INTELLIGENCE BRIEFING
Security Command Center
TLP:CLEAR
2026-06-27 13:43 UTC

Malicious Edge Extension 'Edgecution' Abuses Native Messaging Protocol for Malware Deployment

THREAT CAMPAIGN | **HIGH** | CVSS 7.5

SCC Item ID	SCC-CAM-2026-0591
Type	Threat Campaign
Severity	HIGH
CVSS Base Score	7.5
Affected Products	Microsoft Edge browser users (extension-based attack; no specific Edge version pinned in available data)
Published	2026-06-25
Discovery Source	Gemini

Executive Summary

A malicious Microsoft Edge browser extension named 'Edgecution' has been identified abusing Edge's Native Messaging API to deploy malware on victim systems, bypassing browser sandbox protections. Any organization or individual running Microsoft Edge where users can install extensions is potentially exposed, particularly if extension allow-listing controls are not enforced. The primary business risk is unauthorized code execution on enterprise endpoints, enabling data theft, persistence, or further lateral movement across corporate networks.

Technical Analysis

Edgecution is a malicious browser extension targeting Microsoft Edge that abuses the Native Messaging API (MITRE T1559: Inter-Process Communication, with T1176: Browser Extensions as the primary installation vector) to communicate with a locally installed native host process outside the browser sandbox. This technique effectively bridges the browser security boundary, enabling the extension to pass commands or payloads to the underlying OS. Related techniques include T1543 (Create or Modify System Process) for persistence via the native host, and T1059 (Command and Scripting Interpreter) for downstream execution. Relevant weaknesses are CWE-284 (Improper Access Control), CWE-693 (Protection Mechanism Failure), and CWE-494 (Download of Code Without Integrity Check). No CVE has been assigned. CVSS base score is reported at 7.5 (High); no vector string is available in the supplied data. The item does not appear on CISA KEV as of the configuration date. No specific Edge version is pinned in available data. **Confidence Note: This item is based on

secondary-source reporting. No confirmed IOCs, vendor advisory, or primary-source research have been verified. Technical specifics should be treated as LOW confidence pending official Microsoft Edge Security advisory. Available sources include Microsoft Edge Security Release Notes (T1: https://learn.microsoft.com/en-us/deployedge/microsoft-edge-relnotes-security) and Microsoft Edge Known Issues (T1: https://learn.microsoft.com/en-us/deployedge/microsoft-edge-known-issues); neither has been confirmed to contain specific Edgecution coverage. No patch ID, vendor advisory, or confirmed remediation path is available in the supplied data.**

Action Checklist

1. Step 1: Containment. Audit and enforce extension allow-lists via Microsoft Edge enterprise policy (ExtensionInstallAllowlist / ExtensionInstallBlocklist) to prevent unauthorized extensions from running. Restrict extension installation to approved sources only. Check Microsoft Edge Security Release Notes at https://learn.microsoft.com/en-us/deployedge/microsoft-edge-relnotes-security for any advisory specific to Edgecution or Native Messaging abuse. NIST CM-7 (Least Functionality): prohibit or restrict the use of functions, ports, protocols, and services not required.
2. Step 2: Detection. Query endpoint logs for Edge Native Messaging host process activity: look for unexpected child processes spawned by msedge.exe or edge.exe communicating with native host executables registered under HKCU\Software\Microsoft\Edge\NativeMessagingHosts or HKLM\SOFTWARE\Microsoft\Edge\NativeMessagingHosts. Review Windows Event Log (Security, Sysmon Event ID 1 process creation, Event ID 11 file creation) for native host binaries written to user-writable paths. Audit installed Edge extensions against your approved inventory per CIS 2.1 (Establish and Maintain a Software Inventory) and CIS 1.1 (Establish and Maintain Detailed Enterprise Asset Inventory). Apply D3-SFA (System File Analysis): monitor registry keys and file system paths associated with Native Messaging host registration for unauthorized modifications. Apply D3-LAM (Local Account Monitoring): flag lateral activity originating from browser process trees. No confirmed IOCs are available in the supplied data; behavioral patterns above are the primary detection mechanism.
3. Step 3: Eradication. Remove the Edgecution extension from all affected endpoints via Edge enterprise management or manual uninstallation. Deregister any associated Native Messaging host entries from the Windows registry (HKCU and HKLM NativeMessagingHosts paths). Remove any native host binaries dropped by the extension. No vendor-issued patch ID is available in the supplied data; monitor Microsoft Edge Security Release Notes for a formal advisory. Apply NIST CM-2 (Baseline Configuration): restore and validate against documented baseline for Edge configuration and approved extension list. Apply CIS 2.3 (Address Unauthorized Software): remove the unauthorized extension per your documented process.
4. Step 4: Recovery. Validate that Native Messaging host registry keys return to baseline. Confirm no persistence mechanisms (scheduled tasks, run keys, services) were established by the native host payload using NIST IR-4 (Incident Handling) procedures. For endpoints where Native Messaging host activity or unexpected child processes from msedge.exe are confirmed via detection rules above, consider re-imaging rather than relying solely on artifact removal. For endpoints without confirmed indicators, artifact removal and monitoring may be sufficient. Monitor Edge process trees for 30 days post-remediation using endpoint detection tooling. Apply AU-6 (Audit Record Review, Analysis, and Reporting): schedule a follow-on log review at a defined frequency to confirm no residual indicators.
5. Step 5: Post-Incident. Review and tighten browser extension governance: enforce ExtensionInstallForcelist to allow only vetted extensions, and block all others by default per NIST CM-7 (Least Functionality) and CIS 4.6 (Securely Manage Enterprise Assets and Software). Conduct a

lessons-learned session to assess whether Native Messaging host registration paths are monitored in your SIEM. Update detection rules and playbooks to cover browser-to-native-host abuse patterns (MITRE T1559, T1176). Apply D3-UAP (User Account Permissions): review whether standard users should have write access to Native Messaging host registry paths. Assess whether CIS 6.3 (Require MFA for Externally-Exposed Applications) and CIS 6.5 (Require MFA for Administrative Access) are enforced to reduce post-execution lateral movement risk.

IR / Forensic Enrichment

Triage Priority	URGENT
Escalation Criteria	Escalate immediately to senior IR leadership and legal/compliance if forensic evidence confirms the Edgecution native host payload executed and exfiltrated data (network connections from the native host process to external IPs, or file staging in user-writable paths), or if more than 10 endpoints show confirmed native host execution — triggering potential breach notification assessment under applicable data protection regulations.
Recovery Notes	After eradicating the Edgecution extension and its Native Messaging host artifacts, validate recovery by running the PowerShell NativeMessagingHosts enumeration script against all remediated endpoints and confirming zero non-baseline entries. Endpoints with confirmed native host binary execution must be re-imaged rather than cleaned in place, as the native host process operated outside the browser sandbox with full OS privileges and may have established persistence not yet identified. Maintain elevated Sysmon monitoring on msedge.exe process trees for a minimum of 30 days to detect reinfection or variant deployment targeting the same Native Messaging attack surface.
Forensic Artifacts	Edge extension directory at %LOCALAPPDATA%\Microsoft\Edge\User Data\Default\Extensions\[Edgecution_extension_id] — contains background.js or service_worker.js with Native Messaging API calls (chrome.runtime.connectNative or chrome.runtime.sendNativeMessage) that reveal the host name targeted and data exfiltrated through the sandbox escape | Native Messaging host JSON manifest file (path stored in HKLM\SOFTWARE\Microsoft\Edge\NativeMessagingHosts\[host_name] registry value) — contains allowed_origins field confirming the Edgecution extension ID and the 'path' field pointing to the dropped native host binary location | Native host binary dropped in user-writable path (typically %APPDATA%, %TEMP%, or %USERPROFILE%) — hash and submit for sandbox analysis; Sysmon Event ID 11 (FileCreate) with Image=msedge.exe confirms the browser wrote the binary, establishing the sandbox escape execution chain | Sysmon Event ID 1 (ProcessCreate) logs showing msedge.exe as ParentImage and the native host binary as Image, with full command-line arguments — this is the primary evidence of successful Native Messaging sandbox escape and the timestamp anchors the compromise timeline | Windows registry hive export of HKCU\Software\Microsoft\Edge\NativeMessagingHosts and HKLM\SOFTWARE\Microsoft\Edge\NativeMessagingHosts — documents the attacker-registered host name, and Sysmon Event ID 13 (RegistryValueSet) records when the key was written and by which process, distinguishing legitimate Edge behavior from Edgecution's registration

Per-Action IR Details

Step 1: Containment — Audit and enforce extension allow-lists via Microsoft Edge enterprise policy (ExtensionInstallAllowlist / ExtensionInstallBlocklist) to prevent unauthorized extensions from running. Block

extension sideloading by setting ExtensionInstallSources to approved sources only. Check Microsoft Edge Security Release Notes at https://learn.microsoft.com/en-us/deployedge/microsoft-edge-relnotes-security for any advisory specific to Edgecution or Native Messaging abuse. NIST CM-7 (Least Functionality): prohibit or restrict the use of functions, ports, protocols, and services not required.

NIST Phase: Containment

Reference: NIST 800-61r3 §3.3 — Containment Strategy

Controls: NIST CM-7 (Least Functionality), NIST CM-6 (Configuration Settings), CIS 2.3 (Address Unauthorized Software)

Compensating: For teams without MDM/Intune, deploy Edge group policy (GPO) locally: set ExtensionInstallBlocklist to '*' and ExtensionInstallAllowlist to your approved extension IDs via HKLM\SOFTWARE\Policies\Microsoft\Edge. Validate using 'Get-ItemProperty -Path HKLM:\SOFTWARE\Policies\Microsoft\Edge' in PowerShell. On non-domain systems, push registry settings via a signed PowerShell script deployed through task scheduler with SYSTEM privileges.

Evidence: Before enforcing policy changes that will terminate extension execution and alter browser state, capture: (1) the current contents of HKCU\Software\Microsoft\Edge\Extensions and HKLM\SOFTWARE\Microsoft\Edge\Extensions registry hives via 'reg export'; (2) the Edge extension directory at %LOCALAPPDATA%\Microsoft\Edge\User Data\Default\Extensions — specifically the Edgecution extension manifest.json and background script files; (3) any registered Native Messaging host entries under HKLM\SOFTWARE\Microsoft\Edge\NativeMessagingHosts and HKCU\Software\Microsoft\Edge\NativeMessagingHosts including their JSON manifest paths and referenced executable paths; (4) a live process list showing msedge.exe child processes via 'Get-Process | Where-Object {$_.Parent.Name -eq "msedge"}' before policy enforcement terminates active extension processes.

Step 2: Detection — Query endpoint logs for Edge Native Messaging host process activity: look for unexpected child processes spawned by msedge.exe or edge.exe communicating with native host executables registered under HKCU\Software\Google\Chrome\NativeMessagingHosts or HKLM\SOFTWARE\Microsoft\Edge\NativeMessagingHosts. Review Windows Event Log (Security, Sysmon Event ID 1 process creation, Event ID 11 file creation) for native host binaries written to user-writable paths. Audit installed Edge extensions against your approved inventory per CIS 2.1 (Establish and Maintain a Software Inventory) and CIS 1.1 (Establish and Maintain Detailed Enterprise Asset Inventory). Apply D3-SFA (System File Analysis): monitor registry keys and file system paths associated with Native Messaging host registration for unauthorized modifications. Apply D3-LAM (Local Account Monitoring): flag lateral activity originating from browser process trees. No confirmed IOCs are available in the supplied data; behavioral patterns above are the primary detection mechanism.

NIST Phase: Detection Analysis

Reference: NIST 800-61r3 §3.2 — Detection and Analysis

Controls: NIST AU-6 (Audit Record Review, Analysis, And Reporting), NIST AU-2 (Event Logging), CIS 1.1 (Establish and Maintain Detailed Enterprise Asset Inventory), CIS 2.1 (Establish and Maintain a Software Inventory), CIS 8.2 (Collect Audit Logs)

Compensating: Deploy Sysmon with a configuration that enables Rule Group 'ProcessCreate' filtering on ParentImage containing 'msedge.exe' and TargetImage NOT in your approved native host binary list. Use the following PowerShell one-liner to enumerate all registered Native Messaging hosts and their executable paths: 'Get-ChildItem -Path "HKLM:\SOFTWARE\Microsoft\Edge\NativeMessagingHosts","HKCU:\Software\Microsoft\Edge\NativeMessagingHosts" | ForEach-Object { Get-ItemPropertyValue $_.PSPath -Name "(default)" | Get-Content | ConvertFrom-Json }'. Cross-reference discovered executable paths against known-good hashes using Get-FileHash. Write a Sigma rule targeting Sysmon Event ID 1 where ParentImage is msedge.exe and Image resolves to a path under %APPDATA%, %TEMP%, or %USERPROFILE% — locations consistent with Edgecution dropping a native host binary outside of Program Files.

Evidence: This is an analysis step that does not alter live state; however, before any subsequent containment or eradication action, preserve: (1) Sysmon Event ID 1 (ProcessCreate) logs showing msedge.exe spawning the Native Messaging host process, including full command-line arguments and parent-child PID chain; (2) Sysmon Event ID 11

(FileCreate) records showing native host binary drop location and timestamp; (3) Windows Security Event ID 4688 (Process Creation with command line) for the same process tree if Sysmon is unavailable; (4) the JSON manifest file for the Native Messaging host (path found under NativeMessagingHosts registry key) which contains the allowed_origins field confirming the Edgecution extension ID; (5) network connection state via 'Get-NetTCPConnection | Where-Object {$_.OwningProcess -in (Get-Process msedge).Id}' to identify any C2 channels established by the native host process before isolation.

Step 3: Eradication — Remove the Edgecution extension from all affected endpoints via Edge enterprise management or manual uninstallation. Deregister any associated Native Messaging host entries from the Windows registry (HKCU and HKLM NativeMessagingHosts paths). Remove any native host binaries dropped by the extension. No vendor-issued patch ID is available in the supplied data; monitor Microsoft Edge Security Release Notes for a formal advisory. Apply NIST CM-2 (Baseline Configuration): restore and validate against documented baseline for Edge configuration and approved extension list. Apply CIS 2.3 (Address Unauthorized Software): remove the unauthorized extension per your documented process.

NIST Phase: Eradication

Reference: NIST 800-61r3 §3.4 — Eradication

Controls: NIST CM-2 (Baseline Configuration), NIST CM-6 (Configuration Settings), CIS 2.3 (Address Unauthorized Software)

Compensating: For teams without enterprise MDM, use a PowerShell script executed as SYSTEM to: (1) remove the Edgecution extension directory from '%LOCALAPPDATA%\Microsoft\Edge\User Data\Default\Extensions\[extension_id]'; (2) delete NativeMessagingHosts registry keys for the extension via 'Remove-Item -Path HKLM:\SOFTWARE\Microsoft\Edge\NativeMessagingHosts\[host_name] -Recurse' and the HKCU equivalent; (3) delete the dropped native host binary identified during detection. After removal, run 'Get-FileHash' on remaining Extension directories and compare against a known-clean baseline to detect any additional unauthorized extensions that may have been co-installed by Edgecution.

Evidence: Before removing the extension, native host binary, or registry keys — all of which destroy forensic state — capture: (1) a full RAM acquisition using WinPmem or DumpIt to preserve any in-memory artifacts from the native host executable that may reveal C2 configuration, injected shellcode, or decrypted payload; (2) a complete registry export of HKCU\Software\Microsoft\Edge\NativeMessagingHosts and HKLM\SOFTWARE\Microsoft\Edge\NativeMessagingHosts; (3) a file system image or forensic copy of the Edgecution extension directory (%LOCALAPPDATA%\Microsoft\Edge\User Data\Default\Extensions\[extension_id]) including background.js or service_worker.js files that contain the Native Messaging abuse logic; (4) a copy of the native host binary (hash it before removal) from the user-writable path identified in detection; (5) Sysmon Event ID 13 (RegistryValueSet) logs showing when and by whom the NativeMessagingHosts key was written, establishing the initial compromise timeline.

Step 4: Recovery — Validate that Native Messaging host registry keys return to baseline. Confirm no persistence mechanisms (scheduled tasks, run keys, services) were established by the native host payload using NIST IR-4 (Incident Handling) procedures. Re-image endpoints where confirmed execution occurred rather than relying solely on artifact removal. Monitor Edge process trees for 30 days post-remediation using endpoint detection tooling. Apply AU-6 (Audit Record Review, Analysis, and Reporting): schedule a follow-on log review at a defined frequency to confirm no residual indicators.

NIST Phase: Recovery

Reference: NIST 800-61r3 §3.5 — Recovery

Controls: NIST IR-4 (Incident Handling), NIST AU-6 (Audit Record Review, Analysis, And Reporting), NIST CM-2 (Baseline Configuration)

Compensating: For teams without EDR for 30-day monitoring: deploy a Sysmon configuration permanently capturing ProcessCreate events where ParentImage is msedge.exe, and schedule a weekly PowerShell audit script that enumerates (1) all NativeMessagingHosts registry keys, (2) all Edge extension directories, (3) scheduled tasks with actions pointing to user-writable paths, and (4) HKCU\Software\Microsoft\Windows\CurrentVersion\Run entries —

comparing output against a known-clean baseline and alerting on any delta. For re-imaging candidates, prioritize endpoints where Sysmon Event ID 1 confirmed the native host binary executed (not merely written to disk).

Evidence: Before re-imaging any endpoint, ensure the following volatile and persistent artifacts have been preserved for post-incident review: (1) full RAM acquisition if not yet captured; (2) Windows Security Event Log exported in EVTX format covering the full compromise window; (3) Sysmon log exported in EVTX format; (4) a timeline of scheduled task creation events (Sysmon Event ID 1 showing schtasks.exe spawned from the msedge.exe or native host process tree, or Windows Event ID 4698 — Scheduled Task Created); (5) HKLM and HKCU Run/RunOnce registry keys exported to confirm or rule out persistence established by the native host payload before the image is wiped.

Step 5: Post-Incident — Review and tighten browser extension governance: enforce ExtensionInstallForcelist to allow only vetted extensions, and block all others by default per NIST CM-7 (Least Functionality) and CIS 4.6 (Securely Manage Enterprise Assets and Software). Conduct a lessons-learned session to assess whether Native Messaging host registration paths are monitored in your SIEM. Update detection rules and playbooks to cover browser-to-native-host abuse patterns (MITRE T1559.001, T1176). Apply D3-UAP (User Account Permissions): review whether standard users should have write access to Native Messaging host registry paths. Assess whether CIS 6.3 (Require MFA for Externally-Exposed Applications) and CIS 6.5 (Require MFA for Administrative Access) are enforced to reduce post-execution lateral movement risk.

NIST Phase: Post Incident

Reference: NIST 800-61r3 §4 — Post-Incident Activity

Controls: NIST CM-7 (Least Functionality), NIST IR-4 (Incident Handling), NIST IR-8 (Incident Response Plan), CIS 4.6 (Securely Manage Enterprise Assets and Software), CIS 6.3 (Require MFA for Externally-Exposed Applications), CIS 6.5 (Require MFA for Administrative Access)

Compensating: For teams without a SIEM, create a scheduled osquery query pack targeting: (1) 'SELECT * FROM registry WHERE path LIKE "HKEY_LOCAL_MACHINE\\SOFTWARE\\Microsoft\\Edge\\NativeMessagingHosts\\%"' to detect new Native Messaging host registrations; (2) 'SELECT * FROM chrome_extensions WHERE browser_type = "edge"' to enumerate installed Edge extensions against your approved list. Publish a Sigma rule for community SIEM platforms (Elastic, Splunk free tier) detecting msedge.exe spawning processes outside of '%ProgramFiles%\\Microsoft\\Edge\\' — covering the core Edgecution abuse pattern for any future variant.

Evidence: This post-incident phase does not alter live system state and does not require volatile capture. Document and retain for the lessons-learned record: (1) the complete timeline reconstructed from Sysmon and Windows Event logs showing extension installation, NativeMessagingHosts key creation, native host binary drop, and first execution; (2) the Edgecution extension manifest.json and background script as static artifacts for future YARA rule development; (3) the native host binary (if recovered) submitted to an isolated sandbox (e.g., any.run, Cuckoo) for behavioral analysis to determine full payload capability; (4) the registry export of NativeMessagingHosts keys as a baseline reference artifact for future detection tuning.

Detection Guidance

Primary detection focus is on Native Messaging host registration and process execution. Monitor Windows registry keys HKCU\Software\Microsoft\Edge\NativeMessagingHosts and HKLM\SOFTWARE\Microsoft\Edge\NativeMessagingHosts for new or modified entries not present in your approved baseline (CIS 1.1, CIS 2.1). Use Sysmon or equivalent EDR to alert on process creation (Event ID 1) where the parent process is msedge.exe or edge.exe and the child process is an unexpected executable in a user-writable path (e.g., %APPDATA%, %TEMP%, %LOCALAPPDATA%). Alert on file creation (Sysmon Event ID 11) for new executable files in those same paths coinciding with Edge browser sessions. Query extension installation logs or Edge management telemetry for extensions not on the approved allow-list. Apply D3-SFA (System File Analysis) to monitor native host manifest JSON files, which define the executable path and allowed extension IDs, for unauthorized additions. No confirmed IOCs (hashes, domains, IPs) are available in the

supplied data; detection must rely on behavioral indicators until a primary source advisory is published. **Confidence Note: This item is based on secondary-source reporting. Verify all indicators and detection patterns against Microsoft's official Edge Security Release Notes before operationalizing in production environments.**

Framework Mappings

MITRE-ATTACK

- **T1543** — Create or Modify System Process
- **T1176** — Software Extensions
- **T1059** — Command and Scripting Interpreter
- **T1559.001** — Component Object Model

NIST-800-53R5

- **CM-7** — Least Functionality
- **SI-3** — Malicious Code Protection
- **SI-4** — System Monitoring
- **SI-7** — Software, Firmware, and Information Integrity
- **AC-3** — Access Enforcement
- **CM-3** — Configuration Change Control

OWASP-TOP10-2021

- **A01:2021** — Broken Access Control
- **A08:2021** — Software and Data Integrity Failures

CIS-V8

- **6.1** — Establish an Access Granting Process
- **6.2** — Establish an Access Revoking Process
- **2.5** — Allowlist Authorized Software
- **2.6** — Allowlist Authorized Libraries

SOC2-TSC

- **CC6.1** — The entity implements logical access security software, infrastructure, and architectures over protected information assets

HIPAA-SECURITY

- **164.312(a)(1)** — Access Control

MITRE ATT&CK Mapping

Technique ID	Technique Name	Tactic
T1543	Create or Modify System Process	Persistence

Technique ID	Technique Name	Tactic
T1176	Software Extensions	Persistence
T1059	Command and Scripting Interpreter	Execution
T1559.001	Component Object Model	Execution

Sources

Source	URL	Tier
Release notes for Microsoft Edge Security Updates	https://learn.microsoft.com/en-us/deployedge/microsoft-edge-relnote...	T1
Microsoft Edge known issues	https://learn.microsoft.com/en-us/deployedge/microsoft-edge-known-i...	T1
Microsoft Says Edge Password Security Vulnerability Is 'By Design'	https://www.forbes.com/sites/daveywinder/2026/05/06/microsoft-says-...	T3
New Microsoft Edge Vulnerability Exposes Users to Remote Code ...	https://www.reddit.com/r/pwnhub/comments/1txo0r7/new_microsoft_edge...	T3
Microsoft Browser Vulnerability Research	https://microsoftedge.github.io/edgevr/	T3

Generated 2026-06-27 13:43 UTC by TJS Security Command Center