Threat Hunt Report: Suspicious Internal Network Host Discovery

Detection of cmd.exe or PowerShell used to enumerate hosts (3 or more specific network actions within 5 minutes) on our network via ICMP or SMB from `peter-vm`.

Scenario:

Our Security Team is continuing to harden our cloud environment against attacks, including 'assumed-breach' situations where threat actors may already have access to one of our devices. With that access, attackers could potentially enumerate our inner networks and attempt to move laterally, pivot, or compromise additional accounts. To do this from inside our network, they would first need to discover what machines are on the network, which ports are open, and continue enumerating. Upon the attacker's first access of the compromised machine, they only have whatever applications and tools are present on that machine - there should not be network tools like nmap, rustscan, masscan that would let them check for other hosts/ports.

Without enumeration tools, attackers would either need to download and/or build compatible tools on the compromised host, or use 'living off the land' tactics to check for other hosts on the network that are 'up.' For this hunt we are not looking for enumeration/network tool downloads, though that could be a later hunt. We also want to be careful not to flag administrator or benign network activity, since some legitimate network tools could be in use on the network.

The main goal of this hunt is to detect when suspicious host discovery activities have happened *from* the **`peter-vm`** device, basically is there someone who has accessed that machine who is using it to enumerate 'up' hosts on the network, which could indicate lateral movement, pivoting, or host discovery in our network.

Note: During POC, all alerts apply ONLY to host '**`peter-vm`**'.

High-Level IoC Discovery Plan

- Emulate attacker behavior via 2 methods, from the VM in question.
  - The attacker behavior is very benign, it will entail several pings and TCP commands.
- Look for network behavior originating from the VM in question.
- Filter the network behavior by specific IPs, see which IPs are being contacted.
- Determine an amount of time which is suspicious - how many pings or network "are you up?" checks are needed at minimum for it to be considered suspicious behavior?
- Create a KQL rule that takes all the above into account. Remediations for this will be minimal, for instance there would be no automatic isolation, but an alert will definitely be made so our Security Team can know if possible intra-network enumeration is happening at our company.

***Of note, Windows machines are known for not responding to pings by default. However, we would still want to know of this behavior, because it would indicate malicious intent.

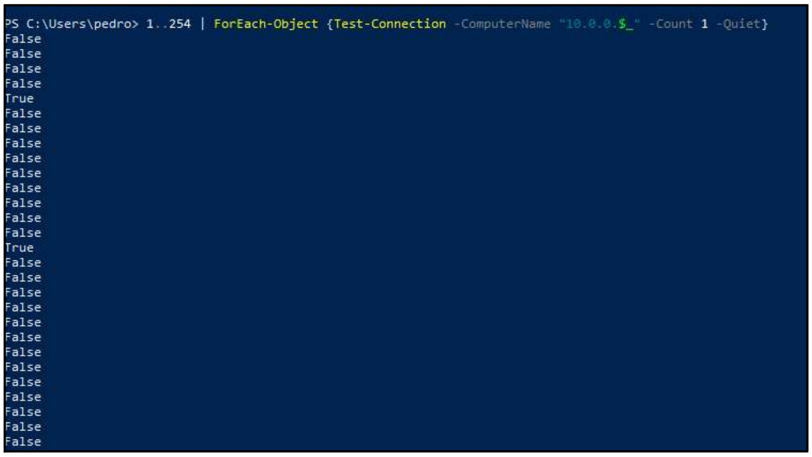
Steps Taken

1. Used a PowerShell script to ping the network:

ICMP Ping Sweep (Discovery via Ping)

```
1..254 | ForEach-Object {Test-Connection -ComputerName "10.0.0.$_" -Count 1 -Quiet}
```

The above command would serve as attacker emulation.

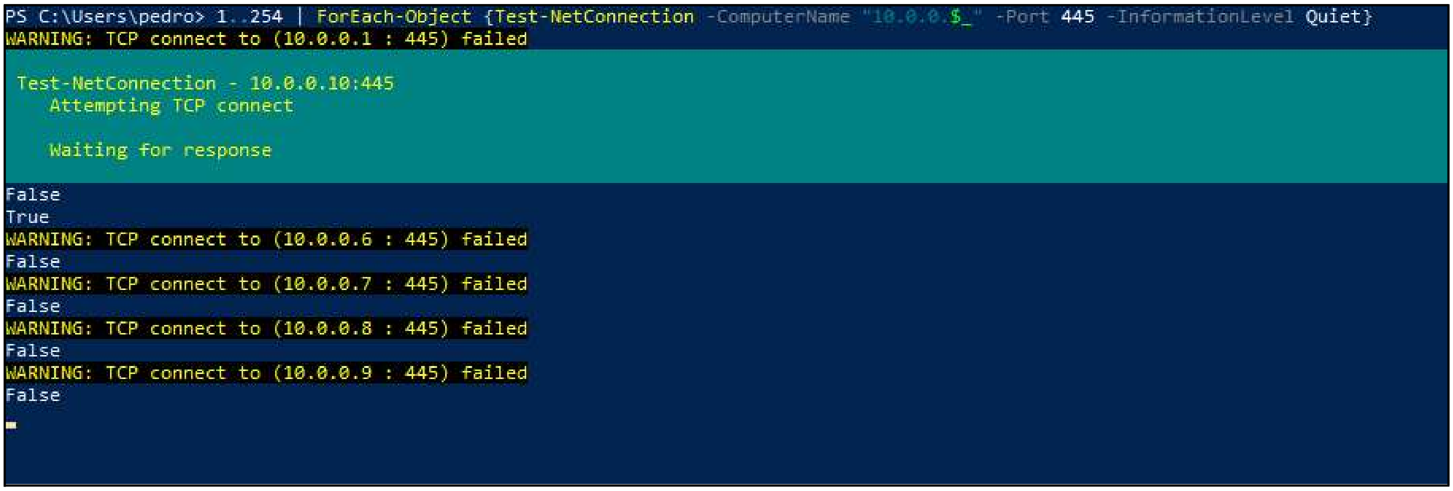


```
PS C:\Users\pedro> 1..254 | ForEach-Object {Test-Connection -ComputerName "10.0.0.$_" -Count 1 -Quiet}
False
False
False
False
True
False
False
False
False
False
False
False
False
False
True
False
False
False
False
False
False
False
False
False
False
False
False
False
```

2. An additional PowerShell script was used to check the network:

2. TCP Port Check on Port 445 (SMB)

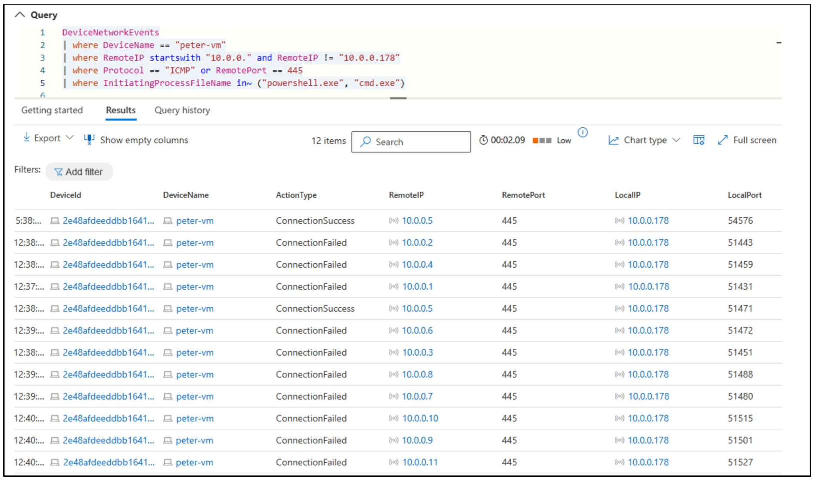
```
1..254 | ForEach-Object {Test-NetConnection -ComputerName "10.0.0.$_" -Port 445 -InformationLevel Quiet}
```



```
PS C:\Users\pedro> 1..254 | ForEach-Object {Test-NetConnection -ComputerName "10.0.0.$_" -Port 445 -InformationLevel Quiet}
WARNING: TCP connect to (10.0.0.1 : 445) failed

 Test-NetConnection - 10.0.0.10:445
    Attempting TCP connect

    Waiting for response

False
True
WARNING: TCP connect to (10.0.0.6 : 445) failed
False
WARNING: TCP connect to (10.0.0.7 : 445) failed
False
WARNING: TCP connect to (10.0.0.8 : 445) failed
False
WARNING: TCP connect to (10.0.0.9 : 445) failed
False
```

- Tries to connect to port 445 on every host in the subnet.

Example query output detecting PowerShell (or cmd.exe) network events (ICMP or port 445/SMB) coming from 'peter-vm':



```
DeviceNetworkEvents
| where DeviceName == "peter-vm"
| where RemoteIP startswith "10.0.0." and RemoteIP != "10.0.0.178"
| where Protocol == "ICMP" or RemotePort == 445
| where InitiatingProcessFileName in~ ("powershell.exe", "cmd.exe")
```

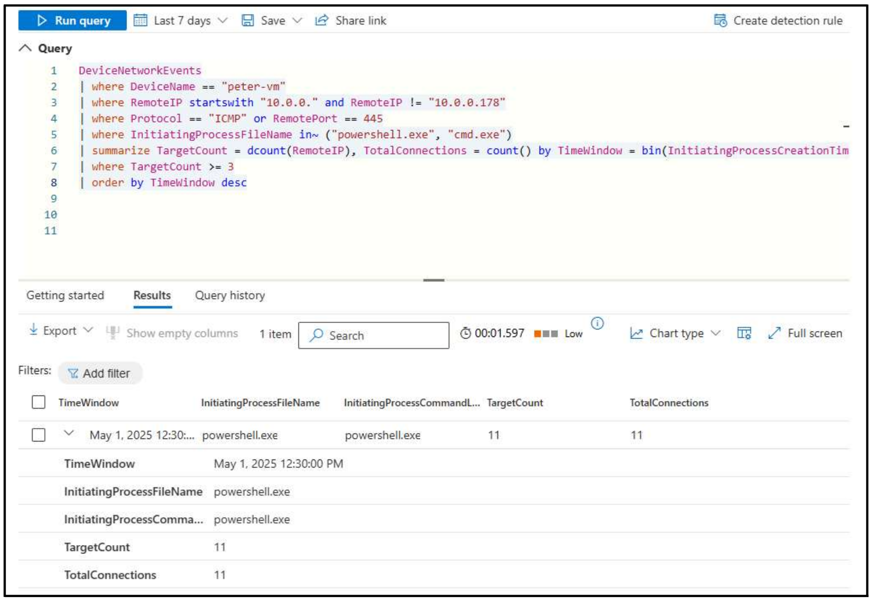
	DeviceId	DeviceName	ActionType	RemoteIP	RemotePort	LocalIP	LocalPort
5:38:...	2e48afdeeddbb1641...	peter-vm	ConnectionSuccess	10.0.0.5	445	10.0.0.178	54576
12:38:...	2e48afdeeddbb1641...	peter-vm	ConnectionFailed	10.0.0.2	445	10.0.0.178	51443
12:38:...	2e48afdeeddbb1641...	peter-vm	ConnectionFailed	10.0.0.4	445	10.0.0.178	51459
12:37:...	2e48afdeeddbb1641...	peter-vm	ConnectionFailed	10.0.0.1	445	10.0.0.178	51431
12:38:...	2e48afdeeddbb1641...	peter-vm	ConnectionSuccess	10.0.0.5	445	10.0.0.178	51471
12:39:...	2e48afdeeddbb1641...	peter-vm	ConnectionFailed	10.0.0.6	445	10.0.0.178	51472
12:38:...	2e48afdeeddbb1641...	peter-vm	ConnectionFailed	10.0.0.3	445	10.0.0.178	51451
12:39:...	2e48afdeeddbb1641...	peter-vm	ConnectionFailed	10.0.0.8	445	10.0.0.178	51488
12:39:...	2e48afdeeddbb1641...	peter-vm	ConnectionFailed	10.0.0.7	445	10.0.0.178	51480
12:40:...	2e48afdeeddbb1641...	peter-vm	ConnectionFailed	10.0.0.10	445	10.0.0.178	51515
12:40:...	2e48afdeeddbb1641...	peter-vm	ConnectionFailed	10.0.0.9	445	10.0.0.178	51501
12:40:...	2e48afdeeddbb1641...	peter-vm	ConnectionFailed	10.0.0.11	445	10.0.0.178	51527

```
DeviceNetworkEvents
| where DeviceName == "peter-vm"
| where RemoteIP startswith "10.0.0." and RemoteIP != "10.0.0.178"
| where Protocol == "ICMP" or RemotePort == 445
| where InitiatingProcessFileName in~ ("powershell.exe", "cmd.exe")
```

3. Updated the query to include actions that happened within 5 minute timeframe, and include more than 3 of the actions of interest (PowerShell or cmd.exe sending ICMP or SMB network requests):

```
DeviceNetworkEvents
| where DeviceName == "peter-vm"
| where RemoteIP startswith "10.0.0." and RemoteIP != "10.0.0.178"
| where Protocol == "ICMP" or RemotePort == 445
| where InitiatingProcessFileName in~ ("powershell.exe", "cmd.exe")
| summarize TargetCount = dcount(RemoteIP), TotalConnections = count() by TimeWindow
= bin(InitiatingProcessCreationTime, 5m), InitiatingProcessFileName,
InitiatingProcessCommandLine
| where TargetCount >= 3
| order by TimeWindow desc
```

KQL output from adjusted query:



```
DeviceNetworkEvents
| where DeviceName == "peter-vm"
| where RemoteIP startswith "10.0.0." and RemoteIP != "10.0.0.178"
| where Protocol == "ICMP" or RemotePort == 445
| where InitiatingProcessFileName in~ ("powershell.exe", "cmd.exe")
| summarize TargetCount = dcount(RemoteIP), TotalConnections = count() by TimeWindow = bin(InitiatingProcessCreationTim
| where TargetCount >= 3
| order by TimeWindow desc
```

TimeWindow	InitiatingProcessFileName	InitiatingProcessCommandL...	TargetCount	TotalConnections
May 1, 2025 12:30:...	powershell.exe	powershell.exe	11	11

TimeWindow	May 1, 2025 12:30:00 PM
InitiatingProcessFileName	powershell.exe
InitiatingProcessComma...	powershell.exe
TargetCount	11
TotalConnections	11

Chronological Events

Our attacker would have first needed to gain access to the `peter-vm` as this is an 'assumed breach' situation. They would then begin checking what other hosts on the internal network are 'up', using either pings (coordinated by a script), or perhaps using connection attempts to port 445 (SMB). Of note, pings may not be accepted by Windows hosts with the default firewall settings, or even potentially Linux hosts. The SMB connection attempts to port 445 would help to identify Windows hosts, so both methods were simulated to show further enumeration.

The simulated chronology here is:

1. Attacker gains credentials or lands on `peter-windows-v`.

- With credentialed access, the attacker can potentially log in and interact with the file system, and begin enumerating the internal network with cmd.exe or powershell.exe.

2. Attacker uses PowerShell or cmd.exe to send pings or connection requests to port 445 on some machines in the network, or maybe the entire subnet.

- Various scripts could be crafted to do this, or the attacker could even enter commands one-by-one for each computer they are interested in, in the IP range.

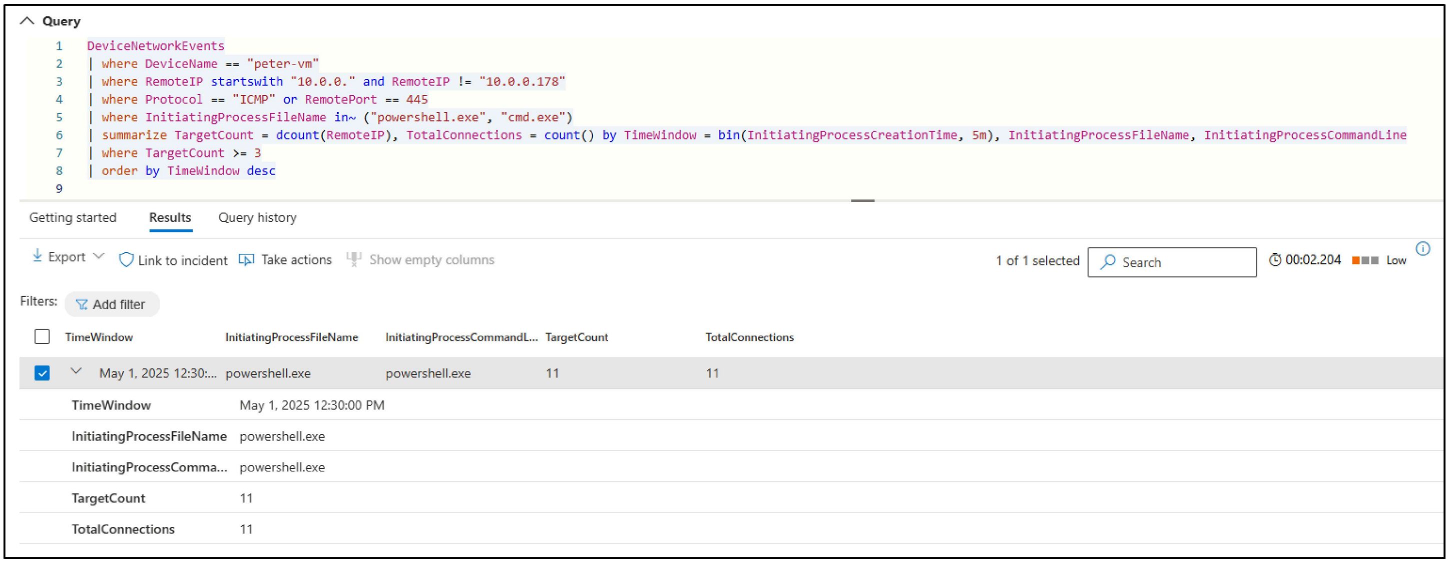
PowerShell:

```
1..254 | ForEach-Object {Test-Connection -ComputerName "10.0.0.$_" -Count
1 -Quiet}
```

and

```
1..254 | ForEach-Object {Test-NetConnection -ComputerName "10.0.0.$_"
-Port 445 -InformationLevel Quiet}
```

KQL query shows matching event starting at TimeWindow `2025-05-01T19:30:00Z`:

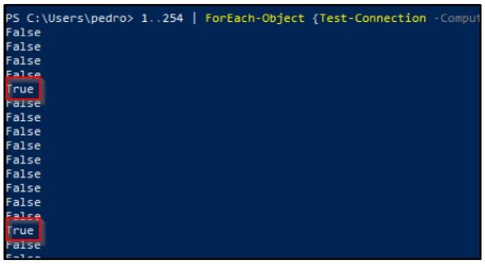


```
DeviceNetworkEvents
| where DeviceName == "peter-vm"
| where RemoteIP startswith "10.0.0." and RemoteIP != "10.0.0.178"
| where Protocol == "ICMP" or RemotePort == 445
| where InitiatingProcessFileName in~ ("powershell.exe", "cmd.exe")
| summarize TargetCount = dcount(RemoteIP), TotalConnections = count() by TimeWindow = bin(InitiatingProcessCreationTime, 5m), InitiatingProcessFileName, InitiatingProcessCommandLine
| where TargetCount >= 3
| order by TimeWindow desc
```

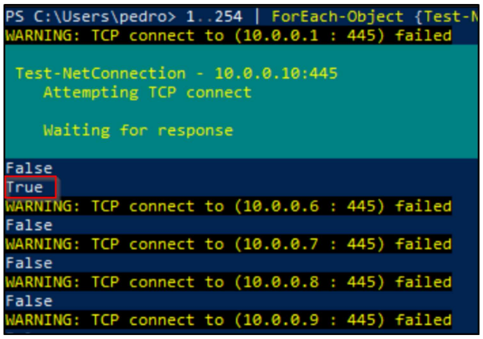
TimeWindow	InitiatingProcessFileName	InitiatingProcessCommandL...	TargetCount	TotalConnections
May 1, 2025 12:30:...	powershell.exe	powershell.exe	11	11

TimeWindow	May 1, 2025 12:30:00 PM
InitiatingProcessFileName	powershell.exe
InitiatingProcessComma...	powershell.exe
TargetCount	11
TotalConnections	11

3. In both the ping sweep and port 445 connect attempts, the fifth attempt appeared to succeed:



```
PS C:\Users\pedro> 1..254 | ForEach-Object {Test-Connection -Comput
False
False
False
False
True
False
False
False
False
False
False
False
False
False
True
False
False
```



```
PS C:\Users\pedro> 1..254 | ForEach-Object {Test-N
WARNING: TCP connect to (10.0.0.1 : 445) failed

 Test-NetConnection - 10.0.0.10:445
    Attempting TCP connect

    Waiting for response

False
True
WARNING: TCP connect to (10.0.0.6 : 445) failed
False
WARNING: TCP connect to (10.0.0.7 : 445) failed
False
WARNING: TCP connect to (10.0.0.8 : 445) failed
False
WARNING: TCP connect to (10.0.0.9 : 445) failed
```

- This could indicate a target for the next host for further reconnaissance.

- **Result:** Due to the network commands that were ran from a workstation inside our perimeter, an attacker could begin to form a 'map' of the network, and enumerate potential targets for lateral movement, pivoting, or more.

Summary

An assumed-breach hunt was examined, in this Azure company environment. With a busy subnet full of VMs, the simulated threat actor did not have network discovery tools like nmap, but wanted to discover active hosts on the network. To do this, there are numerous ways, but two obvious ways would be to ping all hosts on the network, or to attempt connection to high-probability ports like 445. The attacker automated these actions and ping/connection-sweeped the internal network, discovering multiple 'up' hosts. A KQL rule was formed to identify this behavior, to detect when 3 or more of the above network requests are made to hosts on the network, within 5 minutes. These amounts could be tuned, but it is unlikely that a network administrator would be ping sweeping the network at all, so we hope to identify malicious/suspicious behavior in the future with this threat hunt.

Response Taken

Our threat hunt resulted in a Medium-severity alert being generated, which can help our Security Team know when there is a potential Red Team or penetration exercise happening, or in the case of an actual breach, this would alert to internal network reconnaissance activities, warranting full investigation into endpoint/SIEM logs. The host could be isolated after triage depending on the findings, and the hosts receiving network probes via the above methods could also be investigated depending on the outcome.