

Forum:[Hack Forums](#)

Thread: [\[GigaCLR\] \[CUSTOM LOADER\] Run .NET assembly inside any process](#)

gigajew - September 5, 2019, 4:50 pm

Information

Here's my latest native module, that has the capability of loading the .NET framework inside a process and executing a .NET assembly purely from memory. No droppings needed apart from the module itself (which you could manually map). If you were to inject this module into a native process and call it with your parameters, you could achieve a .NET assembly running inside a native process. Otherwise you could just use it as a means of a custom loader inside your .NET stub too.

Disclaimer

This is just for proof of concept and educational purposes. Anything you choose to do with this is your responsibility. This code is based off of the open source C++ project here:

<https://github.com/ChadSki/SharpNeedle>

Pics or it didn't happen

Requirements

It's compiled as 32-bit so naturally you'd have to compile as 32-bit as well. EDIT: Also included 64-bit version of this dll, so now you can use either.

Specifications and technical information

The module exports one function called "GigaRun" and is defined in the header as so:

Code `DllExport GigaRun(void* pBuffer, int dwBufferLength, LPCWSTR strFrameworkVersion, int dwArgumentLength);`

And the parameters would be described like this: `void* pBuffer` - a pointer to your payload (in .net it would be defined as a regular byte array and passed by val, because it's already a reference object

`int dwBufferLength` - the size of your .net payload in bytes (`Array.Length`)

`LPCWSTR strFrameworkVersion` - (regular unicode string) the version of the framework you want your payload to run in (e.g. "v4.0.30319")

`int dwArgumentLength` - the number of arguments of your .NET assembly (traditionally 1 for console apps, and 0 for winforms apps, Quasar for example uses 1)

Examples

C#

VB.NET

Downloads

Download the modules here: <https://www.mediafire.com/file/20zfkaisum1txni/gigaclr.zip/file>

Public virus scan here (gigaclr.dll):

<https://www.virustotal.com/gui/file/076f14aef7bf9d3db8eced4b7a09c078c31dc51d028d75e6cc93d473f7cb6677/detection>

Hashes:

Code gigaclr.dll (application/x-msdownload) - 9728 bytes

MD5

SHA-1 b0efd15ba317ae9e08bdee3a9a1d71394a5e0660

SHA-256 076f14aef7bf9d3db8eced4b7a09c078c31dc51d028d75e6cc93d473f7cb6677

SHA-384

8cc9774eba755d7e564217ba9a2f68ee2fa0f085d614bb99b9eb7099ca1312726eec5823aa5347788

26b0a3541270b2a

SHA-512

f6dde7652756af4de09786dfe9fb368d6a5f16579b569966e86ac463ed70ead4a43f84f34c74d6296fee

3d0b6898f44005c5ae649d16b422f23e38b6ba29f799

Cooooool - September 5, 2019, 8:20 pm

Nice release dude, I'm glad people like you still exist.

riu41776 - September 6, 2019, 7:08 am

>"Anything you choose to do with this is your responsibility."

What can we possibly do with it to risk ourselves?

Suspect. - September 6, 2019, 11:42 am

Nice release Giga. This should be useful.

ixfep - September 7, 2019, 10:58 am

Damn! Nice release! Is there to add parameters?

gigajew - September 7, 2019, 10:58 am

(Sep 7, 2019 10:58 AM)ixfep Wrote: [\[CITE\]](#) Damn! Nice release! Is there to add parameters?

Not at the moment sorry

n0\$f3ratu\$ - September 7, 2019, 11:10 am

(Sep 5, 2019 04:50 PM)gigajew Wrote: [\[CITE\]](#)

You might want to remove this: "C:\Users\Home\Desktop\Bootstrapper\Release\gigaclr."

gigajew - September 7, 2019, 4:21 pm

(Sep 7, 2019 11:10 AM)n0\$f3ratu\$ Wrote: [\[CITE\]](#) You might want to remove this:

"C:\Users\Home\Desktop\Bootstrapper\Release\gigaclr."

Is it pdb paths or something?

Stanley - November 7, 2019, 8:36 pm

I'm Stanley HF's helper bot. This OP (original poster) of this thread wants everyone to see this page and has elected to bump it to the top of the showthread listing. Since HF doesn't want a ton of LQP (Low Quality Posts) this is the best method of bumping your content. There is a small ?yte fee for each bump. Any OP can bump any of their threads on HF from the linked icon on the top of their first post.

If you haven't already read the first post of this thread we encourage you to do so.

Thanks,
Stanley

Paskowsky - November 8, 2019, 7:37 am

Nice idea!! But someone did an upgrade to this concept :

<https://github.com/TheWover/donut>

Quote from github :

"Donut generates x86 or x64 shellcode from VBScript, JScript, EXE, DLL (including .NET Assemblies) files. This shellcode can be injected into an arbitrary Windows processes for in-memory execution. Given a supported file type, parameters and an entry point where applicable (such as Program.Main), it produces position-independent shellcode that loads and runs entirely from memory."

Take a look

Regards,
Paskowsky

gigajew - November 8, 2019, 9:32 pm

(Nov 8, 2019 08:37 AM)Paskowsky Wrote: [\[CITE\]](#) Nice idea!! But someone did an upgrade to this concept :

<https://github.com/TheWover/donut>

Quote from github :

"Donut generates x86 or x64 shellcode from VBScript, JScript, EXE, DLL (including .NET Assemblies) files. This shellcode can be injected into an arbitrary Windows processes for in-memory execution. Given a supported file type, parameters and an entry point where applicable (such as Program.Main), it produces position-independent shellcode that loads and runs entirely from memory."

Take a look

Regards,
Paskowsky
That's very interesting. Does it load the CLR and everything?

Zephyros - November 10, 2019, 5:02 am

Its not that difficult to generate position independent code (aka shellcode) from C: [Exploit Monday: Writing Optimized Windows Shellcode in C](#)

You just need seperate code for PE and .NET loading to generate a stub and come up with some scheme to append and maybe decrypt the payload.

I have done it myself, but only for loading a PE in the same process: [hastebin](#)

AMGTEch - November 10, 2019, 3:55 pm

nice share keep up the good work !!

SIM-4440595 - November 16, 2019, 4:15 pm

Fantastic release. I'm learning this type of shit and this is very helpful for me. Thank you very much!

Arenna - November 17, 2019, 3:10 am

Would you say this provides better runtime detection for AV's over Assembly.Load or Invoke? I'm not sure if those methods are flagged either, cryptography is not really what I'm interested in. However I have made malware in the past that used Load and Invoke hoping it wouldn't cause any

detection's.

gigajew - November 17, 2019, 9:07 am

I'm sure you could make it evade detection if you wanted to yeah

Arrenna - November 17, 2019, 10:03 am

(Nov 17, 2019 10:07 AM)gigajew Wrote: [\[CITE\]](#) I'm sure you could make it evade detection if you wanted to yeah

Okay, would you mind if I incorporated this in a project of mine? I'd give you full credits of course

gigajew - November 17, 2019, 10:47 am

(Nov 17, 2019 11:03 AM)Arrenna Wrote: [\[CITE\]](#) Okay, would you mind if I incorporated this in a project of mine? I'd give you full credits of course

Go right ahead this is now public property basically