الهيئة الوطنية للأمن السيبراني
National Cybersecurity Authority

Destructive Attack “DUSTMAN”

Technical Report

TLP: WHITE

Traffic Light Protocol (TLP) Marking

The TLP system was created by USCERT to enable greater sharing of sensitive material and is widely used across CERTs and industry. There are four colors (traffic lights):

- **RED - Personal for named recipients only**

  Recipients may not share TLP: RED information with any parties outside of the specific exchange, meeting or conversation in which it is originally disclosed.

- **AMBER – limited distribution**

  Recipients may only share TLP: AMBER information with members of their own organization who need to know, and only as widely as necessary to act on that information.

- **GREEN – community wide**

  Recipients may share TLP: GREEN information with peers and partner organizations within their sectors or community, but via publicly accessible channels.

- **WHITE – unlimited:**

  Information may be distributed without restriction, subject to copyright controls.

Contents

1. Overview

Destructive attacks are quite extraordinary as threat actors employ their malware to disrupt or disable availability of the victim's resources by wiping contents on storage devices of the targeted systems. In these attacks, threat actors have already compromised the victim's network and gained privilege access to the internal infrastructure prior to the destruction activities.

In 2019, multiple destructive attacks were observed targeting entities within the Middle East. The National Cyber Security Centre (NCSC), a part of the National Cybersecurity Authority (NCA), detected a new malware named "DUSTMAN" that was detonated on December 29, 2019. Based on analyzed evidence and artifacts found on machines in a victim's network that were not wiped by the malware. NCSC assess that the threat actor behind the attack had some kind of urgency on executing the files on the date of the attack due to multiple OPSEC failures observed on the infected network. NCSC is calling the malware used in this attack "DUSTMAN" after the filename and string embedded in the malware.

"DUSTMAN" has different characteristics when compared to the multiple wiper malwares that have been observed through the years, especially the "Shamoon" variants although they all use the same third-party driver "Eldos RawDisk". Furthermore, "DUSTMAN" varies in terms of techniques and capability when compared to "Shamoon" and from the observed behavior and capabilities, "DUSTMAN" can be considered as a new variant of "ZeroCleare" malware, published in December 2019[(1)].

This report will shed the light on the attack life cycle, technical analysis of the malware, and the preventive recommendation with the Yara rules. It is worth mentioning that NCSC is still coordinating all efforts in understanding the extent of the attack, malware and attribution.

(1): New Destructive Wiper "ZeroCleare" Targets Energy Sector in the Middle East, December 2019, IBM X-Force.

2. Attack Life Cycle

Based on the analysis of the collected artifacts, NCSC found artifacts that shows early signs of compromise of the network that dates back few months prior to the destructive attack. This illustrates the determination and persistence of the threat actor behind this attack that follows the typical attack life cycle of nation-state threat actor. "Figure 1" below demonstrates the high-level phases of the attack life cycle.

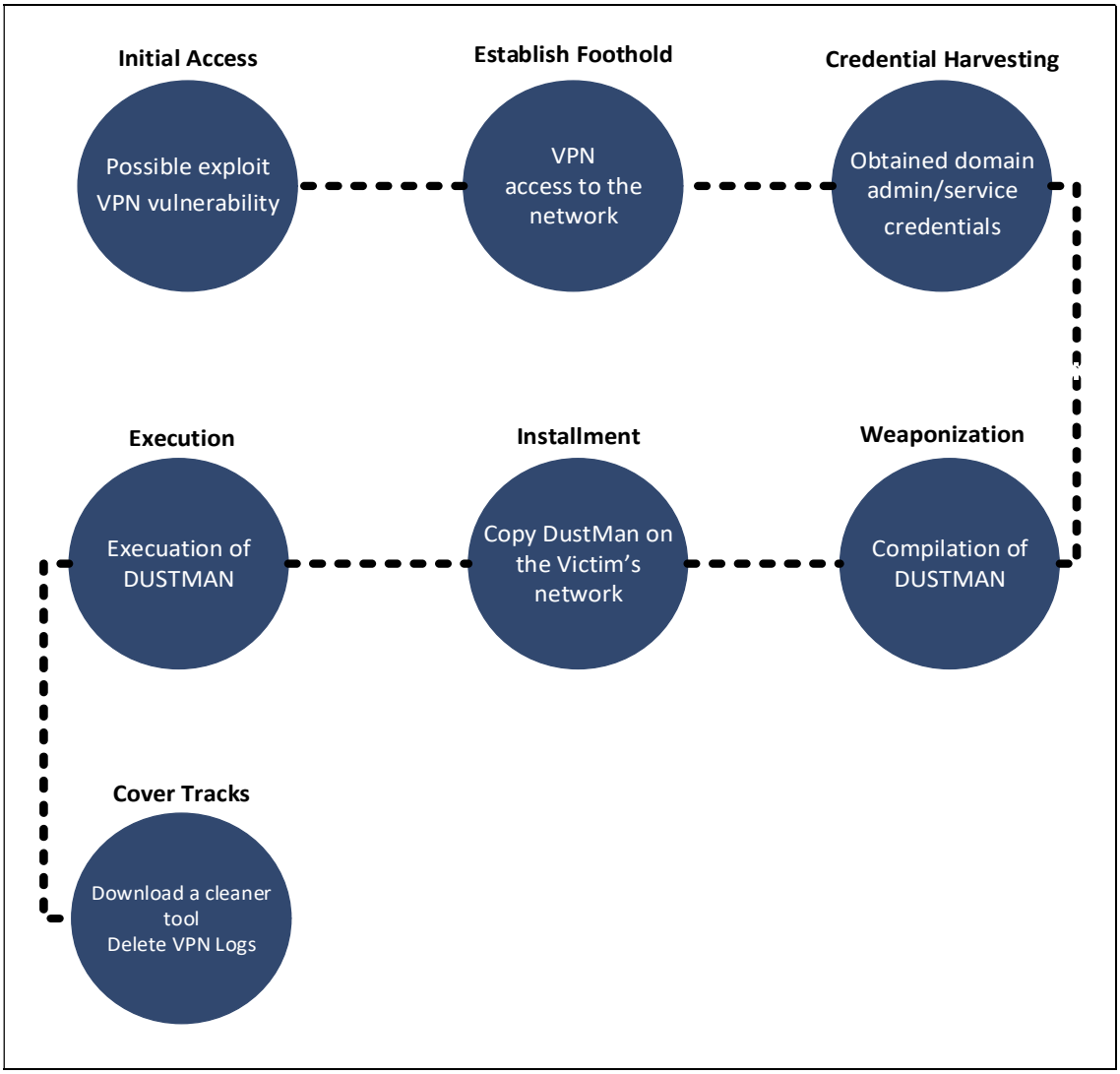

FIGURE 1: ATTACK LIFE CYCLE

2.1. Initial Access:

NCSC assess with moderate confidence that the initial access of the attack occurred by exploiting one of the remote execution vulnerabilities in a VPN appliance that was disclosed in July 2019.

2.2. Establish Foothold:

In this phase, the threat actor has gained access to the VPN server.

2.3. Credential Harvesting and Lateral Movement:

The threat actor obtained domain admin and service accounts on the victim’s network, which was used to run “DUSTMAN” malware on all of the victim’s systems. The attacker utilized the anti-virus management console service account to distribute the malware across the network.

2.4. Weaponization

DUSTMAN malware was compiled, possibly on the threat actor infrastructure, few minutes before deploying it on the victim’s network. This is inconsistent with known destructive attacks as they usually tested f before being deployed.

2.5. Installment

The threat actor accessed the victim’s network and copied the malware and the remote execution tool “PSEXEC” into the anti-virus management console server, which was connected to all machines within the victim’s network due to the nature of its functionality. Few minutes later, the attacker accessed the storage server of the victims and deleted all volumes manually.

2.6. Execution

The attackers then executed a set of commands on the anti-virus management control to distribute the malware to all connected machines, and through (PSEXEC) the malware executed and dropped (3) additional files, two drivers and the wiper. Most of the connected machines were wiped.

NCSC believes that the dropped files from "DUSTMAN" would be detected by the current Anti-Virus, However the files bypassed the antivirus detection and were executed successfully. The threat actor either disabled the Anti-Virus or whitelisted the "Eldos RawDisk" driver in order to allow the wiper execution. Fortunately, few machines were in the sleeping mode at the time of this action and once these machines were started, the anti-virus detected and prevented the execution of the "Eldos RawDisk" driver.

2.7. Covering Tracks

After the wiping mission is accomplished, the attacker logged to the VPN server and deleted the recent VPN access log and downloaded a legitimate file deletion tool and used it to cover their tracks. This download indicates that the threat actor identified few failed wipes on some systems and attempted to clean any left artifact on these systems.

2.8. Wiping Sequence Timeline

Activity
agent.exe was compiled
Dustman.exe was compiled
Accessing the victim domain through VPN
Dustman.exe was copied to the victim's Anti-Virus management console.
Manually deleting all volumes of the storage server.
Disabling the anti-virus on all connected machines
Initiated the attack remotely utilizing PSEXE from the ani-virus management console.
Dustman.exe was copied and executed on all machine connected to the network.
elrawdsk.sys and assisstant.sys were dropped
VBoxDrv driver service was created and utilized to install the unsigned driver elrawdsk.sys

TABLE 1: WIPING SEQUENCE TIMELINE

3. Dustman Analysis Overview

Few wiping activities have been observed in the year 2019. The below “Figure 2” shows the timeline of these activities. The attribution and the threat actor behind these attacks is still not determined. However, commitment, persistence and TTPs observed indicates that these are possibly an act of one nation state actor.

FIGURE 2: TIMELINE OF ACTIVITIES RELATED TO WIPING ATTACKS IN 2019

1. In March 2019, a modified version of “Turla Driver Loader (TDL)” was published. This modified variant exploits a vulnerable legitimate signed driver to bypass “Window’s Driver Signature Enforcement (DSE)” that prevents loading unsigned drivers such as the raw disk driver utilized by the threat actor destructive malware samples.
2. In September 2019, IBM posted an alert on its intelligence portal about “ZeroCleare” attacks as it responded to incidents occurred to organizations within the Middle East.
3. In early December 2019, IBM published a detailed analysis report about “ZeroCleare”.
4. On December 29, 2019, NCSC identified “DUSTMAN” that was used by an adversary, that shares similarities with “ZeroCleare”.

3.1. DUSTMAN vs ZeroCleare

DUSTMAN seems to be a new variant of “ZeroCleare” as the raw disk driver used by it shares the same exact digital fingerprint as the one utilized by “ZeroCleare”. Furthermore, both “DUSTMAN” and “ZeroCleare” utilized a skeleton of the modified “Turla Driver Loader (TDL)” published on March 2019 on GitHub. “Turla Driver Loader (TDL)” exploits a vulnerability in a legitimate singed but vulnerable driver which allows the attackers to overcome the operating system protection against loading unsigned drivers.

Even though the same exact raw disk driver was used in the "ZeroCleare" attack, the techniques utilized by "DUSTMAN" is different. First, an optimization mechanism has been added to "DUSTMAN" that is considered an optimization from "ZeroCleare", where the destructive capability and all needed drivers and loaders are delivered in one executable file as opposed to two files as was the case with "ZeroCleare". Another difference is that "ZeroCleare" wipes the volume by overwriting it with garbage data (0x55) while "DUSTMAN" overwrites the volume.

3.2. Technical Analysis

The malware executable file "dustman.exe" is not the actual wiper, however, it contains all the needed resources and drops three other files upon execution. These files are two drivers and the wiper. These 3 files are resident as encrypted resources in the "DUSTMAN" malware. The malware decrypts these resources and writes them to the same folder where the malware is executed. The characteristics of these files and resources are listed in the "Table 2" below:

File Name	Resource Name	Objective	Resource/File Size (Bytes)
assistant.sys	1	Vulnerable signed driver used to exploit the operating system to load unsigned drivers	68,288
elrawdisk.sys	103	Unsigned driver by Eldos that is used to access and write raw volume data	24,576
agent.exe	106	Wiper process that overwrites a volume passed to it as an argument with a preconfigured string	116,224

TABLE 2: FILES DROPPED BY THE DUSTMAN MALWARE

As those resources can be seen after performing static analysis of the files attributes in "Figure 3" below:

type (2)	name	file-offset (4)	signature	non-standard	size (209233 bytes)
rcdata	1	0x00011100	unknown	-	68288
rcdata	103	0x00021BC0	unknown	-	24576
rcdata	106	0x00027BC0	unknown	-	116224
manifest	1	0x000441C0	manifest	-	145

FIGURE 3: RESOURCES CONTAINED IN THE FILE DUSTMAN.EXE

Once the "DUSTMAN" executable is executed, the two drivers gets dropped and installed and then the agent.exe (the wiper) gets executed with the assistance of the recently installed drivers. The sequence can be seen in the "Figure 4" below:

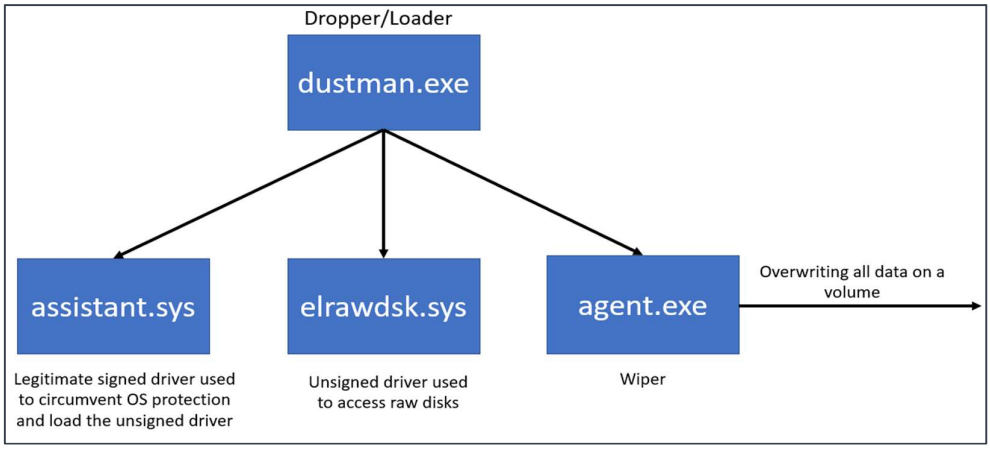

FIGURE 4: TECHNICAL DETAILS

3.2.1. Dustman.exe

The malware sample "dustman.exe" follows the below steps to achieve its destructive purposes:

1. Checks if the malware was executed previously on the system by creating a mutex. If the mutex is found, it means the malware was previously executed on a system, malware jumps to step 8.

```
lea     r8, Name
xor     edx, edx        ; bInitialOwner
xor     ecx, ecx        ; lpMutexAttributes
mov     cs:qword_14000F2C0, rax
call    cs:CreateMutexW
```

FIGURE 5: CREATING MUTEX TO TRACK MACHINES FOR WHICH THE MALWARE WAS EXECUTED

2. Extracts the resource "elrawdsk.sys", decrypts it, and then writes the decrypted resource to a file on the current directory of the malware "dustman.exe".

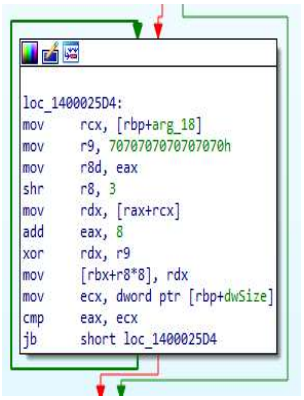

FIGURE 6: DECRYPTION ROUTINE USED TO DECRYPT THE DIFFERENT RESOURCED EMBEDDED IN THE MALWARE

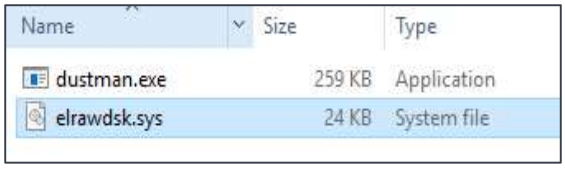

FIGURE 7: ELRAWDSK.SYS IS THE FIRST FILE TO BE DROPPED

3. Checks if "VirtualBox" service is running on the system, If the service is running, the malware tries to stop it. The checks happens by looking for the registry key "Software\Oracle\VirtualBox" in the registry.

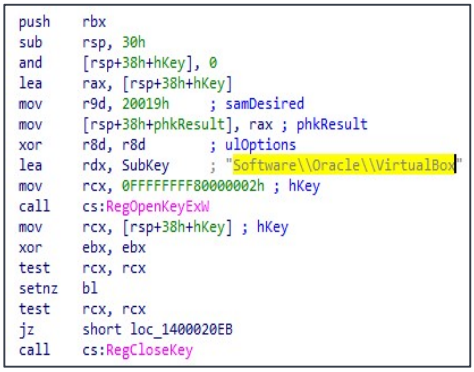

FIGURE 8: ROUTINE TO CHECK IF VIRTUALBOX IS INSTALLED BY QUERYING THE REGISTRY

4. Extracts the vulnerable singed driver resource ("assistant.sys"), decrypts it, and then writes the decrypted resource to a file on the current directory of the dropper malware "dustman.exe".

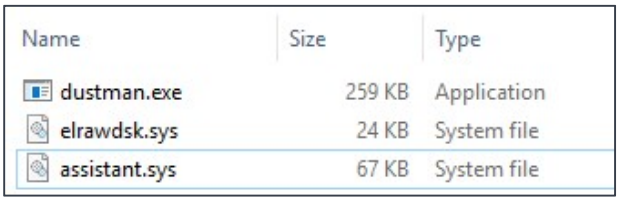

FIGURE 9: ASSISTANT.SYS IS THE SECOND FILE TO BE DROPPED

5. Creates and starts a driver service “VBoxDrv” backed by signed vulnerable driver “assistant.sys”.

```
C:\WINDOWS\system32>driverquery | findstr VBoxDrv
VBoxDrv      VBoxDrv                 Kernel         5/31/2008 5:18:53 AM
```

FIGURE 10: LISTING VBOXDRV UPON ITS INSTALLATION

6. Exploits the vulnerable service driver “assistant.sys” to overcome “Windows Driver Signature Enforcement (DSE)” and loads the unsigned raw disk driver “elrawdsk.sys” by writing shellcode to the vulnerable signed driver which would cause the unsigned driver to be loaded into the system.

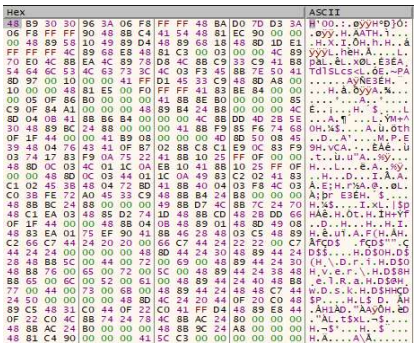

FIGURE 11: SHELLCODE USED TO EXPLOIT THE VULNERABLE SINGED DRIVER

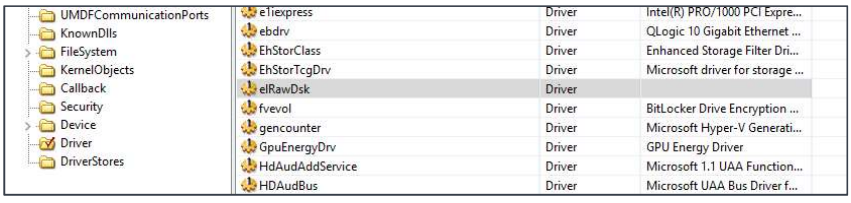

FIGURE 12: UNSIGNED DRIVER ELRAWDSK UPON ITS LOADING IN THE SYSTEM. SCREENSHOT FROM WINOBJEX64.EXE.

7. Unloads the vulnerable signed driver and removes the driver service that was created.

```
C:\WINDOWS\system32>driverquery | findstr VBoxDrv

C:\WINDOWS\system32>
```

FIGURE 13: UNSIGNED DRIVER ELRAWDSK UPON ITS LOADING IN THE SYSTEM. SCREENSHOT FROM WINOBJEX64.EXE.

8. Extracts the wiper malware resource "agent.exe", decrypts it, and then writes the resource to a file on the current directory of the dropper malware "dustman.exe".

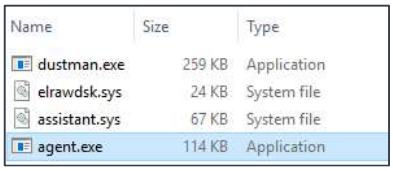

FIGURE 14: AGENT.EXE IS THE LAST FILE TO BE DROPPED

9. Enumerates the different volumes on the system using the system call "GetLogicalDriveStringsW".

```
mov     rbx, rax
call    cs:GetLogicalDriveStringsW
test    eax, eax
```

FIGURE 15: GETLOGICALDRIVESTRINGW IS USED TO ENUMERATE ALL VOLUMES ON A WINDOWS SYSTEM

10. Starts the wiping process "agent.exe" for every volume on the system. The command used to wipe the volume will have the volume letter appended to the command, for example, to wipe the C drive, the command would be "$:\windows\system32\cmd.exe /c agent.exe C" where the "C" at the end of the command indicates the volume to be wiped.

```
"C:\\windows\\system32\\cmd.exe"
"/c agent.exe C"
```

FIGURE 16: DUSTMAN.EXE STARTING AGENT.EXE TO WIPE OUT THE C VOLUME

3.2.2. Agent.exe

The wiper "agent.exe" follows the below steps to achieve its destructive purposes:

1. Reads the command line arguments to determine the disk to be wiped.

```
call    cs:GetCommandLineA
mov     cs:qword_14001C518, rax
call    cs:GetCommandLineW
mov     cs:qword_14001C520, rax
```

FIGURE 17: SYSTEM CALL TO READ THE COMMAND LINE ARGUMENTS

2. Initialize a buffer that contains a message.

3. Creates a handle to the Eldo RawDisk driver. The driver was already loaded by "dustman.exe" as a system driver and should be ready for usage. When creating a handle to the driver, the string: (b4b615c28ccd059cf8ed1abf1c71fe03c0354522990af63adf3c911e2287a4b906d47d) is appended to the driver name "ElRawDisk" and the volume to be accessed utilizing the drive "C". We believe this string to be the license key for Eldos.

4. Retrieves information about the disk to be wiped including its geometry and length using multiple calls of "DeviceIOControl" with different control codes.

5. Starts a loop to overwrite all data in the volume utilizing the Eldos RawDisk driver and the system call "DeviceIOControl". In each loop iteration, 40,960 (40 KB) bytes are overwritten.

```
00007FF67E892390   4C:8D0D 39AA0100   lea r9,qword ptr ds:[7FF67E8ACDD0]
00007FF67E892397   44:8BC6            mov r8d,esi
00007FF67E89239A   48:8BD7            mov rdx,rdi
00007FF67E89239D   49:8BCF            mov rcx,r15
00007FF67E8923A0   E8 4BEEFFFF        call agent.7FF67E8911F0
00007FF67E8923A5   48:2BFE            sub rdi,rsi
00007FF67E8923A8   48:83EB 01         sub rbx,1
00007FF67E8923AC   79 E2              jns agent.7FF67E892390
```

FIGURE 18: LOOP TO OVERWRITE THE VOLUME

6. If successful, a blue screen will be displayed to the user as shown in "Figure 19".

FIGURE 19: COMPUTER SCREEN UPON CRASHING THE SYSTEM AFTER IT IS WIPED

4. Host Indicator of Comprise

Name	dustman.exe
MD5 Hash	8AFA8A59EEBF43EF223BE52E08FCDC67
SHA-1 Hash	E3AE32EBE8465C7DF1225A51234F13E8A44969CC
SHA-256 Hash	F07B0C79A8C88A5760847226AF277CF34AB5508394A58820DB4DB5A8D0340FC7
Size	264,704 (bytes)
Type	64-bit EXE
Compilation Date	Sun Dec 29 08:57:19 2019 (GMT+3)

Name	elrawdsk.sys
MD5 Hash	993E9CB95301126DEBDEA7DD66B9E121
SHA-1 Hash	A7133C316C534D1331C801BBCD3F4C62141013A1
SHA-256 Hash	36A4E35ABF2217887E97041E3E0B17483AA4D2C1AEE6FEADD48EF448BF1B9E6C
Size	24,576 (bytes)
Type	64-bit EXE
Compilation Date	Sun Oct 14 10:43:19 2012(GMT+3)

Name	assistant.sys
MD5 Hash	EAEA9CCB40C82AF8F3867CD0F4DD5E9D
SHA-1 Hash	7C1B25518DEE1E30B5A6EAA1EA8E4A3780C24D0C
SHA-256 Hash	CF3A7D4285D65BF8688215407BCE1B51D7C6B22497F09021F0FCE31CBEB78986
Size	68,288 (bytes)
Type	64-bit EXE
Compilation Date	Sat May 31 05:18:53 2008 (GMT+3)

Name	agent.exe
MD5 Hash	F5F8160FE8468A77B6A495155C3DACEA
SHA-1 Hash	20D61C337653392EA472352931820DC60C37B2BC
SHA-256 Hash	44100C73C6E2529C591A10CD3668691D92DC0241152EC82A72C6E63DA299D3A2
Size	116,224 (bytes)
Type	64-bit EXE
Compilation Date	Sun Dec 29 08:56:27 2019 (GMT+3)

5. Tactical Recommendations:

Preventative measures:

1. Update all security devices, windows servers and workstations with the latest updates and signatures.
2. Ensure to block VirtualBox's driver (i.e. vboxdrv.sys) from loading on all systems. This can be done by blocking the following hash on EDR/Anti-virus system:
    - eaea9ccb40c82af8f3867cd0f4dd5e9d, The driver might be renamed so blocking the filename would not be sufficient.
3. Ensure that service accounts are not member of the Domain/Enterprise Admins or Administrators groups.
4. Ensure that privileged accounts are not shared or used as service accounts.
5. Implement privileged access workstations (PAWs) for personnel with administrative privileges. PAWs should be physically secured devices, and cannot be accessed remotely. Administrative tasks, such as Remote access, can only be performed from these workstations.
6. Implement Local Administrator Password Solution (LAPS) on servers and workstations to prevent Pass-the-hash attacks and other similar internal pivoting attacks, it mitigates the risk of lateral escalation that results when you have the same administrative local account and password combination on many computers.
7. Prevent the local admin account from authenticating over the network as specified in: https://blogs.technet.microsoft.com/secguide/2014/09/02/blocking-remote-use-of-local-accounts/
8. Validate the business requirements and the risks of allowing Virtual Private Network (SSL VPN) connections. By checking the following:
    - Ensure the access control list is properly configured (Unused accounts, default accounts, resigned users, test accounts.)
    - Block access for administrators and management users with high privileges.
    - Implement Multi-factor authentication (MFA) for allowed users.
    - Prevent access from countries known for malicious activities.
    - Implement a Lockout mechanism for failed attempts after a specific number e.g 3 failed attempts.
    - Enable access logging for successful and failed attempts.
    - Review (if enabled) the access logs for at least six months for any uncommon or suspicious logs. For example: unusual access times, unusual source locations, unusual users from different location, high number of failed attempts to logon, access from external VPN services or infrastructure of other compromised organizations.

- If the used SSLVPN solution is a product of Palo Alto, Pulse Secure or Fortinet, Perform a comprehensive review on logs, account usage and any configuration changes.

9. Reset passwords for all privileged accounts including domain administrators' accounts, network devices accounts.
10. Prevent workstation-to-workstation communications using host-based firewalls. The default policy should be to deny all incoming connection requests. Exceptions should be made for connections originating from domain controllers, patch management or configuration servers, systems belonging to helpdesk personnel, and other systems that have a business need to connect to workstations.
11. Disable and delete PowerShell V2, and Ensure to only allow PowerShell V5, or PowerShell V6 across the organization. If possible, prevent the execution of PowerShell on machines not requiring the usage of PowerShell.
12. Ensure that all privileged activities are logged, also ensure to enable Command-line Logging through: https://docs.microsoft.com/en-us/windows-server/identity/adds/manage/componentupdates/command-line-process-auditing
13. It's critical to review any generated data to detect active compromises
14. If MFA is not present, it is highly recommended to disable all Remote Access Services until it is properly implemented.
15. If SMS is used as an MFA method, ensure that users cannot change the assigned number without being reviewed and verified by an administrator.
16. Backup all Critical servers/data and make sure to have an offline backup isolated from the network and perform a restore test to validate the backups

Detective measures:

1. Review and validate any recently created accounts (normal users and high privileged accounts).
2. Review and validate all Domain/Enterprise Admin accounts and service accounts; all unused accounts should be disabled or deleted.
3. Perform Yara Rules scanning on all internet facing and critical servers.
4. Search of any new scheduled tasks on critical servers in which suspicious commands/activities are configured to run.
    - Enable PowerShell's module logging, script block logging, and transcription logging and review all generated logs.
    - Enable PowerShell constrained-mode for both interactive input and user-authorized scripts.
5. Review/scan for any usage of PSExec or any similar tool within the network in the last two months.
6. Increase monitoring activities looking for the following events:

- Usage of Guest and/or VMware accounts.
- Creation of high-privileged accounts.
- Unusual authentication activities on multiple servers by high-privileged accounts.
- Network devices logs (DNS, DHCP, Netflow and proxy).
- Newly registered schedule task :EventID:106
- usage of WinRM (if enabled):
    - EventID: 6 - Creating WSMAN Session
    - EventID: 169 - User Authentication: authenticated successfully
- Usage of RDP Authentication: EventID 21: Remote Desktop Services: Session logon succeeded
- Account logon: EventID: 4624 - An account was successfully logged on. possibly dangerous Logon types are:
    - 3 = Network,
    - 10 =Remote Interactive
- A specified logon attempt by particular account: EventID 4648 - A logon was attempted using explicit credentials. possibly dangerous Logon types are:
    - 3 = Network,
    - 10 =Remote Interactive
- If Sysmon is installed and deployed, review Sysmon log for EventID
    - 1 = Process Creation
    - 8 = CreateRemoteThread detected
- Deletion of Windows Event logs, Event ID 1102.

7. Ensure proper configuration of the Security Information Event Management (SIEM) or log analytics system so that events can be correlated from all devices on the network. This should also include VPN logs.
8. Ensure that alerts are generated for all suspicious activity on the network (e.g. port scanning, remote connections, using local administrator accounts over the network).
9. Configure all IIS logs, and other similar systems, to log the external IP addresses instead of the load balancer/proxy internal IP by enabling the X-forwarded function.
10. Export security related logs to secure offline location periodically.
11. Perform regular scanning for unauthorized software. This includes alerting when unrecognized binaries (executable files, DLL's and other libraries, etc.) are found on a system, even inside of compressed archives.
12. Ensure that all IOCs shared by NCSC and other security vendors have been consumed by the available security products.

Appendix

YARA Rules

Dustman.exe

```
import "pe"

rule dustman {
  meta:
    hash1 = "f07b0c79a8c88a5760847226af277cf34ab5508394a58820db4db5a8d0340fc7"
  strings:
    $x1 = "C:\\windows\\system32\\cmd.exe" fullword ascii
    $x2 = "C:\\Users\\Admin\\Desktop\\Dustman\\x64\\Release\\Dustman.pdb" fullword ascii
    $s3 = "AppPolicyGetProcessTerminationMethod" fullword ascii
    $s4 = "elrawdsk.sys" fullword wide
    $s5 =
"qppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppp
ppppppppppppppppppppppppppppqphppphpp" fullword ascii
    $s6 =
"Wppppppppppppppppqpppfppprppprpppsppprppptppphpppuppp}pppvpppypppwppp|pppxppp|pppyppp|pp
pzpppwppp{pppxppp|pppfppp}pppfppp" fullword ascii
    $s7 =
"ipppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppp
pppppppppppppppppppppppppppppppppOp0qppp" fullword ascii
    $s8 =
"!q0qpppyPppppppp\"q0qpppzPpppppp`\"q0qpppKPppppppP\"q0qpppqTpppppp@\"q0qpppyTpppppp0\"q0
qpppzTpppppp \"q0qpppKTpppppp" fullword ascii
    $s9 =
"qppppp{0x.pppp~ppppppppppppppppqppppppppppppppppppppppppppppppppppppppppppppppppppp
pppppppppppppppppppppppppppppppp0" fullword ascii
    $s10 =
"ppLSpppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppppp
pppppppppppppppppppppp8" fullword ascii
    $s11 =
"\"q0qpppqDppppppp#q0qpppyDpppppp`#q0qpppzDppppppP#q0qpppqHpppppp@#q0qpppzHpppppp0#q0q
pppqLpppppp #q0qpppzLpppppp" fullword ascii
    $s12 = "zpppppp" fullword ascii /* reversed goodware string 'ppppppz' */
    $s13 = "ppprppp`ppp}pppapppbpppbppprpppQppp}pppEppprppp1ppp}ppp3ppprppp
pppappp\"ppp}ppp#ppp}ppp'pppfppp)ppp{ppp" fullword ascii
    $s14 = "/c agent.exe A" fullword ascii
    $s15 =
"<q0qpppOtppppppp=q0qppp0tpppppp`=q0qppp1tppppppP=q0qppp3tpppppp@=q0qppp4tpppppp8=q0qppp
5tpppppp(=q0qppp6tpppppp" fullword ascii
    $s16 = "<pp[=ppq?pppeeeqeeereeeseeeteeeueeeveeeweeexeeeyeeezeee{eee|eee}eee~eee" fullword
ascii
    $s17 = "\\assistant.sys" fullword wide
    $s18 =
":q0qpppitppppppx;q0qpppjtpppppph;q0qpppktppppppX;q0qpppltppppppH;q0qpppmtpppppp8;q0qpppntpp
pppp(;q0qpppotpppppp" fullword ascii
```

```
    $s19 =
">q0qpppjxppppppx?q0qpppmxpppppppP?q0qppp\\xppppppp@?q0qpppKxppppppp8?q0qpppNxppppppp(?q0qpp
p3xppppppp" fullword ascii
    $s20 = ";q0qpppZtppppppppx<q0qppp[tppppppph<q0qppp\\tppppppX<q0qppp]tppppppp0<q0qppp_tppppppp
<q0qpppBtppppppp" fullword ascii
  condition:
    ( uint16(0) == 0x5a4d and
      filesize < 800KB and
      pe.imphash() == "47cb8a71a145ac31ea5df1b531c7fa09" and
      ( 1 of ($x*) or 4 of ($s*) )
    ) or ( all of them )
}
```

elrawdsk.sys

```
import "pe"

rule elrawdsk {
  meta:
    hash1 = "36a4e35abf2217887e97041e3e0b17483aa4d2c1aee6feadd48ef448bf1b9e6c"
  strings:
    $x1 = "c:\\projects\\rawdisk\\bin\\wnet\\fre\\amd64\\elrawdsk.pdb" fullword ascii
    $s2 = "elrawdsk.sys" fullword wide
    $s3 = "RawDisk Driver. Allows write access to files and raw disk sectors for user mode applications in
Windows 2000 and later." fullword wide
    $s4 = "\\DosDevices\\ElRawDisk" fullword wide
    $s5 = "Copyright (C) 2007-2012, EldoS Corporation " fullword wide
    $s6 = "IoGetDiskDeviceObject" fullword wide
    $s7 = "\\#{9A6DB7D2-FECF-41ff-9A92-6EDA696613DF}#" fullword wide
    $s8 = "\\#{8A6DB7D2-FECF-41ff-9A92-6EDA696613DE}#" fullword wide
    $s9 = "EldoS Corporation" fullword wide
    $s10 = "{25EC4453-AB06-4b3f-BCF0-B260A68B64C9}" fullword ascii
    $s11 = "\\Device\\ElRawDisk" fullword wide
    $s12 = "###ElRawDiskAMD64###" fullword ascii
  condition:
    ( uint16(0) == 0x5a4d and
      filesize < 70KB and
      pe.imphash() == "6863bacaac5428e1e55a107a613c0717" and
      ( 1 of ($x*) or 4 of ($s*) )
    ) or ( all of them )
}
```

assistant.sys

```
import "pe"

rule assistant {
  meta:
    hash1 = "cf3a7d4285d65bf8688215407bce1b51d7c6b22497f09021f0fce31cbeb78986"
  strings:
    $x1 = "C:\\vbox\\branch\\w64-
1.6\\out\\win.amd64\\release\\obj\\src\\VBox\\HostDrivers\\VBoxDrv\\VBoxDrv.pdb" fullword ascii
    $s2 = "C:\\vbox\\branch\\w64-1.6\\src\\VBox\\Runtime\\r0drv\\memobj-r0drv.cpp" fullword ascii
```

```
    $s3 = "VBoxDrv.sys" fullword ascii
    $s4 = "vboxdrv: Bad ioctl request header; cbIn=%#lx cbOut=%#lx fFlags=%#lx" fullword ascii
    $s5 = "SUP_IOCTL_COOKIE: Version mismatch. Requested: %#x  Min: %#x  Current: %#x" fullword ascii
    $s6 = "SUP_IOCTL_QUERY_FUNCS: Invalid input/output sizes. cbIn=%ld expected %ld. cbOut=%ld expected
%ld." fullword ascii
    $s7 = "SUP_IOCTL_PAGE_ALLOC: Invalid input/output sizes. cbIn=%ld expected %ld. cbOut=%ld expected
%ld." fullword ascii
    $s8 = "SUP_IOCTL_LOW_ALLOC: Invalid input/output sizes. cbIn=%ld expected %ld. cbOut=%ld expected
%ld." fullword ascii
    $s9 = "SUP_IOCTL_LDR_LOAD: Invalid input/output sizes. cbIn=%ld expected %ld. cbOut=%ld expected
%ld." fullword ascii
    $s10 = "SUP_IOCTL_PAGE_LOCK: Invalid input/output sizes. cbIn=%ld expected %ld." fullword ascii
    $s11 = "SUP_IOCTL_CALL_VMMR0: Invalid input/output sizes. cbIn=%ld expected %ld. cbOut=%ld expected
%ld." fullword ascii
    $s12 = "VBoxDrvLinuxIOCtl: too much output! %#x > %#x; uCmd=%#x!" fullword ascii
    $s13 = "supdrvLdrFree: Image '%s' has %d dangling objects!" fullword ascii
    $s14 = "SUP_IOCTL_PAGE_LOCK: Invalid input/output sizes. cbOut=%ld expected %ld." fullword ascii
    $s15 = "!supdrvCheckInvalidChar(pReq->u.In.szName, \";:()[]{}/\\\\\|&*%#@!~`\\\"'\")" fullword ascii
    $s16 = "\\DosDevices\\VBoxDrv" fullword wide
    $s17 = "SUP_IOCTL_LDR_GET_SYMBOL: %s" fullword ascii
    $s18 = "pReq->Hdr.cbIn <= SUP_IOCTL_PAGE_ALLOC_SIZE_IN" fullword ascii
    $s19 = "pReq->Hdr.cbIn <= SUP_IOCTL_LOW_ALLOC_SIZE_IN" fullword ascii
    $s20 = "SUP_IOCTL_LDR_LOAD: sym #%ld: unterminated name! (%#lx / %#lx)" fullword ascii
  condition:
    ( uint16(0) == 0x5a4d and
      filesize < 200KB and
      pe.imphash() == "b262e8d078ede007ebd0aa71b9152863" and pe.exports("AssertMsg1") and
pe.exports("RTAssertDoBreakpoint") and pe.exports("RTMpDoesCpuExist") and pe.exports("SUPR0ContAlloc")
and pe.exports("SUPR0ContFree") and pe.exports("SUPR0GipMap") and
      ( 1 of ($x*) or 4 of ($s*) )
    ) or ( all of them )
}
```

agent.exe

```
import "pe"

rule agent {
  meta:
    hash1 = "44100c73c6e2529c591a10cd3668691d92dc0241152ec82a72c6e63da299d3a2"
  strings:
    $x1 = "C:\\Users\\Admin\\Desktop\\Dustman\\Furutaka\\drv\\agent.plain.pdb" fullword ascii
    $s2 = "********** “The Political Statement” ********** " fullword ascii
    $s3 = "api-ms-win-core-synch-l1-2-0.dll" fullword wide
    $s4 = "AppPolicyGetProcessTerminationMethod" fullword ascii
    $s5 = "b4b615c28ccd059cf8ed1abf1c71fe03c0354522990af63adf3c911e2287a4b906d47d" fullword wide
    $s6 = "operator co_await" fullword ascii
    $s7 = "api-ms-win-appmodel-runtime-l1-1-2" fullword wide
    $s8 = "bad array new length" fullword ascii
    $s9 = ".CRT$XIAC" fullword ascii
    $s10 = ".?AVERDError@@" fullword ascii
    $s11 = ".?AVbad_array_new_length@std@@" fullword ascii
    $s12 = "\\\\?\\ElRawDisk" fullword wide
```

```
    $s13 = "api-ms-win-core-file-l1-2-2" fullword wide
    $s14 = ".CRT$XCL" fullword ascii
  condition:
    ( uint16(0) == 0x5a4d and
      filesize < 300KB and
      pe.imphash() == "75f159bf634600808810849f244592eb" and
      ( 1 of ($x*) or 4 of ($s*) )
    ) or ( all of them )
}
```