# SOC Playbook: PowerShell Abuse Detection (T1059.001)

## 1. Objective

Detect and respond to unauthorized, suspicious, or malicious usage of PowerShell, including obfuscated or encoded scripts, suspicious execution contexts, and post-exploitation behaviors.

## 2. Scope

- Detection of PowerShell abuse across Windows environments.
- Targeting base64-encoded commands, AMSI bypasses, obfuscated payloads.
- Identification of fileless malware, lateral movement, and enumeration tools executed via PowerShell.
- Response automation for fast containment and investigation.

## 3. Log Sources

| Platform | Log Source | Description |
|---|---|---|
| Windows | PowerShell Operational Logs (Event ID 4104) | Captures script blocks for analysis |
| Windows | Security Event Logs (Event ID 4688) | Logs process creation |
| Windows | Sysmon Logs (Event ID 1, 7, 11, 13) | Logs detailed process/file/registry activity |
| All | EDR/XDR Logs | Advanced behavior and telemetry visibility |
| All | File Integrity Monitoring | Detects unauthorized script or binary changes |

## 4. Detection Rules / Alerts

| Alert Name | Description | Conditions / Triggers |
|---|---|---|
| Suspicious PowerShell Script Execution | PowerShell run from temp folders, user profiles, or via suspicious parent process | powershell.exe launched from %TEMP%, %APPDATA%, or with parent process winword.exe, outlook.exe, etc. |
| PowerShell Obfuscated Command | Encoded or obfuscated PowerShell commands | Event ID 4104 with base64 strings, string concatenation, or variable aliasing |
| AMSI Bypass Attempt | PowerShell script disabling AMSI | Script block contains strings like amsiInitFailed, Reflection, FromBase64String |

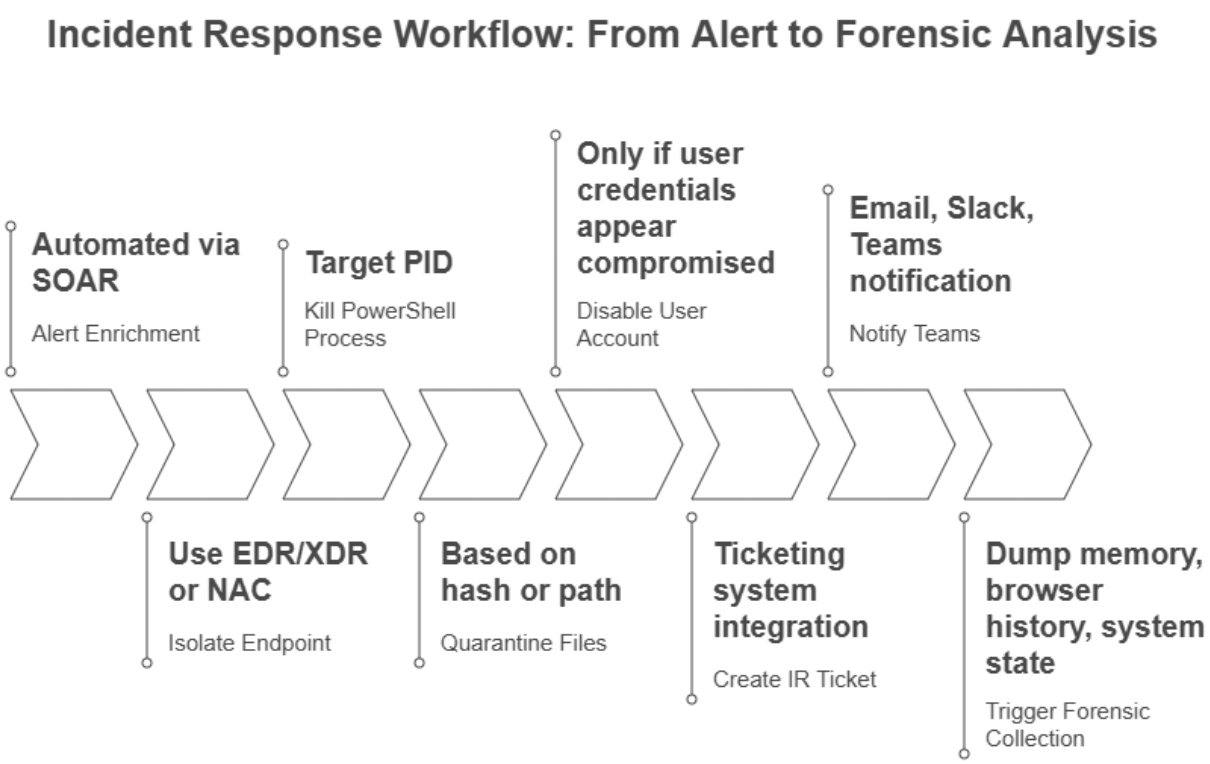
| | | |
|---|---|---|
| Download Cradle Detected | Use of PowerShell to download and execute code | Invoke-WebRequest, Invoke-Expression, IEX, Net.WebClient.DownloadFile |
| Unusual Parent Process for PowerShell | Execution of PowerShell by Office apps, browsers, or unknown binaries | Parent process not in known-good baseline (e.g., winword.exe → powershell.exe) |
| Encoded Command Flag Used | PowerShell started with -EncodedCommand flag | Command line contains powershell.exe -EncodedCommand |
| Suspicious PowerShell Script Execution | PowerShell run from temp folders, user profiles, or via suspicious parent process | powershell.exe launched from %TEMP%, %APPDATA%, or with parent process winword.exe, outlook.exe, etc. |

## 5. Automated Enrichment

| Enrichment Task | Details |
|---|---|
| Hash Lookup | Check script or parent process hash in VirusTotal/ReversingLabs |
| GeoIP Resolution | If external connection observed, enrich with GeoIP |
| User Context | Resolve username, logon type, domain/workgroup |
| Host Info | OS version, hostname, asset criticality tag |
| Parent Process Chain | Trace execution lineage via Sysmon or EDR |

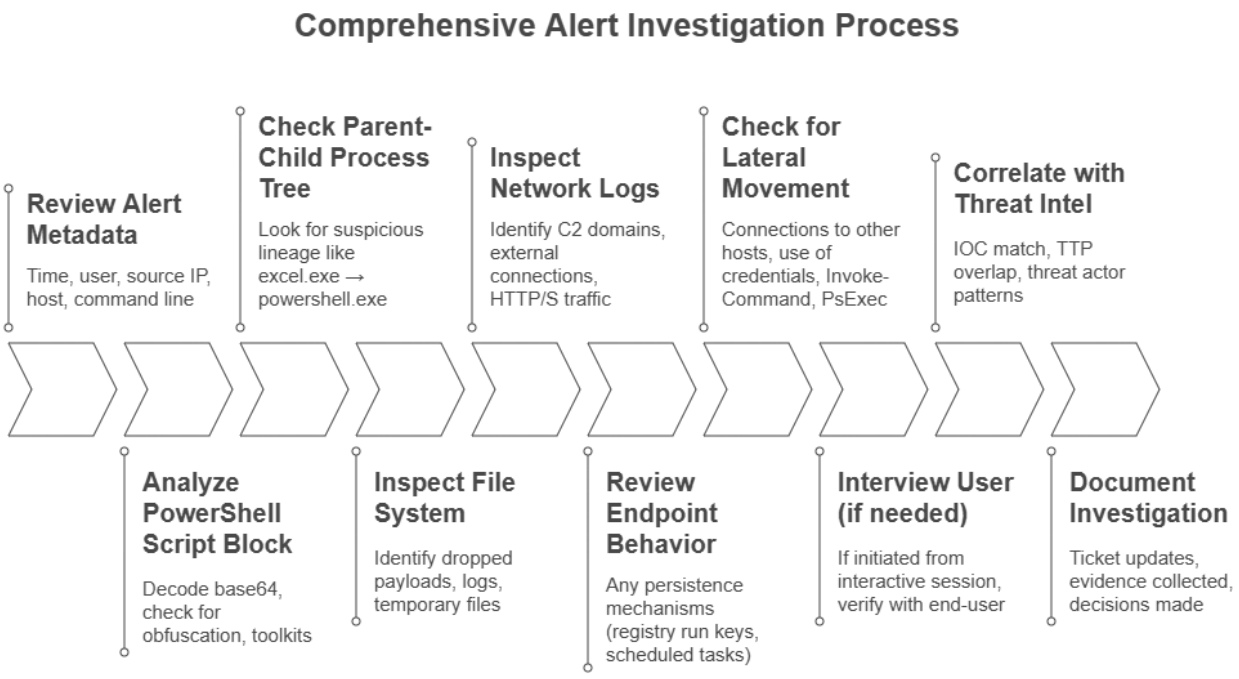
## 6. Automated Response Play

| Step | Action | Notes |
|---|---|---|
| Alert Enrichment | Automated via SOAR | Hash check, threat intel, user and host context |
| Isolate Endpoint | Use EDR/XDR or NAC | Blocks lateral movement |
| Kill PowerShell Process | Target PID | Use remote command or EDR |
| Quarantine Files | Quarantine dropped or involved scripts | Based on hash or path |
| Disable User Account | Temporarily disable if compromise suspected | Only if user credentials appear compromised |
| Create IR Ticket | Ticketing system integration | Include artifacts, impacted systems |
| Notify Teams | Email, Slack, Teams notification | Summary with host/user info and severity |
| Trigger Forensic Collection | Dump memory, browser history, system state | Retain for deeper analysis if necessary |




## 7. Investigation Checklist

| Step | Description |
|---|---|
| 1. Review Alert Metadata | Time, user, source IP, host, command line |
| 2. Analyze PowerShell Script Block | Decode base64, check for obfuscation, toolkits (Empire, PowerSploit, etc.) |
| 3. Check Parent-Child Process Tree | Look for suspicious lineage like excel.exe → powershell.exe |
| 4. Inspect File System | Identify dropped payloads, logs, temporary files |
| 5. Inspect Network Logs | Identify C2 domains, external connections, HTTP/S traffic |
| 6. Review Endpoint Behavior | Any persistence mechanisms (registry run keys, scheduled tasks) |
| 7. Check for Lateral Movement | Connections to other hosts, use of credentials, Invoke-Command, PsExec |
| 8. Interview User (if needed) | If initiated from interactive session, verify with end-user |
| 9. Correlate with Threat Intel | IOC match, TTP overlap, threat actor patterns |
| 10. Document Investigation | Ticket updates, evidence collected, decisions made |




## 8. Playbook Notes

- Tune detection to reduce false positives from legitimate admin automation.
- Enable script block logging (4104) via GPO or endpoint security settings.
- Baseline known good behaviors (scheduled scripts, admin toolkits).
- Monitor usage of -EncodedCommand, IEX, DownloadString, Add-Type.
- Keep threat intel feeds and IOCs updated to catch latest abuse methods.