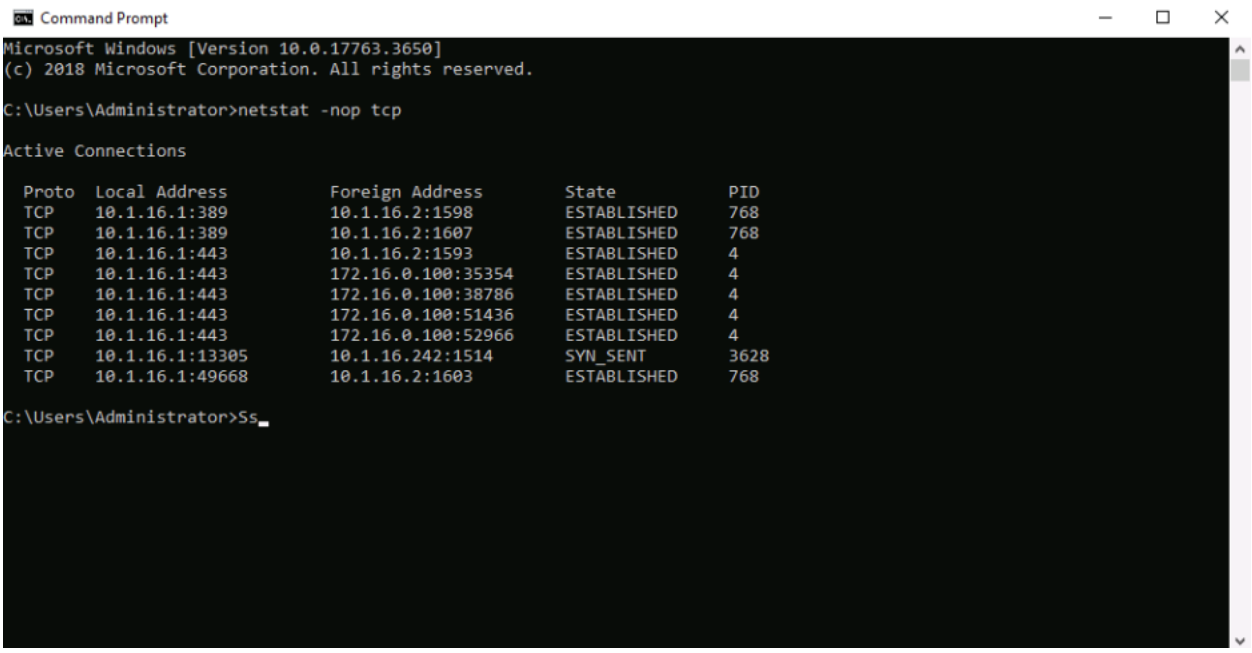
Threat Hunting New IoC

Adrian Cortez

In this project, I performed threat hunting within a segmented network environment. The DC10 and MS10 systems are located on the internal LAN, while the Kali system resides in a screened subnet. A new Indicator of Compromise (IoC) from the threat intelligence service indicated that secure websites were being targeted in a resource exhaustion attack, originating from any system capable of accessing the site. System monitoring tools had flagged repeated connection activity on DC10's web service, consuming significant resources. The goal of this project was to investigate the IoC, identify the source and pattern of the attack, and determine whether the environment was being impacted.

Access DC10 (Domain Controller VM) as an administrator and begin the investigation by running the following command:



```
Command Prompt
Microsoft Windows [Version 10.0.17763.3650]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\Administrator>netstat -nop tcp

Active Connections

  Proto  Local Address          Foreign Address        State           PID
  TCP    10.1.16.1:389          10.1.16.2:1598         ESTABLISHED     768
  TCP    10.1.16.1:389          10.1.16.2:1607         ESTABLISHED     768
  TCP    10.1.16.1:443          10.1.16.2:1593         ESTABLISHED     4
  TCP    10.1.16.1:443          172.16.0.100:35354     ESTABLISHED     4
  TCP    10.1.16.1:443          172.16.0.100:38786     ESTABLISHED     4
  TCP    10.1.16.1:443          172.16.0.100:51436     ESTABLISHED     4
  TCP    10.1.16.1:443          172.16.0.100:52966     ESTABLISHED     4
  TCP    10.1.16.1:13305        10.1.16.242:1514       SYN_SENT        3628
  TCP    10.1.16.1:49668        10.1.16.2:1603         ESTABLISHED     768

C:\Users\Administrator>Ss_
```

The netstat command is used to view current network connections and their status.

- The -n parameter forces numbers only to be displayed instead of hostnames, FQDNs, and protocol acronyms (such as TCP or HTTP).
- The -o parameter displays the associated process ID (PID).
- The -p tcp parameter limits the display to the selected protocol, in this instance: TCP.

You consult your network configuration documentation to determine the following details:

Hostname	IPv4 address
DC10	10.1.16.1
MS10	10.1.16.2
PC10	10.1.24.101
Kali	172.16.0.100
Wazuh	10.1.16.242
LAMP	172.16.0.201

Based on the netstat output and the network configuration documentation, you elect to investigate MS10 next.

The PID of secure services connected is 4. So, do

Enter:

```
tasklist /FI "PID eq 4"
```

to see only the process associated with the PID of concern.

The name of the process associated with this PID is System. Given this information, I decided to investigate MS10 instead.

```
Command Prompt
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:\Users\jaime>netstat -nop tcp

Active Connections

  Proto  Local Address          Foreign Address        State           PID
  TCP    10.1.16.2:1593         10.1.16.1:443          ESTABLISHED     4288
  TCP    10.1.16.2:1598         10.1.16.1:389          ESTABLISHED     2004
  TCP    10.1.16.2:1603         10.1.16.1:49668        ESTABLISHED     2004
  TCP    10.1.16.2:1607         10.1.16.1:389          ESTABLISHED     2004
  TCP    10.1.16.2:1639         10.1.16.1:135          TIME_WAIT       0
  TCP    10.1.16.2:1640         10.1.16.1:49668        ESTABLISHED     676
  TCP    10.1.16.2:1645         10.1.16.1:49699        TIME_WAIT       0

C:\Users\jaime>tasklist /FI "PID eq 4288"

Image Name                     PID Session Name        Session#    Mem Usage
========================= ======== ================ =========== ============
powershell.exe                4288 Services                   0     29,296 K

C:\Users\jaime>
```

We perform a similar process to view the current session and see the process associated with PID of the secure connection.

On MS10, Open PowerShell and run a command to export to a file the details about the process of concern.

```
Administrator: Windows PowerShell
Windows PowerShell
Copyright (C) 2016 Microsoft Corporation. All rights reserved.

PS C:\Windows\system32> Get-WmiObject win32_process -filter 'processid=4288' > outfile.txt
PS C:\Windows\system32> S
```

Enter:

`Select-String -Path .\outfile.txt -Pattern 'CommandLine'`

to view the CommandLine element of the PowerShell process to see the name of the script is running.

The script that is executing in the PowerShell process with PID: 4288 is C:\Users\jaime\lab04demo3.ps1

Run the following command to view the contents of the script:

```
PS C:\Windows\system32> type C:\Users\jaime\lab04demo3.ps1
$counter = 0

while ($counter -lt 10) {
  Invoke-WebRequest -Uri "https://ca.ad.structureality.com" | Out-Null
  Start-Sleep -Seconds 10
  #$counter++ #By commenting this line, the counter does not increment, and thus runs indefinitely.
}
```

You suspect this is a regularly scheduled task and confirm it by running the following command:

```
PS C:\Windows\system32> Get-ScheduledTask -TaskPath "\" | Where-Object {$_.State -eq "Running" -and $_.Principal.UserID
-eq "SYSTEM"}

TaskPath                                       TaskName                       State
--------                                       --------                       -----
\                                              Lab04Demo3.ps1                 Running


PS C:\Windows\system32>
PS C:\Windows\system32> S
```

Now that you have eliminated MS10 as a cause of IoC-related connections, you will shift your attention over to Kali.

In Kali,

Enter:

`netstat -np --protocol=inet`

to view the active processes on Kali related to IPv4 connections.

The netstat command on Linux is similar to but not exactly the same as the command on Windows. To view the full syntax, enter **`netstat -h.`** The parameters used here are:

- The -n parameter forces numbers only to be displayed instead of hostnames, FQDNs, and protocol acronyms (such as TCP or HTTP).
- The -p parameter displays the PID and process/program name
- The --protocol=inet parameter limits the display to only IPv4 protocols.

Notice how there are potentially several secure web sessions from Kali to DC10.

```
┌──(root㉿kali)-[~]
└─# netstat -np protocol=inet
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 172.20.22.129:22        172.20.0.11:41700       ESTABLISHED 942/sshd: root@nott
tcp        0      0 172.16.0.100:45892      10.1.16.1:443           ESTABLISHED 20626/nc
tcp        0      0 172.20.22.129:22        172.20.0.11:41384       ESTABLISHED 890/sshd: root@nott
tcp        0      0 172.16.0.100:48250      10.1.16.1:443           ESTABLISHED 20172/nc
tcp        0      0 172.20.22.129:22        172.20.0.11:41224       ESTABLISHED 755/sshd: root@nott
tcp        0      0 172.16.0.100:53414      10.1.16.1:443           ESTABLISHED 20120/nc
tcp        0      0 172.20.22.129:22        172.20.0.11:41552       ESTABLISHED 917/sshd: root@nott
tcp        0      0 172.16.0.100:59570      10.1.16.1:443           ESTABLISHED 21199/nc
udp        0      0 172.16.0.100:68         172.16.0.254:67         ESTABLISHED 431/NetworkManager
udp        0      0 172.20.22.129:68        172.20.0.1:67           ESTABLISHED 431/NetworkManager
Active UNIX domain sockets (w/o servers)
Proto RefCnt Flags       Type       State         I-Node   PID/Program name     Path
unix  3      [ ]         STREAM     CONNECTED     15625    410/dbus-daemon
unix  3      [ ]         STREAM     CONNECTED     19144    1213/xfce4-notifyd
unix  3      [ ]         STREAM     CONNECTED     19915    1162/xfdesktop
unix  3      [ ]         STREAM     CONNECTED     13427    1/init               /run/systemd/journal/stdout
unix  2      [ ]         DGRAM      CONNECTED     15673    469/ModemManager
unix  3      [ ]         STREAM     CONNECTED     15569    1/init               /run/systemd/journal/stdout
unix  3      [ ]         STREAM     CONNECTED     20052    1239/nm-applet
unix  3      [ ]         STREAM     CONNECTED     19067    560/Xorg             @/tmp/.X11-unix/X0
unix  3      [ ]         STREAM     CONNECTED     15234    560/Xorg             @/tmp/.X11-unix/X0
unix  3      [ ]         STREAM     CONNECTED     13691    392/haveged
unix  3      [ ]         STREAM     CONNECTED     19140    1213/xfce4-notifyd
unix  3      [ ]         STREAM     CONNECTED     19903    780/dbus-daemon      /run/user/0/bus
unix  3      [ ]         STREAM     CONNECTED     13271    1/init               /run/systemd/journal/stdout
unix  3      [ ]         STREAM     CONNECTED     15553    410/dbus-daemon
unix  2      [ ]         DGRAM      CONNECTED     13280    339/hv_kvp_daemon
unix  3      [ ]         STREAM     CONNECTED     19902    1157/Thunar
unix  3      [ ]         STREAM     CONNECTED     19185    780/dbus-daemon      /run/user/0/bus
unix  2      [ ]         DGRAM                    15660    431/NetworkManager
unix  3      [ ]         STREAM     CONNECTED     15593    1/init               /run/systemd/journal/stdout
unix  2      [ ]         DGRAM      CONNECTED     14627    409/cron
unix  3      [ ]         STREAM     CONNECTED     19999    1031/xfce4-session   @/tmp/.ICE-unix/1031
unix  3      [ ]         STREAM     CONNECTED     19919    560/Xorg             @/tmp/.X11-unix/X0
unix  3      [ ]         DGRAM      CONNECTED     12925    1/init               /run/systemd/notify
unix  3      [ ]         STREAM     CONNECTED     15592    415/systemd-logind
unix  3      [ ]         STREAM     CONNECTED     14671    413/polkitd
```

Having identified the system from where the suspicious secure web connections are originating, you need to compare your findings to the elements of the IoC: observable.

In this exercise, you have used an IoC to perform threat hunting. You traced the unwanted activity from a secure website host (i.e., DC10) to the origins of the abuse. You were able to eliminate MS10 as a suspected host of malware. Then you confirmed that Kali was the host of the abusive connections.