Exploiting trust in the vendor – cybersecurity provided binary turned malicious

Author: Piotr Bienias
No of pages: 31
Read time: ~18 min

Contents

1 Summary

This article documents a real-world intrusion in which threat actors abused a legitimate, digitally signed Trend Micro component to deploy a sophisticated, multi-stage Remote Access Trojan (RAT) inside an enterprise network. The investigation began after abnormal internal authentication activity that revealed a compromised web-facing server running an outdated component. Following initial access, the attackers conducted extensive internal discovery across the environment, harvested credentials, mapped Active Directory and selectively compromised high-value systems.

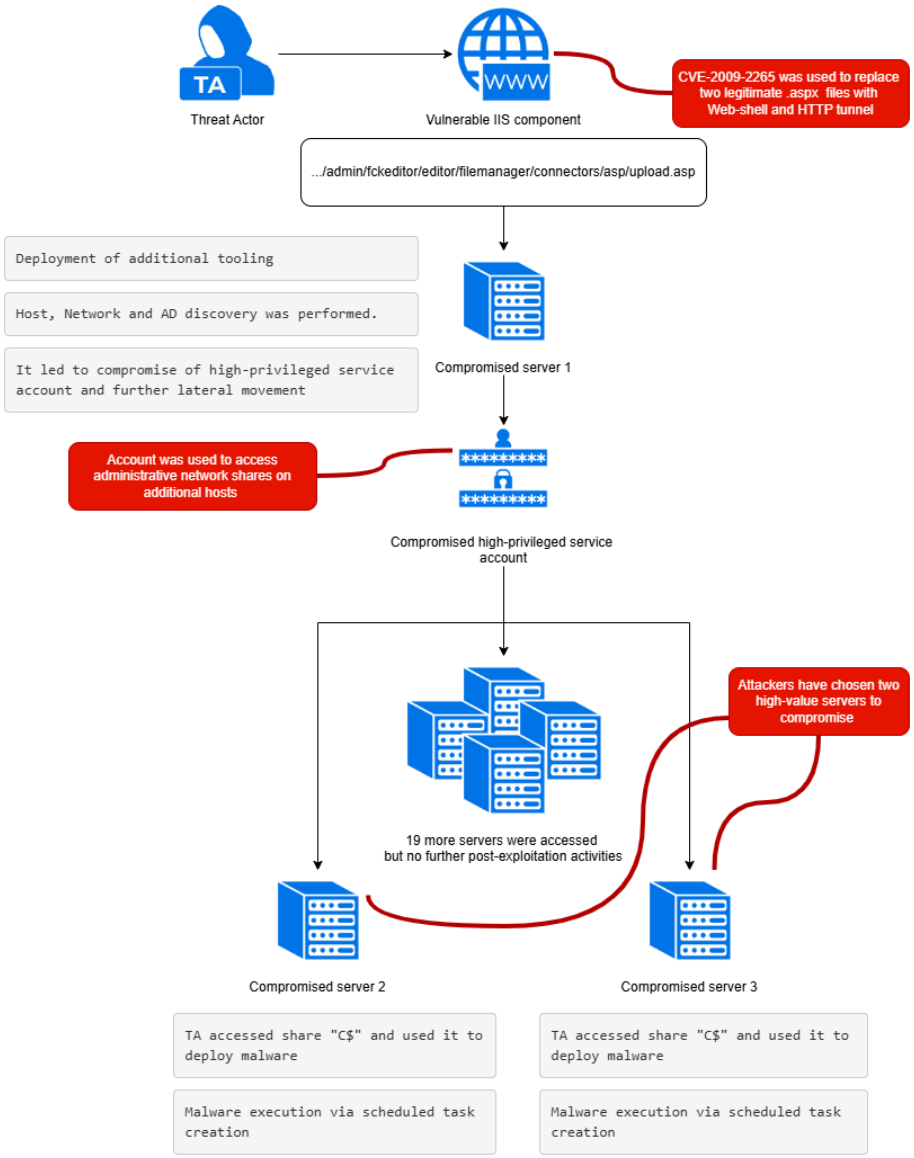
Rather than deploying a clearly malicious loader, the attackers repurposed Trend Micro's ActiveUpdate binary (Build.exe), renamed to LockAp.exe and leveraging DLL side-loading to run a malicious dbghelp.dll. This DLL functioned as a stealthy in-memory loader, decrypting an RC4-encrypted payload stored in a separate .apx file and executing it without writing the final malware to disk. The resulting backdoor established persistence via services, scheduled tasks and registry RUN keys, injected code into trusted Windows processes and communicated with command-and-control infrastructure using encrypted multi-protocol channels, including named pipes.

The campaign demonstrates how attackers can weaponize trust in security vendors by abusing signed binaries to evade detection and bypass application trust controls. While the techniques observed during this breach- DLL side-loading, in-memory execution and living-off-the-land reconnaissance - are well known, their combination with a trusted cybersecurity vendor binary significantly increased stealth and dwell time. The case highlights, yet again, the importance of behavior-based detection and post-exploitation monitoring, as static indicators and signature-based defenses alone are insufficient against modern intrusions.

2 Analysis

2.1 Background and Initial Access

Atos uncovered this activity during an incident involving a compromise of a **web-facing server** running an outdated FCKEditor component, which has known vulnerability (CVE-2009-2265), leading to Remote Command Execution.



High-level overview of the attack

After initial compromise, the attackers dropped and executed numerous tools, including:

- Enumeration utilities: fscan.exe, nltest.exe, systeminfo.exe
- Active Directory reconnaissance: setspn.exe, ldapbrute.exe, cmdkey.exe, Kerbrute, qscan.exe
- File archivers and custom droppers
- Legitimate system tools (LOLBins)

Attackers maintained access for over 48 hours before deploying:

- LockAp.exe (renamed legitimate Trend Micro executable: Build.exe)
- Malicious dbghelp.dll
- Encrypted payload LockAp.apx

According to what Atos Researchers have been able to establish, “Build.exe” is related to Trend Micro’s ActiveUpdate module, built-in automatic system used by many Trend Micro’s security products to obtain latest detection patterns, scan engines, program updates, and other security components.

By using crafted DLL attackers were able to bypass detection and establish various persistence mechanisms on the hosts.

2.2 Discovery stage

Comprehensive analysis of attacker actions shows a structured, multi-phase discovery process consistent with advanced threat actor tradecraft. The activity unfolded in several sequential stages:

2.2.1 Initial Host Profiling and Environment Baseline

Attackers began by establishing context on the compromised system:

- **Identity and privilege assessment** – using whoami.exe to determine the active user's token and security groups.
- **System fingerprinting** – through systeminfo.exe to gather OS details, hardware profile, and patch level.
- **Tooling preparation** – leveraging curl.exe to retrieve additional components from attacker-controlled infrastructure.

2.2.2 Network and Session Discovery

Once host characteristics were established, discovery expanded to the surrounding environment:

- **Local network mapping** – querying ARP tables via arp.exe and validating reachability using ping.exe.
- **Session enumeration** – running quser.exe to identify active and dormant user sessions, looking for privileged accounts logged into the system.

2.2.3 Application and infrastructure profiling

Mid-stage discovery targeted systems supporting web services and configuration management:

- **IIS inspection** – through tools like iissetup.exe and inetmgr.exe to analyze web server configuration and application pool contexts.
- **Provisioning and cloud-hybrid visibility** – using provtool.exe to inspect provisioning packages and azurearcsystray.exe to understand the system's Azure Arc integration.

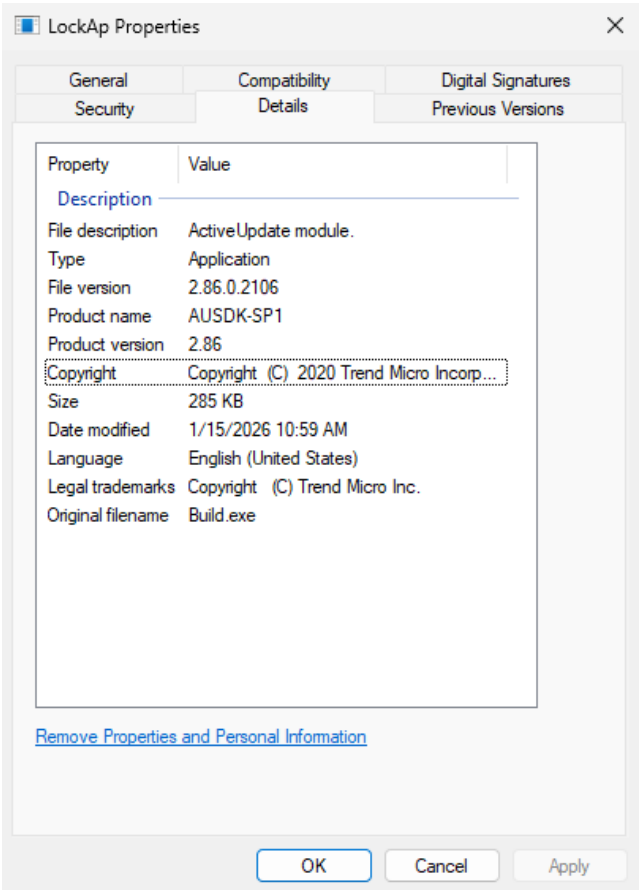
2.2.4 Advanced Active Directory and Service Enumeration

The final discovery phase focused on domain-level discovery and identifying accounts or systems for further compromise:

- **Domain and trust mapping** – via nltest.exe to enumerate Domain Controllers and trust relationships.
- **Aggressive LDAP enumeration** – using ldapbrute.exe to obtain a detailed directory structure.
- **Credential and service account targeting** – using setspn.exe to locate Kerberoastable accounts and cmdkey.exe to extract stored credentials.
- **High-speed internal scanning** – with fscan.exe and qscan (under the name "q.exe") to rapidly survey reachable hosts and services.

2.3 High-level overview of malware behavior

Legitimate and signed Trend Micro's application, used by threat actors, has following metadata:

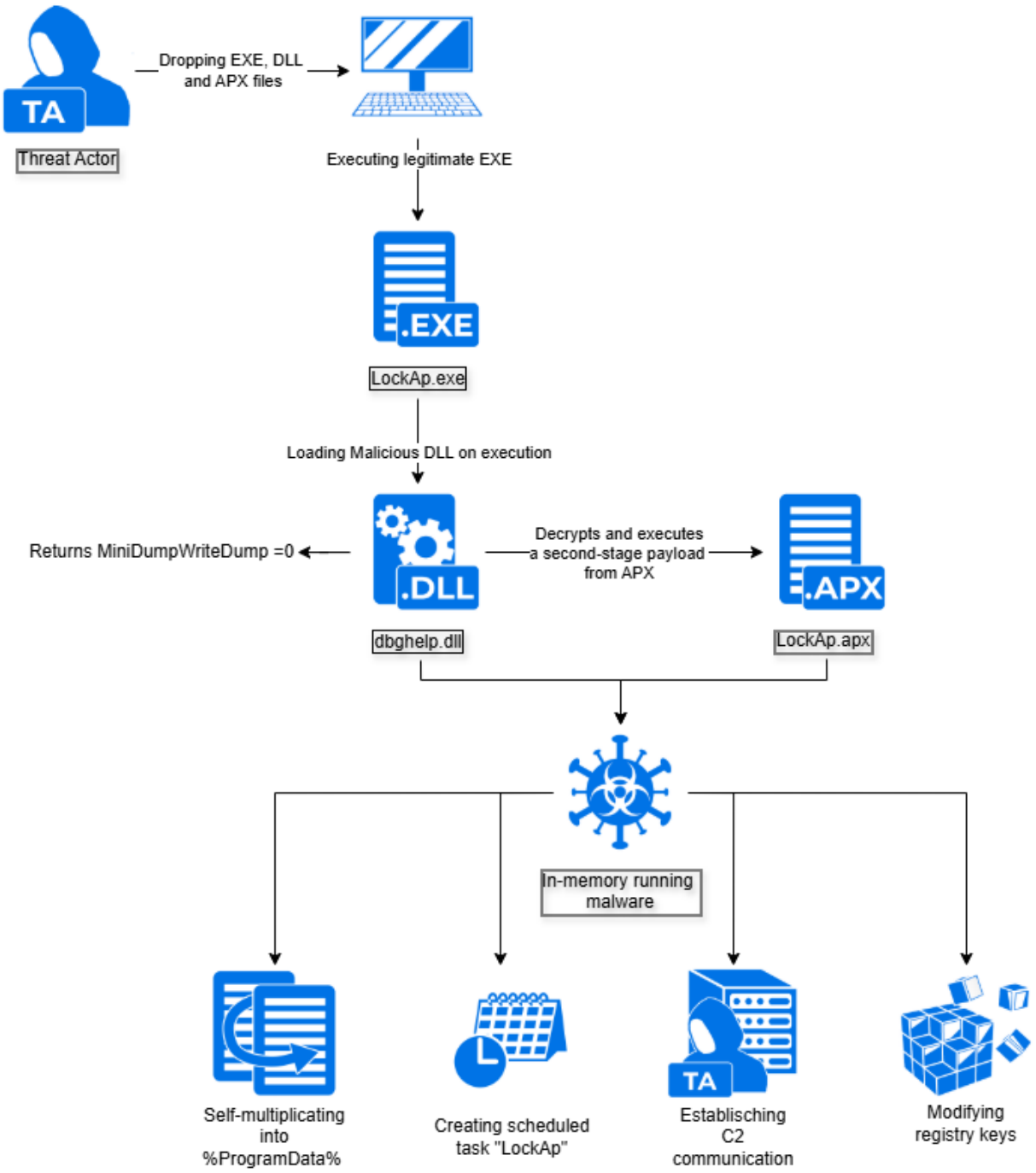


LockAp.exe properties

property	value
certificate	
revision	0x0200 (WIN_CERT_REVISION_2_0)
type	0x0002 (WIN_CERT_TYPE_X509)
file-offset (from)	0x00043800
file-offset (to)	0x000475B0
size-certificate	0x3DB0 (15792 bytes)
size-PKCS7	0x3DA2 (15778 bytes)
size-PKCS7-null-padding	2 bytes
certificate > sha256	CBF903AB55242559C904A03D46CF3CEF83100A5A5BE0CD67C122EE3B02FD7D6A
details	
name	**Trend Micro, Inc.**
signature-info	This digital signature is OK.
issued-by	DigiCert EV Code Signing CA
stamp > signing	Wed Jul 08 09:55:51 2020
valid-from	Mon Jun 22 01:00:00 2020
valid-to	Wed Oct 27 13:00:00 2021
serial-number	040828C3999F80971F3E8D63429AC5EB
thumbprint	0B67CBAEF32CE19ADD0C2ACD852D764BAE685B25
signature-algorithm	sha1RSA
program-name	n/a
email	n/a
more-info-url	n/a
tail	
n/a	-

LockAp certificate details

Static and dynamic analysis of all the files created by attackers showed them to be sophisticated and highly obfuscated malware with multiple steps of execution.



Simplified malware behavior graph

Following is a high-level overview of malware behavior:

1. Legitimate LockAP.exe (Build.exe) binary is executed.

2. LockAp.exe loads malicious dbghelp.dll on execution
3. DLL performs several operations:
    a. API Hashing – to evade static analysis and signature-based detection by hiding API calls
    b. Code Injection and hooking – function is being hooked to ntdll.dll in memory
    c. Decrypting malicious payload – using RC4 key “drturyTTteu7345Q” it extracts malicious binary from LockAp.apx
    d. Shellcode execution – using previously allocated memory buffer it executes the decrypted payload in memory
    e. Exporting fake export – to masquerade as legitimate dbghelp.dll it exports MiniDumpWriteDump = 0
4. Malicious payload loaded into memory does:
    a. File system activity – Copies itself to “C:\ProgramData\LockAp”
    b. Persistence Mechanisms – Creates scheduled task and installs Windows service
    c. Process Injection and Privilege Escalation – Spawns “msiexec.exe”, creates remote thread inside of it and accesses “Explorer.exe” multiple times.
    d. Network Activity – queries DNS for local proxy, tries to access it via local IP and connects to C2 under 64.190.113[.]170:443

2.4 Analysis of the extracted malicious code.

The analyzed malware is composed of three primary components that work together to enable stealthy execution, payload decryption, and long-term persistence:

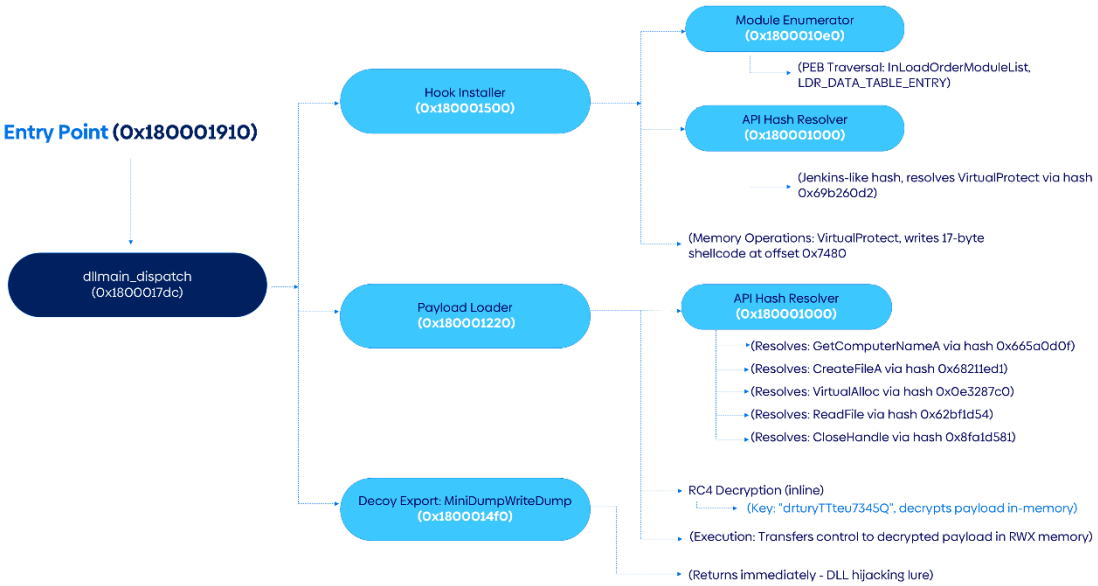
1) LockAp.exe
2) dbghelp.dll
3) LockAp.apx

2.4.1 LockAp.exe — Signed Host Used Only for DLL Side-Loading

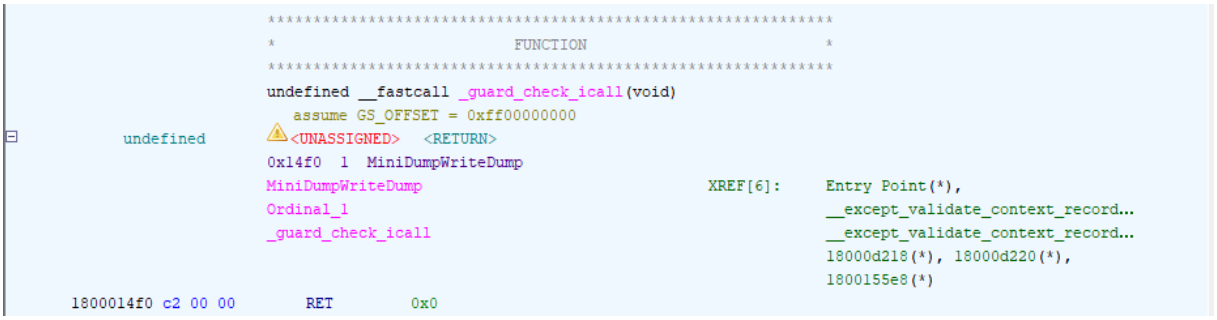
LockAp.exe is a legitimate, signed 64-bit application and remains unmodified. It serves exclusively as the trusted container used to side-load the attacker's dbghelp.dll. Although it contains crypto-related strings, Base64 tables and extensive localization data, it contains no malicious logic of its own and simply provides the execution context needed for DLL loading.

2.4.2 Malicious dbghelp.dll — Primary Loader

This dbghelp.dll is a trojanized replacement of the legitimate Windows debugging library, deployed via DLL side-loading to establish stealthy code execution. While exporting a non-functional MiniDumpWriteDump to mimic legitimate functionality, its true purpose is to patch memory in loaded modules during DLL attachment, redirecting execution to a payload loader that retrieves RC4-encrypted files, decrypts them in-memory using API resolution by hash to evade detection, and executes arbitrary code without touching disk, a fileless malware delivery mechanism.



dbghelp.dll function tree



MiniDumpWriteDump function

MiniDumpWriteDump (0x1800014f0)

Decoy export function that immediately returns without performing any operation. Acts as a lure to disguise the DLL as a legitimate debugging library replacement.

entry (0x180001910)

Primary entry point that initializes security cookies and dispatches to dllmain_dispatch for payload execution.

dllmain_dispatch (0x1800017dc)

Core orchestration function handling DLL lifecycle events. On DLL_PROCESS_ATTACH, installs memory hooks and initiates CRT initialization. Implements Control Flow Guard checks and manages thread attach/detach events.

FUN_180001500 (Hook Installer)

Locates first loaded module via PEB traversal, resolves VirtualProtect by hash (0x69b260d2), and patches 17 bytes at offset 0x7480 with shellcode redirecting execution to the payload loader. Uses PAGE_EXECUTE_READWRITE permissions for code modification.

FUN_180001220 (Payload Loader)
Multi-stage loader that: (1) constructs file path using computer name, (2) reads encrypted content via hashed API calls, (3) performs RC4 decryption with hardcoded key "drturyTTteu7345Q", (4) allocates RWX memory, (5) executes decrypted payload in-memory. Implements fileless malware execution pattern.

FUN_180001000 (API Hash Resolution)
Custom GetProcAddress implementation using Jenkins-like hash algorithm. Iterates module exports, computes hashes, and resolves functions without direct API calls, evading API monitoring and static analysis.

FUN_1800010e0 (Module Enumeration)
Traverses PEB InLoadOrderModuleList to locate modules by name, performing case-insensitive string comparison. Returns module base addresses for subsequent memory manipulation.

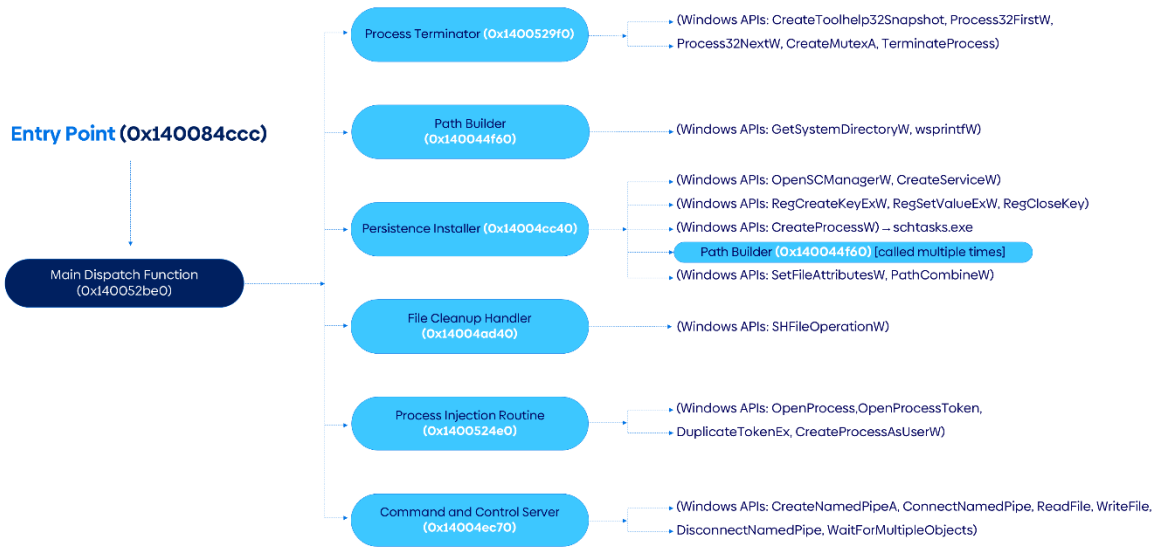
2.4.3 LockAp.apx – Encrypted Payload

The LockAp.apx file serves as the encrypted payload container and is the core malicious component of the LockAp attack, deliberately separated from the execution and loading logic to evade detection. Rather than containing executable code in a directly runnable form, LockAp.apx stores the second-stage RAT encrypted with RC4, preventing static scanners and signature-based defenses from identifying the malware on disk.

2.4.4 Second Stage — Memory-Resident Backdoor

This is a sophisticated Remote Access Trojan (RAT) that establishes multi-vector persistence through service creation (service name: LockAp), registry modification (HKLM\System\CurrentControlSet\Services\LockAp with Index

value, and Software\Microsoft\Windows\CurrentVersion\Run), and scheduled task deployment. The malware drops its main payload at C:\ProgramData\LockAp\LockAp.exe along with additional components (LockAp.apx, K, rt, kl), acquires SeDebugPrivilege for elevated operations, implements named pipe-based C2 communication, and uses position-dependent XOR encryption (keys at 0x1400c5430, 0x1400c53f0, 0x1400c5458, 0x1400c5470) to obfuscate strings including registry paths, service names, and privilege names. It supports process injection, token manipulation, and includes potential keylogger functionality.



LockAp malware function tree

Entry Point (0x140084ccc)

Program entry point that initializes stack security cookies, calls GetConsoleWindow() and ShowWindow(hWnd, 0) to hide window, opens current process token with TOKEN_ADJUST_PRIVILEGES | TOKEN_QUERY (0x28), decrypts and acquires privilege name "SeDebugPrivilege" through LookupPrivilegeValueA and AdjustTokenPrivileges, parses command-line with CommandLineToArgvW, and dispatches to main orchestrator.

Main Dispatch Function (0x140052be0)

Main dispatcher that decrypts 16-byte privilege string "SeDebugPrivilege" using XOR key from DAT_1400bf488, decrypts mutex name (14 bytes from 0x1400c5458), checks argc for /install or -u parameters via lstrcmpiW, creates named mutex and event using decrypted identifiers, calls persistence installer 0x14004cc40 on -u parameter, drops files via PathCombineW to "C:\ProgramData\LockAp\LockAp.exe", "LockAp.apx", "K", "rt", "kl", and calls process injection routine 0x1400524e0.

Process Terminator (0x1400529f0)

Enumerates running processes with CreateToolhelp32Snapshot/Process32FirstW/Process32NextW, creates multiple mutexes with decrypted names for single-instance enforcement, compares process names using lstrcmpiW against target list, and calls TerminateProcess on matches.

Path Builder (0x140044f60)

Calls GetSystemDirectoryW to retrieve system path, decrypts wide-character strings using 23-word XOR key at 0x1400c5430, decrypted output includes path components for "Software\Microsoft\Windows\CurrentVersion\Run", uses wsprintfW to format final paths returned to caller.

File Cleanup Handler (0x14004ad40)

Initializes SHFILEOPSTRUCTW with wFunc=FO_DELETE (0x03), sets fFlags to FOF_NO_UI (0x0614) for silent operation, calls SHFileOperationW for file deletion, used to remove dropped payloads during uninstall.

Command and Control Server (0x14004ec70)

Decrypts pipe name using XOR key at 0x1400c5458, formats as "\\.\pipe\<[a-z]{2}><[0-9]{2}><[a-z]{1}><[0-9]{6,}>", creates named pipe with CreateNamedPipeA using PIPE_ACCESS_DUPLEX | FILE_FLAG_OVERLAPPED, sets PIPE_TYPE_MESSAGE mode, enters loop with ConnectNamedPipe/WaitForMultipleObjects, reads

commands with ReadFile, dispatches to handlers, and cleans up with DisconnectNamedPipe.

Process Injection Routine (0x1400524e0)

Takes process ID and flag parameters, calls OpenProcess with PROCESS_ALL_ACCESS (0x001F0FFF), opens target token with OpenProcessToken, duplicates token with DuplicateTokenEx, spawns process using CreateProcessAsUserW with duplicated token in target session, configures STARTUPINFOW with hidden window (wShowWindow=0), and closes handles.

Persistence Installer (0x14004cc40)

Decrypts and acquires "SeDebugPrivilege" using 49-word XOR key at 0x1400c5470, decrypts service name "LockAp", creates Windows service with CreateServiceW at "C:\ProgramData\LockAp\LockAp.exe" with SERVICE_AUTO_START type, creates registry key "HKLM\System\CurrentControlSet\Services\LockAp" with "Index" DWORD value (0x00000000), decrypts and creates "HKCU\Software\Microsoft\Windows\CurrentVersion\Run" entry, decrypts and executes schtasks /create /tn LockAp /tr "C:\ProgramData\LockAp\LockAp.exe" /ru "SYSTEM" /sc onstart /F via CreateProcessW to create scheduled task that runs at boot with SYSTEM privileges, creates registry path "SOFTWARE\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Tree\LockAp", drops files (LockAp.exe, LockAp.apx, K, rt, kl) to *C:\ProgramData\LockAp* directory, and sets FILE_ATTRIBUTE_HIDDEN | FILE_ATTRIBUTE_SYSTEM (0x06) on dropped files.

1. Defining Technical Traits

The malware's signature traits include signed binary side-loading, a fake minimal export table, a non-standard API hashing method, a user-mode trampoline inside ntdll, host-specific payload naming, an **embedded RC4 key**

("drturyTTteu7345Q"),

```
                        s__1800145b0                                    XREF[1,1]:   FUN_180001220:180001372(R),
                        s_drturyTTteu7345Q_1800145a0                                 FUN_180001220:180001379(R)
1800145a0 64 72 74        ds          "drturyTTteu7345Q"
          75 72 79
          54 54 74 ...
```

Hardcoded RC4 key from dbghelp.dll

and execution of the second stage from RWX memory without disk artifacts. Its C2 design blends standard crypto with custom TLS and multi-transport support, reflecting a modern and adaptable architecture.

3 Conclusions

In today's world companies are strongly reliant on automated detections of threats, antivirus software and EDR platforms. They come to trust them with their day-to-day security and there is implicit trust in any applications released by providers of such solutions. This trust is so strong that it even impacts the logic behind mechanisms used to differentiate between which applications are safe to use, and which are a security risk, as applications signed by one cybersecurity vendor are most of the time whitelisted across the other vendors' solutions. This is exactly what attackers attempted to exploit when they took an older binary, released by Trend Micro, and with the use of malicious DLL and additional payload placed on the host tried to turn it into Remote Access Tool.

Although DLL sideloading is not a new technique it is still favored by many attackers as it proves to still be effective with bypassing detection and allowing for execution of malicious programs. In this particular case attackers used quite unique combination of legitimate cybersecurity vendor binary, malicious DLL with additional payload hidden inside encrypted ".apx" file. In such a scenario, signature-based detection is ineffective because the initial executable is trusted and signed, the DLL can closely mimic legitimate components, and the final payload is encrypted, leaving little to no static indicators to reliably match against. This shows that effective defense requires a strong focus on behavioral and post-exploitation detections that can identify abnormal execution chains and misuse of trusted binaries rather than relying on static signatures alone.

4 Appendixes

4.1 Indicators of compromise

File / Path Indicators

Indicator Type	Value	Notes
File Path	C:\ProgramData\LockAp\LockAp.exe	Renamed signed binary used as execution host
File Path	C:\ProgramData\LockAp\dbghelp.dll	Unsigned malicious DLL (~93 KB), minimal exports, single stub
File Path	C:\ProgramData\LockAp\LockAp.apx	RC4-encrypted payload container

Registry Keys & Scheduled Tasks & Services

Indicator Type	Registry / Task Path	Description
Registry Run Key	HKCU\Software\Microsoft\Windows\CurrentVersion\Run → C:\ProgramData\LockAp\LockAp.exe	User-level persistence
Scheduled Task	HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Tree\LockAp	SYSTEM-level scheduled task
Service	HKLM\SYSTEM\CurrentControlSet\Services\LockAp	Attempted service persistence

Strings & Cryptographic Artifacts

Type	Value	Context
RC4 Key	drturyTTteu7345Q	Embedded in malicious dbghelp.dll

Network Indicators

Indicator Type	Value	Notes
IP Address	64.190.113[.]170:443	Observed C2 endpoint
IPC	Named Pipes following name pattern: \ \\.\pipe\<[a-z]{2}><[0-9]{2}><[a-z]{1}><[0-9]{6,}>	Local C2 / inter-process communication

Cryptographic Hashes

SHA-256	File Name	Description
a89702559e4c29198bfd265b415055996c56f9e8f992671456c958d931c2a668	Binary extracted from memory	Decrypted RAT payload
1f06a60e216b64a7574fb7774f4d5503b61f12fe241b71c747d2c2c417af6991	dbghelp.dll	Malicious sideloaded DLL
e3ee61d06b6febb735f5f819ca578bbd824e784d5ef225a92360e7cf37dab797	LockAp.apx	Encrypted payload container
67d1dc50383e4e0936f4c226ddac08ebb0ebe80df346dbd8d9c5228fc19cc5b9	LockAp.exe (Build.exe)	Legitimate Trend Micro binary used for DLL-

		sideloading (not malicious)
a91b42e5062fef608f285002debaff9358162b25	weave.exe	VLC binary (not malicious)
9f80418170ebb067fd931d756698ba1f1b9bdc3693f4790c55ddb90097db49df	k.exe	Kerbrute tool
35e0b542bed4170fcb5404eca84719554a7c21697c2bdbc74b09c155c8747a44	q.exe/if.exe	Intranet comprehensive scanning tool (original name: qscan.exe)
e2ca3ec168ae9c0b4115cd4fe220145ea9b2dc4b6fc79d765e91f415b34d00de	7z.exe	Legitimate 7zip executable (not malicious)

SHA-1	File Name	Description
adb09168c9b8dd25581efcd9f85948f31378de02	gggod.exe	Unknown tool used during discovery
afc7e0759080e4f285f1c78dfc5d8a7e09e3d20a	good.exe	Unknown tool used during discovery
fdb1f597b0adc16b73f42172b5261d02dbd84f31	fscan.exe	Intranet comprehensive scanning tool

1db5b2ad86bdd71f26d35c5301dbad0089de6e93	system_info.exe	Exploit validator fro CVE-2025-62215
02af222ee40bd764e070f4288b9654978d090705	ldapbrute.exe	Tool for LDAP operations (not verified)
c4dc9ebac62529bf58f898b5ab981ab30ed84d61	god.exe	Unknown tool used during discovery

4.2 Detection

As usual to detect this type of attack we encourage defenders to properly focus on behavioral aspects of post-exploitation activities. Good detection coverage for well-known TTPs that often overlaps in arsenal of Threat Actors, and their campaigns gives the highest chances to reveal malicious activities in the environment, even if malware used during breach is highly sophisticated and avoid signature-based detections.
Based on TTPs observed in this incident we can highlight following patterns:

- lateral movement and remote execution with built-in Windows mechanisms; remote service creation, remote scheduled task creation, usage of administrative network shares for malware transfer

- persistence mechanisms: creation of registry RUN keys, scheduled task and services with suspicious arguments

- not-expected host or domain reconnaissance

- internal network scans for well-known lateral movement ports

- loading DLLs from non-standard or user-writable paths

- executing trusted or signed binaries from uncommon locations (ProgramData, Temp, staging directories)

Detection of LockAp malware should focus on the four core behaviors observed in this case:\ **(1) Creation of LockAp directories, (2) DLL loading of dbghelp.dll from outside of system location, (3) presence of persistence mechanisms and (4) the use of legitimate but renamed TrendMicro's binary. To be more specific:**

- Presence of the C:\ProgramData\LockAp\ directory, especially the combination of LockAp.exe, dbghelp.dll, and LockAp.apx; this directory should not exist on legitimate systems and is a strong on-disk indicator.
- DLL loading of dbghelp.dll from a non-system path, particularly when loaded by a signed or trusted executable from non-standard location instead of System32/SysWOW64.
- Creation of persistence mechanisms pointing to C:\ProgramData\LockAp\LockAp.exe, including a Run key under HKCU\Software\Microsoft\Windows\CurrentVersion\Run, a scheduled task named LockAp, or a Windows service creation with the same name.
- TrendMicro is unlikely to rename their binary after it has been compiled. That alone is suspicious, and when paired with file being placed outside of typical TrendMicro's directories on the host is a strong indicator of misuse.

Sigma Rules

```
title: dbghelp.dll Loaded from Non-Default Directory
id: b4b355d3-f182-4160-918f-a910425566ff
status: experimental
description: Detects loading of dbghelp.dll from a non-default path, which may indicate DLL sideloading or search order hijacking.
author: TRC
date: 2026-03-02
tags:
 - attack.defense-evasion
```

```
  - attack.t1574.001
logsource:
  product: windows
  category: image_load
detection:
  selection:
    ImageLoaded|endswith: '\dbghelp.dll'
  filter_system32:
    ImageLoaded|startswith: 'C:\Windows\System32\'
  filter_syswow64:
    ImageLoaded|startswith: 'C:\Windows\SysWOW64\'
  condition: selection and not 1 of filter_*
falsepositives:
  - Developer/debugger tooling bundling dbghelp.dll alongside an application
  - Third-party applications shipping a private dbghelp.dll copy
level: high
```

```
title: Scheduled Task Created or Updated - LockAp
id: 5fdc3338-053b-4473-b8f6-cc9bbba05597
status: experimental
description: Detects creation/registration/update of a scheduled task named
LockAp (SYSTEM-level persistence observed).
author: TRC
date: 2026-03-02
tags:
  - attack.persistence
  - attack.t1053.005
logsource:
  product: windows
  service: taskscheduler
detection:
  selection_task_name:
```

```
  TaskName|contains: '\LockAp'
 selection_event_ids:
  EventID:
   - 106
   - 140
   - 141
 condition: selection_task_name and selection_event_ids
falsepositives:
 - Rare legitimate task using the same name (validate task XML, author, and action path)
level: high
```

```
title: APX File Created in ProgramData Subdirectory
id: 7d9201a2-72f7-4772-9f62-a8ddbaf3856b
status: experimental
description: Detects creation of files with .apx extension in any subdirectory of C:\ProgramData\, which is uncommon and may indicate malware staging or encrypted payload containers.
author: TRC
date: 2026-03-02
tags:
 - attack.defense-evasion
 - attack.t1027
 - attack.command-and-control
logsource:
 product: windows
 category: file_event
detection:
 selection_path:
  TargetFilename|startswith: 'C:\ProgramData\'
 selection_ext:
  TargetFilename|endswith: '.apx'
```

```
  condition: selection_path and selection_ext
falsepositives:
  - Legitimate applications storing files with .apx extension under ProgramData (rare, environment dependent)
  - Vendor-specific cache/packaging formats using .apx (uncommon)
level: medium
```

```
title: Trend Micro Build.exe Executed Under Different Filename (OriginalFileName Mismatch)
id: 814aa4cd-79f3-4a66-8518-cccb783b675e
status: experimental
description:
  Detects execution of a binary whose PE OriginalFileName metadata indicates Build.exe,  but which is executed under a different on-disk filename. This can indicate masquerading or abuse of a legitimate Trend Micro executable as an execution host
author: TRC
date: 2026-03-02
tags:
  - attack.defense-evasion
  - attack.t1036.003
logsource:
  product: windows
  category: process_creation
detection:
  selection_originalfilename:
    OriginalFileName: 'Build.exe'
  filter_expected_imagename:
    Image|endswith: '\Build.exe'
  condition: selection_originalfilename and not filter_expected_imagename
falsepositives:
  - Legitimate enterprise workflows that intentionally rename the binary
```

```
  - Software packaging or deployment systems that rename executables
  - Vendor updates or internal tooling that re-wraps or rebrands executables
level: high
```

```
title: Run Key Persistence Created or Updated - LockAp (ProgramData Path)
id: 794ee2ba-29c9-4f24-ae2a-ef3582890088
status: experimental
description: Detects creation or modification of a HKCU Run key value that
points to   C:\ProgramData\LockAp\LockAp.exe (user-level persistence).
author: TRC
date: 2026-03-02
tags:
  - attack.persistence
  - attack.t1547.001
logsource:
  product: windows
  category: registry_set
detection:
  selection_key:
    TargetObject|contains: '\Software\Microsoft\Windows\CurrentVersion\Run\'
  selection_value_path:
    Details|contains: 'C:\ProgramData\LockAp\LockAp.exe'
  selection_hive:
    TargetObject|startswith: 'HKCU\'
  condition: selection_hive and selection_key and selection_value_path
falsepositives:
  - Unlikely; validate the value name, parent process, and signer of the
referenced binary
  - Legitimate software incorrectly installed to this exact path (rare)
level: high
```

```
title: Windows Service Created - LockAp (ProgramData Path)
```

```
id: c154ee30-4d75-4a6f-b080-afa4ab8e674b
status: experimental
description: Detects creation of a Windows service named LockAp where the service binary path points to C:\ProgramData\LockAp\LockAp.exe (SYSTEM-level persistence).
author: TRC
date: 2026-03-02
tags:
 - attack.persistence
 - attack.t1543.003
logsource:
 product: windows
 service: security
detection:
 selection_eventid:
  EventID: 4697
 selection_service_name:
  ServiceName: 'LockAp'
 selection_service_path:
  ServiceFileName|contains: 'C:\ProgramData\LockAp\LockAp.exe'
 condition: selection_eventid and selection_service_name and selection_service_path
falsepositives:
 - Rare; validate service command line, signer, and creation source (installer, admin tooling)
level: high
```

4.3 MITRE ATT&CK Mapping (key techniques)

Category	Technique	MITRE ID
Initial Access	Exploit Public-Facing Application	T1190

Execution/ Defense Evasion	DLL Search Order Hijacking	T1574.001
Execution/ Defense Evasion	DLL Side-Loading	T1574.002
Execution	Native API	T1106
Persistence	Registry Run Keys	T1547.001
Persistence	Scheduled Task	T1053.005
Persistence/ Privilege Escalation	Create or Modify System Process: Windows Service	T1543.003
Privilege Escalation	Access Token Manipulation	T1134
Defense Evasion	Obfuscated Files or Information	T1027
Defense Evasion	Deobfuscate/Decode Files or Information	T1140
Defense Evasion	Masquerading	T1036
Discovery	System Information Discovery	T1082
Discovery	Process Discovery	T1057
Discovery	Query Registry	T1012
Discovery	System Service Discovery	T1007
Command and Control	Application Layer Protocol: Web Protocols	T1071.001
Command and Control	Encrypted Channels	T1573

About Atos Group

Atos Group is a global leader in digital transformation with c. 56,000 employees and annual revenue of c. €7.2 billion (at the go-forward perimeter), operating in 54 countries under two brands – Atos for services and Eviden for products and systems. European number one in cybersecurity and a leader in cloud, Atos Group is committed to a secure and decarbonized future and provides tailored AI-powered, end-to-end solutions for all industries. Atos Group is the brand under which Atos SE (Societas Europaea) operates. Atos SE listed on Euronext Paris.

The purpose of Atos Group is to help design the future of the information space. Its expertise and services support the development of knowledge, education and research in a multicultural approach and contribute to the development of scientific and technological excellence. Across the world, the Group enables its customers and employees, and members of societies at large to live, work and develop sustainably, in a safe and secure information space.

Find out more about us

atos.net
atos.net/career