flag the implant. Persistence therefore comes from the scheduled task rather than from the backdoor itself, which is the sense in which the title calls it persistent.

In mid-July, the actor deployed follow-on tooling. GTIG reports Adaptix C2, an open-source post-exploitation framework, delivered through components disguised as Nitro PDF software, with a second scheduled task named `NitroRuntime` [1]. This suggests that SHINYROAD served as an initial access foothold from which the operator selected systems for further activity, though the report does not describe what the operator did on those systems after deployment.

## Credential coercion and evasion on July 20

The July 20 window added two techniques. First, the download URL moved to `storage.googleapis.com`, which GTIG assesses may help the initial fetch pass proxy controls that block newly registered domains, while later command-and-control traffic continued to use the actor's own domains [1]. Second, the modified `Notifycfg.ini` defined a 1×1-pixel pop-up pointing to a `file://` WebDAV path on an attacker-controlled IP address [1]. When the client rendered that pop-up, Windows attempted to authenticate to the remote server, which could expose a NetNTLMv2 challenge response without any code executing on the endpoint [1]. The pop-up channel required no malware to be present: a client that merely polled during the window could leak authentication material. NetNTLMv2 responses can be subjected to offline cracking or, in some environments, relayed, so affected organizations should treat accounts that were active on exposed hosts as potentially exposed [1].

## Why the protocol design mattered

The update scheme combined three properties that together made a short write to a server sufficient for code execution on clients. The configuration files were not signed, the integrity value was carried in the same file as the download URL, and the pop-up mechanism could launch local programs and reach `file://` paths [1]. An MD5 value placed next to the URL it protects confirms only that the download matches what the file author intended, so an attacker who controls the file controls both. GTIG's recommendations for vendors follow from this: sign configuration files with asymmetric keys and verify before parsing, verify Authenticode signatures on executables and archives, pin download domains, restrict URI schemes in embedded web handlers, separate security updates from promotional content, and monitor origin servers for file integrity changes with immediate alerting [1].

For defenders, the broader lesson is that a trusted software publisher's update path is a trust relationship that most enterprises do not inspect. A backup agent is typically installed widely, runs with elevated rights, and is permitted to reach the internet, so network and endpoint policy often leaves it unrestricted. Those same properties make it an effective carrier.

## Recommendations

### Immediate Actions

Inventory every host running AOMEI Backupper or AnyViewer, including copies installed by individual users and by managed service providers. On each, look for the scheduled tasks `RadminComponentRuntime` and `NitroRuntime`, the directories `%LOCALAPPDATA%\Radmin\` and `%LOCALAPPDATA%\Nitro Pro\`, unsigned DLLs alongside signed Radmin executables (notably `BlkScrDl.dll` and `Eztwain4.dll`), and `upgrade-now.exe` or `upgrade-now.zip` in AOMEI install paths [1]. Preserve any cached `down_html.ini` and `Notifycfg.ini` files under the AOMEI `Tmphtml` directories before remediation, because they may be the only record of a tampered download [1]. Use the YARA rules published by GTIG and the Sigma rules published by METAL for sweeps [1][2].

Block the published command-and-control domains and IP addresses at egress, and review historical proxy and DNS logs for connections to them and to the `aomeiupdate` Google Cloud Storage path from the tampering dates [1][3]. Restricting outbound SMB (TCP 445) and WebDAV to external addresses at the perimeter, including the WebClient service's HTTP fallback, reduces exposure to NTLM coercion of this kind [3]. If any host contacted the actor's WebDAV address, reset credentials for accounts that authenticated on that host and review for relay or cracking activity. Treat confirmed infections as full compromise of the host and any credentials it held, and rebuild from known-good media, since a backup agent may have had access to stored backup credentials and repositories.

### Short-Term Mitigations

Apply alerting for signed binaries that load unsigned DLLs from user-writable paths, for scheduled tasks created by processes launched from application-update directories, and for PowerShell started with `- WindowStyle Hidden` from `%TEMP%` or application data folders. Add detections for self-deletion behavior such as NTFS alternate-data-stream renaming. Constrain the update behavior of third-party

utilities where possible: route their updates through a managed software distribution system, disable built-in updaters for tools that do not need them, and require that updated binaries come from approved publishers with valid signatures.

When responding to alerts about supply chain compromise, record the vendor-side state at alert time and do not rely on a later live fetch. Where egress proxies are in use, log full URLs and response hashes for software-update domains so that a short-lived substitution can be reconstructed afterward. Ask software vendors directly whether their update metadata is signed, and treat a negative answer as a finding in the vendor risk record.

## Strategic Considerations

Organizations should extend third-party risk practices to cover the integrity of the update channel, not only the integrity of the installer. Procurement and vendor-assessment questionnaires can ask whether manifests are signed, whether keys are separate from the web-hosting environment, how origin changes are monitored, and whether the vendor can detect a modification that lasts only minutes. Where a vendor cannot answer, compensating controls such as application allowlisting, restricted egress for the product, and managed updates become the primary defense.

Security teams should also plan for a class of incident in which evidence disappears quickly. Endpoint telemetry that preserves process lineage and file paths proved decisive here, since the dropper deleted itself and the vendor's URLs later served clean content [2]. Retention policies for EDR and proxy data should be set with that in mind. Finally, since this is a single reported campaign with undisclosed initial access, defenders should avoid inferring that AOMEL is unique; the combination of unsigned update metadata and privileged agents likely exists in other utilities (this is the author's inference, with the Notepad++ case [4] as a second documented example of update-channel abuse), and an inventory of auto-updating privileged software is a reasonable starting point for review.

## CSA Resource Alignment

CSA's *Mastra npm Takeover: 144 AI Packages Backdoored* examines a June 2026 incident in which a trusted publishing channel was used to distribute backdoored packages [5]. The parallel is the abuse of a legitimate distribution path to reach downstream users who have no reason to distrust it. In both cases the defender's problem is that the malicious content arrived through an authorized channel, which supports the recommendation here to log and verify update content independently of the vendor's own signals.

CSA's *Software Transparency: Securing the Digital Supply Chain* addresses SBOM adoption and supply chain security implementation strategies [6]. An inventory of installed software and its update mechanisms, the starting point recommended above, depends on this kind of transparency, but an SBOM does not state whether update metadata is signed, so vendor questionnaires should ask about it explicitly. The AI Controls Matrix v1.1 provides a control framework for threat and vulnerability management and for supply chain management domains that can be used to structure these vendor and endpoint requirements [7].

# References

- [1] Google Threat Intelligence Group. "[Investigating Ephemeral Supply Chain Tampering](#)." Google Cloud Blog, October 15, 2026.
- [2] ManageEngine Threat Analysis Lab. "[AOMEI Backupper supply-chain compromise: Investigation and analysis](#)." ManageEngine, October 16, 2026.
- [3] The Hacker News. "[Attackers Tampered with AOMEI Backupper's Update Servers for Minutes to Deliver a Backdoor](#)." The Hacker News, October 16, 2026.
- [4] eSecurity Planet. "[Notepad++ Update Servers Hijacked in Targeted Supply Chain Attack](#)." eSecurity Planet, February 2, 2026.
- [5] Cloud Security Alliance. "[Mastra npm Takeover: 144 AI Packages Backdoored](#)." CSA Labs, June 18, 2026.
- [6] Cloud Security Alliance. "[Software Transparency: Securing the Digital Supply Chain](#)." CSA, 2025.
- [7] Cloud Security Alliance. "[AI Controls Matrix \(AICM\) v1.1](#)." CSA.