BazarISO Analysis - Loading with Advpack.dll

forensicitguy.github.io/bazariso-analysis-advpack/

January 22, 2022

By *Tony Lambert*
Posted *2022-01-22* Updated *2022-03-28* *8 min* read

Malware comes in all shapes and sizes, and in the case of BazarISO it comes in the form of an ISO file that contains a malicious shortcut and an executable. In this post I'll tear apart the ISO to show how one of the more recent BazarISO samples works. If you want to follow along at home, I'm using the sample from MalwareBazaar here: https://bazaar.abuse.ch/sample/38cf92de5c97f9f79ddfb5632ac92f2670f3aa25414943735ddbe24507ad49f3/

Triage and Unpack the ISO

First, let's make sure the file is an ISO with `file`.

```
remnux@remnux:~/cases/bazariso$ file Documents-
17.iso
Documents-17.iso: ISO 9660 CD-ROM filesystem data
''
```

Alright, let's take a stab at unpacking with `7z`.

```
remnux@remnux:~/cases/bazariso$ 7z x Documents-17.iso

7-Zip [64] 16.02 : Copyright (c) 1999-2016 Igor Pavlov : 2016-05-21
p7zip Version 16.02 (locale=en_US.UTF-8,Utf16=on,HugeFiles=on,64 bits,2 CPUs
Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz (806EA),ASM,AES-NI)

Scanning the drive for archives:
1 file, 743424 bytes (726 KiB)

Extracting archive: Documents-17.iso
--
Path = Documents-17.iso
Type = Iso
Physical Size = 743424
Created = 2022-01-21 09:15:47

Everything is Ok

Files: 2
Size:       689865
Compressed: 743424
```

And 7-zip gave us the very life-affirming message that `Everything is OK` since both files in the ISO unpacked properly. Let’s see what files we have.

```
remnux@remnux:~/cases/bazariso$ ls -lah
total 1.4M
drwxrwxr-x  2 remnux remnux 4.0K Jan 22 17:44 .
drwxrwxr-x 10 remnux remnux 4.0K Jan 22 17:24 ..
-rw-rw-r--  1 remnux remnux 673K Jan 21 09:15 autorun.exe
-rw-rw-r--  1 remnux remnux 1.2K Jan 21 09:15 docs.lnk
-rw-r--r--  1 remnux remnux 726K Jan 22  2022 Documents-
17.iso
```

It looks like the ISO contained two files, an `autorun.exe` and a `docs.lnk` file.

Inspecting the LNK File

Thinking through the chain of actions a victim is likely to take, the victim will try to double-click the ISO file and Windows will mount it as a removable drive. The victim then will click either `autorun.exe` or `docs.lnk`. Shortcut LNK files often contain shady material because they allow a creator to specify command line arguments in the shortcut to perform arbitrary actions. We can triage this file and analyze it using `file` and `exiftool`.

```
remnux@remnux:~/cases/bazariso$ file docs.lnk
docs.lnk: MS Windows shortcut, Item id list present, Points to a file or
directory, Has Relative path, Has command line arguments, Icon number=5,
Archive, ctime=Mon Dec 27 02:31:16 2021, mtime=Fri Jan 21 17:35:47 2022,
atime=Mon Dec 27 02:31:16 2021, length=71680, window=hide

remnux@remnux:~/cases/bazariso$ exiftool docs.lnk
ExifTool Version Number         : 12.30
File Name                       : docs.lnk
Directory                       : .
File Size                       : 1225 bytes
File Modification Date/Time     : 2022:01:21 09:15:47-05:00
File Access Date/Time           : 2022:01:22 17:51:01-05:00
File Inode Change Date/Time     : 2022:01:22 17:42:19-05:00
File Permissions                : -rw-rw-r--
File Type                       : LNK
File Type Extension             : lnk
MIME Type                       : application/octet-stream
Flags                           : IDList, LinkInfo, RelativePath, CommandArgs,
IconFile, Unicode, TargetMetadata
File Attributes                 : Archive
Create Date                     : 2021:12:26 16:31:16-05:00
Access Date                     : 2022:01:21 07:35:47-05:00
Modify Date                     : 2021:12:26 16:31:16-05:00
Target File Size                : 71680
```

```
Icon Index                      : 5
Run Window                      : Normal
Hot Key                         : (none)
Target File DOS Name            : rundll32.exe
Drive Type                      : Fixed Disk
Volume Label                    :
Local Base Path                 : C:\Windows\System32\rundll32.exe
Relative Path                   : ..\Windows\System32\rundll32.exe
Command Line Arguments          : advpack.dll,RegisterOCX autorun.exe
Icon File Name                  : %systemroot%\system32\imageres.dll
Machine ID                      : desktop-i8bn9qk
```

Alright we definitely have a LNK shortcut file! Inspecting with `exiftool` it looks like the shortcut is fairly small at 1225 bytes. Larger shortcut files may indicate very large PowerShell or scripting command line properties. In this case, it looks like the command the shortcut executes is `C:\Windows\System32\rundll32.exe advpack.dll,RegisterOCX autorun.exe`. This command is a way to execute `autorun.exe` using `rundll32.exe` as a LOLBIN. The icon is one from a default Windows installation, but some LNK files I've seen have had ones distributed with the files as well. The last bit of detail in this output is the Machine ID. This property shows the computer name of the system that created the shortcut. So, the adversary either created this shortcut on a system named `desktop-i8bn9qk` or knows how to modify the shortcut file to that name.

Triage and Estimate Capabilities

Alright, let's see if we can estimate some of the capabilities of the `autorun.exe` binary using `capa` and `yara`.

```
remnux@remnux:~/cases/bazariso$ capa autorun.exe

+------------------------+------------------------------------------------------------
------------------------------+
| md5                    | 6d583d7666ffbc439f86f8954cc3e0ec
|
| sha1                   | d17ff6f48a3e3693ee61b79341ed282087df2e71
|
| sha256                 |
667753d0c33cf7874b3d4cf05be4cf245558515e73330e133c60da63554471d8
|
| path                   | autorun.exe
|
+------------------------+------------------------------------------------------------
------------------------------+

+------------------------+------------------------------------------------------------
------------------------------+
| ATT&CK Tactic          | ATT&CK Technique
|
|------------------------+------------------------------------------------------------
------------------------------|
| COLLECTION             | Input Capture::Keylogging T1056.001
|
| DEFENSE EVASION        | Modify Registry:: T1112
|
|                        | Obfuscated Files or Information:: T1027
|
|                        | Obfuscated Files or Information::Indicator Removal
from Tools T1027.005           |
| DISCOVERY              | File and Directory Discovery:: T1083
|
|                        | Query Registry:: T1012
```

```
|
|                        | System Information Discovery:: T1082
|
| EXECUTION              | Shared Modules:: T1129
|
+------------------------+-------------------------------------------------------
------------------------------+

+-----------------------------+--------------------------------------------------
------------------------------+
| MBC Objective               | MBC Behavior
|
|-----------------------------+--------------------------------------------------
------------------------------|
| COLLECTION                  | Keylogging::Polling [F0002.002]
|
| DATA                        | Encode Data::XOR [C0026.002]
|
| DEFENSE EVASION             | Obfuscated Files or Information::Encoding-
Standard Algorithm [E1027.m02]      |
| DISCOVERY                   | Application Window Discovery::Window Text
[E1010.m01]                         |
| FILE SYSTEM                 | Delete File:: [C0047]
|
|                             | Read File:: [C0051]
|
|                             | Writes File:: [C0052]
|
| OPERATING SYSTEM            | Environment Variable::Set Variable [C0034.001]
|
|                             | Registry::Create Registry Key [C0036.004]
|
|                             | Registry::Delete Registry Key [C0036.002]
|
|                             | Registry::Open Registry Key [C0036.003]
|
|                             | Registry::Query Registry Value [C0036.006]
|
|                             | Registry::Set Registry Key [C0036.001]
|
| PROCESS                     | Set Thread Local Storage Value:: [C0041]
|
|                             | Terminate Process:: [C0018]
|
+-----------------------------+--------------------------------------------------
------------------------------+

+------------------------------------------------------+-------------------------
------------------------------+
| CAPABILITY                                           | NAMESPACE
|
|------------------------------------------------------+-------------------------
------------------------------|
| log keystrokes via polling                           | collection/keylog
|
| encode data using XOR (2 matches)                    | data-
manipulation/encoding/xor                       |
| contain a resource (.rsrc) section                   |
executable/pe/section/rsrc                             |
| extract resource via kernel32 functions (8 matches)  | executable/resource
|
| query environment variable                           | host-
```

```
interaction/environment-variable                |
| set environment variable                             | host-
interaction/environment-variable                |
| get common file path                                 | host-interaction/file-
system                         |
| delete file                                          | host-interaction/file-
system/delete                  |
| enumerate files via kernel32 functions (2 matches)   | host-interaction/file-
system/files/list              |
| get file size                                        | host-interaction/file-
system/meta                    |
| read .ini file (2 matches)                           | host-interaction/file-
system/read                    |
| read file                                            | host-interaction/file-
system/read                    |
| write file (2 matches)                               | host-interaction/file-
system/write                   |
| get graphical window text                            | host-
interaction/gui/window/get-text                 |
| get disk information                                 | host-
interaction/hardware/storage                    |
| set thread local storage value                       | host-
interaction/process                             |
| terminate process                                    | host-
interaction/process/terminate                   |
| query or enumerate registry value (3 matches)        | host-
interaction/registry                            |
| set registry value                                   | host-
interaction/registry/create                     |
| delete registry key (2 matches)                      | host-
interaction/registry/delete                     |
| access PEB ldr_data (3 matches)                      | linking/runtime-linking
|
| link function at runtime (15 matches)                | linking/runtime-linking
|
| resolve function by hash (2 matches)                 | linking/runtime-linking
|
| parse PE exports (3 matches)                         | load-code/pe
|
| parse PE header (10 matches)                         | load-code/pe
|
+------------------------------------------------------+-----------------------
------------------------------+
```

From the `capa` output there are already some capabilities in here that will likely increase analysis time. These include:

- access PEB ldr_data
- link function at runtime
- resolve function by hash

The PEB ldr_data rule indicates the binary contains some assembly instructions that resolve DLL module lists from the process environment block of the process while it is running. This is a method used by shellcode to resolve DLL imports and pain-in-the-can malware to hide their imports. Linking functions at runtime means the sample likely issues `LoadLibrary()` or similar calls to import DLLs at runtime instead of when the program first runs. Finally, resolving functions by hash means it'll be a hassle to potentially see what functions are being resolved as they'll be hashed strings rather than the clear strings. Let's see what we get from YARA.

```
remnux@remnux:~/cases/bazariso$ yara-rules
autorun.exe
Check_OutputDebugStringA_iat autorun.exe
anti_dbg autorun.exe
win_hook autorun.exe
screenshot autorun.exe
keylogger autorun.exe
win_registry autorun.exe
win_files_operation autorun.exe
IsPE64 autorun.exe
IsWindowsGUI autorun.exe
HasRichSignature autorun.exe
Microsoft_Visual_Cpp_80_DLL autorun.exe
```

YARA thinks the sample has some anti-debugging, so that might become a hassle while doing further analysis later. From here my analysis style would be to toss this sucker into a sandbox to see what it does because further analysis is going to produce diminishing returns for me.

Thanks for reading!