INTELLIGENCE BRIEFING
Security Command Center
TLP:CLEAR
2026-06-20 13:35 UTC

Microsoft ClickOnce Weaponized as Malware Delivery Channel, First In-Depth Abuse Analysis Published

SECURITY ANALYSIS | HIGH | CVSS 7.5

SCC Item ID	SCC-STY-2026-0229
Type	Security Analysis
Severity	HIGH
CVSS Base Score	7.5
Affected Products	Microsoft ClickOnce (Windows), Visual Studio, Windows Endpoints
Discovery Source	Rss:T1 Threatintel

Executive Summary

CrowdStrike has published the first documented in-depth technical analysis of Microsoft ClickOnce as a malware delivery mechanism, revealing that threat actors can exploit the framework's native design, no admin privileges required, minimal user friction, auto-update functionality, to deploy malicious payloads on Windows endpoints while evading standard endpoint defenses. The research documents both previously known and newly disclosed abuse vectors, catalogued under CWE-494 and CWE-345, with an associated CVSS score of 7.5 (High). This disclosure signals a broader pattern of adversaries repurposing trusted, built-in Windows deployment infrastructure to bypass security controls, a strategy that compounds detection difficulty and increases dwell time.

Technical Analysis

ClickOnce is a Microsoft deployment technology built into the Windows ecosystem, used by developers to distribute and auto-update applications without requiring administrative privileges. Its design priorities, frictionless installation, user-space execution, and transparent updates, are precisely the characteristics threat actors exploit to deliver malware with low user interaction and reduced security tooling visibility.

CrowdStrike's Part 1 analysis provides the first comprehensive technical breakdown of ClickOnce internals from an adversarial perspective. The research maps two core weakness classes: CWE-494 (Download of Code Without Integrity Check) and CWE-345 (Insufficient Verification of Data Authenticity). Together, these weaknesses mean that a weaponized ClickOnce manifest can deliver and execute a malicious payload without triggering the same scrutiny applied to traditional executable downloads.

The MITRE ATT&CK techniques in play span the delivery-through-execution chain. T1204.002 (Malicious File) captures the user interaction component, a target clicks a ClickOnce link or manifest, often delivered via phishing or a compromised site. T1036 (Masquerading) and T1553.002 (Code Signing) reflect how attackers dress malicious ClickOnce applications in the appearance of legitimacy, including the possibility of signing packages with obtained or misused certificates. T1218 (System Binary Proxy Execution) is relevant because ClickOnce executes through trusted Windows components, such as dfsvc.exe, which may be allowlisted by default policy. T1072 (Software Deployment Tools) maps to the abuse of the deployment mechanism itself. T1105 (Ingress Tool Transfer) covers the payload retrieval step embedded in the ClickOnce update mechanism.

The defensive gap this research highlights is structural. ClickOnce applications run in user space, do not require UAC elevation, and leverage trusted Microsoft infrastructure and processes. Enterprise security stacks that rely on privilege escalation events or unsigned-binary detections as primary tripwires will not fire. Similarly, organizations that allowlist Microsoft deployment binaries, a common configuration practice, may inadvertently create a permissive lane for ClickOnce-delivered payloads.

Part 2 of the CrowdStrike series is expected to address detection strategies and platform-specific protections, which means the defensive picture is currently incomplete. Security teams should treat this as an active research gap, not a resolved finding. The absence of a CVE ID and KEV listing reflects that this is an architectural abuse pattern rather than a patchable discrete vulnerability, meaning vendor remediation is not the primary mitigation path.

Action Checklist

1. Step 1: Assess exposure, audit whether ClickOnce-delivered applications are in use across your Windows endpoint fleet; query asset inventory for applications installed via dfsvc.exe or with .application and .appref-ms file associations (CIS 1.1: Establish and Maintain Detailed Enterprise Asset Inventory)
2. Step 2: Review application allowlisting policy, determine whether dfsvc.exe and other ClickOnce execution components are unconditionally allowlisted in your EDR or application control policy; evaluate whether execution can be scoped to known-good manifest sources (NIST CM-7: Least Functionality)
3. Step 3: Audit code signing and manifest trust, verify that your endpoint policy enforces signature validation on ClickOnce manifests; review whether CWE-494 and CWE-345 exposure exists in your current deployment trust model (NIST CM-14: Signed Components)
4. Step 4: Update threat model, incorporate T1218 (System Binary Proxy Execution via ClickOnce/dfsvc.exe) and T1553.002 (Code Signing abuse) into your threat register and detection engineering backlog
5. Step 5: Review logging coverage, confirm that process creation events for dfsvc.exe and child processes, as well as network connections initiated from ClickOnce application directories, are captured and forwarded to your SIEM (NIST AU-2: Event Logging; CIS 8.2: Collect Audit Logs)
6. Step 6: Monitor for Part 2, track the CrowdStrike blog for the follow-up publication covering detection strategies and Falcon-specific protections; assign a team member to incorporate those findings into detection rules when published

IR / Forensic Enrichment

Triage Priority	URGENT

Escalation Criteria	Escalate to incident commander and legal/compliance immediately if Sysmon or SIEM logs confirm dfsvc.exe spawning interactive shells or making outbound connections from `%APPDATA%\Local\Apps\2.0\` directories on endpoints handling PII, PHI, or financial data, as this would trigger breach notification obligations and indicates active exploitation rather than theoretical exposure.
Recovery Notes	After containing any confirmed ClickOnce-delivered malware payload, verify integrity of the ClickOnce application cache by hashing all files under `%USERPROFILE%\AppData\Local\Apps\2.0\` on affected hosts and comparing against known-good hashes from your software inventory before restoring normal operations. Monitor dfsvc.exe execution and child process telemetry for a minimum of 30 days post-remediation, as ClickOnce's native auto-update mechanism could re-fetch a malicious payload from a still-trusted manifest source if the deployment URL and certificate trust are not explicitly revoked. Confirm that registry keys controlling ClickOnce trust behavior under `HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Deployment` are hardened to their most restrictive values before returning affected hosts to production.
Forensic Artifacts	ClickOnce application cache directory: `%USERPROFILE%\AppData\Local\Apps\2.0\` — contains staged payload binaries, DLLs, and deployment manifests (.application files) fetched by dfsvc.exe; hash all contents and inspect manifests for non-enterprise publisher certificates or suspicious deployment URLs (CWE-494/CWE-345 evidence) | Sysmon Event ID 1 (Process Create) logs filtering on `dfsvc.exe` as ParentImage — reveals the full child process tree spawned by ClickOnce execution, identifying any post-exploitation activity such as cmd.exe, powershell.exe, or LOLBin invocations launched by the malicious payload | Windows Security Event Log Event ID 4688 (Process Creation with full command line auditing enabled) — captures dfsvc.exe invocation context including the deployment URL passed as a command-line argument, which identifies the malicious manifest source | ClickOnce deployment registry keys at `HKCU\Software\Microsoft\Windows\CurrentVersion\Deployment` and `HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Deployment` — records trusted deployment sources, auto-update configuration, and trust prompt suppression settings that reveal whether the attack relied on a pre-trusted publisher or exploited a permissive trust policy | Sysmon Event ID 3 (Network Connection) for processes executing from `%APPDATA%\Local\Apps\2.0\` paths — captures C2 or payload-staging outbound connections initiated by the ClickOnce-delivered malware after execution, distinct from the initial dfsvc.exe manifest fetch

Per-Action IR Details

Step 1: Assess exposure — audit whether ClickOnce-delivered applications are in use across your Windows endpoint fleet; query asset inventory for applications installed via dfsvc.exe or with .application and .appref-ms file associations (CIS 1.1: Establish and Maintain Detailed Enterprise Asset Inventory)

NIST Phase: Preparation

Reference: NIST 800-61r3 §2 — Preparation: Establishing IR Capability and Asset Visibility

Controls: CIS 1.1 (Establish and Maintain Detailed Enterprise Asset Inventory), CIS 2.1 (Establish and Maintain a Software Inventory), NIST CM-8 (System Component Inventory)

Compensating: Run the following on each Windows endpoint via PowerShell to enumerate ClickOnce-installed apps and their deployment sources: `Get-ChildItem 'C:\Users\*\AppData\Local\Apps\2.0\' -Recurse -Include '*.application','*.appref-ms' | Select-Object FullName,LastWriteTime`. Cross-reference dfsvc.exe execution history using Sysmon Event ID 1 (Process Create) with `ParentImage` or `Image` matching `C:\Windows\System32\dfsvc.exe`. Use osquery: `SELECT name, path, install_date FROM programs WHERE path LIKE '%AppData%Local%Apps%2.0%';`

Evidence: This step is an inventory action and does not alter live state. Before querying endpoints, note that ClickOnce stores deployment manifests and cached payloads in `%USERPROFILE%\AppData\Local\Apps\2.0\` — capture directory listings (including hidden files) and file hashes of all `.application`, `.appref-ms`, and associated `.exe` or `.dll` payloads found there. Also preserve the ClickOnce application cache index at `%USERPROFILE%\AppData\Local\Deployment\` prior to any remediation action.

Step 2: Review application allowlisting policy — determine whether dfsvc.exe and other ClickOnce execution components are unconditionally allowlisted in your EDR or application control policy; evaluate whether execution can be scoped to known-good manifest sources (NIST CM-7: Least Functionality)

NIST Phase: Preparation

Reference: NIST 800-61r3 §2 — Preparation: Hardening and Reducing Attack Surface Prior to Incidents

Controls: NIST CM-7 (Least Functionality), CIS 4.6 (Securely Manage Enterprise Assets and Software)

Compensating: If no EDR is available, use Windows Software Restriction Policies (SRP) or AppLocker to restrict dfsvc.exe execution to administrator accounts only: in AppLocker, create a Publisher rule for `%WINDIR%\System32\dfsvc.exe` scoped exclusively to a named admin group, blocking standard user execution. Document the policy change in a change ticket. Validate with: `Get-AppLockerPolicy -Effective | Test-AppLockerPolicy -Path C:\Windows\System32\dfsvc.exe -User domain\standarduser`.

Evidence: This step modifies endpoint policy and does not directly alter process or memory state. However, before restricting dfsvc.exe, capture a snapshot of current AppLocker/SRP effective policy (`Get-AppLockerPolicy -Effective -Xml > applocker_baseline.xml`) and enumerate all currently running or recently executed ClickOnce applications (`Get-WinEvent -LogName 'Microsoft-Windows-AppLocker/EXE and DLL' | Where-Object {$_.Message -match 'dfsvc'}`) to avoid breaking legitimate workflows during the policy change.

Step 3: Audit code signing and manifest trust — verify that your endpoint policy enforces signature validation on ClickOnce manifests; review whether CWE-494 and CWE-345 exposure exists in your current deployment trust model (NIST CM-14: Signed Components)

NIST Phase: Preparation

Reference: NIST 800-61r3 §2 — Preparation: Establishing Trust Controls and Policy Baselines

Controls: NIST CM-14 (Signed Components), NIST CM-6 (Configuration Settings), CIS 2.2 (Ensure Authorized Software is Currently Supported)

Compensating: Use `sigcheck.exe` (Sysinternals, free) to batch-validate signatures on all `.application` manifest files found in `%USERPROFILE%\AppData\Local\Apps\2.0\`: `sigcheck.exe -r -s C:\Users\\AppData\Local\Apps\2.0\ > sigcheck_results.txt`. Flag any manifest signed by an untrusted, expired, or self-signed certificate — these represent direct CWE-345 (Insufficient Verification of Data Authenticity) exposure. Cross-check publisher thumbprints against your approved certificate inventory.

Evidence: This is a policy audit step and does not alter live state. Prior to making any trust policy changes, export the current Trusted Publishers and Trusted Root Certification Authorities certificate stores: `certutil -store TrustedPublisher > trusted_publishers_baseline.txt`. Also capture the ClickOnce trust prompt behavior settings from the registry at `HKCU\Software\Microsoft\Windows\CurrentVersion\WinTrust\Trust Providers\Software Publishing` and `HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Deployment` — these keys control whether unsigned or low-trust manifests are silently permitted, which is the root of CWE-494 exposure in this threat.

Step 4: Update threat model — incorporate T1218 (System Binary Proxy Execution via ClickOnce/dfsvc.exe) and T1553.002 (Code Signing abuse) into your threat register and detection engineering backlog

NIST Phase: Preparation

Reference: NIST 800-61r3 §2 — Preparation: Threat Modeling and Detection Engineering Readiness

Compensating: Document the threat model update in a shared wiki or ticket. Write a Sigma rule targeting dfsvc.exe spawning unexpected child processes (e.g., cmd.exe, powershell.exe, wscript.exe, mshta.exe) and save to your detection backlog: `detection: Image|endswith: '\dfsvc.exe' AND CommandLine|contains: any of ('cmd','powershell','wscript','mshta')`. Reference the CrowdStrike ClickOnce abuse analysis as the source authority for the threat model entry. Flag for re-evaluation when Part 2 of the CrowdStrike research publishes.

Evidence: This is a planning and documentation step that does not alter live state and requires no pre-capture of volatile evidence. Note that MITRE ATT&CK technique IDs T1218 and T1553.002 are referenced here as threat actor behavior descriptors for detection scoping — they are not defensive controls and are correctly excluded from the controls field.

Step 5: Review logging coverage — confirm that process creation events for dfsvc.exe and child processes, as well as network connections initiated from ClickOnce application directories, are captured and forwarded to your SIEM (NIST AU-2: Event Logging; CIS 8.2: Collect Audit Logs)

NIST Phase: Detection Analysis

Reference: NIST 800-61r3 §3.2 — Detection and Analysis: Monitoring and Log Collection for Adverse Event Identification

Controls: NIST AU-2 (Event Logging), NIST AU-3 (Content Of Audit Records), NIST AU-6 (Audit Record Review, Analysis, And Reporting), NIST AU-12 (Audit Record Generation), CIS 8.2 (Collect Audit Logs)

Compensating: Deploy Sysmon with a configuration that captures Event ID 1 (Process Create) filtered on `dfsvc.exe` and its children, Event ID 3 (Network Connection) for processes executing from `%APPDATA%\Local\Apps\2.0\`, and Event ID 7 (Image Load) for DLLs loaded by ClickOnce-spawned processes. Use the SwiftOnSecurity Sysmon config as a baseline and add: `dfsvc.exe`. Forward Sysmon logs to a central syslog server (e.g., Graylog CE or Elastic free tier) if no SIEM is available.

Evidence: This step establishes monitoring and does not alter live state on endpoints. Before modifying Sysmon or logging configuration, export the current Sysmon effective configuration (`sysmon.exe -c > current_sysmon_config.xml`) and verify existing Windows Security Event Log retention settings (`wevtutil gl Security`). For ClickOnce-specific forensic baseline: confirm whether Windows Security Event ID 4688 (Process Creation) with audit process tracking enabled is already capturing dfsvc.exe invocations — if not, this is a critical logging gap for detecting this threat's execution chain.

Step 6: Monitor for Part 2 — track the CrowdStrike blog for the follow-up publication covering detection strategies and Falcon-specific protections; assign a team member to implement those findings into detection rules when published

NIST Phase: Post Incident

Reference: NIST 800-61r3 §4 — Post-Incident Activity: Lessons Learned, Intelligence Integration, and Detection Improvement

Controls: NIST IR-4 (Incident Handling), NIST IR-8 (Incident Response Plan)

Compensating: Set a browser alert or RSS feed monitor on the CrowdStrike adversary intelligence blog for terms 'ClickOnce' and 'dfsvc'. Assign a named owner in your team's issue tracker with a due date 30 days out to review and translate any published Falcon-specific IOCs or YARA/Sigma rules into tool-agnostic equivalents compatible with your stack (Sysmon + free SIEM or manual log review). Document the assigned owner, expected publication window, and integration acceptance criteria in your threat register entry created in Step 4.

Evidence: This is a planning and intelligence-tracking step that does not alter live state. No volatile evidence capture is required. When Part 2 publishes, treat newly disclosed IOCs (e.g., specific ClickOnce manifest hashes, C2 URLs, payload staging paths under `%APPDATA%\Local\Apps\2.0\`) as retrospective hunt queries against logs already captured per Step 5 — this is why logging coverage established in Step 5 must precede this intelligence integration step.

Detection Guidance

The primary behavioral signal to hunt is dfsvc.exe, the Windows ClickOnce service host, spawning unexpected child processes or initiating outbound network connections to non-Microsoft, non-enterprise-managed domains. In environments where ClickOnce is not used at all, any execution of dfsvc.exe warrants immediate investigation.

For environments where ClickOnce is legitimately deployed: baseline the known manifest sources and application update URLs, then alert on deviations. Unexpected .application or .appref-ms file execution originating from user download directories, browser cache paths, or email attachment staging locations (e.g., %LOCALAPPDATA%\Temp) is a high-fidelity behavioral indicator.

Log sources to check: Windows Security event log (process creation, Event ID 4688 with command line auditing enabled), Sysmon Event ID 1 (process create) for dfsvc.exe parent-child chains, and network proxy or DNS logs for domains contacted during ClickOnce manifest resolution and payload retrieval.

Hunting hypothesis: search for dfsvc.exe child process trees that include scripting hosts (wscript.exe, cscript.exe, powershell.exe, mshta.exe) or common post-exploitation binaries. Also hunt for .appref-ms files dropped in user-writable paths outside of standard ClickOnce application deployment directories.

Applicable D3FEND countermeasures: D3-SFA (System File Analysis, monitor ClickOnce manifest and configuration files for tampering or unexpected modification), D3-UAP (User Account Permissions, restrict which users or roles may execute ClickOnce-deployed applications where not operationally required).

Relevant NIST controls: AU-2 (Event Logging, ensure dfsvc.exe process events are in scope), AU-12 (Audit Record Generation), CM-7 (Least Functionality, restrict ClickOnce execution to necessary endpoints).

Note: CrowdStrike has indicated that Part 2 of this research will include specific detection strategies and Falcon platform signatures. Until that publication, treat current detection coverage as potentially incomplete and prioritize behavioral hunting over signature-based approaches.

Indicators of Compromise

Type	Value	Context	Confidence
TOOL	`dfsvc.exe`	dfsvc.exe (Windows ClickOnce service host) leveraged via malicious .application manifest or .appref-ms file to execute untrusted payloads in user space without administrative privileges	HIGH
TOOL	`Pending — refer to CrowdStrike blog Part 1 and forthcoming Part 2 for published indicators`	CrowdStrike's analysis documents specific weaponized ClickOnce manifest structures and payload delivery mechanisms; concrete hashes, domains, and URLs associated with observed malicious ClickOnce deployments are expected to be published in Part 2 of the series	LOW

Framework Mappings

MITRE-ATTACK

- **T1204.002** — Malicious File
- **T1036** — Masquerading
- **T1072** — Software Deployment Tools
- **T1553.002** — Code Signing

- **T1218** — System Binary Proxy Execution
- **T1105** — Ingress Tool Transfer

NIST-800-53R5

- **CA-7** — Continuous Monitoring
- **SC-7** — Boundary Protection
- **SI-3** — Malicious Code Protection
- **SI-4** — System Monitoring
- **SI-7** — Software, Firmware, and Information Integrity
- **CM-3** — Configuration Change Control

OWASP-TOP10-2021

- **A08:2021** — Software and Data Integrity Failures

CIS-V8

- **2.5** — Allowlist Authorized Software
- **2.6** — Allowlist Authorized Libraries
- **8.2** — Collect Audit Logs

NIST-CSF-2

- **DE.CM-01** — Networks and network services are monitored

MITRE ATT&CK Mapping

Technique ID	Technique Name	Tactic
T1204.002	Malicious File	Execution
T1036	Masquerading	Defense-Evasion
T1072	Software Deployment Tools	Execution
T1553.002	Code Signing	Defense-Evasion
T1218	System Binary Proxy Execution	Defense-Evasion
T1105	Ingress Tool Transfer	Command-And-Control

Sources

Source	URL	Tier
Blog	https://www.crowdstrike.com/en-us/blog/new-abuse-of-the-clickonce-t...	T3
	https://www.crowdstrike.com/en-us/blog/how-the-infrastructure-inves...	T3

Source	URL	Tier
	https://www.crowdstrike.com/en-us/blog/reasons-why-nonprofits-are-t...	T3
	https://www.crowdstrike.com/en-us/blog/crowdstrike-and-industry-par...	T3
New Abuse of the ClickOnce Technology: Part 1 - CrowdStrike	https://www.crowdstrike.com/content/crowdstrike-www/locale-sites/us...	T3

Generated 2026-06-20 13:35 UTC by TJS Security Command Center