Scenario 1 — Privilege Escalation & Discovery Detection

This report presents research and detection logic for identifying attacker behavior related to privilege escalation and account discovery using native Windows tools such as whoami.exe and net.exe. The research focuses on arguments commonly leveraged by adversaries and provides a KQL rule suitable for Microsoft Sentinel or Microsoft Defender for Endpoint (MDE).

1. Tools Overview

The following Windows binaries are often abused for discovery and privilege escalation:

• whoami.exe — used to display user, group, and privilege information.
- /all, /priv, /groups, /logonid, /upn, /fqdn, /claims

• net.exe / net1.exe — used for user, group, and share enumeration, as well as modification of group memberships.
- net user /domain
- net group /domain
- net localgroup administrators
- net view /domain, net share, net session, net use
- Privilege escalation attempts: net localgroup administrators <user> /add, net group "Domain Admins" <user> /add /domain

2. MITRE ATT&CK Mapping

• T1033: System Owner/User Discovery
• T1087: Account Discovery
• T1098: Account Manipulation

3. KQL Rule for Detection

Below is a sample detection rule in KQL using the DeviceProcessEvents table (Microsoft Defender for Endpoint data).

```
let TimeWindow = 1h;
let WhoamiArgs =
dynamic(["/all","/priv","/groups","/logonid","/upn","/fqdn","/claim
s"]);
let NetReconTokens = dynamic([" user "," group "," localgroup ","
view "," share "," session "," accounts "]);

// whoami discovery
let WhoamiDiscovery =
DeviceProcessEvents
```

```
| where Timestamp > ago(TimeWindow)
| where FileName =~ "whoami.exe"
| where ProcessCommandLine has_any (WhoamiArgs)
| extend Detection = "Account/Privilege discovery via whoami",
      ATTCK = "T1033/T1087";

// net.exe discovery
let NetDiscovery =
DeviceProcessEvents
| where Timestamp > ago(TimeWindow)
| where FileName in~ ("net.exe","net1.exe")
| where ProcessCommandLine has_any (NetReconTokens)
| where ProcessCommandLine has "/domain"
| where ProcessCommandLine !has "/add" and
ProcessCommandLine !has "/delete"
| extend Detection = "User/Group/Share discovery via net",
      ATTCK = "T1033/T1087";

// net.exe privilege escalation
let NetPrivEsc =
DeviceProcessEvents
| where Timestamp > ago(TimeWindow)
| where FileName in~ ("net.exe","net1.exe")
| where ProcessCommandLine has_all
("localgroup","administrators")
| where ProcessCommandLine has "/add"
| extend Detection = "Privilege escalation attempt: add to local
Administrators",
      ATTCK = "T1098";

union WhoamiDiscovery, NetDiscovery, NetPrivEsc
| project Timestamp, DeviceName, InitiatingProcessAccountName,
      FileName, ProcessCommandLine,
      Detection, ATTCK
```

---

4. Rule Trigger Conditions

The detection will be triggered under the following conditions:

- Execution of whoami.exe with discovery-related flags (/all, /priv, /groups, etc.).
- Execution of net.exe or net1.exe with domain-focused enumeration commands (user, group, localgroup, view, share, session, accounts) combined with /domain.
- Attempts to escalate privileges by adding users to the local Administrators group or to domain-level privileged groups using /add.

5. Tuning & Considerations

• Include both net.exe and net1.exe in the rule to cover compatibility variants.
• Pay attention to the parent process (cmd.exe, powershell.exe, wscript.exe, etc.).
• Add allowlists for known administrative automation accounts (e.g., SCCM, Ansible).
• Group multiple recon commands within a short timeframe to increase alert fidelity.

Another Task

Alerting & Detection Strategies (ADS) – Registry Persistence, Firewall Disable, Defender Disable

---

Scenario 1 – Registry Persistence via Run/RunOnce Keys

Goal
Detect when an attacker establishes persistence by adding entries to the Windows Run or RunOnce registry keys.

Categorization
Persistence / Registry Run Keys / Startup Folder (T1547.001)

Strategy Abstract
This alert looks for modifications to HKCU\Software\Microsoft\Windows\CurrentVersion\Run or HKLM\Software\Microsoft\Windows\CurrentVersion\Run registry keys. Detection is based on registry event logs ingested into Microsoft Defender for Endpoint or Windows Event Forwarding.

Technical Context
Attackers use Run/RunOnce registry keys to execute programs at user logon.
Example Command:

reg add HKCU\Software\Microsoft\Windows\CurrentVersion\Run /v MyApp /t REG_SZ /d "C:\malware\evil.exe"

KQL Rule

event.code: "4657" and event.action: "registry-value-modified"

Blind Spots and Assumptions

- Does not catch persistence via WMI, scheduled tasks, or services.
- Assumes registry logging is enabled.

False Positives

- Legitimate applications adding startup entries (e.g., OneDrive, Teams).

Validation

New-ItemProperty -Path "HKCU:\Software\Microsoft\Windows\CurrentVersion\Run" -Name "PersistenceTest" -Value "calc.exe" -PropertyType String -Force

Priority
High – Persistence often indicates adversary foothold.

Response

- Identify the process responsible for registry modification.
- Check if file path corresponds to known software.
- If malicious, isolate host and remove persistence.

Scenario 2 – Windows Firewall Disabled via Obfuscated PowerShell

Goal
Detect when an attacker disables the Windows Firewall using PowerShell, including obfuscated commands.

Categorization
Defense Evasion / Disable or Modify System Firewall (T1562.004)

Strategy Abstract
This alert detects PowerShell commands that attempt to disable Windows Firewall, even if obfuscation is used. Data source: PowerShell operational logs and command-line telemetry.

Technical Context
Common attacker commands:

Set-NetFirewallProfile -Profile Domain,Public,Private -Enabled False

netsh advfirewall set allprofiles state off

KQL Rule

powershell.file.script_block_text : "*Set-NetFirewallProfile*"

Blind Spots and Assumptions

- Cannot detect if firewall is disabled via Group Policy.
- Obfuscation with Base64 encoding may evade simple keyword searches unless expanded detection patterns are added.

False Positives

- Rare, but could occur if IT scripts legitimately modify firewall rules.

Validation
Run in admin PowerShell:

Set-NetFirewallProfile -Profile Domain,Public,Private -Enabled False

Confirm detection.

Priority
High – Disabling firewall significantly increases risk.

Response

- Validate if command was expected IT activity.
- If not, isolate device and re-enable firewall.
- Review process tree for lateral movement indicators.

Scenario 3 – Windows Defender Disabled via PowerShell

Goal
Detect attempts to disable Microsoft Defender AV via PowerShell.

Categorization
Defense Evasion / Disable or Modify Tools (T1562.001)

Strategy Abstract
This alert looks for PowerShell commands that modify Defender preferences, e.g., disabling real-time protection.

Technical Context
Attackers use commands like:

Set-MpPreference -DisableRealtimeMonitoring $true

Set-MpPreference -DisableIOAVProtection $true

KQL Rule

event.code : "4104" and

event.provider : "Microsoft-Windows-PowerShell"

Blind Spots and Assumptions

- If attacker tampers with Defender registry keys directly, this rule may not catch it.

- Assumes Defender logs are collected.

False Positives

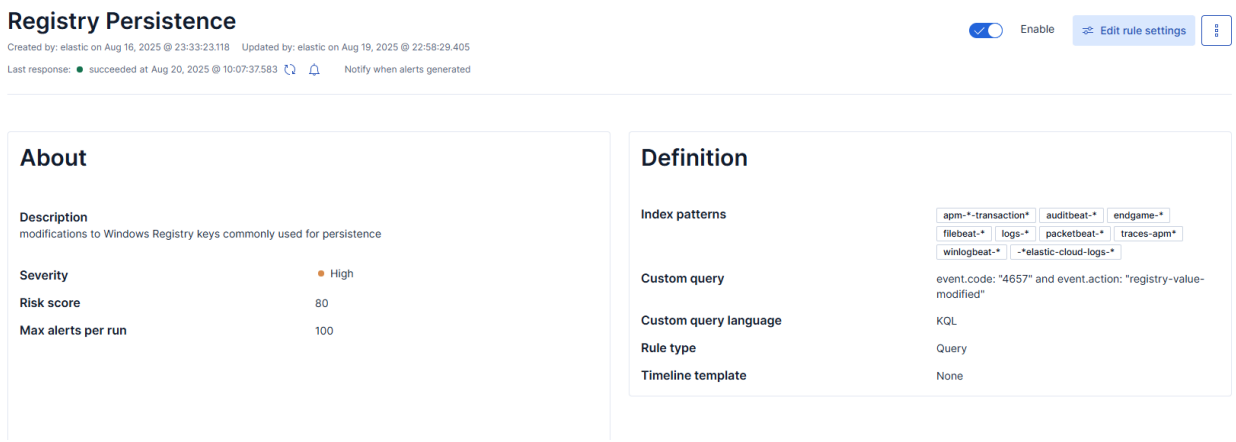
- Possible during legitimate IT maintenance/testing.

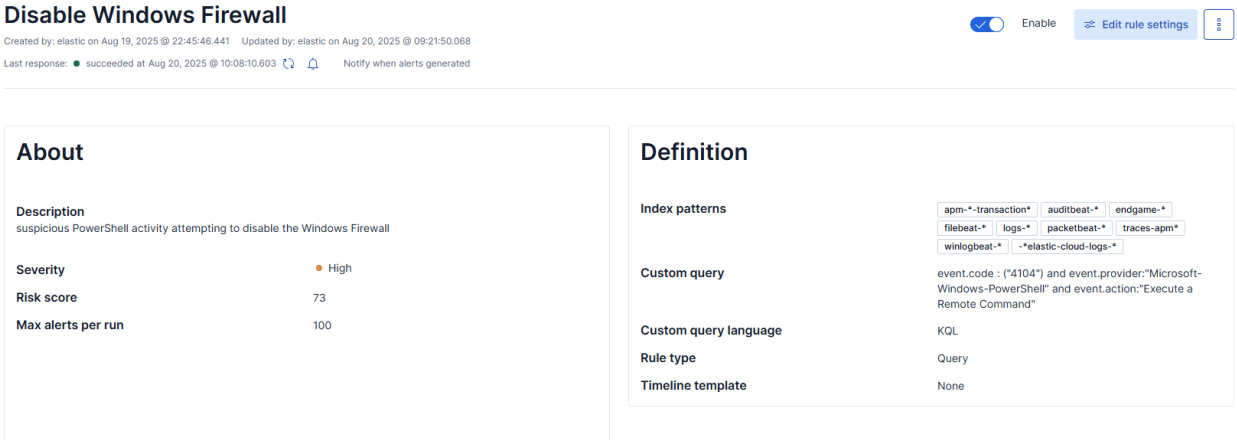
Priority

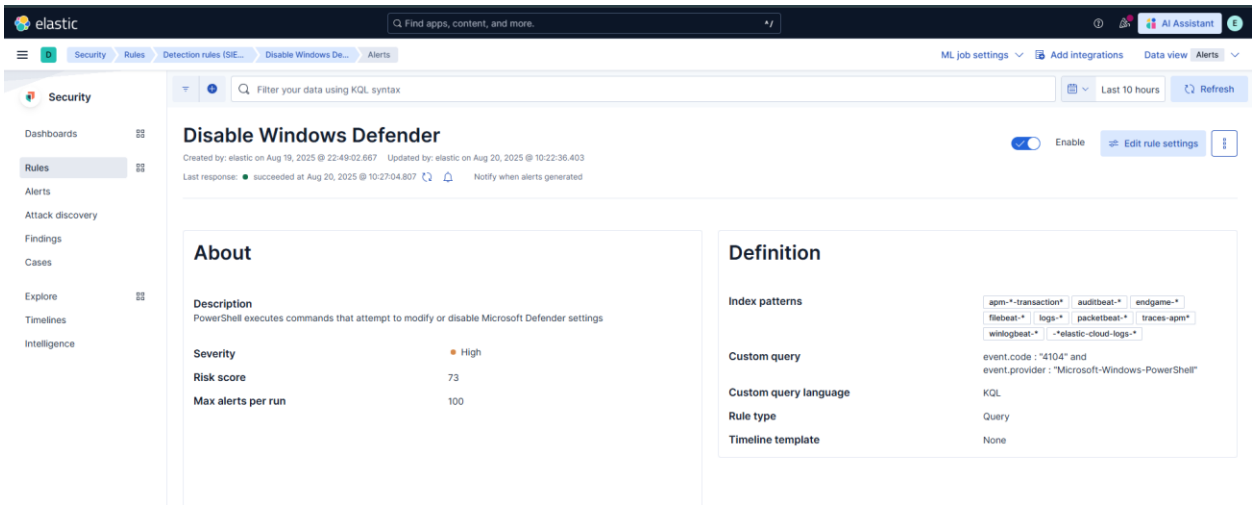
High – Disabling Defender removes core endpoint protections.

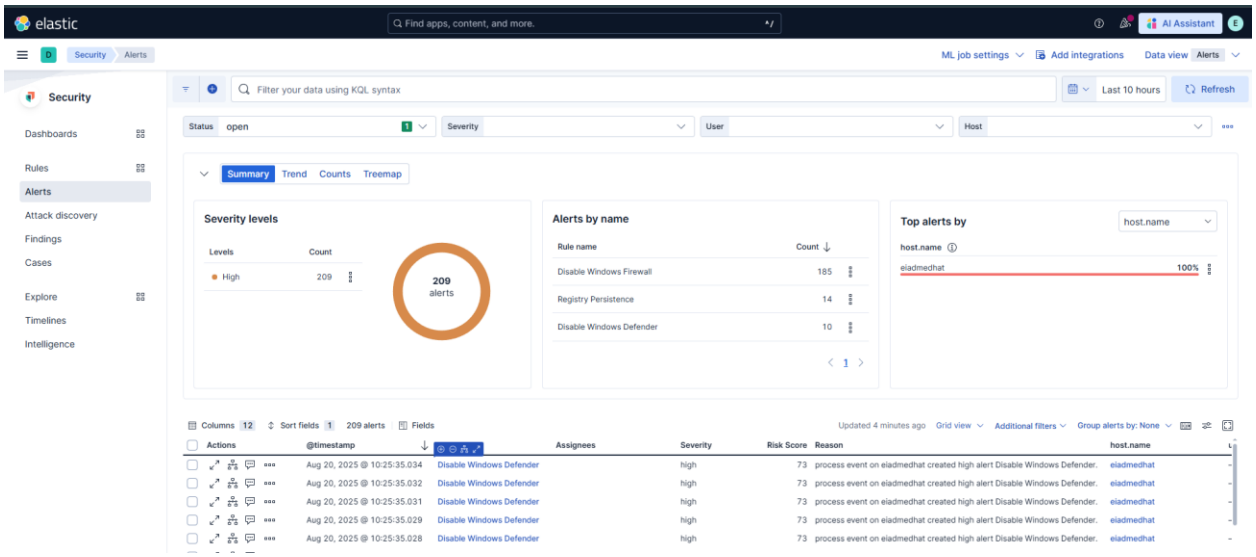
Response

- Validate legitimacy of Defender configuration change.
- If unauthorized, re-enable protections.
- Investigate parent process for further compromise.






elastic
Find apps, content, and more.
AI Assistant
Security
Rules
Detection rules (SIE...
Disable Windows De...
Alerts
ML job settings
Add integrations
Data view
Alerts
Security
Filter your data using KQL syntax
Last 10 hours
Refresh
Dashboards
Rules
Alerts
Attack discovery
Findings
Cases
Explore
Timelines
Intelligence
Disable Windows Defender
Created by: elastic on Aug 19, 2025 @ 22:49:02.667
Updated by: elastic on Aug 20, 2025 @ 10:22:36.403
Last response: succeeded at Aug 20, 2025 @ 10:27:04.807
Notify when alerts generated
Enable
Edit rule settings
About
Description
PowerShell executes commands that attempt to modify or disable Microsoft Defender settings
Severity
High
Risk score
73
Max alerts per run
100
Definition
Index patterns
apm-*-transaction*
auditbeat-*
endgame-*
filebeat-*
logs-*
packetbeat-*
traces-apm*
winlogbeat-*
-*elastic-cloud-logs-*
Custom query
event.code : "4104" and
event.provider : "Microsoft-Windows-PowerShell"
Custom query language
KQL
Rule type
Query
Timeline template
None


elastic
Find apps, content, and more.
AI Assistant
Security
Alerts
ML job settings
Add integrations
Data view
Alerts
Security
Filter your data using KQL syntax
Last 10 hours
Refresh
Dashboards
Rules
Alerts
Attack discovery
Findings
Cases
Explore
Timelines
Intelligence
Status
open
Severity
User
Host
Summary
Trend
Counts
Treemap
Severity levels
Levels
Count
High
209
209
alerts
Alerts by name
Rule name
Count
Disable Windows Firewall
185
Registry Persistence
14
Disable Windows Defender
10
1
Top alerts by
host.name
host.name
eladmedhat
100%
Columns
12
Sort fields
1
209 alerts
Fields
Updated 4 minutes ago
Grid view
Additional filters
Group alerts by: None
Actions
@timestamp
Assignees
Severity
Risk Score
Reason
host.name
Aug 20, 2025 @ 10:25:35.034
Disable Windows Defender
high
73
process event on eladmedhat created high alert Disable Windows Defender.
eladmedhat
Aug 20, 2025 @ 10:25:35.032
Disable Windows Defender
high
73
process event on eladmedhat created high alert Disable Windows Defender.
eladmedhat
Aug 20, 2025 @ 10:25:35.031
Disable Windows Defender
high
73
process event on eladmedhat created high alert Disable Windows Defender.
eladmedhat
Aug 20, 2025 @ 10:25:35.029
Disable Windows Defender
high
73
process event on eladmedhat created high alert Disable Windows Defender.
eladmedhat
Aug 20, 2025 @ 10:25:35.028
Disable Windows Defender
high
73
process event on eladmedhat created high alert Disable Windows Defender.
eladmedhat