Decrypt MassLogger 2.4.0.0 configuration

By NexusFuzzy

Published: 2020-08-19 · Archived: 2026-04-05 17:53:36 UTC

Press enter or click to view image in full size

The malware MassLogger has been around for some time and different analysis approaches have been published in the past — for example by FireEye.

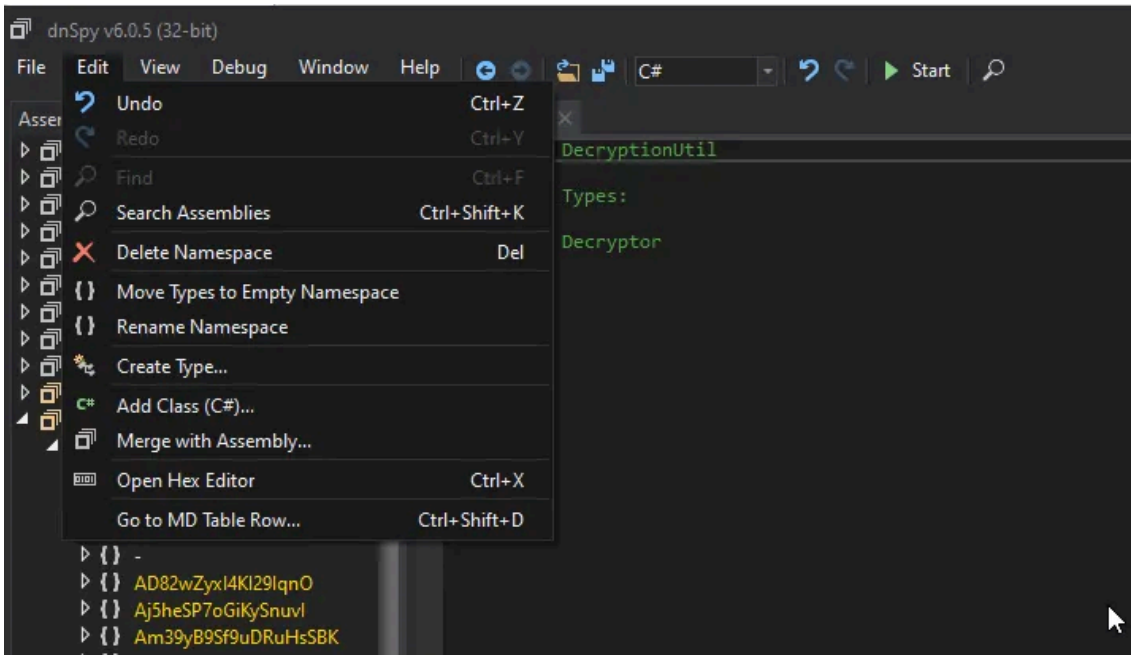
Unfortunately, this approach didn't work in my case mainly because I later realized that I was dealing with a MassLogger sample with version 2.4.0.0 while the one analyzed by FireEye seemed to be version 1.3.4.0.

So, what now? If you are just interested in the used config itsself to find out local and network IOCs I have some good news for you.

Note: Those steps only work on the MassLogger binary itsself. In most cases it has to get unpacked before you can start to decrypt the config. One easy check is to search with dnSpy for "**FtpEnable**" or any other value you know will be present in the config. If you are able to find these references you are good to go!

To do this open up dnSpy, load the binary and **Edit** > **Search Assemblies**

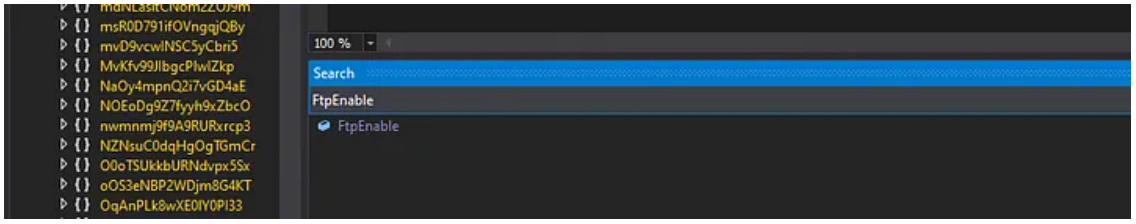
Press enter or click to view image in full size



Edit > Search Assemblies

Now search for “FtpEnable”:

Press enter or click to view image in full size



Search for FtpEnable was successful and you are looking at the MassLogger sample

Now that we are certain that we are looking at the MassLogger sample itsself (you may have used Yara rules to be certain it is MassLogger, too), we can now apply our trick to get all the decrypted config values.

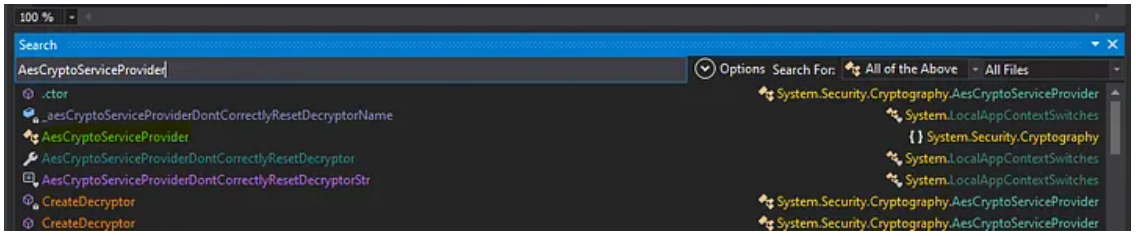
Get NexusFuzzy’s stories in your inbox

Join Medium for free to get updates from this writer.

Remember me for faster sign in

Just add “AesCryptoServiceProvider” to the search field we used before and open the corresponding search result:

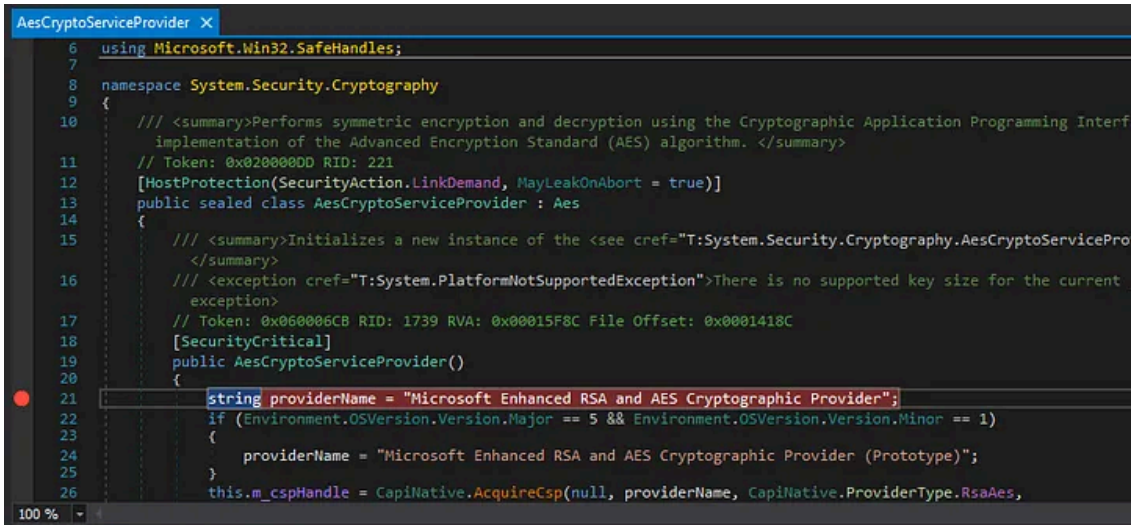
Press enter or click to view image in full size



Open AesCryptoServiceProvider from System.Security.Cryptography with a double click

Once you opened up the file, you set a breakpoint:

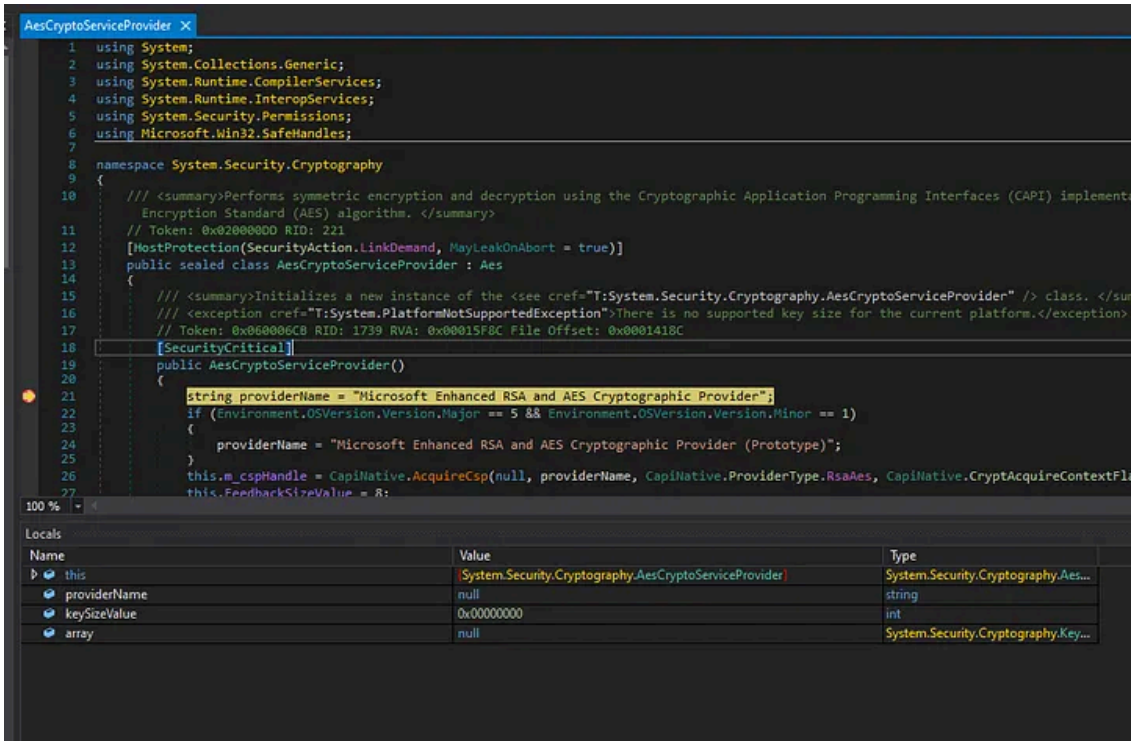
Press enter or click to view image in full size



This way, you will always hit this breakpoint when MassLogger makes use of AES to decrypt its config strings. We will need this breakpoint just once to get back to the calling function in the MassLogger binary itsself.

Now you can run the sample (in a sandbox of course) and after some seconds the breakpoint we created will be hit.

Press enter or click to view image in full size

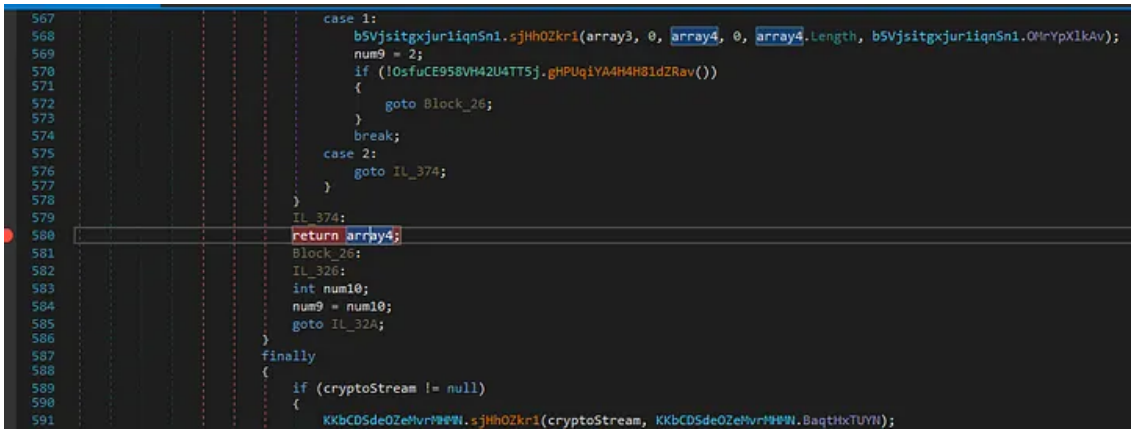


Breakpoint was hit

Once the breakpoint was hit, press Shift + F11 (Step out of function) and you'll be right back in your MassLogger binary where all the decryption takes place.

Once you are back, you have to find where the function you are currently in returns. In my case it looks like this:

Press enter or click to view image in full size

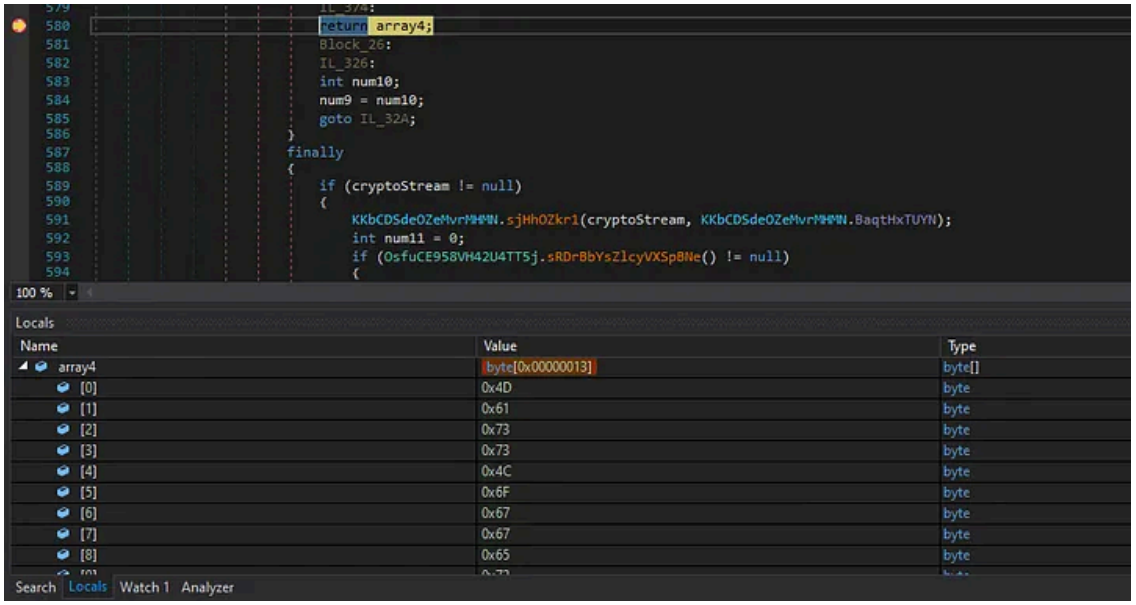


Add a breakpoint on the return of the function

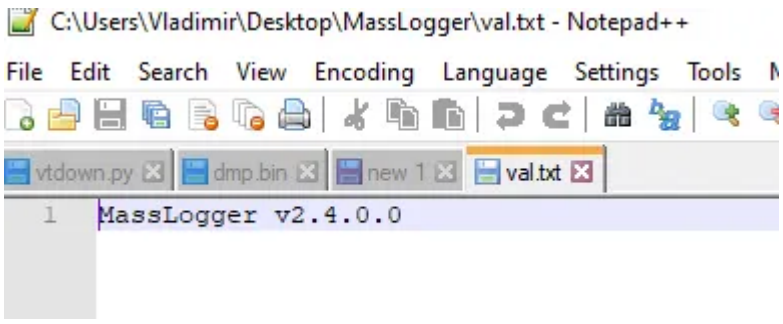
Now add a breakpoint to this line. This way, you'll hit this breakpoint whenever a part of the configuration file gets decrypted. You can now also safely disable the breakpoint in System.Security.Cryptography since we only needed that one to jump back into MassLogger to set our "real" breakpoint.

If you've set your breakpoint and investigate the value of array4 you'll see:

Press enter or click to view image in full size



Which can be dumped and viewed in an editor:



Dumped config value

This process can be repeated for every config item you’re interested in!

Source: https://medium.com/@mariohenkel/decrypt-masslogger-2-4-0-0-configuration-eff3ee0720a7