MrRobot Lab

Case Summary

This report documents a multi-stage compromise affecting an end-user workstation (Target1 / Front Desk), lateral movement to a second workstation (Target2 / Gideon), access to a Domain Controller (AD01), and a separate compromise of a POS system. Evidence is derived from Windows memory image analysis using Volatility and includes confirmed artifacts for initial access, malware execution, persistence, credential theft, internal reconnaissance, lateral movement, data staging, and POS malware activity. The analysis confirms two malware families: ExtremeRAT on Target1 and Dexter on the POS system.

Scope and Evidence Sources

The investigation is based on memory image analysis with Volatility. The memory profile was identified as Win7SP1x86_23418 with KDBG at 0x82765be8, supporting confidence that process, network, registry, and file artifacts were parsed correctly. Evidence types reviewed include Outlook process memory artifacts, extracted files via filescan/dumpfiles, process injection indicators via hollowfind/malfind, registry persistence artifacts via printkey, mutex/handle artifacts via handles, command history via consoles, and network sessions via netscan.

Attack Narrative and Findings

Initial Access (Target1 / Front Desk)

Initial access was achieved through phishing delivered to the user via Microsoft Outlook. In-memory email header artifacts from OUTLOOK.EXE indicate the sender address was th3wh1t3r0s3@gmail.com. The email carried an executable attachment named AnyConnectInstaller.exe, consistent with masquerading as a Cisco AnyConnect-related installer. The attachment was recovered and identified as ExtremeRAT (XTREMERAT). This establishes a clear user-targeted phishing vector delivering a remote access trojan.

Execution and Defense Evasion (Target1)

After delivery, the malware executed with stealth via process hollowing. Volatility hollowfind flagged iexplore.exe (PID 2996) with indicators consistent with a hollowed process, including mismatched base address information between memory regions and process structures. This strongly indicates the attacker used Internet Explorer as a host process to blend malicious code execution into a legitimate Windows process context.

Persistence (Target1)

Persistence on Target1 was established through a registry Run key. Registry artifacts show an entry named “MrRobot” under HKLM…\Run, indicating the malware (or a related launcher) was configured to start automatically at system boot or user logon. Additionally, a mutex named fsociety0.dat was found in the process handle list for the compromised context, consistent with malware single-instance control and providing a reliable indicator of compromise.

Post-Exploitation Tooling and Credential Access (Target1)

Following persistence, the attacker staged additional tools in the Windows temp area. Recovered artifacts confirm the presence of getlsasrvaddr.exe, nbtscan.exe, and wce.exe. Console history artifacts reveal the password flagadmin@1234, indicating that local administrative credentials for the Front Desk endpoint were obtained or exposed during attacker activity. The presence of wce.exe

and subsequent credential-related artifacts strongly support credential dumping from memory as part of the attacker's escalation and movement preparation.

Internal Reconnaissance (Target1)
Network reconnaissance was performed using nbtscan.exe, and results were written to a file named nbs.txt. The extracted results indicate the discovery of a critical internal host at 10.1.1.2 identified as AD01, consistent with a Domain Controller in the ALLSAFECYBERSEC domain context. A timestamp associated with nbtscan.exe is recorded as 2015-10-09 10:45:12 UTC, providing a temporal anchor for reconnaissance staging and supporting sequencing of attacker actions on Target1.

Command and Control and Additional Remote Access (Target1)
Network session artifacts confirm external connectivity consistent with command-and-control. A connection to 180.76.254.120 over port 22 was observed, which is consistent with C2 over a port commonly associated with SSH but frequently used by malware for blending and egress reliability. In addition to the RAT channel, TeamViewer.exe was found running on Target1, indicating the attacker deployed a legitimate remote administration tool as a secondary access mechanism that can persist independently of the original malware.

Lateral Movement (Target1 to Target2)
Evidence shows lateral movement was executed using Remote Desktop. Netscan artifacts indicate mstsc.exe initiated an RDP session to 10.1.1.21 over port 3389, which corresponds to the Target2 workstation (Gideon). This confirms internal pivoting using built-in Windows tooling rather than exclusively relying on the RAT's remote execution capabilities.

Credential Dumping and Persistence (Target2 / Gideon)
On Target2, credential dumping output was recovered from a file named w.tmp, which contains the password t76fRJhS. This indicates that once the attacker accessed Target2, they performed local credential extraction to broaden access and prepare for higher-value asset access. Persistence on Target2 was established through the Windows Task Scheduler. A task artifact (At1) was recovered that references execution of C:\Users\gideon\1.bat, indicating an automated re-entry mechanism on that host.

Domain Controller Access and Data Staging (Target2 to AD01)
Console artifacts show that the attacker accessed data via a mapped drive Z: and staged "crown jewels" into an encrypted RAR archive. The archive name is crownjewlez.rar and was created with a password 123qwe!@#, with three files added to the archive. This demonstrates confirmed access to sensitive data on the DC or a DC-accessible share and the attacker's intent to package and protect stolen content for exfiltration or later retrieval.

POS Compromise and Dexter Malware Activity
Separately, the POS system shows evidence of infection with the Dexter POS malware family. Memory analysis indicates a suspicious external connection to 54.84.237.92 over port 80 from iexplore.exe (PID 3208), consistent with HTTP-based command infrastructure or data transfer. The malware family is identified as Dexter via malfind and associated indicators described in the evidence set. Strings extracted from memory reference allsafe_protector.exe, suggesting the malware contains environment-specific configuration such as whitelisting or process targeting exclusions. Additional artifacts identify allsafe_update.exe as an execution vector on the POS host, consistent with a masqueraded updater used to introduce or launch the malware.

Confirmed Indicators of Compromise (IOCs)
The investigation confirms multiple high-confidence IOCs. Email sender used for delivery was th3wh1t3r0s3@gmail.com. The phishing attachment name was AnyConnectInstaller.exe and was identified as ExtremeRAT. The hollowed process on Target1 was iexplore.exe (PID 2996). Persistence on Target1 included HKLM Run value “MrRobot” and a mutex fsociety0.dat. Observed external endpoints include 180.76.254.120:22 (Target1 C2) and 54.84.237.92:80 (POS activity). Lateral movement used RDP from mstsc.exe to 10.1.1.21:3389. Target2 persistence referenced C:\Users\gideon\1.bat via a scheduled task, and data staging produced crownjewlez.rar with password 123qwe!@# containing three files. POS-related filenames include allsafe_update.exe and the configuration-related reference allsafe_protector.exe.

Impact Assessment

The compromise affected at least two user endpoints and extended to Domain Controller-accessible data. Credential theft is confirmed by recovered passwords from console history and WCE output artifacts, and lateral movement is confirmed by RDP session evidence. Data staging on the DC path is confirmed by creation of an encrypted archive containing three files, demonstrating probable theft of sensitive information. Separately, POS compromise with Dexter introduces potential risk of payment card data exposure, depending on the POS application stack and Dexter’s scraping targets.

Recommended Response Actions

Immediate containment should prioritize isolating Target1, Target2, and the POS host from the network and blocking the external IPs 180.76.254.120 and 54.84.237.92 at egress points and perimeter controls. Persistence removal should include deleting the “MrRobot” Run entry, removing the scheduled task referencing C:\Users\gideon\1.bat, and investigating/removing TeamViewer where unauthorized, including reviewing TeamViewer configuration for unattended access and resetting any associated credentials. Credential hygiene actions should include resetting exposed passwords and rotating domain credentials that may have been captured, with particular attention to local admin credentials on endpoints and any privileged AD accounts potentially accessed after discovery of AD01. The DC should be examined for access logs and evidence of file access corresponding to the three staged files, and the environment should be searched for crownjewlez.rar or copies of it, including any outbound transfer traces. For the POS environment, an incident response track should validate whether Dexter performed memory scraping of payment applications; this includes forensic acquisition, scope determination across POS fleet, and coordination with compliance and payment processor requirements if card data exposure is suspected.

Conclusion

The evidence confirms a targeted intrusion beginning with phishing on Target1, followed by ExtremeRAT execution using process hollowing, persistence via registry Run, credential theft with WCE tooling, internal reconnaissance identifying the Domain Controller, lateral movement via RDP to Target2, persistence via scheduled task on Target2, and data staging into an encrypted archive likely intended for exfiltration. In parallel, the POS host shows a separate compromise involving Dexter POS malware with external HTTP communication and environment-specific configuration references. The presence of both a RAT-based enterprise intrusion path and a POS malware path indicates either multiple objectives by the same actor or overlapping compromise activity and warrants full enterprise scoping and coordinated containment and eradication.

Event Timeline (end-to-end kill chain)

Step	System	What happened	Evidence / artifact
1	Target1 (Front Desk)	A phishing “security update” email was received	Email headers found in OUTLOOK.EXE memory (sender: th3wh1t3r0s3@gmail.com)
2	Target1	The email delivered attachment **AnyConnectInstaller.exe**	Outlook memory strings showing the attachment name
3	Target1	The attachment was extracted/identified as **XTREMERAT**	filescan/dumpfiles + family identification
4	Target1	RAT execution was hidden via **process hollowing** into **iexplore.exe (PID 2996)**	hollowfind indicators (VAD/PEB/base mismatch)
5	Target1	Persistence via registry **Run key “MrRobot”**	printkey on HKLM…Run
6	Target1	Single-instance/IOC via mutex **fsociety0.dat**	handles output for PID 2996
7	Target1	Post-exploitation tools dropped: **getlsasrvaddr.exe, nbtscan.exe, wce.exe** (3 total)	files found in temp via filescan/dumpfiles
8	Target1	Local admin password discovered: **flagadmin@1234**	console history (consoles/cmd output)
9	Target1	Network discovery activity timestamp anchor: **2015-10-09 10:45:12 UTC**	nbtscan.exe timestamp
10	Target1	nbtscan results saved to **nbs.txt**; first key host was **10.1.1.2 (AD01 / DC)**	nbs.txt extracted and reviewed
11	Target1	RAT C2 connection observed: **180.76.254.120:22**	netscan connection
12	Target1	Secondary remote access installed: **TeamViewer.exe**	pslist showing TeamViewer.exe
13	Target1 → Target2	Lateral movement via RDP: **mstsc.exe → 10.1.1.21:3389**	netscan RDP session evidence
14	Target2 (Gideon)	Credential dumping output revealed password **t76fRJhS**	w.tmp extracted (dumpfiles) containing the password
15	Target2 → DC	Data staged and encrypted: **crownjewlez.rar** created with password **123qwe!@#**, containing **3 files**	console commands/output showing RAR with password and 3 added files
16	Target2	Persistence via Scheduled Task pointing to **C:\Users\gideon\1.bat**	extracted task (At1) referencing 1.bat
17	POS	POS malware CNC/command traffic: **54.84.237.92:80** (iexplore.exe PID 3208)	netscan evidence
18	POS	POS malware family identified as **Dexter**	malfind + family identification
19	POS	Malware config/strings show a whitelist entry **allsafe_protector.exe**	strings from memory dump
20	POS	Initial execution vector on POS appears as **allsafe_update.exe**	iehistory / execution artifact

2) Malware and Tools Used (summary table)

Type	Name	Where observed	Purpose in the attack	Evidence / artifact
Malware (RAT)	**ExtremeRAT (XTREMERAT)**	Target1	Remote control, C2, staging post-exploit actions	AnyConnectInstaller.exe identified as XTREMERAT
Malware (POS)	**Dexter**	POS	POS data theft (card-stealing malware family)	malfind + Dexter identification
Tool (cred dump)	**wce.exe**	Target1/Target2	Extract credentials from memory	wce.exe present + w.tmp output includes password
Tool (recon)	**nbtscan.exe**	Target1	NetBIOS discovery / network mapping	nbtscan.exe timestamp + nbs.txt results
Tool (LSA helper)	**getlsasrvaddr.exe**	Target1	Assists credential theft by interacting with LSA	presence in temp directory set
Legit remote access	**TeamViewer.exe**	Target1	Backup remote access channel	pslist shows TeamViewer.exe
Built-in Windows	**mstsc.exe (RDP)**	Target1 → Target2	Lateral movement via RDP	netscan shows 10.1.1.21:3389 session
Utility	**rar**	Target2 → DC	Encrypt/stage stolen data for exfiltration	crownjewlez.rar + password + 3 files

---

3) What the Malware Did (behavior mapped to evidence)

Malware	Action	How it's proven	Key IOC / indicator
ExtremeRAT	Delivered via phishing attachment	Outlook email memory shows AnyConnectInstaller.exe	sender th3wh1t3r0s3@gmail.com; attachment name
ExtremeRAT	Stealth execution via process hollowing	hollowfind flags in iexplore.exe (PID 2996)	iexplore.exe PID 2996; VAD/PEB mismatch
ExtremeRAT	Persistence via Run key	registry Run value "MrRobot"	HKLM…Run: MrRobot
ExtremeRAT	Ensures only one instance runs	mutex discovered	fsociety0.dat
ExtremeRAT (as foothold)	Drops and runs post-exploit tools	tools present in temp	getlsasrvaddr.exe, nbtscan.exe, wce.exe
ExtremeRAT (as foothold)	Network reconnaissance	nbs.txt extracted and shows DC	10.1.1.2 (AD01)
ExtremeRAT	Maintains C2 communication	netscan shows external connection	180.76.254.120:22
Dexter	Communicates with CNC over HTTP	netscan shows external :80 endpoint	54.84.237.92:80
Dexter	Environment-aware config (whitelist)	memory strings show whitelist entry	allsafe_protector.exe
Dexter	Initial execution via	iehistory/execution artifact	allsafe_update.exe

Malware	Action	How it's proven	Key IOC / indicator
	fake updater		

Short Chronology

1. Target1 receives a phishing email from th3wh1t3r0s3@gmail.com containing AnyConnectInstaller.exe.
2. The attachment is ExtremeRAT, which runs stealthily via process hollowing inside iexplore.exe (PID 2996).
3. The RAT persists via HKLM Run "MrRobot" and uses mutex fsociety0.dat as an indicator/single-instance control.
4. The attacker drops tools (wce/nbtscan/getlsasrvaddr), discovers flagadmin@1234, and performs network discovery (nbs.txt).
5. RAT C2 traffic is observed to 180.76.254.120:22, and TeamViewer appears as an additional access method.
6. The attacker moves laterally via RDP to 10.1.1.21, dumps t76fRJhS on Target2, and adds persistence via a scheduled task pointing to 1.bat.
7. Data is staged and encrypted into crownjewlez.rar using password 123qwe!@# (3 files).
8. On POS, a separate infection involves Dexter, communicating to 54.84.237.92:80, launched via allsafe_update.exe, and referencing allsafe_protector.exe in its config/strings.

1. Machine:Target1 What email address tricked the front desk employee into installing a security update?

First, we determine the Windows version. Identifying this profile is crucial, as using the wrong profile can lead to misinterpretation of data and erroneous conclusions during our investigation. We also determine the KDBG value 0x82765be8, which will be useful for subsequent analysis commands. Once we determine the appropriate profile, we move on to examining the processes running at the time of the memory capture.

vol2 -f "/work/target1/Target1-1dd8701f.vmss" imageinfo

```
ruslan@pop-os:~/Downloads/88-Grrcon2015/temp_extract_dir$ vol2 -f "/work/target1/Target1-1dd8701f.vmss" imageinfo
Volatility Foundation Volatility Framework 2.6.1
INFO    : volatility.debug    : Determining profile based on KDBG search...
          Suggested Profile(s) : Win7SP1x86_23418, Win7SP0x86, Win7SP1x86_24000, Win7SP1x86
                     AS Layer1 : IA32PagedMemoryPae (Kernel AS)
                     AS Layer2 : VMWareAddressSpace (Unnamed AS)
                     AS Layer3 : FileAddressSpace (/work/target1/Target1-1dd8701f.vmss)
                      PAE type : PAE
                           DTB : 0x185000L
                          KDBG : 0x82765be8L
          Number of Processors : 2
     Image Type (Service Pack) : 0
                KPCR for CPU 0 : 0x82766c00L
                KPCR for CPU 1 : 0x807c5000L
             KUSER_SHARED_DATA : 0xffdf0000L
           Image date and time : 2015-10-09 12:53:02 UTC+0000
     Image local date and time : 2015-10-09 08:53:02 -0400
```

Win7SP1x86_23418, 0x82765be8

```
ruslan@pop-os:~/Downloads/88-Grrcon2015/temp_extract_dir$ vol2 -f "/work/target1/Target1-1dd8701f.vmss" --profile=Win7SP1x86_23418 -g 0x82765be8 pstree
Volatility Foundation Volatility Framework 2.6.1
Name                                                  Pid   PPid   Thds   Hnds Time
-------------------------------------------------- ------ ------ ------ ------ ----
 0x84ecbb18:csrss.exe                                 368    360      9    366 2015-10-09 11:30:47 UTC+0000
 0x84f97628:wininit.exe                               420    360      3     77 2015-10-09 11:30:48 UTC+0000
. 0x84e979f8:services.exe                             528    420      9    200 2015-10-09 11:30:48 UTC+0000
.. 0x85ae0cb0:dllhost.exe                            1888    528     13    196 2015-10-09 11:30:54 UTC+0000
.. 0x8586fd40:svchost.exe                             644    528     11    351 2015-10-09 11:30:48 UTC+0000
.. 0x85ae3030:vmtoolsd.exe                           1432    528      8    274 2015-10-09 11:30:54 UTC+0000
.. 0x85935030:svchost.exe                             796    528     19    446 2015-10-09 11:30:51 UTC+0000
.. 0x85d01510:svchost.exe                            3232    528      9    131 2015-10-09 11:31:34 UTC+0000
.. 0x858b69e8:msdtc.exe                              1980    528     12    145 2015-10-09 11:30:55 UTC+0000
.. 0x85978940:svchost.exe                             864    528     30   1036 2015-10-09 11:30:52 UTC+0000
.. 0x85969030:svchost.exe                             836    528     17    405 2015-10-09 11:30:52 UTC+0000
... 0x85c09968:dwm.exe                               2088    836      3     93 2015-10-09 11:31:04 UTC+0000
.. 0x85c39030:taskhost.exe                           2252    528      7    150 2015-10-09 11:31:04 UTC+0000
.. 0x8582c8d8:spoolsv.exe                            1228    528     12    273 2015-10-09 11:30:53 UTC+0000
.. 0x84e01448:svchost.exe                             720    528      6    276 2015-10-09 11:30:50 UTC+0000
.. 0x85a138f0:svchost.exe                            1124    528     16    484 2015-10-09 11:30:53 UTC+0000
.. 0x85a55d40:svchost.exe                            1256    528     17    304 2015-10-09 11:30:53 UTC+0000
.. 0x85b43a58:sppsvc.exe                             3900    528      4    153 2015-10-09 11:32:54 UTC+0000
.. 0x859cc2c0:svchost.exe                            1008    528     13    650 2015-10-09 11:30:52 UTC+0000
.. 0x8598c920:SearchIndexer.                         2544    528     13    670 2015-10-09 11:31:10 UTC+0000
.. 0x85976318:svchost.exe                            1784    528      5     99 2015-10-09 11:30:54 UTC+0000
. 0x8583b030:lsass.exe                                536    420      9    851 2015-10-09 11:30:48 UTC+0000
. 0x8583d960:lsm.exe                                  544    420     10    163 2015-10-09 11:30:48 UTC+0000
 0x83d334e8:System                                      4      0     94    500 2015-10-09 11:30:44 UTC+0000
. 0x84edcbf0:smss.exe                                 276      4      2     30 2015-10-09 11:30:44 UTC+0000
 0x84013598:TeamViewer.exe                           2680   1696     28    632 2015-10-09 12:08:46 UTC+0000
. 0x858bc278:TeamViewer_Des                          1092   2680     16    405 2015-10-09 12:10:56 UTC+0000
. 0x84017d40:tv_w32.exe                              4064   2680      2     83 2015-10-09 12:08:47 UTC+0000
 0x85c1e5f8:explorer.exe                             2116   2060     23    912 2015-10-09 11:31:04 UTC+0000
. 0x83eb5d40:cmd.exe                                 2496   2116      1     22 2015-10-09 11:33:42 UTC+0000
. 0x83f1ed40:mstsc.exe                               2844   2116     11    484 2015-10-09 12:12:03 UTC+0000
. 0x83fb86a8:cmd.exe                                 3064   2116      1     22 2015-10-09 11:37:32 UTC+0000
. 0x859281f0:vmtoolsd.exe                            2388   2116      7    164 2015-10-09 11:31:04 UTC+0000
. 0x85cd3d40:OUTLOOK.EXE                             3196   2116     22   1678 2015-10-09 11:31:32 UTC+0000
 0x855f6d40:csrss.exe                                 432    412     11    366 2015-10-09 11:30:48 UTC+0000
. 0x83f13d40:conhost.exe                             1624    432      3     81 2015-10-09 11:35:15 UTC+0000
. 0x83fa9030:conhost.exe                              676    432      3     83 2015-10-09 11:37:32 UTC+0000
. 0x83e5cd40:conhost.exe                              916    432      3     83 2015-10-09 11:33:42 UTC+0000
. 0x83fc7c08:conhost.exe                             1824    432      3     85 2015-10-09 11:39:22 UTC+0000
 0x8561d030:winlogon.exe                              480    412      3    115 2015-10-09 11:30:48 UTC+0000
 0x85d0d030:iexplore.exe                             2996   2984      6    463 2015-10-09 11:31:27 UTC+0000
. 0x83f105f0:cmd.exe                                 1856   2996      1     33 2015-10-09 11:35:15 UTC+0000
 0x83fb2d40:cmd.exe                                  3784   2196      1     24 2015-10-09 11:39:22 UTC+0000
```

Email clients such as Microsoft Outlook are common vectors for social engineering attacks, as attackers often use specially crafted emails to trick users into executing malicious code. The presence of OUTLOOK.EXE in the process list is relevant to our investigation, as the incident report mentions a suspicious email related to a security update. The process list shows OUTLOOK.EXE with PID 3196 running in a parent process with a timestamp suggesting the email client was active around the time of the incident. We can use the memdump plugin to extract the memory of the OUTLOOK.EXE process. The memdump command with the process ID parameter allows us to dump the memory of only this specific process. This targeted approach focuses our analysis on the email client that likely received the suspicious message. The memory dump is saved to a file named after the process ID,

creating the file 3196.dmp, which contains potential evidence. After extracting the process memory, we search the email headers to determine the source of the suspicious email.

```
ruslan@pop-os:~/Downloads/88-Grrcon2015/temp_extract_dir$ vol2 -f "/work/target1/Target1-1dd8701f.vmss" --profile=Win7SP1x86_23418 -g 0x82765be8 memdump --pid=3196 -D /work/3196
Volatility Foundation Volatility Framework 2.6.1
************************************************************************
Writing OUTLOOK.EXE [  3196] to 3196.dmp
ruslan@pop-os:~/Downloads/88-Grrcon2015/temp_extract_dir$ ls -lah 3196
strings 3196/3196.dmp | grep -F "From: "
total 226M
drwxrwxrwx 2 ruslan          ruslan  4.0K Jan 21 22:51 .
drwxrwxr-x 6 ruslan          ruslan  4.0K Jan 21 22:50 ..
-rw-r--r-- 1 systemd-resolve nogroup 226M Jan 21 22:51 3196.dmp
From: The Whit3R0s3 <th3wh1t3r0s3@gmail.com>
```

Answer: th3wh1t3r0s3@gmail.com

2. Machine:Target1 What is the filename that was delivered in the email?

vol2y -f "/work/target1/Target1-1dd8701f.vmss" --profile=Win7SP0x86 yarascan --yara-rules=".exe" --wide -p 3196

```
Owner: Process OUTLOOK.EXE Pid 3196
0x085e11e2  2e 65 78 65 22 3e 68 74 74 70 3a 2f 2f 31 38 30   .exe">http://180
0x085e11f2  2e 37 36 2e 32 35 34 2e 31 32 30 2f 41 6e 79 43   .76.254.120/AnyC
0x085e1202  6f 6e 6e 65 63 74 49 6e 73 74 61 6c 6c 65 72 2e   onnectInstaller.
0x085e1212  65 78 65 3c 2f 61 3e 3c 2f 64 69 76 3e 3c 64 69   exe</a></div><di
0x085e1222  76 3e 3c 62 72 3e 3c 2f 64 69 76 3e 3c 64 69 76   v><br></div><div
0x085e1232  3e 49 66 20 79 6f 75 20 68 61 76 65 20 61 6e 79   >If.you.have.any
0x085e1242  20 71 75 65 73 74 69 6f 6e 73 20 70 6c 65 61 73   .questions.pleas
0x085e1252  65 20 64 6f 6e 27 74 20 68 65 73 69 74 61 74 65   e.don't.hesitate
0x085e1262  20 74 6f 20 63 6f 6e 74 61 63 74 20 49 54 20 73   .to.contact.IT.s
0x085e1272  75 70 70 6f 72 74 2e 3c 2f 64 69 76 3e 3c 64 69   upport.</div><di
0x085e1282  76 3e 3c 62 72 3e 3c 2f 64 69 76 3e 3c 64 69 76   v><br></div><div
0x085e1292  3e 54 68 61 6e 6b 73 20 61 6e 64 20 68 61 76 65   >Thanks.and.have
0x085e12a2  20 61 20 67 72 65 61 74 20 64 61 79 21 3c 2f 64   .a.great.day!</d
0x085e12b2  69 76 3e 3c 64 69 76 3e 41 6c 6c 53 61 66 65 20   iv><div>AllSafe.
0x085e12c2  49 54 20 53 75 70 70 6f 72 74 20 44 65 73 6b 3c   IT.Support.Desk<
0x085e12d2  2f 64 69 76 3e 3c 2f 64 69 76 3e 0d 0a 03 00 00   /div></div>.....
```

or we can use: strings 3196/3196.dmp | grep -i '\.exe'

```
ruslan@pop-os:~/Downloads/88-Grrcon2015/temp_extract_dir$ strings 3196/3196.dmp | grep -i '\.exe'
C:\Program Files\Microsoft Office\Office15\OUTLOOK.EXE
ComSpec=C:\Windows\system32\cmd.exe
PATHEXT=.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC
ComSpec=C:\Windows\system32\cmd.exe
ComSpec=C:\Windows\system32\cmd.exe
PATHEXT=.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC
"C:\Program Files\Microsoft Office\Office15\OUTLOOK.EXE"
C:\PROGRA~1\MICROS~1\Office15\OUTLOOK.EXE
am Files\Microsoft Office\Office15\OUTLOOK.EXE
        C:\Program Files\Microsoft Office\Office15\OUTLOOK.EXE
OUTLOOK.EXE
C:\Program Files\Microsoft Office\Office15\OUTLOOK.EXE
ComSpec=C:\Windows\system32\cmd.exe
PATHEXT=.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC
2015/10/09 08:08:47.466  2680  2416 H32  teamviewer.exe: SharedMem Connected (seg = 0x2b30000, refcnt = 2)
2015/10/09 08:11:15.885  2116  2120 H32  explorer.exe: ResumeAllThreads: resumed 30 threads, max count 30
2015/10/09 08:11:15.885  2116  2120 H32  explorer.exe: DragInterceptor: interception successful (new interface)
infopath.exe
excel.exe
OUTLOOK.EXE
outlook.exe
outlook.exe
<meta http-equiv="Content-Type" content="text/html; charset=utf-8"><div dir="ltr">Hello Mr. Wellick,<div><br></div><div>In order to provide the best service, in the most secure manner, AllSa
fe has recently updated our remote VPN software. Please download the update from the link below.</div><div><br></div><div><a href="http://180.76.254.120/AnyConnectInstaller.exe">http://180.7
6.254.120/AnyConnectInstaller.exe</a></div><div><br></div><div>If you have any questions please don't hesitate to contact IT support.</div><div><br></div><div>Thanks and have a great day!</d
iv><div>AllSafe IT Support Desk</div></div>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8"><div dir="ltr">Hello Mr. Wellick,<div><br></div><div>In order to provide the best service, in the most secure manner, AllSa
fe has recently updated our remote VPN software. Please download the update from the link below.</div><div><br></div><div><a href="http://180.76.254.120/AnyConnectInstaller.exe">http://180.7
6.254.120/AnyConnectInstaller.exe</a></div><div><br></div><div>If you have any questions please don't hesitate to contact IT support.</div><div><br></div><div>Thanks and have a great day!</d
iv><div>AllSafe IT Support Desk</div></div>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8"><div dir="ltr">Hello Mr. Wellick,<div><br></div><div>In order to provide the best service, in the most secure manner, AllSa
fe has recently updated our remote VPN software. Please download the update from the link below.</div><div><br></div><div><a href="http://180.76.254.120/AnyConnectInstaller.exe">http://180.7
6.254.120/AnyConnectInstaller.exe</a></div><div><br></div><div>If you have any questions please don't hesitate to contact IT support.</div><div><br></div><div>Thanks and have a great day!</d
iv><div>AllSafe IT Support Desk</div></div>
```

Answer: AnyConnectInstaller.exe

3. Machine:Target1 What is the name of the rat's family used by the attacker?

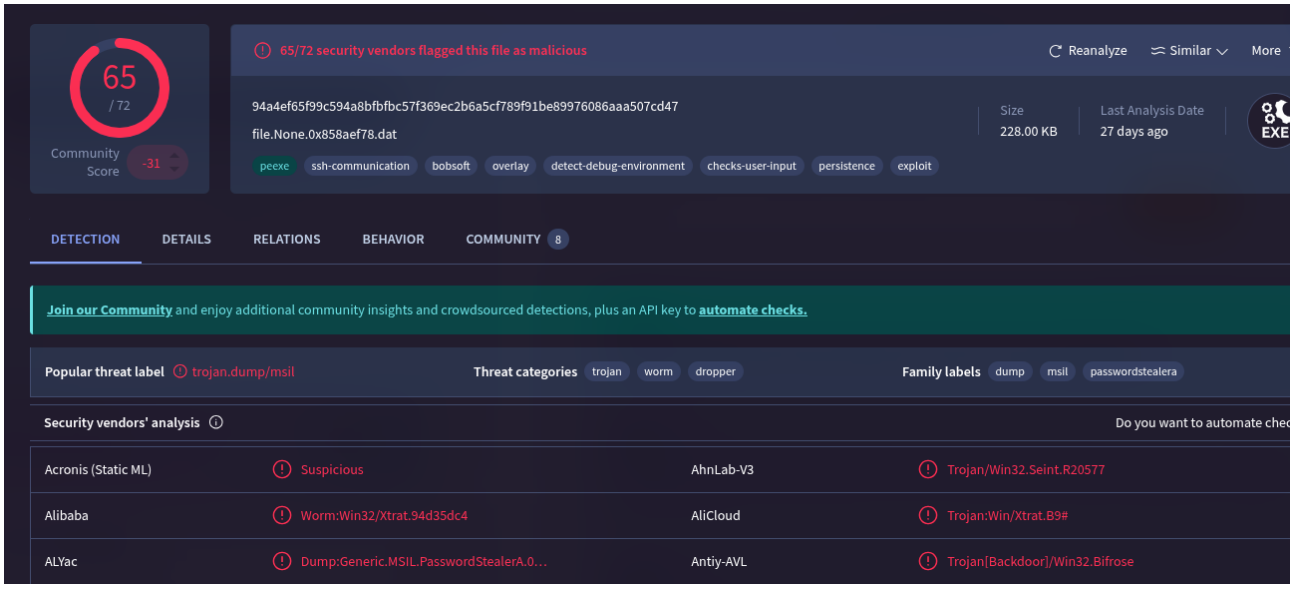
vol2 -f "/work/target1/Target1-1dd8701f.vmss" --profile=Win7SP0x86 filescan | grep "AnyConnectInstaller.exe"

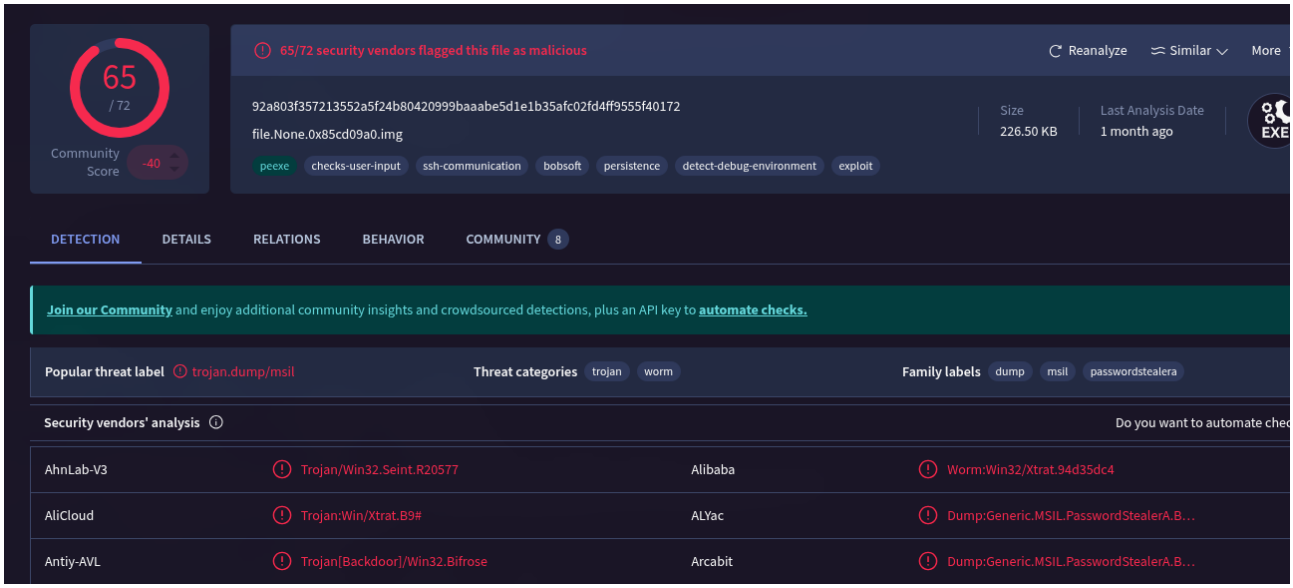
```
ruslan@pop-os:~/Downloads/88-Grrcon2015/temp_extract_dir$ vol2 -f "/work/target1/Target1-1dd8701f.vmss" --profile=Win7SP0x86 filescan | grep "AnyConnectInstaller.exe"
0x000000003df12dd0      2      0 RW-rwd \Device\HarddiskVolume2\Users\anyconnect\AnyConnect\AnyConnectInstaller.exe
0x000000003df1cf00      4      0 R--r-d \Device\HarddiskVolume2\Users\anyconnect\AnyConnect\AnyConnectInstaller.exe
0x000000003e0bc5e0      7      0 R--r-d \Device\HarddiskVolume2\Users\frontdesk\Downloads\AnyConnectInstaller.exe
0x000000003e2559b0      8      0 R--rwd \Device\HarddiskVolume2\Users\frontdesk\Downloads\AnyConnectInstaller.exe
0x000000003e2ae8e0      8      0 RWD--- \Device\HarddiskVolume2\Users\anyconnect\AnyConnect\AnyConnectInstaller.exe
0x000000003ed57968      4      0 R--r-d \Device\HarddiskVolume2\Users\frontdesk\Downloads\AnyConnectInstaller.exe
```

vol2 -f "/work/target1/Target1-1dd8701f.vmss" --profile=Win7SP0x86 dumpfiles -Q 0x000000003e0bc5e0 -D /work/dumped

```
ruslan@pop-os:~/Downloads/88-Grrcon2015/temp_extract_dir$ chmod 777 dumped
ruslan@pop-os:~/Downloads/88-Grrcon2015/temp_extract_dir$ vol2 -f "/work/target1/Target1-1dd8701f.vmss" --profile=Win7SP0x86 dumpfiles   -Q 0x000000003e0bc5e0 -D /work/dumped
\\Volatility Foundation Volatility Framework 2.6.1
ImageSectionObject 0x3e0bc5e0   None   \Device\HarddiskVolume2\Users\frontdesk\Downloads\AnyConnectInstaller.exe
DataSectionObject 0x3e0bc5e0   None   \Device\HarddiskVolume2\Users\frontdesk\Downloads\AnyConnectInstaller.exe
ruslan@pop-os:~/Downloads/88-Grrcon2015/temp_extract_dir$
```

```
ruslan@pop-os:~/Downloads/88-Grrcon2015/temp_extract_dir$ md5sum dumped/*
23a9329505c6eb16840901524ca7bdc9  dumped/file.None.0x858aef78.dat
165a952830dbd91509c48a3275edc379  dumped/file.None.0x85cd09a0.img
ruslan@pop-os:~/Downloads/88-Grrcon2015/temp_extract_dir$
```





Answer: XTREMERAT

4. Machine:Target1 The malware appears to be leveraging process injection. What is the PID of the process that is injected?

After identifying a malicious executable and determining its affiliation with the Extreme RAT family, we need to further analyze how this malware operates on the system. One common technique used by sophisticated malware is process injection, which allows malicious code to execute in the address space of a legitimate process. This method helps malware evade detection by security tools, which primarily focus on suspicious executables rather than on abnormal behavior in legitimate processes. Process injection encompasses various methods that allow attackers to inject and execute their code in the address space of another process. By injecting malicious code into legitimate processes, attackers can bypass security controls, hide their presence, and gain the same privileges as the host process. This method is effective because malicious activity appears to originate from a trusted process, making it more difficult for security tools that rely on the process's reputation to detect. The hollowfind plugin is designed specifically to detect process hollowing, a specific type of this technique. Process hollowing occurs when malware creates a legitimate process in a suspended state, replaces its memory with malicious code, and then resumes execution. This technique is insidious because the process appears legitimate from the outside, but executes completely different code than expected.

vol2hollowfind -f "/work/target1/Target1-1dd8701f.vmss" --profile=Win7SP0x86 hollowfind

```
Hollowed Process Information:
        Process: iexplore.exe PID: 2996
        Parent Process: NA PPID: 2984
        Creation Time: 2015-10-09 11:31:27 UTC+0000
        Process Base Name(PEB): iexplore.exe
        Command Line(PEB): "C:\Program Files\Internet Explorer\iexplore.exe"
        Hollow Type: Process Base Address and Memory Protection Discrepancy

VAD and PEB Comparison:
        Base Address(VAD): 0x12d0000
        Process Path(VAD): \Program Files\Internet Explorer\iexplore.exe
        Vad Protection: PAGE_EXECUTE_WRITECOPY
        Vad Tag: Vad

        Base Address(PEB): 0x13400000
        Process Path(PEB): C:\Program Files\Internet Explorer\iexplore.exe
        Memory Protection: PAGE_READWRITE
        Memory Tag: VadS

0x13400000  4d 5a 50 00 02 00 00 00 04 00 0f 00 ff ff 00 00   MZP.............
0x13400010  b8 00 00 00 00 00 00 00 40 00 1a 00 00 00 00 00   ........@.......
0x13400020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0x13400030  00 00 00 00 00 00 00 00 00 00 00 00 01 00 00 00   ................

Similar Processes:
        iexplore.exe(2996) Parent:NA(2984) Start:2015-10-09 11:31:27 UTC+0000
```

iexplore.exe (PID 2996) явный признак process hollowing: Hollow Type: Process Base Address and Memory Protection Discrepancy. "Hollow Type: Process Base Address and Memory Protection Discrepancy" прямо говорит в несоответсвие. Так же VAD Base ≠ PEB Base (0x12d0000 ≠ 0x13400000)

The plugin compares the expected contents of virtual address descriptor (VAD) entries with the contents of the process environment block (PEB), highlighting inconsistencies that indicate tampering. Process intrusion is a complex technique involving several stages. First, the malware creates a legitimate process (in this case, Internet Explorer) in a suspended state. Then, it deletes the source code from the process's address space. Afterward, the malware allocates new memory within the suspended process and writes its own malicious code to this newly allocated space. Finally, it adjusts the suspended process's entry point to point to the malicious code and resumes execution. When the process starts, it appears to be Internet Explorer in the process list, but in reality, it executes malicious code. This method offers attackers several advantages. It allows malware to run in the context of a legitimate process, potentially inheriting its privileges and trust level. It evades detection by basic security tools that only check process names or paths. Furthermore, it can bypass application checklists that allow Internet Explorer to run but block unknown executables.

Therefore, based on our analysis, we can determine that the PID of the process into which the malicious code was injected is 2996. This process disguises itself as Internet Explorer (iexplore.exe), but has been modified and filled with Extreme RAT code.

Answer: 2996

5. Machine:Target1 What is the unique value the malware is using to maintain persistence after reboot?

After identifying the process injection method used by the Extreme RAT malware, it is necessary to determine how the malware maintains persistent presence on the infected system. Persistence is a critical capability of

malware, allowing it to survive a system reboot and continue operating without the user having to re-run the initial infection vector. Without a persistence mechanism, the malware will be deleted upon system reboot, significantly limiting its effectiveness as an attack tool. Malware authors use various persistence techniques to ensure their malicious code runs automatically after a system reboot. Common methods include modifying Windows registry keys such as Run and RunOnce, creating scheduled tasks, installing services, modifying the startup folder, using WMI event subscriptions, creating browser helper objects, or using DLL hijacking techniques. Each of these methods serves a single purpose: to ensure that the malware runs automatically at system startup without any user interaction.

The Windows registry is one of the most commonly used locations for persistence due to its role in system configuration and startup processes. In particular, the Run keys located in the HKEY_CURRENT_USER and HKEY_LOCAL_MACHINE keys are often misused, as any executable files listed in these keys are automatically launched at user logon or system boot, respectively. To investigate potential registry-based data persistence mechanisms, we can use Volatility's printkey plugin to examine the Windows registry stored in memory. The printkey plugin allows you to view the contents of specific registry keys, including their values and data. We'll focus on the Run key in HKEY_LOCAL_MACHINE\Software, as it's a common location for system-wide data persistence mechanisms.

vol2 -f "/work/target1/Target1-1dd8701f.vmss" --profile=Win7SP0x86 printkey -K "MICROSOFT\WINDOWS\CURRENTVERSION\RUN"

Answer: MrRobot

6. Machine:Target1 Malware often uses a unique value or name to ensure that only one copy runs on the system. What is the unique name the malware is using?

Having discovered the persistence mechanism used by the Extreme RAT malware, we need to examine another common malware characteristic: the use of mutexes to ensure that only one instance of the malware can run on the system at any given time. This is a common trait among malware authors, as running multiple instances simultaneously can lead to resource conflicts, system instability, or increased visibility for security monitoring tools. The malware typically ensures that only one instance runs by creating a mutex (mutual exclusion object) with a unique name at startup. A mutex is a synchronization primitive in Windows that can only be owned by one process at a time. Upon startup, the malware attempts to create or open a mutex with its unique identifier. If the mutex already exists (indicating that another instance is running), the new instance typically terminates to avoid running multiple copies. These mutex names are often hardcoded strings within the malware and can serve as valuable indicators of compromise. They are often unique to specific malware families or even campaigns, making them useful for identification and attribution. By examining mutexes created by suspicious processes, we can gain insight into the malware's identity and operational characteristics. To examine mutexes associated with our infected process, we can use Volatility's handles plugin. Since we previously determined that process ID 2996 (the closed Internet Explorer process) was compromised, we specifically look for mutex objects created by this process. The handles plugin displays open handles (references to system resources) for a given process, including files, registry keys, events, and mutexes.

vol2 -f "/work/target1/Target1-1dd8701f.vmss" --profile=Win7SP0x86 handles -t Mutant -p 2996

```
ruslan@pop-os:~/Downloads/88-Grrcon2015/temp_extract_dir$ vol2 -f "/work/target1/Target1-1dd8701f.vmss" --profile=Win7SP0x86 handles -t Mutant -p 2996
Volatility Foundation Volatility Framework 2.6.1
Offset(V)     Pid     Handle     Access Type             Details
---------- ------ ---------- ---------- ---------------- -------
0x85c80238   2996       0x18   0x1f0001 Mutant
0x8560f0c0   2996       0xa4   0x100000 Mutant           RasPbFile
0x85d1be20   2996       0xe4   0x1f0001 Mutant
0x85d1bd90   2996       0xec   0x1f0001 Mutant
0x85d11500   2996      0x118   0x1f0001 Mutant
0x85d118d0   2996      0x124   0x1f0001 Mutant
0x85d1b0f0   2996      0x14c   0x1f0001 Mutant
0x85d11700   2996      0x150   0x1f0001 Mutant           fsociety0.dat
0x85c76b80   2996      0x36c   0x1f0001 Mutant           ZonesCounterMutex
0x85c73da8   2996      0x3ac   0x1f0001 Mutant           ZoneAttributeCacheCounterMutex
0x85c81270   2996      0x3b4   0x1f0001 Mutant           ZonesCacheCounterMutex
0x85c73da8   2996      0x3b8   0x1f0001 Mutant           ZoneAttributeCacheCounterMutex
0x85928fe0   2996      0x3bc   0x1f0001 Mutant           ZonesLockedCacheCounterMutex
0x83e99318   2996      0x588   0x1f0001 Mutant
0x83fc4450   2996      0x5b4   0x1f0001 Mutant           TeamViewerHooks_LogBuffer
0x84016860   2996      0x5b8   0x1f0001 Mutant           TeamViewerHooks_Mutex4
0x84009200   2996      0x5bc   0x1f0001 Mutant           TeamViewerHooks_Mutex1
0x8402ca90   2996      0x5c4   0x1f0001 Mutant           TeamViewerHooks_Mutex5
0x84015b98   2996      0x5d4   0x1f0001 Mutant           TeamViewerHooks_DynamicMemMutex
0x84015b38   2996      0x5d8   0x1f0001 Mutant           TeamViewerHooks_DirectXBufferMutex
```

The output shows several mutex handles belonging to this process, providing valuable information about the malware's operation. The handle output shows numerous mutexes with various system names, such as ZonesCacheCounterMutex and TeamViewerHooks_Mutex4. Many of these appear to be legitimate mutexes used by regular applications or Windows components. However, one mutex stands out as particularly unusual and likely associated with the malware: fsociety0.dat.

Answer: fsociety0.dat

7. Machine:Target1 It appears that a notorious hacker compromised this box before our current attackers. Name the movie he or she is from.

vol2 -f "/work/target1/Target1-1dd8701f.vmss" --profile=Win7SP0x86 filescan | grep -oP '(?<=\\Users\\)[^\\]+(?=\\|$)' | sort | uniq

```
ruslan@pop-os:~/Downloads/88-Grrcon2015/temp_extract_dir$ vol2 -f "/work/target1/Target1-1dd8701f.vmss" --profile=Win7SP0x86 filescan | grep -oP '(?<=\\Users\\)[^\\]+(?=\\|$)' | sort | uniq
Administrator.front-desk-PC
anyconnect
desktop.ini
FRONTD-1
front-desk
frontdesk
frontdesk
gideon
Public
zerocool
```

Answer: Hacker

8. Machine:Target1 What is the NTLM password hash for the administrator account?

vol2 -f "/work/target1/Target1-1dd8701f.vmss" --profile=Win7SP0x86 hashdump

```
zerocool
ruslan@pop-os:~/Downloads/88-Grrcon2015/temp_extract_dir$ vol2 -f "/work/target1/Target1-1dd8701f.vmss" --profile=Win7SP0x86 hashdump
Volatility Foundation Volatility Framework 2.6.1
Administrator:500:aad3b435b51404eeaad3b435b51404ee:79402b7671c317877b8b954b3311fa82:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
front-desk:1000:aad3b435b51404eeaad3b435b51404ee:2ae4c526659523d58350e4d70107fc11:::
```

Administrator is the username
500 is the SID, which is always 500 for the built-in Administrator account

aad3b435b51404eeaad3b435b51404ee is the LM hash (this specific value indicates that LM hashing is disabled)

```
zeroc00l
ruslan@pop-os:~/Downloads/88-Grrcon2015/temp_extract_dir$ vol2 -f "/work/target1/Target1-1dd8701f.vmss" --profile=Win7SP0x86 hashdump
Volatility Foundation Volatility Framework 2.6.1
Administrator:500:aad3b435b51404eeaad3b435b51404ee:79402b7671c317877b8b954b3311fa82:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
front-desk:1000:aad3b435b51404eeaad3b435b51404ee:2ae4c526659523d58350e4d70107fc11:::
```

79402b7671c317877b8b954b3311fa82 is the NTLM hash of the Administrator password

Answer: 79402b7671c317877b8b954b3311fa82

9. Machine:Target1 The attackers appear to have moved over some tools to the compromised front desk host. How many tools did the attacker move?

After examining user credentials and system access levels, it's time to investigate the specific actions the attackers took after gaining access. Understanding the tools used by attackers provides valuable insight into their goals, technical capabilities, and the potential impact of a breach. Experienced attackers often use their own toolsets to maintain a persistent presence on a system, navigate the network, and achieve their goals. When analyzing a compromised system, they often check for suspicious files in the Windows temporary directory (C:\Windows\Temp). Attackers often use this directory to temporarily store their tools because it is accessible to all users and is typically excluded from routine security scans. Furthermore, files in temporary directories may not attract as much attention as files located in more visible locations. By examining the command history and file system artifacts obtained using the console plugin, we see evidence that someone has accessed the Windows temporary directory and viewed its contents. The directory listing reveals several suspicious executable files that appear to be non-legitimate Windows components. These files are distinguished by their names, sizes, and the fact that they are executable files located in a location typically reserved for temporary system files.

vol2 -f "/work/target1/Target1-1dd8701f.vmss" --profile=Win7SP0x86 filescan | grep '\\Windows\\Temp\\'

```
ruslan@pop-os:~/Downloads/88-Grrcon2015/temp_extract_dir$ vol2 -f "/work/target1/Target1-1dd8701f.vmss" --profile=Win7SP0x86 filescan | grep '\\Windows\\Temp\\'
0x000000003df31038      8      0 R--r-- \Device\HarddiskVolume2\Windows\Temp\wce.exe
0x000000003e1eee10      7      0 R--r-d \Device\HarddiskVolume2\Windows\Temp\getlsasrvaddr.exe
0x000000003e25eca8      5      0 R--r-d \Device\HarddiskVolume2\Windows\Temp\wce.exe
0x000000003eca37f8      8      0 -W-r-- \Device\HarddiskVolume2\Windows\Temp\w.tmp
0x000000003fa633f0      1      0 R--rw- \Device\HarddiskVolume2\Windows\Temp\Rar.exe
0x000000003fc3fb80      6      0 R--r-d \Device\HarddiskVolume2\Windows\Temp\nbtscan.exe
0x000000003fc5af80      7      0 R--r-d \Device\HarddiskVolume2\Windows\Temp\Rar.exe
0x000000003fcaa598      8      0 -W-rw- \Device\HarddiskVolume2\Windows\Temp\MpCmdRun.log
0x000000003fdb7808      8      0 -W-r-- \Device\HarddiskVolume2\Windows\Temp\nbs.txt
0x000000003fdd4ca0      7      0 R--r-- \Device\HarddiskVolume2\Windows\Temp\nbtscan.exe
```

Among the files in the temporary directory, several suspicious executables can be found, which appear to be attacker tools:

getlsasrvaddr.exe (50,176 bytes) - This filename indicates a tool designed to retrieve the address of the Local Security Authority Subsystem Service (LSASS) in memory. LSASS is a Windows process that handles authentication and stores credential information, making it a common target for credential theft.

nbtscan.exe (36,864 bytes) - NBTScan is a legitimate network discovery tool that scans open NetBIOS name servers on a local or remote TCP/IP network. While it has legitimate administrative functions, attackers often use it for network reconnaissance to identify potential targets for lateral movement.

wce.exe (199,168 bytes) appears to be Windows Credential Editor (WCE), a known security tool that can be used to extract cleartext passwords, hashes, and Kerberos tickets from memory. Evidence of running it with the "-w" flag (which typically outputs Windows credentials) further confirms its malicious use.

Answer: 3

10. Machine:Target1 What is the password for the front desk local administrator account?

After identifying the tools used by the attackers to breach the system, it's necessary to investigate what they did with them. Command history stored in memory can reveal the specific actions taken by the attackers after gaining access to the system. Command line history is valuable evidence during forensic analysis, as it accurately reveals the commands entered and executed, allowing us to understand the attackers' behavior, their goals, and the scope of the breach. During our previous analysis, we identified the Windows Credential Editor (WCE) in the attackers'

toolkit. The presence of WCE in the temporary directory indicates that the primary goal was credential theft, but we need to determine whether the attackers successfully used this tool and what information they obtained.

vol2 -f "/work/target1/Target1-1dd8701f.vmss" --profile=Win7SP0x86 consoles | grep -i 'administrator'

```
ruslan@pop-os:~/Downloads/88-Grrcon2015/temp_extract_dir$ vol2 -f "/work/target1/Target1-1dd8701f.vmss" --profile=Win7SP0x86 consoles | grep -i 'administrator'
Title: Administrator: cmd
Title: Administrator: C:\Program Files\Internet Explorer\iexplore.exe
Title: Administrator: C:\Windows\System32\cmd.exe
Cmd #3 at 0x3487b8: runas /profile /user:Administrator
Cmd #4 at 0x34e500: runas /profile /user:Administrator cmd
C:\Windows\Temp>runas /profile /user:Administrator
> runas /noprofile /user:mymachine\administrator cmd
C:\Windows\Temp>runas /profile /user:Administrator cmd
Enter the password for Administrator:
Attempting to start cmd as user "FRONT-DESK-PC\Administrator" ...
OriginalTitle: cmd (running as FRONT-DESK-PC\Administrator)
Title: Administrator: cmd (running as FRONT-DESK-PC\Administrator)
Administrator\front-desk-PC:flagadmin@1234
```

Answer: flagadmin@1234

11. Machine:Target1 What is the std create data timestamp for the nbtscan.exe tool?

As we continue our investigation of a compromised system, we need to establish a timeline of events to better understand when the various components of the attack were deployed. Pinpointing the precise moment the attacker's tools were placed on the system can help correlate the attack with other security events and potentially reveal the full attack sequence. File timestamps are essential for this purpose, as they can reveal when files were created, modified, accessed, or executed. Windows stores detailed file metadata in the Master File Table (MFT), which is part of the NTFS file system. The MFT contains entries for every file and directory on an NTFS volume, including various timestamps associated with each file. These timestamps include Standard Information (SI) timestamps (record creation, modification, access, and modification times) and File Name (FN) timestamps. Standard Information attributes are important for forensic analysis, as they track when files were created, modified, and accessed. To analyze the attacker's toolkit timestamps, we can use Volatility's mftparser plugin, which extracts and analyzes MFT records from memory. This plugin provides detailed information about files, including their various timestamps, which can help establish the attack's chronology. By focusing specifically on the nbtscan.exe tool, which we previously identified as part of the attacker's toolkit, we can determine when this tool was first installed on the system. When running the command, Volatility searches the MFT records in memory and filters the results to show only those records related to nbtscan.exe. The output displays detailed timestamp information for this file.

vol2 -f "/work/target1/Target1-1dd8701f.vmss" --profile=Win7SP0x86 timeliner | grep -i "nbtscan"

or

vol2 -f "/work/target1/Target1-1dd8701f.vmss" --profile=Win7SP0x86 mftparser | grep -i "nbtscan"

```
ruslan@pop-os:~/Downloads/88-Grrcon2015/temp_extract_dir$ vol2 -f "/work/target1/Target1-1dd8701f.vmss" --profile=Win7SP0x86 timeliner | grep -i "nbtscan"
2015-10-09 10:45:12 UTC+0000|[SHIMCACHE]| \??\C:\Windows\Temp\nbtscan.exe|
```

```
ruslan@pop-os:~/Downloads/88-Grrcon2015/temp_extract_dir$ vol2 -f "/work/target1/Target1-1dd8701f.vmss" --profile=Win7SP0x86 mftparser | grep -i "nbtscan"
2015-10-09 10:45:12 UTC+0000 2015-10-09 10:45:12 UTC+0000   2015-10-09 10:45:12 UTC+0000   2015-10-09 10:45:12 UTC+0000   Windows\Temp\nbtscan.exe
2015-10-09 10:47:07 UTC+0000 2015-10-09 10:47:07 UTC+0000   2015-10-09 10:47:07 UTC+0000   2015-10-09 10:47:07 UTC+0000   Windows\Prefetch\NBTSCAN.EXE-44BD0B89.pf
```

Thus, based on this analysis, the Standard Information (SI) creation timestamp for the nbtscan.exe tool is 2015-10-09 10:45:12 UTC. This timestamp indicates the moment the malicious tool was first deployed on the compromised system, allowing this action to be placed within the overall attack timeline. The proximity of this timestamp to the creation times of other malicious files suggests that the attackers deployed multiple tools in rapid succession as part of a coordinated attack strategy. This discovery helps us establish a partial attack timeline, confirming that the nbtscan.exe tool was deployed on the system on October 9, 2015, at approximately 10:45 UTC. This information can be correlated with other system events and network logs to gain a comprehensive understanding of the attack's progression and potentially identify other compromised systems that may exhibit similar activity patterns.

Answer: 2015-10-09 10:45:12 UTC

12. Machine:Target1 The attackers appear to have stored the output from the nbtscan.exe tool in a text file on a disk called nbs.txt. What is the IP address of the first machine in that file?

We know that the file “nbs.txt” is located in the Temp directory based on the results from Q.9. We can use the dumpfiles plugin to extract the “nbs.txt” file.

Having established exactly when the attackers' tools were deployed on the compromised system, we need to investigate how these tools were used and what information the attackers collected. Network reconnaissance is a common stage of sophisticated attacks, allowing attackers to identify potential targets for lateral movement within a network. The nbtscan tool we previously identified is specifically designed for this purpose, as it can quickly enumerate NetBIOS information from systems on the local network. To fully understand the scope of the compromise, we need to determine what network information the attackers collected using nbtscan. Attackers often save the results of their reconnaissance tools to files for later use, allowing them to methodically plan their lateral movement strategy. By examining any saved nbtscan results, we can potentially identify other systems that may have been targeted or compromised as part of this attack campaign.

vol2 -f "/work/target1/Target1-1dd8701f.vmss" \ --profile=Win7SP0x86 \ dumpfiles -n -D /work/dumped -Q 0x000000003fdb7808

```
ruslan@pop-os:~/Downloads/88-Grrcon2015/temp_extract_dir$ vol2 -f "/work/target1/Target1-1dd8701f.vmss" \
     --profile=Win7SP0x86 \
     dumpfiles -n -D /work/dumped -Q 0x000000003fdb7808
Volatility Foundation Volatility Framework 2.6.1
DataSectionObject 0x3fdb7808   None   \Device\HarddiskVolume2\Windows\Temp\nbs.txt
```

cat dumped/file.None.0x83eda598.nbs.txt.dat

```
ruslan@pop-os:~/Downloads/88-Grrcon2015/temp_extract_dir$ ls -lah dumped
total 468K
drwxrwxrwx 2 ruslan          ruslan  4.0K Jan 23 00:22 .
drwxrwxr-x 7 ruslan          ruslan  4.0K Jan 22 22:52 ..
-rw-r--r-- 1 systemd-resolve nogroup 4.0K Jan 23 00:22 file.None.0x83eda598.nbs.txt.dat
-rw-r--r-- 1 systemd-resolve nogroup 228K Jan 22 00:16 file.None.0x858aef78.dat
-rw-r--r-- 1 systemd-resolve nogroup 227K Jan 22 00:16 file.None.0x85cd09a0.img
ruslan@pop-os:~/Downloads/88-Grrcon2015/temp_extract_dir$ cat dumped/file.None.0x83eda598.nbs.txt.dat
10.1.1.2        ALLSAFECYBERSEC\AD01            SHARING DC
10.1.1.3        ALLSAFECYBERSEC\EX01            SHARING
10.1.1.20       ALLSAFECYBERSEC\FRONT-DESK-PC   SHARING
10.1.1.21       ALLSAFECYBERSEC\GIDEON-PC       SHARING
ruslan@pop-os:~/Downloads/88-Grrcon2015/temp_extract_dir$
```

After successfully extracting the file, we examine its contents using a simple text viewing command. The output reveals a list of machines on the network along with their NetBIOS information. This output is typical of what we would expect from the nbtscan tool, which queries systems for their NetBIOS names and services. The file contains entries for several systems within the ALLSAFECYBERSEC domain, displaying their IP addresses, NetBIOS names, and sharing status. The first entry in the file shows IP address 10.1.1.2 corresponding to ALLSAFECYBERSEC\AD01, which appears to be a domain controller (indicated by "SHARING DC" in the output). This suggests the attackers were able to identify the domain controller, which would be a high-value target for further compromise as it typically contains authentication information for all users in the domain. This information provides the attackers with a map of the network, including key servers and potential targets for lateral movement. The identification of the domain controller and an Exchange server is particularly concerning, as compromising these systems would give attackers broad access to the organization's resources and data.

Answer: 10.1.1.2

13. Machine:Target1 What is the full IP address and the port was the attacker's malware using?

After identifying the attackers' internal network reconnaissance, we need to examine the malware's external communications. Understanding how malware interacts with its command and control (C2) infrastructure is

crucial for threat analysis, network security, and the potential identification of other compromised systems within the organization. Malware typically needs to communicate with external servers to receive commands, extract data, or download additional payloads. To identify these external communications, we can examine the network connections established by compromised processes.

vol2 -f "/work/target1/Target1-1dd8701f.vmss" --profile=Win7SP0x86 netscan | grep '10.1.1.20'

```
ruslan@pop-os:~/Downloads/88-Grrcon2015/temp_extract_dir$ vol2 -f "/work/target1/Target1-1dd8701f.vmss" --profile=Win7SP0x86 netscan | grep '10.1.1.20'
0x3deba9a0         UDPv4    10.1.1.20:56813                *:*                                    3232     svchost.exe    2015-10-09 11:32:55 UTC+0000
0x3e143978         UDPv4    10.1.1.20:1900                 *:*                                    3232     svchost.exe    2015-10-09 11:32:55 UTC+0000
0x3e25bc60         UDPv4    10.1.1.20:138                  *:*                                    4        System         2015-10-09 11:30:49 UTC+0000
0x3e2b0f50         UDPv4    10.1.1.20:137                  *:*                                    4        System         2015-10-09 11:30:49 UTC+0000
0x3e2b08a8         TCPv4    10.1.1.20:139                  0.0.0.0:0              LISTENING       4        System
0x3de98df8         TCPv4    10.1.1.20:49261                10.1.1.21:445          ESTABLISHED     4        System
0x3e0d0df8         TCPv4    10.1.1.20:49208                10.1.1.3:80            ESTABLISHED     3196     OUTLOOK.EXE
0x3e0eedf8         TCPv4    10.1.1.20:49205                180.76.254.120:22      ESTABLISHED     2996     iexplore.exe
0x3e1e5008         TCPv4    10.1.1.20:49330                10.1.1.2:139           CLOSED          4        System
0x3e1f0df8         TCPv4    10.1.1.20:49207                10.1.1.3:80            ESTABLISHED     3196     OUTLOOK.EXE
0x3e1fadf8         TCPv4    10.1.1.20:49314                10.1.1.3:443           CLOSED          3196     OUTLOOK.EXE
0x3fa4dbf8         TCPv4    10.1.1.20:49333                10.1.1.3:443           CLOSED          3196     OUTLOOK.EXE
0x3fa8d1d8         TCPv4    10.1.1.20:49336                10.1.1.3:443           CLOSED          3196     OUTLOOK.EXE
0x3fa95df8         TCPv4    10.1.1.20:49297                192.96.201.138:5938    ESTABLISHED     2680     TeamViewer.exe
0x3fb7a560         TCPv4    10.1.1.20:49301                10.1.1.21:3389         ESTABLISHED     2844     mstsc.exe
0x3fc426a8         TCPv4    10.1.1.20:49291                107.6.97.19:5938       ESTABLISHED     2680     TeamViewer.exe
```

The remote IP address 180.76.254.120 matches the domain found in the phishing email that delivered the initial payload (180.76.254.129 is the server hosting AnyConnectInstaller.exe). The slight difference in the last octet (120 vs. 129) suggests that the attackers control multiple IP addresses in this range, possibly for different stages of their attack infrastructure. Destination port 22 is also significant. Port 22 is typically associated with SSH (Secure Shell) traffic, which provides encrypted communication between systems. However, in this context, it is unlikely that the compromised Internet Explorer process would legitimately establish SSH connections. Instead, the malware likely uses port 22 for command and control communications to disguise itself as legitimate SSH traffic, making it difficult to detect using simple port filtering. This established connection represents the command and control channel that the Extreme RAT malware uses to communicate with its operator. Through this channel, attackers can send commands to the malware, receive execution results, download additional tools, download stolen data, or issue instructions for further movement within the network.

Answer: 180.76.254.120:22

14. Machine:Target1 It appears the attacker also installed legit remote administration software. What is the name of the running process?

During further forensic analysis of the compromised system, we need to identify all potential methods the attackers could have used to maintain access. While we have already detected Extreme RAT malware, which utilizes process injection and persistent access via registry modification, sophisticated attackers often employ multiple access methods to ensure they retain control even if one method is detected and eliminated. A common tactic in more sophisticated attacks is to install legitimate remote administration software alongside malicious tools. This approach offers attackers several advantages: legitimate software is less likely to trigger antivirus warnings, it provides a fallback access method if malware is detected, and it can be passed off as authorized administration software if it is discovered during a cursory inspection. To identify all running processes on the system at the time of the memory hijacking, we can use the pslist plugin from Volatility. This plugin provides a complete list of processes active on the system at the time the memory dump was created, including information such as process IDs, parent processes, thread count, handle count, and process startup time.

vol2 -f "/work/target1/Target1-1dd8701f.vmss" --profile=Win7SP0x86 pslist

```
ruslan@pop-os:~/Downloads/88-Grrcon2015/temp_extract_dir$ vol2 -f "/work/target1/Target1-1dd8701f.vmss" --profile=Win7SP0x86 pslist
Volatility Foundation Volatility Framework 2.6.1
Offset(V)  Name                    PID   PPID   Thds     Hnds   Sess  Wow64 Start                          Exit
---------- -------------------- ------ ------ ------ -------- ------ ------ ------------------------------ ------------------------------
0x83d334e8 System                    4      0     94      500 ------      0 2015-10-09 11:30:44 UTC+0000
0x84edcbf0 smss.exe                276      4      2       30 ------      0 2015-10-09 11:30:44 UTC+0000
0x84ecbb18 csrss.exe               368    360      9      366      0      0 2015-10-09 11:30:47 UTC+0000
0x84f97628 wininit.exe             420    360      3       77      0      0 2015-10-09 11:30:48 UTC+0000
0x855f6d40 csrss.exe               432    412     11      366      1      0 2015-10-09 11:30:48 UTC+0000
0x8561d030 winlogon.exe            480    412      3      115      1      0 2015-10-09 11:30:48 UTC+0000
0x84e979f8 services.exe            528    420      9      200      0      0 2015-10-09 11:30:48 UTC+0000
0x8583b030 lsass.exe               536    420      9      851      0      0 2015-10-09 11:30:48 UTC+0000
0x8583d960 lsm.exe                 544    420     10      163      0      0 2015-10-09 11:30:48 UTC+0000
0x8586fd40 svchost.exe             644    528     11      351      0      0 2015-10-09 11:30:48 UTC+0000
0x84e01448 svchost.exe             720    528      6      276      0      0 2015-10-09 11:30:50 UTC+0000
0x85935030 svchost.exe             796    528     19      446      0      0 2015-10-09 11:30:51 UTC+0000
0x85969030 svchost.exe             836    528     17      405      0      0 2015-10-09 11:30:52 UTC+0000
0x85978940 svchost.exe             864    528     30     1036      0      0 2015-10-09 11:30:52 UTC+0000
0x859cc2c0 svchost.exe            1008    528     13      650      0      0 2015-10-09 11:30:52 UTC+0000
0x85a138f0 svchost.exe            1124    528     16      484      0      0 2015-10-09 11:30:53 UTC+0000
0x8582c8d8 spoolsv.exe            1228    528     12      273      0      0 2015-10-09 11:30:53 UTC+0000
0x85a55d40 svchost.exe            1256    528     17      304      0      0 2015-10-09 11:30:53 UTC+0000
0x85ae3030 vmtoolsd.exe           1432    528      8      274      0      0 2015-10-09 11:30:54 UTC+0000
0x85976318 svchost.exe            1784    528      5       99      0      0 2015-10-09 11:30:54 UTC+0000
0x85ae0cb0 dllhost.exe            1888    528     13      196      0      0 2015-10-09 11:30:54 UTC+0000
0x858b69e8 msdtc.exe              1980    528     12      145      0      0 2015-10-09 11:30:55 UTC+0000
0x85c09968 dwm.exe                2088    836      3       93      1      0 2015-10-09 11:31:04 UTC+0000
0x85c1e5f8 explorer.exe           2116   2060     23      912      1      0 2015-10-09 11:31:04 UTC+0000
0x85c39030 taskhost.exe           2252    528      7      150      1      0 2015-10-09 11:31:04 UTC+0000
0x859281f0 vmtoolsd.exe           2388   2116      7      164      1      0 2015-10-09 11:31:04 UTC+0000
0x8598c920 SearchIndexer.         2544    528     13      670      0      0 2015-10-09 11:31:10 UTC+0000
0x85d0d030 iexplore.exe           2996   2984      6      463      1      0 2015-10-09 11:31:27 UTC+0000
0x85cd3d40 OUTLOOK.EXE            3196   2116     22     1678      1      0 2015-10-09 11:31:32 UTC+0000
0x85d01510 svchost.exe            3232    528      9      131      0      0 2015-10-09 11:31:34 UTC+0000
0x85b43a58 sppsvc.exe             3900    528      4      153      0      0 2015-10-09 11:32:54 UTC+0000
0x83eb5d40 cmd.exe                2496   2116      1       22      1      0 2015-10-09 11:33:42 UTC+0000
0x83e5cd40 conhost.exe             916    432      3       83      1      0 2015-10-09 11:33:42 UTC+0000
0x83f105f0 cmd.exe                1856   2996      1       33      1      0 2015-10-09 11:35:15 UTC+0000
0x83f13d40 conhost.exe            1624    432      3       81      1      0 2015-10-09 11:35:15 UTC+0000
0x83fb86a8 cmd.exe                3064   2116      1       22      1      0 2015-10-09 11:37:32 UTC+0000
0x83fa9030 conhost.exe             676    432      3       83      1      0 2015-10-09 11:37:32 UTC+0000
0x83fb2d40 cmd.exe                3784   2196      1       24      1      0 2015-10-09 11:39:22 UTC+0000
0x83fc7c08 conhost.exe            1824    432      3       85      1      0 2015-10-09 11:39:22 UTC+0000
0x84013598 TeamViewer.exe         2680   1696     28      632      1      0 2015-10-09 12:08:46 UTC+0000
0x84017d40 tv_w32.exe             4064   2680      2       83      1      0 2015-10-09 12:08:47 UTC+0000
0x858bc278 TeamViewer_Des         1092   2680     16      405      1      0 2015-10-09 12:10:56 UTC+0000
0x83f1ed40 mstsc.exe              2844   2116     11      484      1      0 2015-10-09 12:12:03 UTC+0000
```

Analyzing the pslist command output, we see the expected Windows system processes, such as System (PID 4), smss.exe, csrss.exe, winlogon.exe, services.exe, lsass.exe, and various instances of svchost.exe. We also see the previously identified OUTLOOK.EXE (PID 3196) and iexplore.exe (PID 2996) processes, associated with the initial infection vector and the process containing the Extreme RAT malware. However, another set of processes is particularly interesting: TeamViewer.exe (PID 2680), tv_x32.exe (PID 4064), and TeamViewer_Desktop (PID 1092). TeamViewer is a legitimate remote desktop access and remote control software widely used by IT administrators for legitimate support purposes. However, its presence on this system is suspicious given the other indicators of compromise we found. The TeamViewer process appears to have started on October 9, 2015, at 12:08:46 UTC, which falls within the attack window and occurs after the attacker's tools were deployed to the temporary directory. This timing indicates that TeamViewer was installed by the attackers, rather than being pre-installed legitimate software. TeamViewer provides full remote desktop access capabilities, allowing the user to view and control the system remotely, as if they were sitting in front of it. For attackers, this is a powerful and convenient way to interact with a compromised system, potentially bypassing some security controls that might block more obviously malicious remote access tools. Furthermore, TeamViewer traffic is often allowed through corporate firewalls because it is used for legitimate business purposes, making it an ideal covert channel for attackers.

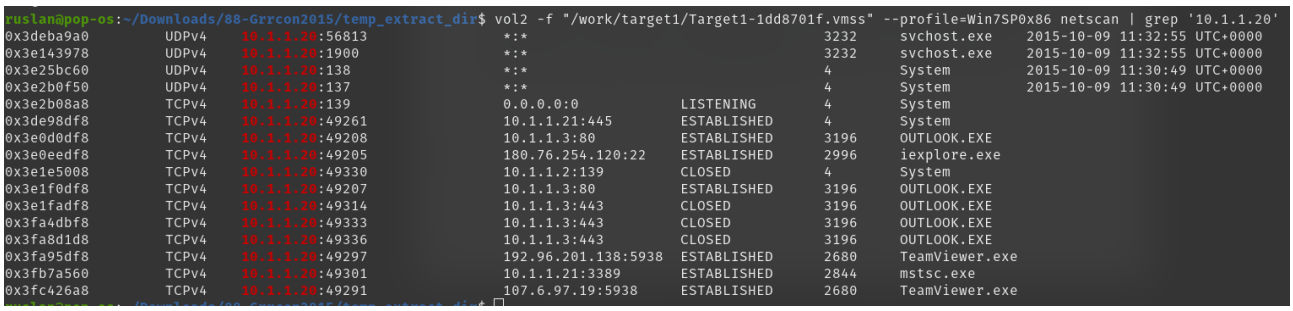
Answer: TeamViewer.exe

15. Machine:Target1 It appears the attackers also used a built-in remote access method. What IP address did they connect to?

vol2 -f "/work/target1/Target1-1dd8701f.vmss" --profile=Win7SP0x86 netscan | grep 'mstsc.exe'

As we near the end of our forensic investigation on the first machine, we need to identify all the methods the attackers used to access and control the compromised system. In addition to malware and third-party remote administration tools, attackers often exploit Windows' built-in remote access capabilities to maintain access. The advantage of these built-in methods is that they leave fewer obvious traces than third-party software installations and can persist even after malware is detected and removed. To identify all network connections established by

the compromised system, we can use Volatility's netscan plugin without filtering by specific processes. This approach provides a comprehensive overview of all network activity on the system at the time of the memory seizure, potentially revealing additional communication channels used by the attackers. The only filter we use is to display connections associated with the compromised acceptance control system (IP 10.1.1.20). This approach helps us focus on relevant connections while simultaneously capturing all processes that may be involved in suspicious communications.



The netscan results show numerous network connections associated with various processes on the system. We've already identified an Extreme RAT connection to 180.76.254.120:22 via the blocked iexplore.exe process and a TeamViewer connection to 107.6.97.19:5938. However, another interesting connection stands out in the results. We see a TCP connection between the compromised system (10.1.1.20:49301) and another internal system at 10.1.1.21:3389. Port 3389 is the default port used by Windows Remote Desktop Protocol (RDP), a built-in Windows feature that enables graphical remote access to Windows systems. The connection displays as ESTABLISHED, indicating an active RDP session at the time of the memory capture. The target IP address, 10.1.1.21, matches the GIDEON-PC system we observed previously in the nbtscan output, suggesting that the attackers established an RDP connection from the compromised system at the front desk to this other internal workstation. This indicates lateral movement within the network, a common phase of sophisticated attacks when attackers move from the initially compromised systems to other targets within the environment. The connection is associated with the mstsc.exe process (PID 2844), which is a Microsoft Terminal Services client, the standard Windows RDP client application. This confirms that the attackers used the legitimate built-in Windows Remote Desktop feature to move laterally within the network after compromising the system at the front desk. This discovery is important because it demonstrates that the attackers did not simply establish a foothold on one system, but were actively expanding their presence within the organization's network. Using RDP for lateral movement is a common tactic because it leverages built-in Windows features, requires no additional software to be installed, and provides a graphical interface that makes it easy to explore and hack additional systems.

Answer: 10.1.1.21

16. Machine:Target2 It appears the attacker moved latterly from the front desk machine to the security admins (Gideon) machine and dumped the passwords. What is Gideon's password?

vol2 -f "/work/target2/target2-6186fe9f.vmss" --profile=Win7SP0x86 consoles

```
Cmd #0 at 0xe6030: cd C:\Users
Cmd #1 at 0xe6ea8: dir
Cmd #2 at 0xee3d0: wce.exe -w > gideon/w.tmp
Cmd #3 at 0xe0170: who ami
Cmd #4 at 0xe0188: whoami
Cmd #5 at 0xea3c8: net use z: \\10.1.1.2\c$
Cmd #6 at 0xe01b8: cd z:
Cmd #7 at 0xe6ed8: dir
Cmd #8 at 0xe6070: cd gideon
Cmd #9 at 0xe6ef8: dir
Cmd #10 at 0xe6f08: z:
Cmd #11 at 0xe6f18: dir
Cmd #12 at 0xf2418: copy c:\users\gideon\rar.exe z:\crownjewels
Cmd #13 at 0xe0cb8: cd crownjewels
Cmd #14 at 0xe6f28: dir
Cmd #15 at 0xe6f38: rar
Cmd #16 at 0xf2478: rar crownjewlez.rar *.txt -hp123qwe!@#
Cmd #17 at 0xf24d0: rar a -hp123!@#qwe crownjewlez.rar *.txt
----
```

Windows Credentials Editor - a tool for dumping passwords from memory
The wce.exe -w > gideon\w.tmp command is concerning because it indicates that the attackers ran Windows Credentials Editor with the -w parameter, which extracts cleared Windows credentials from memory and redirects the output to a file named w.tmp in the gideon directory. This confirms that credential theft was the primary goal after hacking this machine. To recover the contents of this credential dump, we first need to locate the file in memory. Using the filescan plugin with a filter on the target file path, we can obtain its physical offset. Once the file is located, we can extract it using the dumpfiles plugin to extract the credential dump file to our analysis directory.

vol2 -f "/work/target2/target2-6186fe9f.vmss" --profile=Win7SP0x86 filescan | grep 'w.tmp'

```
ruslan@pop-os:~/Downloads/88-Grrcon2015/temp_extract_dir$ vol2 -f "/work/target2/target2-6186fe9f.vmss" --profile=Win7SP0x86 filescan | grep 'w.tmp'
0x000000003fcf2798      8      0 -W-r-- \Device\HarddiskVolume2\Users\gideon\w.tmp
```

vol2 -f "/work/target2/target2-6186fe9f.vmss" --profile=Win7SP0x86 dumpfiles -D /work/dumped -Q 0x000000003fcf2798

```
ruslan@pop-os:~/Downloads/88-Grrcon2015/temp_extract_dir$ vol2 -f "/work/target2/target2-6186fe9f.vmss" --profile=Win7SP0x86 dumpfiles -D /work/dumped -Q 0x000000003fcf2798
Volatility Foundation Volatility Framework 2.6.1
DataSectionObject 0x3fcf2798   None   \Device\HarddiskVolume2\Users\gideon\w.tmp
```

cat dumped/file.None.0x85a35da0.dat

```
ruslan@pop-os:~/Downloads/88-Grrcon2015/temp_extract_dir$ cat dumped/file.None.0x85a35da0.dat
WCE v1.42beta (Windows Credentials Editor) - (c) 2010-2013 Amplia Security - by Hernan Ochoa (hernan@ampliasecurity.com)
Use -h for help.


gideon\ALLSAFECYBERSEC:t76fRJhS
GIDEON-PC$\ALLSAFECYBERSEC:s9O3t%sd1q>:u5Za8Xrx_3Eg;(\qapu<"Rn$#QQJlsD m#;z2hbJkr*tLe>0)F[S)'USh3BKJILn3-?vt]q=s-Cp.ws9wVik[]5?#F\*l/J19+`PYco:au;T
```

Answer: t76fRJhS

17. Machine:Target2 Once the attacker gained access to "Gideon," they pivoted to the AllSafeCyberSec domain controller to steal files. It appears they were successful. What password did they use?

vol2 -f "/work/target2/target2-6186fe9f.vmss" --profile=Win7SP0x86 consoles

```
Cmd #0 at 0xe6030: cd C:\Users
Cmd #1 at 0xe6ea8: dir
Cmd #2 at 0xee3d0: wce.exe -w > gideon/w.tmp
Cmd #3 at 0xe0170: who ami
Cmd #4 at 0xe0188: whoami
Cmd #5 at 0xea3c8: net use z: \\10.1.1.2\c$
Cmd #6 at 0xe01b8: cd z:
Cmd #7 at 0xe6ed8: dir
Cmd #8 at 0xe6070: cd gideon
Cmd #9 at 0xe6ef8: dir
Cmd #10 at 0xe6f08: z:
Cmd #11 at 0xe6f18: dir
Cmd #12 at 0xf2418: copy c:\users\gideon\rar.exe z:\crownjewels
Cmd #13 at 0xe0cb8: cd crownjewels
Cmd #14 at 0xe6f28: dir
Cmd #15 at 0xe6f38: rar
Cmd #16 at 0xf2478: rar crownjewlez.rar *.txt -hp123qwe!@#
Cmd #17 at 0xf24d0: rar a -hp123!@#qwe crownjewlez.rar *.txt
----
```

Successful execution of subsequent commands on drive Z confirms the successful connection attempt. This means the attackers were able to authenticate to the domain controller and mount its drive, gaining direct access to all files on the system. This level of access is highly concerning, as it effectively grants the attackers complete control over the domain environment. After successfully connecting to the domain controller, we see signs of a data leak. The commands include:
copy c:\users\gideon\rar.exe z:\crownjewels

cd crownjewels

dir

rar

rar crownjewelz.rar *.txt -hp123qwe!@#

rar a -hp123!@#qwe crownjewelz.rar *.txt

These commands show that the attackers copied the RAR archiving tool to the crownjewlez.rar directory on the domain controller, navigated to that directory, and created an encrypted archive of the text files located there. Using an encrypted archive indicates the attackers' intention to protect the stolen data from easy access if intercepted during exfiltration. Most importantly, we can see the password they used to encrypt the archive: 123qwe!@#.

Answer: 123qwe!@#

18. Machine:Target2 What was the name of the RAR file created by the attackers?

Answer: crownjewlez.rar

19. Machine:Target2 How many files did the attacker add to the RAR archive?

In the final phase of our forensic investigation, we need to determine the full scope of the data breach from the domain controller. Having established that the attackers created an encrypted RAR archive of files from the crownjewlez directory, we now need to determine exactly how many files were included in this archive. Understanding the scope of the data theft is crucial for incident response teams to assess the potential impact and inform affected parties. When attackers create archives of stolen data, the contents of these archives can reveal their specific objectives and the types of information they were targeting. In this case, the command history revealed that the attackers used wildcards (*.txt) when creating the archive, indicating that they specifically targeted text files, which often contain sensitive information such as credentials, configuration details, or proprietary data.

vol2 -f "/work/target2/target2-6186fe9f.vmss" --profile=Win7SP0x86 cmdscan

```
^CInterrupted
ruslan@pop-os:~/Downloads/88-Grrcon2015/temp_extract_dir$ vol2 -f "/work/target2/target2-6186fe9f.vmss" --profile=Win7SP0x86 cmdscan
Volatility Foundation Volatility Framework 2.6.1
**************************************************
CommandProcess: conhost.exe Pid: 2888
CommandHistory: 0x2d9ff0 Application: cmd.exe Flags: Allocated, Reset
CommandCount: 1 LastAdded: 0 LastDisplayed: 0
FirstCommand: 0 CommandCountMax: 50
ProcessHandle: 0x60
Cmd #0 @ 0x2d77a0: ipconfig
Cmd #1 @ 0x2d0031: ???
Cmd #12 @ 0x2d0032: ???
Cmd #17 @ 0x2d0035: ?
Cmd #36 @ 0x2a00c4: -?-?*???*
Cmd #37 @ 0x2d6be0: -?*???????-
**************************************************
CommandProcess: conhost.exe Pid: 3048
CommandHistory: 0xe9198 Application: cmd.exe Flags: Allocated, Reset
CommandCount: 18 LastAdded: 17 LastDisplayed: 17
FirstCommand: 0 CommandCountMax: 50
ProcessHandle: 0x60
Cmd #0 @ 0xe6030: cd C:\Users
Cmd #1 @ 0xe6ea8: dir
Cmd #2 @ 0xee3d0: wce.exe -w > gideon/w.tmp
Cmd #3 @ 0xe0170: who ami
Cmd #4 @ 0xe0188: whoami
Cmd #5 @ 0xea3c8: net use z: \\10.1.1.2\c$
Cmd #6 @ 0xe01b8: cd z:
Cmd #7 @ 0xe6ed8: dir
Cmd #8 @ 0xe6070: cd gideon
Cmd #9 @ 0xe6ef8: dir
Cmd #10 @ 0xe6f08: z:
Cmd #11 @ 0xe6f18: dir
Cmd #12 @ 0xf2418: copy c:\users\gideon\rar.exe z:\crownjewels
Cmd #13 @ 0xe0cb8: cd crownjewels
Cmd #14 @ 0xe6f28: dir
Cmd #15 @ 0xe6f38: rar
Cmd #16 @ 0xf2478: rar crownjewlez.rar *.txt -hp123qwe!@#
Cmd #17 @ 0xf24d0: rar a -hp123!@#qwe crownjewlez.rar *.txt
Cmd #36 @ 0xb00c4: ??
                   ???

Cmd #37 @ 0xe5d48: ?
                    ???????
```

There are two conhost.exe processes, but the one with PID 3048 is responsible for processing the command when the attacker adds all .txt files to the RAR archive.

vol2 -f "/work/target2/target2-6186fe9f.vmss" --profile=Win7SP0x86 memdump --pid=3048 -D /work/dumped

```
^CInterrupted
ruslan@pop-os:~/Downloads/88-Grrcon2015/temp_extract_dir$ vol2 -f "/work/target2/target2-6186fe9f.vmss" --profile=Win7SP0x86 memdump --pid=3048 -D /work/dumped
Volatility Foundation Volatility Framework 2.6.1
************************************************************************
Writing conhost.exe [  3048] to 3048.dmp
```

grep -F '\crownjewels\' 3048.txt

```
ruslan@pop-os:~/Downloads/88-Grrcon2015/temp_extract_dir$ grep -F '\crownjewels\' 3048.txt
Z:\crownjewels\
Z:\crownjewels\Rar.exe
\10.1.1.2\c$\crownjewels\Rar.exe
\10.1.1.2\c$\crownjewels\Rar.exe
\crownjewels\rar.lng
\crownjewels\CRYPTBASE.dll
\crownjewels\rar.ini
\crownjewels\SecretSauce2.txt
\crownjewels\crownjewlez.rar
\10.1.1.2\c$\crownjewels\Rar.exe
\crownjewels\crownjewlez.rar
\crownjewels\SecretSauce1.txt
\crownjewels\CRYPTBASE.dll
\crownjewels\CRYPTSP.dll
\crownjewels\en
\crownjewels\Rar.exe
\crownjewels\SecretSauce3.txt
\crownjewels\Rar.exe
\crownjewels\ui\SwDRM.dll
```

Answer: 3

20. Machine:Target2 The attacker appears to have created a scheduled task on Gideon's machine. What is the name of the file associated with the scheduled task?

During further forensic analysis of compromised systems, we need to investigate potential persistence mechanisms beyond those already identified. Experienced attackers often use multiple methods to maintain access to compromised systems, ensuring they can regain control even if some of these methods are discovered and removed. Windows scheduled tasks are a common persistence mechanism, as they can be configured to execute programs at specific times or upon specific events, allowing malware to reactivate even after a system reboot. To identify any scheduled tasks created by attackers, we need to examine the Windows Task Scheduler directories in the memory dump.

vol2 -f "/work/target2/target2-6186fe9f.vmss" --profile=Win7SP0x86 filescan | grep '\\Windows\\System32\\Tasks\\'

```
0x000000003fb10408      2      0 R--r-d \Device\HarddiskVolume2\Windows\System32\Tasks\Microsoft\Windows\CertificateServicesClient\UserTask
0x000000003fb21e88      2      0 R--r-d \Device\HarddiskVolume2\Windows\System32\Tasks\Microsoft\Windows\TextServicesFramework\MsCtfMonitor
0x000000003fb37bc0      2      0 R--r-d \Device\HarddiskVolume2\Windows\System32\Tasks\Microsoft\Windows\User Profile Service\HiveUploadTask
0x000000003fb395d8      2      0 R--r-d \Device\HarddiskVolume2\Windows\System32\Tasks\Microsoft\Windows\Media Center\SqlLiteRecoveryTask
0x000000003fb587c0      2      0 R--r-d \Device\HarddiskVolume2\Windows\System32\Tasks\Microsoft\Windows\Power Efficiency Diagnostics\AnalyzeSystem
0x000000003fb58ec8      8      0 R--r-d \Device\HarddiskVolume2\Windows\System32\Tasks\Microsoft\Windows\RemoteAssistance\RemoteAssistanceTask
0x000000003fb5b470      2      0 R--r-d \Device\HarddiskVolume2\Windows\System32\Tasks\Microsoft\Windows\Media Center\RecordingRestart
0x000000003fb92038      2      0 R--r-d \Device\HarddiskVolume2\Windows\System32\Tasks\Microsoft\Office\OfficeTelemetryAgentLogOn
0x000000003fb92708      8      0 R--r-d \Device\HarddiskVolume2\Windows\System32\Tasks\Microsoft\Windows\CertificateServicesClient\SystemTask
0x000000003fbaef80      2      0 R--r-d \Device\HarddiskVolume2\Windows\System32\Tasks\Microsoft\Windows\Tcpip\IpAddressConflict1
0x000000003fbb7e00      8      0 R--r-d \Device\HarddiskVolume2\Windows\System32\Tasks\GoogleUpdateTaskMachineCore
0x000000003fbb7f80      2      0 R--r-d \Device\HarddiskVolume2\Windows\System32\Tasks\Microsoft\Windows\MUI\LPRemove
0x000000003fc399b8      8      0 R--r-d \Device\HarddiskVolume2\Windows\System32\Tasks\At1
0x000000003fc436a8      2      0 R--r-d \Device\HarddiskVolume2\Windows\System32\Tasks\Microsoft\Windows\Media Center\MediaCenterRecoveryTask
0x000000003fc447b0      2      0 R--r-d \Device\HarddiskVolume2\Windows\System32\Tasks\Microsoft\Windows\Customer Experience Improvement Program\KernelCeipTask
```

This is striking because it doesn't match the generally accepted naming convention for legitimate Windows tasks, which typically includes the name of the component or function. The name At1 is suspicious and requires further investigation. To examine the contents of the suspicious task file, we can use the filescan plugin to determine its physical offset, and then use the dumpfiles plugin to extract it for analysis.

vol2 -f "/work/target2/target2-6186fe9f.vmss" --profile=Win7SP0x86 dumpfiles -Q 0x000000003fc399b8 -D /work/dumped

```
ruslan@pop-os:~/Downloads/88-Grrcon2015/temp_extract_dir$ vol2 -f "/work/target2/target2-6186fe9f.vmss" --profile=Win7SP0x86 dumpfiles -Q 0x000000003fc399b8 -D /work/dumped
Volatility Foundation Volatility Framework 2.6.1
DataSectionObject 0x3fc399b8   None   \Device\HarddiskVolume2\Windows\System32\Tasks\At1
```

cat dumped/file.None.0x85a86af0.dat

```
ruslan@pop-os:~/Downloads/88-Grrcon2015/temp_extract_dir$ cat dumped/file.None.0x85a86af0.dat
<?xml version="1.0" encoding="UTF-16"?>
<Task version="1.0" xmlns="http://schemas.microsoft.com/windows/2004/02/mit/task">
  <RegistrationInfo />
  <Triggers>
    <TimeTrigger>
      <StartBoundary>2015-10-09T08:00:00</StartBoundary>
    </TimeTrigger>
  </Triggers>
  <Principals>
    <Principal id="Author">
      <UserId>@AtServiceAccount</UserId>
      <LogonType>InteractiveTokenOrPassword</LogonType>
      <RunLevel>HighestAvailable</RunLevel>
    </Principal>
  </Principals>
  <Actions Context="Author">
    <Exec>
      <Command>c:\users\gideon\1.bat</Command>
    </Exec>
  </Actions>
```

Analysis of the extracted file reveals information about a scheduled task, including a link to a batch file located at C:\Users\gideon\1.bat. This indicates that the scheduled task was configured to execute this batch file, likely as a mechanism to ensure persistent access for the attackers. This discovery reveals another layer of the attackers' persistent access strategy, revealing that they not only installed multiple remote access methods (Extreme RAT, TeamViewer, and RDP) but also created a scheduled task to ensure the ability to maintain access to the system over an extended period.

Answer: 1.bat

21. Machine:POS What is the malware CNC's server?

After analyzing the registration system (Target 1) and the security administrator's computer (Target 2), we need to expand the investigation to include the POS (Point of Sale) system, which was likely compromised as part of the same attack campaign. POS systems are high-value targets for attackers because they process payment card information, making them lucrative sources of financial data that can be monetized through card fraud or sold on underground markets. To determine how the POS system was compromised and what data may have been stolen, we need to examine the network connections established by processes on this system. Malware targeting payment card data typically requires a command and control (CNC) server to receive the stolen information and issue commands to the malware. By identifying these connections, we can determine both the nature of the compromise and the destination of any stolen data. Using the Volatility netscan plugin, we can identify all network connections and listening ports in the POS system at the time the memory dump was captured. When analyzing the network scan results, it's important to look for suspicious connections, especially those established by processes that don't typically require external network access, or those connecting to unfamiliar or suspicious IP addresses.

vol2 -f "/work/pos01/POS-01-c4e8f786.vmss" --profile=Win7SP0x86 netscan

```
0x3e6cf058         TCPv6    :::49154                  :::0                  LISTENING        900      svchost.exe
0x3e6cf270         TCPv4    0.0.0.0:49154             0.0.0.0:0             LISTENING        900      svchost.exe
0x3e0f90e8         TCPv4    10.1.1.10:64532           10.1.1.3:80           ESTABLISHED      3376     OUTLOOK.EXE
0x3e135df8         TCPv4    10.1.1.10:58751           54.84.237.92:80       CLOSE_WAIT       3208     iexplore.exe
0x3e24c7d0         TCPv4    10.1.1.10:49201           23.203.149.112:443    CLOSE_WAIT       2464     jusched.exe
0x3e611b10         TCPv4    -:49887                   108.162.232.200:49155 CLOSED           536      lsass.exe
```

Netscan results reveal numerous network connections associated with various processes on the system. Most of these appear to be legitimate internal connections used by Windows services, but several entries stand out as suspicious. In particular, we notice a TCP connection from 10.1.1.10:58751 (likely the local POS system) to 54.84.237.92:80. This connection is associated with iexplore.exe (PID 3208) and is in the CLOSE_WAIT state, indicating that the connection was active but is now in the process of terminating. Several characteristics make this connection suspicious. First, Internet Explorer is typically not running on a POS system, where minimal software should be installed and internet access should be limited to reduce the attack surface. Second, the connection was established to a remote server on port 80, which is typically used for HTTP traffic but is also frequently used by malware for command and control, blending in with regular web traffic. Third, the IP address 54.84.237.92 appears to belong to an Amazon Web Services (AWS) instance, which is often used by attackers to host malicious infrastructure due to its reliability and relative anonymity. Fourth, we have already seen the iexplore.exe process used by attackers on an earlier machine. Therefore, based on the characteristics of the connection and the associated process, the connection to 54.84.237.92:80 from iexplore.exe is the most likely candidate for the malware's command and control server. This IP address is running a service on port 80, which suggests it is configured to accept HTTP traffic, which is a common protocol used by malware to communicate with its command and control servers, while disguising itself as normal web traffic to avoid detection.

22. Machine:POS What is the common name of the malware used to infect the POS system?

Having identified the command and control server for the malware on the Point of Sale (POS) system, we need to determine the specific type of malware used in the attack. Identifying the malware family provides crucial information about its capabilities, potential data exfiltration methods, and appropriate remediation strategies. POS malware typically targets payment card data stored temporarily in memory during transaction processing, making it particularly dangerous for retail environments. To analyze the malware, we need to first locate potentially malicious code in the system's memory. We already identified a suspicious network connection from Internet Explorer (PID 3208) to an external server, suggesting this process may have been compromised or is hosting the malware. Advanced malware often uses memory injection techniques to hide within legitimate processes, making them harder to detect by conventional security tools. The
Volatility malfind plugin is designed to identify signs of memory injection by scanning for memory regions with suspicious characteristics, such as executable permissions in unexpected areas or memory segments allocated through suspicious API calls.

vol2 -f "/work/pos01/POS-01-c4e8f786.vmss» --profile=Win7SP0x86 malfind

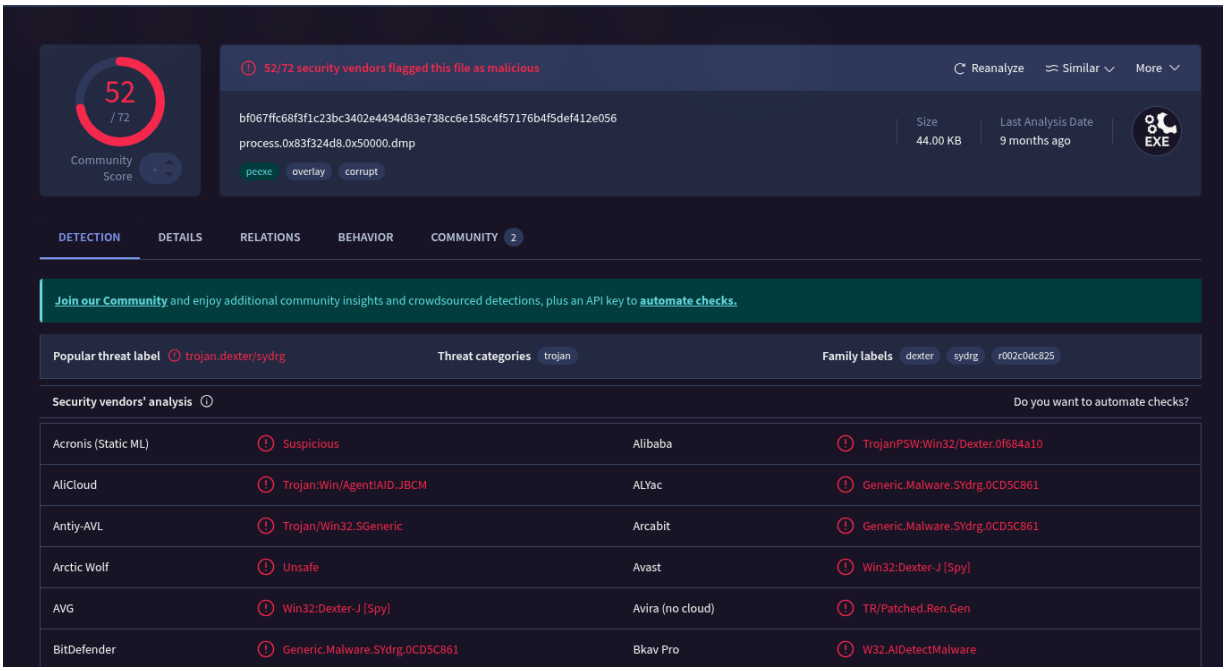
```
Process: iexplore.exe Pid: 3208 Address: 0x50000
Vad Tag: VadS Protection: PAGE_EXECUTE_READWRITE
Flags: CommitCharge: 11, MemCommit: 1, PrivateMemory: 1, Protection: 6

0x0000000000050000  4d 5a 90 00 03 00 00 00 04 00 00 00 ff ff 00 00   MZ..............
0x0000000000050010  b8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00   ........@.......
0x0000000000050020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0x0000000000050030  00 00 00 00 00 00 00 00 00 00 00 00 d8 00 00 00   ................
```

```
Process: iexplore.exe Pid: 3136 Address: 0x50000
Vad Tag: VadS Protection: PAGE_EXECUTE_READWRITE
Flags: CommitCharge: 11, MemCommit: 1, PrivateMemory: 1, Protection: 6

0x0000000000050000  4d 5a 90 00 03 00 00 00 04 00 00 00 ff ff 00 00   MZ..............
0x0000000000050010  b8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00   ........@.......
0x0000000000050020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0x0000000000050030  00 00 00 00 00 00 00 00 00 00 00 00 d8 00 00 00   ................
```

vol2 -f "/work/pos01/POS-01-c4e8f786.vmss" --profile=Win7SP0x86 malfind -p 3208 -D /work/dumped



A VirusTotal report indicates that most security vendors classify this malware as the Dexter Trojan. In the section on specific malware families, two main classifications are found: Dexter and Sydug, with Dexter being the more common designation. Dexter is a well-known family of POS malware, first discovered in 2012. It is designed to attack POS systems and steal payment card data from memory during processing. Dexter operates by searching process memory for patterns of payment card tracking data and then transmitting this data to a command and control server. This malware has been responsible for numerous data breaches at retail organizations, resulting in the theft of millions of payment card records. Dexter malware typically infects POS systems through a variety of methods, including compromised remote administration tools, phishing emails, or by lateral movement from other compromised systems on the network. Its presence on this POS system, connected to a remote command and control server, strongly suggests the theft of payment card data from this organization. The common name for the malware used to infect the POS system is Dexter. This discovery completes our forensic analysis of all three compromised systems, providing a comprehensive understanding of the attack chain, beginning with the initial phishing email, through lateral movement and credential theft, and ending with the ultimate goal of stealing payment card data from the organization's POS infrastructure.

Answer: Dexter

23. Machine:POS In the POS malware whitelist. What application was specific to Allsafecybersec?

After detecting the Dexter malware in a POS system, we needed to delve deeper into its configuration and behavior to understand how it operated in this specific environment. Malware authors often tailor their payloads to specific targets, incorporating features to avoid detection in certain environments. One common technique is to implement a whitelist of processes or applications that the malware will ignore or avoid infecting to prevent disruption of critical system functions and reduce the likelihood of detection. To gain more information about the malware's configuration, we can examine the strings present in the memory dump of the suspect process. String analysis is a fundamental malware analysis technique that can reveal human-readable text embedded within the malware, including configuration parameters, hardcoded IP addresses, file paths, and other valuable artifacts.

strings dumped/process.0x83f324d8.0x50000.dmp | grep -E '_.*\.exe'

This filename is notable for containing the organization name "allsafe" (as in AllSafeCybersec), suggesting it is a custom or organization-specific application rather than a standard Windows component. It's important to note the presence of this application name in what appears to be a whitelist of processes within the malware. Malware often includes whitelists to avoid interfering with certain security products or critical system processes, either to prevent system instability that could alarm administrators, or to bypass certain security tools. In this case, allsafe_protector.exe may represent a security or monitoring application specific to the AllSafeCybersec environment.

Answer: allsafe_protector.exe

24. Machine:POS What is the name of the file the malware was initially launched from?

To complete the investigation of the POS system breach, we need to determine how the Dexter malware was initially introduced into the system. Understanding the initial infection vector is crucial for strengthening security measures and preventing similar attacks in the future. Although we have already identified the malware's command and control server and its internal configuration, we still need to determine how it first gained access to the POS system. Internet Explorer often serves as the initial infection vector for malware through downloads from third-party services or malicious websites. The Volatility iehistory plugin is designed to retrieve Internet Explorer browsing history from memory, which can reveal websites visited and files downloaded prior to infection. This information can provide important context about how the malware was initially delivered to the system.

vol2 -f "/work/pos01/POS-01-c4e8f786.vmss" --profile=Win7SP0x86 iehistory | grep '54.84.237.92'

The domain 54.84.237.92 matches the IP address of the command and control server we identified earlier. The file name allsafe_update.exe is deceptive, intended to appear to be a legitimate update for AllSafeCybersec systems.

Answer: allsafe_update.exe