Results of Live Response Inquiry vs. Memory Image Analysis

Maksim Ogur [1], Alexey Dmitrienko [1] *and* Mikhail Kotlov [1]

[1] *North-Caucasus Federal University, Prospect Kulakova, 2, 355000, Stavropol, Russia*

Abstract

People responsible for computer security incident response and digital forensic examination need to continually update their skills, tools, and knowledge to keep pace with changing technology. No longer able to simply unplug a computer and evaluate it later, examiners must know how to capture an image of the running memory and perform volatile memory analysis using various tools, such as PsList, ListDLLs, Handle, Netstat, FPort, Userdump, Strings, and PSLoggedOn. This paper presents a live response scenario and compares various approaches and tools used tocapture and analyze evidence from computer memory.

Keywords

Forensic, Live Response, Memory image analysis, security operations, files, sessions, dashboard, volatile memory, drawbacks.

1. Introduction

Live response gives security operations teams instantaneous access to a device (also referred to as a machine) using a remote shell connection. This gives you the power to do in-depth investigative work and take immediate response actions to promptly contain identified threats in real time.

Live response is designed to enhance investigations by enabling your security operations team to collect forensic data, run scripts, send suspicious entities for analysis, remediate threats, and proactively hunt for emerging threats.

As in [19] authors says, that a special place among IT is occupied by information technology security, which represent the methods and resources necessary to prevent unauthorized access, usage, disclosure, distortion, modification or destruction of information. These information technologies have found application in electronic payment systems (EPS), which process electronic network-based money. Thanks to the use of authentication protocols with zero knowledge disclosure, EPS provide a high protection degree of information transmitted via open Internet channels, anonymity of money owners and security of transactions.

In [27] The proposed logical-probabilistic model is associated with the use of the specifics of the regulatory and legal framework in the field of SCII safety; structured detailing of the CII subject, taking into account the specifics of the subject; stages of the life cycle of the SCII information protection system; highlighted destructive malicious influences of an infrastructural nature; interrelationships of the selected destructs with a number of vulnerabilities on CII objects.

Authors of [22] says about attacks on CPS`s, based on the analysis of changes in network node parameters. One of the main purposes of their work is development a methodology for evaluating the ability of the catch to demonstrate trusted behavior in the normal operation of the network and during attacks.

All of this suggests that the problem of choice and optimizing between live response and static analysis is quite extensive in the tasks of computer forensics.

AISMA-2021: International Workshop on Advanced in Information Security Management and Applications, October 1, 2021, Stavropol, Krasnoyarsk, Russia

EMAIL: ogur26@mail.com (Maksim Ogur); admitrienko770@gmail.com (Alexey Dmitrienko); mikhailits161@yandex.ru (Michail Kotlov)
ORCID: 0000-0002-2387-0901 (Maksim Ogur); 0000-0002-3219-6672 (Alexey Dmitrienko); 0000-0001-5114-1012 (Michail Kotlov)

2. Basic tasks

With live response, analysts can do all of the following tasks:

- Run basic and advanced commands to do investigative work on a device.
- Download files such as malware samples and outcomes of PowerShell scripts.
- Download files in the background (new!).
- Upload a PowerShell script or executable to the library and run it on a device from a tenant level.
- Take or undo remediation actions.

2.1. Live response dashboard overview

During initiation of a live response session on a device, a dashboard opens. The dashboard provides information about the session such as the following:

- Who created the session
- When the session started
- The duration of the session

The dashboard also gives you access to:

- Disconnect session
- Upload files to the library
- Command console
- Command log

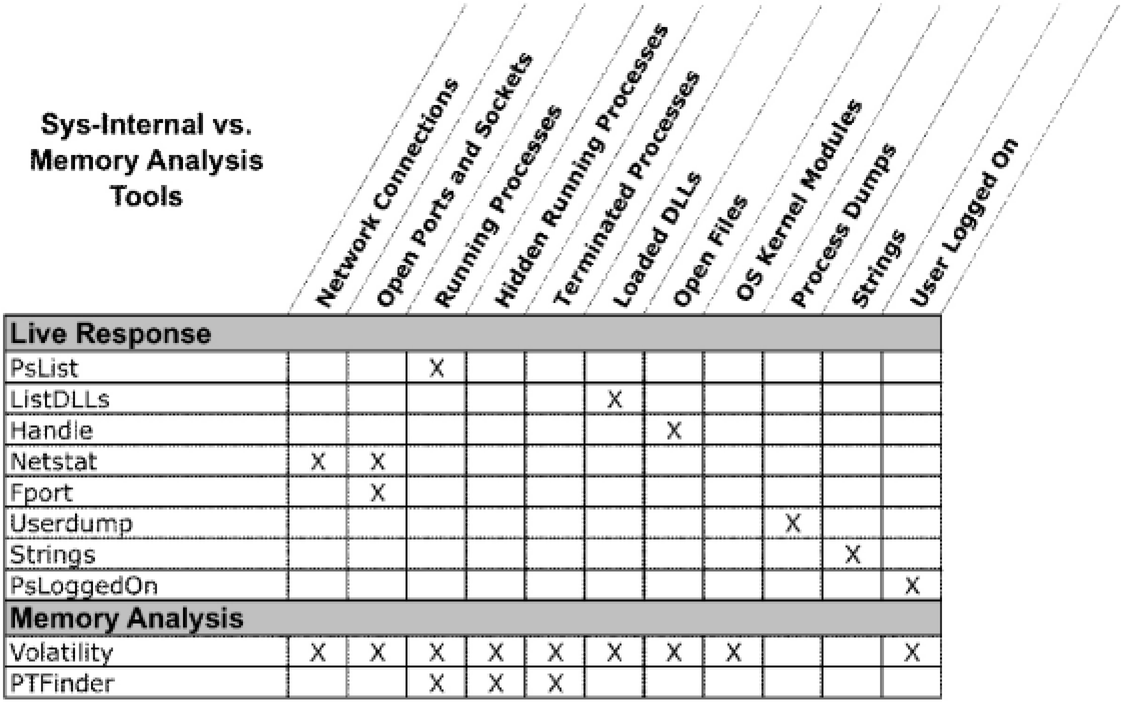
2.2. Initiation of a live response session on a device (Windows platform)

- Sign in to Microsoft 365 Defender portal.
- Navigate to Endpoints > Device inventory and select a device to investigate. The devices page opens.
- Launch the live response session by selecting Initiate live response session. A command console is displayed. Wait while the session connects to the device.
- Use the built-in commands to do investigative work. For more information, see Live response commands.
- After completing your investigation, select Disconnect session, then select Confirm.

2.3. Live response dashboard overview

The traceability matrix of Table 1 is a mapping of the capabilities of live response and memory analysis tools during an investigation of a memory image (or running memory). The Live Response part of Figure 1 lists the tools used in live response, and the Memory Analysis part shows tools that analyze physical memory dumps. This section contains hints for creating and maintaining Word files and suggestions for avoiding common mistakes.

Table 1
Live response with Sys-Internal tools vs. memory analysis on a static memory dump

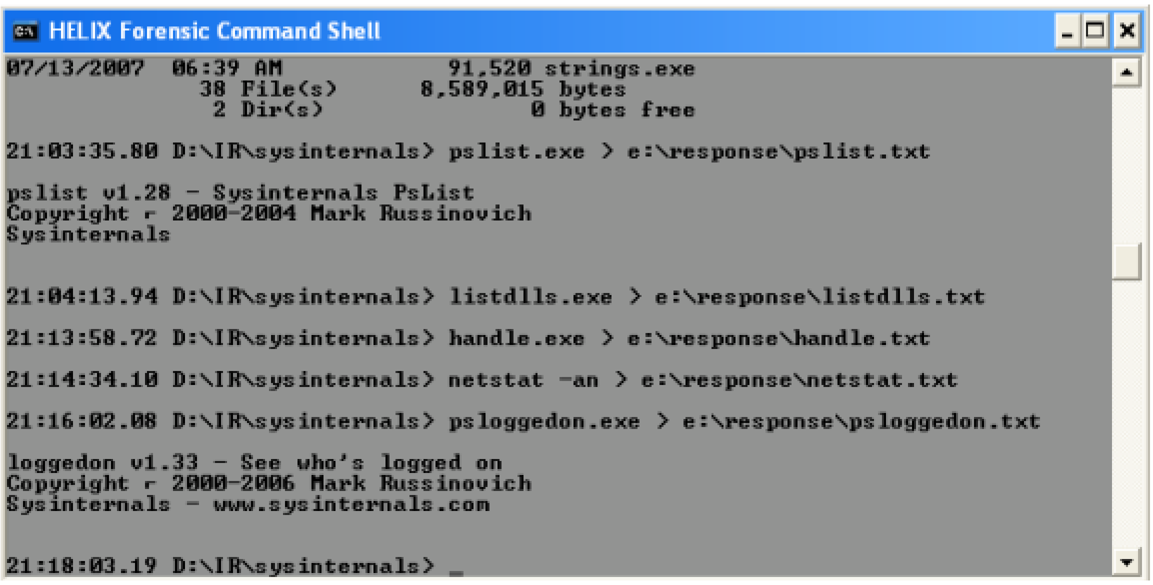


Sys-Internal vs. Memory Analysis Tools	Network Connections	Open Ports and Sockets	Running Processes	Hidden Running Processes	Terminated Processes	Loaded DLLs	Open Files	OS Kernel Modules	Process Dumps	Strings	User Logged On
Live Response											
PsList			X								
ListDLLs						X					
Handle							X				
Netstat	X	X									
Fport		X									
Userdump									X		
Strings										X	
PsLoggedOn											X
Memory Analysis											
Volatility	X	X	X	X	X	X	X	X			X
PTFinder			X	X	X						

In our virtual environment scenario, we start with a Windows XP Service Pack 2 virtual machine with an IP address of 192.168.203.132. Netcat was used to establish a telnet connection on port 4444 (PID: 3572) with a second machine at 192.168.203.133. MACSpoof was also installed and running (PID: 3008). This machine was then compromised by installing the FUTo rootkit and a ProRat server listening on port 5110. The netcat and MACSpoof processes were then hidden using the FUTo rootkit. In the following sections, we present two possible techniques to approach the compromised system and we discuss what details are visible and invisible concerning the various compromises using each approach. The first approach we present is a live response process using sys-internal style tools. The second is a static memory dump analysis using open source memory analysis tools. Finally, we discuss the benefits and drawbacks of both approaches.

2.4. Live Response

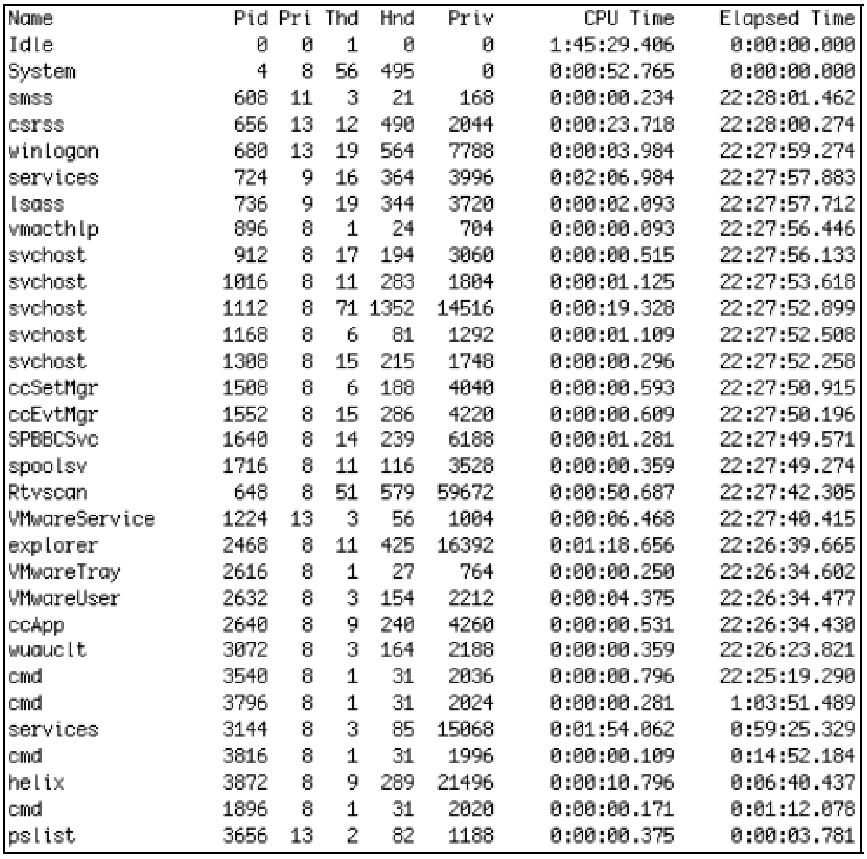
The first approach is live response. Here an investigator would first establish a trusted command shell. In addition, they would establish a method for transmitting and storing the information on a data collection system of some sort. One option is to redirect the output of the commands on the compromised system to the data collection system. One popular tool is netcat, a network utility that transmits data across network connections. Another approach would be to insert a USB drive and write all query results to that external drive. Finally, investigators would attempt to bolster the credibility of the tool output in court. During a live interrogation of a system, it is important to realize that the state of the running machine is not static. This could lead to the same query producing different results based on when it is run. Therefore, hashing the memory is not effective. Rather, an investigator could compute a cryptographic checksum of the tool outputs and make a note of this hash value in the log. This would help dispel any notion that the results had been altered after the fact. In this exercise, HELIX (a live response and Linux bootable CD), was used to establish a trusted command shell.



```
07/13/2007  06:39 AM            91,520 strings.exe
              38 File(s)      8,589,015 bytes
               2 Dir(s)               0 bytes free

21:03:35.80 D:\IR\sysinternals> pslist.exe > e:\response\pslist.txt

pslist v1.28 - Sysinternals PsList
Copyright ⌐ 2000-2004 Mark Russinovich
Sysinternals


21:04:13.94 D:\IR\sysinternals> listdlls.exe > e:\response\listdlls.txt

21:13:58.72 D:\IR\sysinternals> handle.exe > e:\response\handle.txt

21:14:34.10 D:\IR\sysinternals> netstat -an > e:\response\netstat.txt

21:16:02.08 D:\IR\sysinternals> psloggedon.exe > e:\response\psloggedon.txt

loggedon v1.33 - See who's logged on
Copyright ⌐ 2000-2006 Mark Russinovich
Sysinternals - www.sysinternals.com


21:18:03.19 D:\IR\sysinternals> _
```

Figure 1: Trusted command shell established using HELIX

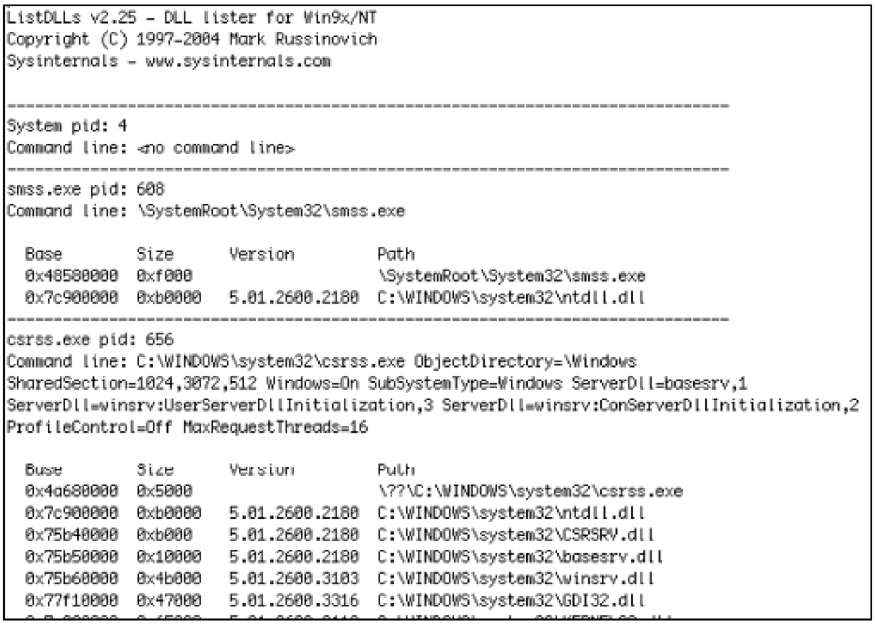
Once the above data collection setup is complete, an investigator can begin to collect evidence from the compromised system. The sys-internal style tools used in this exercise are not meant to be an exhaustive list. Rather, they are representative of the types of tools available. The common thread for the tools used is that each relies on native API calls to some degree, and thus the results are filtered through the operating system. The tools used in this case were PsList, ListDLLs, Handle, Netstat, FPort, Userdump, Strings, and PSLoggedOn.



```
Name              Pid Pri Thd  Hnd   Priv       CPU Time    Elapsed Time
Idle                0   0   1    0      0    1:45:29.406    0:00:00.000
System              4   8  56  495      0    0:00:52.765    0:00:00.000
smss              608  11   3   21    168    0:00:00.234   22:28:01.462
csrss             656  13  12  490   2044    0:00:23.718   22:28:00.274
winlogon          680  13  19  564   7788    0:00:03.984   22:27:59.274
services          724   9  16  364   3996    0:02:06.984   22:27:57.883
lsass             736   9  19  344   3720    0:00:02.093   22:27:57.712
vmacthlp          896   8   1   24    704    0:00:00.093   22:27:56.446
svchost           912   8  17  194   3060    0:00:00.515   22:27:56.133
svchost          1016   8  11  283   1804    0:00:01.125   22:27:53.618
svchost          1112   8  71 1352  14516    0:00:19.328   22:27:52.899
svchost          1168   8   6   81   1292    0:00:01.109   22:27:52.508
svchost          1308   8  15  215   1748    0:00:00.296   22:27:52.258
ccSetMgr         1508   8   6  188   4040    0:00:00.593   22:27:50.915
ccEvtMgr         1552   8  15  286   4220    0:00:00.609   22:27:50.196
SPBBCSvc         1640   8  14  239   6188    0:00:01.281   22:27:49.571
spoolsv          1716   8  11  116   3528    0:00:00.359   22:27:49.274
Rtvscan           648   8  51  579  59672    0:00:50.687   22:27:42.305
VMwareService    1224  13   3   56   1004    0:00:06.468   22:27:40.415
explorer         2468   8  11  425  16392    0:01:18.656   22:26:39.665
VMwareTray       2616   8   1   27    764    0:00:00.250   22:26:34.602
VMwareUser       2632   8   3  154   2212    0:00:04.375   22:26:34.477
ccApp            2640   8   9  240   4260    0:00:00.531   22:26:34.430
wuauclt          3072   8   3  164   2188    0:00:00.359   22:26:23.821
cmd              3540   8   1   31   2036    0:00:00.796   22:25:19.290
cmd              3796   8   1   31   2024    0:00:00.281    1:03:51.489
services         3144   8   3   85  15068    0:01:54.062    0:59:25.329
cmd              3816   8   1   31   1996    0:00:00.109    0:14:52.184
helix            3872   8   9  289  21496    0:00:10.796    0:06:40.437
cmd              1896   8   1   31   2020    0:00:00.171    0:01:12.078
pslist           3656  13   2   82   1188    0:00:00.375    0:00:03.781
```

Figure 2: Results from PSList

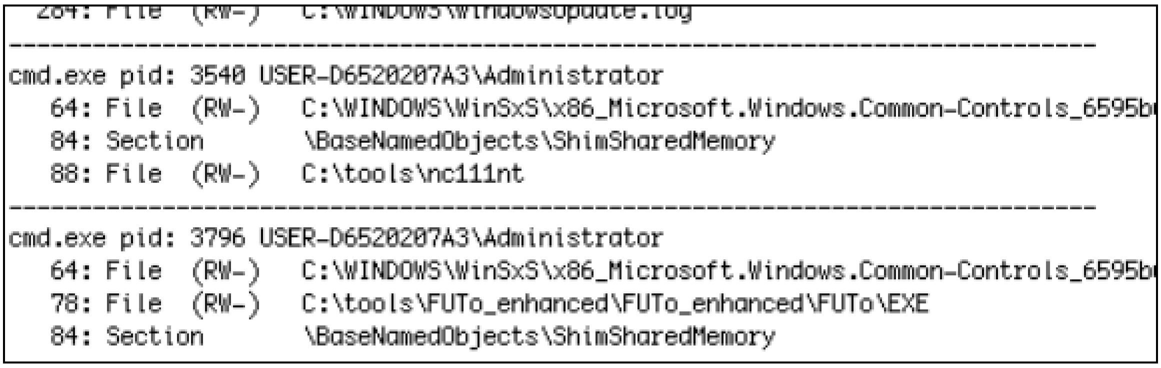
PsList allows investigators to view process and thread statistics on a system. Applying PsList reveals all running processes on the system but does not reveal the presence of the rootkit or the other processes that the rootkit has hidden (netcat and MACSpoof).



```
ListDLLs v2.25 - DLL lister for Win9x/NT
Copyright (C) 1997-2004 Mark Russinovich
Sysinternals - www.sysinternals.com

------------------------------------------------------------------------------
System pid: 4
Command line: <no command line>
------------------------------------------------------------------------------
smss.exe pid: 608
Command line: \SystemRoot\System32\smss.exe

  Base        Size      Version         Path
  0x48580000  0xf000                    \SystemRoot\System32\smss.exe
  0x7c900000  0xb0000   5.01.2600.2180  C:\WINDOWS\system32\ntdll.dll
------------------------------------------------------------------------------
csrss.exe pid: 656
Command line: C:\WINDOWS\system32\csrss.exe ObjectDirectory=\Windows
SharedSection=1024,3072,512 Windows=On SubSystemType=Windows ServerDll=basesrv,1
ServerDll=winsrv:UserServerDllInitialization,3 ServerDll=winsrv:ConServerDllInitialization,2
ProfileControl=Off MaxRequestThreads=16

  Base        Size      Version         Path
  0x4a680000  0x5000                    \??\C:\WINDOWS\system32\csrss.exe
  0x7c900000  0xb0000   5.01.2600.2180  C:\WINDOWS\system32\ntdll.dll
  0x75b40000  0xb000    5.01.2600.2180  C:\WINDOWS\system32\CSRSRV.dll
  0x75b50000  0x10000   5.01.2600.2180  C:\WINDOWS\system32\basesrv.dll
  0x75b60000  0x4b000   5.01.2600.3103  C:\WINDOWS\system32\winsrv.dll
  0x77f10000  0x47000   5.01.2600.3316  C:\WINDOWS\system32\GDI32.dll
```

Figure 3: Excerpt from ListDLLs output

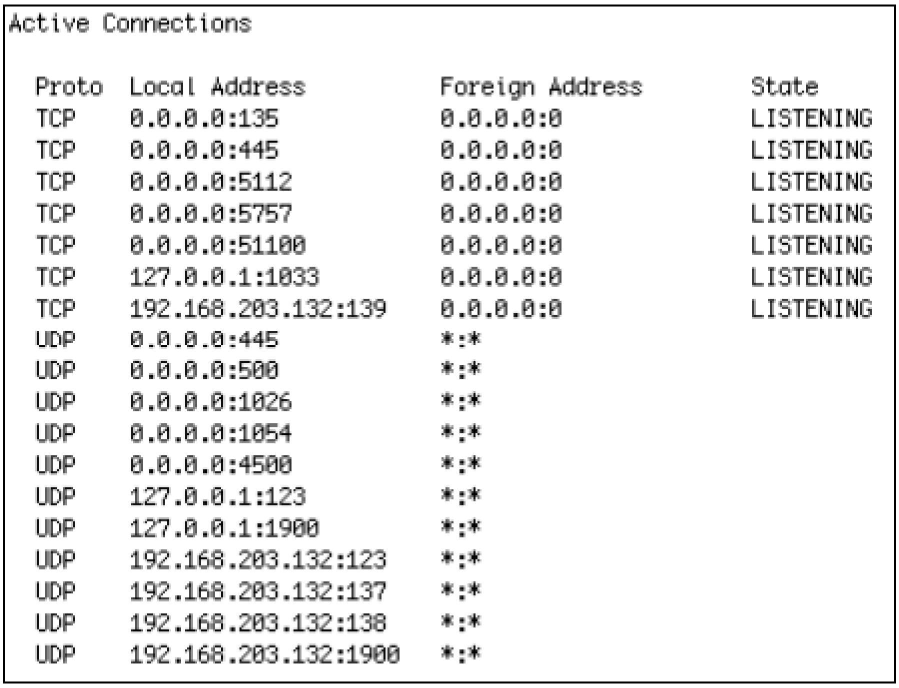
ListDLLs allows investigators to view the currently loaded DLLs for a process. Applying ListDLLs reveals the DLLs loaded by all running processes. However, since there are processes that are hidden, ListDLLs cannot show the DLLs loaded for them. Thus, critical evidence that could reveal the presence of the rootkit is missed. The problem is that an attacker may have compromised the Windows API upon which an investigator's toolkit depends. To a degree, this is the case with our scenario. As a result, rootkit manipulation cannot be easily detected with these tools. A more sophisticated and non-intrusive approach is necessary to find what could be critical evidence.



```
 284: File  (RW-)   C:\WINDOWS\windowsupdate.log
------------------------------------------------------------------------------
cmd.exe pid: 3540 USER-D6520207A3\Administrator
   64: File  (RW-)   C:\WINDOWS\WinSxS\x86_Microsoft.Windows.Common-Controls_6595b
   84: Section       \BaseNamedObjects\ShimSharedMemory
   88: File  (RW-)   C:\tools\nc111nt
------------------------------------------------------------------------------
cmd.exe pid: 3796 USER-D6520207A3\Administrator
   64: File  (RW-)   C:\WINDOWS\WinSxS\x86_Microsoft.Windows.Common-Controls_6595b
   78: File  (RW-)   C:\tools\FUTo_enhanced\FUTo_enhanced\FUTo\EXE
   84: Section       \BaseNamedObjects\ShimSharedMemory
```

Figure 4: Excerpt from Handle output

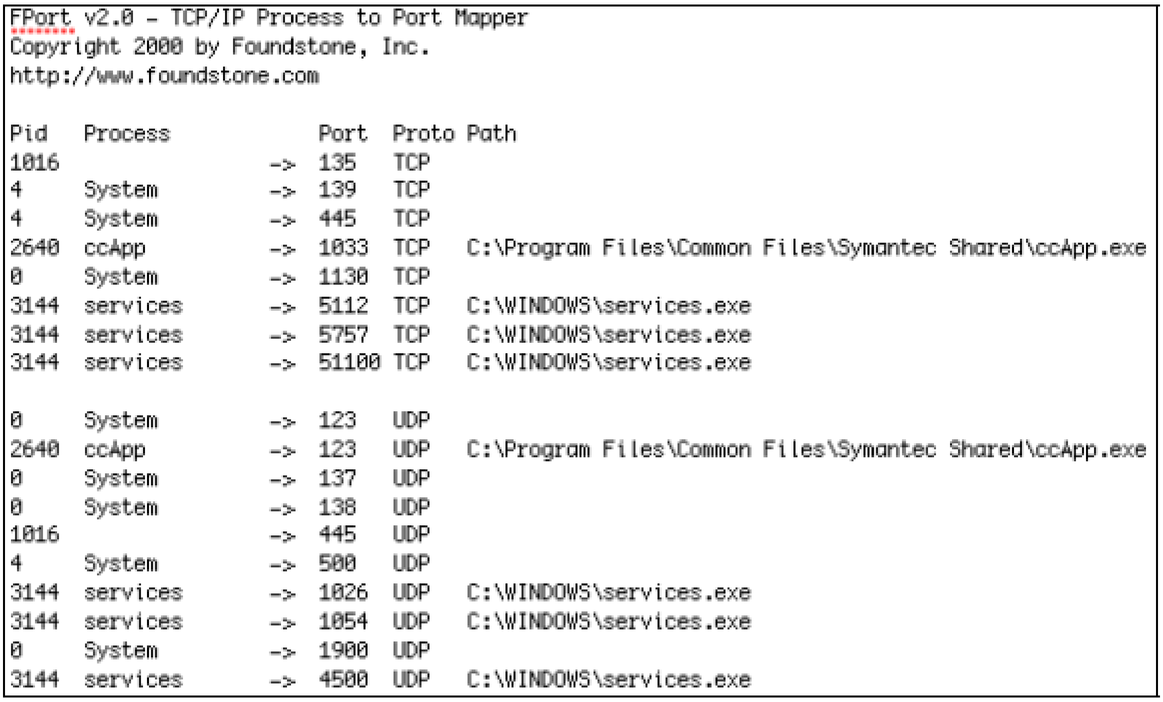
The Handle utility allows investigators to view open handles for any process. It reveals the open files for all the running processes, which includes the path to the file. In this case, one of the command shells is running from a directory labeled …\FUTo\EXE. This is a strong hint of the presence of the FUTo rootkit. Similarly, there is another instance of cmd.exe running from C:\tools\nc11nt. The nc11nt folder is a default for the windows distribution of netcat. While it is useful to show the implications of the tool results, it is important to remember that simply renaming these directories or running the cmd.exe from a different directory would have prevented these disclosures.



```
Active Connections

  Proto  Local Address          Foreign Address        State
  TCP    0.0.0.0:135            0.0.0.0:0              LISTENING
  TCP    0.0.0.0:445            0.0.0.0:0              LISTENING
  TCP    0.0.0.0:5112           0.0.0.0:0              LISTENING
  TCP    0.0.0.0:5757           0.0.0.0:0              LISTENING
  TCP    0.0.0.0:51100          0.0.0.0:0              LISTENING
  TCP    127.0.0.1:1033         0.0.0.0:0              LISTENING
  TCP    192.168.203.132:139    0.0.0.0:0              LISTENING
  UDP    0.0.0.0:445            *:*
  UDP    0.0.0.0:500            *:*
  UDP    0.0.0.0:1026           *:*
  UDP    0.0.0.0:1054           *:*
  UDP    0.0.0.0:4500           *:*
  UDP    127.0.0.1:123          *:*
  UDP    127.0.0.1:1900         *:*
  UDP    192.168.203.132:123    *:*
  UDP    192.168.203.132:137    *:*
  UDP    192.168.203.132:138    *:*
  UDP    192.168.203.132:1900   *:*
```

Figure 5: Netstat results

The Netstat utility allows investigators to view the network connections of a running machine. Nestat (with the –an option) reveals nothing immediately suspicious in this case.



```
FPort v2.0 - TCP/IP Process to Port Mapper
Copyright 2000 by Foundstone, Inc.
http://www.foundstone.com

Pid   Process            Port  Proto Path
1016                 ->  135   TCP
4     System         ->  139   TCP
4     System         ->  445   TCP
2640  ccApp          ->  1033  TCP   C:\Program Files\Common Files\Symantec Shared\ccApp.exe
0     System         ->  1130  TCP
3144  services       ->  5112  TCP   C:\WINDOWS\services.exe
3144  services       ->  5757  TCP   C:\WINDOWS\services.exe
3144  services       ->  51100 TCP   C:\WINDOWS\services.exe

0     System         ->  123   UDP
2640  ccApp          ->  123   UDP   C:\Program Files\Common Files\Symantec Shared\ccApp.exe
0     System         ->  137   UDP
0     System         ->  138   UDP
1016                 ->  445   UDP
4     System         ->  500   UDP
3144  services       ->  1026  UDP   C:\WINDOWS\services.exe
3144  services       ->  1054  UDP   C:\WINDOWS\services.exe
0     System         ->  1900  UDP
3144  services       ->  4500  UDP   C:\WINDOWS\services.exe
```

Figure 6: Results of FPort

3. Analysis

Thus far, was described incident response approaches to the scenario discussed. The approach is the well-known live response where an investigator surveys the crime scene, collects the evidence, and at the same time probes for suspicious activity. The approach is the relatively new field of volatile memory analysis where an investigator collects the memory dump and performs analysis in an isolated environment. In different approaches, was described what types of information gave an investigator insight into the scenario. Now, it will be discussed some of the issues with live response that hinder effective analysis of a digital crime scene. It also will be discussed why volatile memory analysis should be the ideal approach to investigating cyber crime. While the purpose of live response is to collect all relevant evidence from the system that will likely be used to confirm whether an incident occurred, the implementation of the process has significant setbacks, including the following:

- First Responder toolkit may rely on Windows API: The problem is that if an attacker compromises the system and changes system files without an investigator suspecting, then an investigator could collect a large amount of evidence that is based on compromised sources. As a result, this would damage the credibility of the analysis in a court of law.
- Live response is not repeatable: The information in memory is volatile and with every passing second, bytes are being overwritten. As we saw in our scenario, the tools may produce the correct output and in themselves can be verified by a third-party expert. However, the input data supplied to them can never be reproduced. As a result, this puts the evidence collected at risk in a court of law. Therefore, it becomes difficult for investigators to prove the correctness of their analysis of the evidence. [Walters 2007].
- Investigators cannot ask new questions later: The live response process does not support examination of the evidence in a new way. This is mainly because the same inputs to the tools from the collection phase cannot be reproduced. As a result, investigators cannot ask new questions later on in the analysis phase of the investigation [Walters 2007]. By the analysis phase, it becomes impossible to learn anything new about the compromise. In addition, as we saw in our scenario, once critical evidence is missed during collection, it can never be recovered again. It damages the case against the attacker.

On the other hand, a volatile memory analysis shows promise in that the only source of evidence is the physical memory dump. Moreover, collection of physical memory has become more commonly practiced. An investigator can then build the case by analyzing the memory dump in an isolated environment that is non-obtrusive to the evidence. Thus, volatile memory analysis addresses the drawbacks facing live response as follows:

- It limits impact to the compromised system: Unlike live response, memory analysis uses a simplified approach to investigating a crime scene. It involves merely extracting the memory dump and minimizes the fingerprint left on the compromised system. In addition, the nature of live response puts the analysis of the evidence at risk in a court of law. As a result, an investigator gets the added benefit of analyzing the memory dump fully confident that the impact to the data is minimal.
- Analysis is repeatable: Since the memory dumps are analyzed directly and in isolated environments, this allows for multiple sources to validate and repeat the analysis. We saw this in our scenario, where the hidden malware processes were identified by the two tools. In addition, it allows for conclusions made by investigators to be verified by third-party experts. Essentially, it improves the credibility of the analysis in a court of law.
- Nature of analysis supports asking new questions later: Contrary to live response, memory analysis allows investigators with more expertise, technique, or understanding to ask new questions later on in the investigation [Walters 2007]. We saw this in our scenario. Our initial analysis of the memory dump with Volatility gave us some suspicion of a rootkit being present on the system. We later confirmed this with evidence of the terminated rootkit process using the Lsproc script. This important evidence may have been missed in a live response.

One of the greatest drawbacks with volatile memory analysis is that the tools’ support has not matured enough. This is because with every release of a new operating system, the physical memory structure changes. Development of memory analysis tools has been gaining velocity recently, but the kinks still remain. This is an emerging field and new ground is being broken across the area of study.

4. Acknowledgements

The work was carried out using the equipment of the Center for Collective Use of North-Caucasus Federal University with financial support from the Ministry of Science and Higher Education of Russian Federation, unique project identifier RF —— 2296.61321X0029 (agreement no. 075-15-2021-687).

5. References

[1] Carrier, B. 2019 Open Source Digital Forensics Tools
[2] Olajide, F.; Savage, N.; Ndzi, D.; Al-Sinani, H. 2018 Forensic Live Response and Event Reconstruction Methods in Linux Systems
[3] Thomas, D.S.; Forcht, K.A. 2004 Legal methods of using computer forensics techniques for computer crime analysis and investigation (Issues Inf. Syst. 2004, 5) pp 692-698.
[4] Harrell, C. 2011 What's a Timeline
[5] Esposito, S.; Peterso, G. 2013 Creating Super Timelines in Windows Investigations (Proceedings of the 9th International Conference on Digital Forensics, Orlando, FL, USA, 28-30 January 2013) pp. 135-144.
[6] James, J.I.; Gladyshev, P. Automated inference of past action instances in digital investigations) Int. J. Inf. Secur. 2015, 14) pp 249-261, https://doi.org/10.1007/s10207-014-0249-6.
[7] Inglot, B.; Liu, L. 2014 Enhanced Timeline Analysis for Digital Forensic Investigations. (Inf. Secur. J. Glob. Perspect. 2014, 23) pp 32-44, https://doi.org/10.1080/19393555.2014.897401.
[8] Guðjónsson, K. 2010 Mastering the Super Timeline Who Am I?
[9] Sitompul, O.S.; Handoko, A.; Rahmat, R.F. 2018 File Reconstruction in Digital Forensic (TELKOMNIKA Indones. J. Electr. Eng. 2018, 16) pp 776-794, https://doi.org/10.12928/TELKOMNIKA.v16i2.8230
[10] Cho, G.S. 2013 A computer forensic method for detecting timestamp forgery in NTFS (Comput. Secur. 2013, 34) pp 36-46, https://doi.org/10.1016/j.cose.2012.11.003.
[11] Kalber, S.; Dewald, A.; Freiling, F.C. 2013 Forensic application-fingerprinting based on file system metadata (Proceedings of the Seventh International Conference on IT Security Incident Management and IT Forensics, Nuremberg, Germany, 12-14 March 2013) pp 98-112.
[12] Bang, J.; Yoo, B.; Kim, J.; Lee, S. 2009 Analysis of time information for digital investigation (Proceedings of the Fifth International Joint Conference on INC, IMS and IDC, Seoul, Korea, 25-27 August 2009) pp 1858-1864.
[13] Chabot, Y.; Bertaux, A.; Nicolle, C.; Kechadi, M.T. 2014 A complete formalized knowledge representation model for advanced digital forensics timeline analysis (Digit. Investig. 2014, 11) pp 95-105, https://doi.org/10.1016/j.diin.2014.05.009.
[14] Hargreaves, C.; Patterson, J. 2012 An automated timeline reconstruction approach for digital forensic investigations (Digit. Investig. 2012, 9) pp 69-79, https://doi.org/10.1016/j.diin.2012.05.006.
[15] Brady, O.; Overill, R. 2015 DESO: Addressing volume and variety in large scale criminal cases (Digit. Investig. 2015, 88) pp 72-825, https://doi.org/10.1016/j.diin.2015.10.002.
[16] Brady, O.; Overill, R.; Keppens, J. 2014 Addressing the increasing volume and variety of digital evidence using an ontology (Proceedings of the 2014 IEEE Joint Intelligence and Security Informatics Conference, Hague, The Netherlands, 24-26 September 2014) pp 176-183.
[17] Debinski, M.; Breitinger, F.; Mohan, P. 2019 Timeline2GUI: A Log2timeline CSV parser and training scenarios (Digit. Investig. 2019, 28) pp 34-43, https://doi.org/10.1016/j.diin.2018.12.004.
[18] Soltani, S.; Seno, S.A.H.; Yazdi, H.S. 2019 Event reconstruction using temporal pattern of file system modification (IET Inf. Secur. 2019, 13) pp 201-212, DOI:10.1049/iet-ifs.2018.5209.
[19] Chistousov, N.K., Kalmykov, I.A., Lapina, M.A., Kalmykov, M.I. Application of Information Security Technologies for Improving the Imitation Resistance of Low-Orbital Satellite Communication Systems. Lecture Notes in Networks and System, 2021, 228, pp. 54-63, https://doi.org/10.1007/978-3-030-77448-6_6

[20] Grigoryan, K., Olefirenko, E., Basan, E., Lapina, M., Mecella, M. Analysis of Security Problems in Groups of Intelligent Sensors. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2021, 12689 LNCS, pp. 26-37, https://doi.org/10.1007/978-3-030-78743-1_3

[21] Basan, E., Lapina, M., Mudruk, N., Abramov, E. Intelligent Intrusion Detection System for a Group of UAVs, Lecture Notes in Computer Science, 2021, 12690 LNCS, pp. 230-240, https://doi.org/10.1007/978-3-030-78811-7_22

[22] Basan, A.S., Basan, E.S., Lapina, M.A., Lapin, V.G. Behavior-Based Assessment of Trust in a Cyber-Physical System. Communications in Computer and Information Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2021, 1395 CCIS, pp. 190-201, https://doi.org/10.1007/978-981-16-1480-4_17

[23] Privalov, A.N., Bogatyreva, Y.I., Lapina, M.A., Lapin, V.G., Mysina, Y.A. Decision support information system for patient treatment procedures in hospital // CEUR Workshop Proceedings, 2021, 2914, pp. 441-448, http://ceur-ws.org/Vol-2914/paper44.pdf.

[24] Maria Lapina, Ksenia Lokhacheva, Denis Parfenov. Designing of Information System for Semantic Analysis and Classification of Issues in Service Desk System // YRID-2020 Proceedings of the International Workshop on Data Mining and Knowledge Engineering Stavropol, Russia, October 15-16, 2020. CEUR Workshop Proceedings, 2021, 2842, Pp. 70-76, http://ceur-ws.org/Vol-2842/paper_8.pdf.

[25] Parfenov, D.I., Bolodurina, I.P., Lapina, M.A. Development of a model for detecting security incidents in event flows from various components in a network of telecommunication service providers // IOP Conference Series: Materials Science and Engineering 2020, 873(1), 012020, https://doi.org/10.1088/1757-899X/873/1/012020.

[26] Proshkin, N.A., Basan, E.S., Lapina, M.A., Klepikova, A.G., Lapin, V.G. Developing models of IoT infrastructures to identify vulnerabilities and analyse threats // IOP Conference Series: Materials Science and Engineering, 2020, 873(1),012018, https://doi.org/10.1088/1757-899X/873/1/012018.

[27] E. A. Maksimova, M. A. Lapina, V. V. Baranov and O. S. Lauta The logical-probabilistic model for assessing the information security assessing of the critical information infrastructure subject under destructive influences. 2nd Scientific Conference on Fundamental Information Security Problems in terms of the Digital Transformation (FISP 2020) 30 November 2020, Stavropol, Russian Federation. – IOP Conference Series: Materials Science and Engineering, Vol. 873, 2021, 012035. https://doi.org/10.1088/1757-899X/1069/1/012035.