

Forum:[Hack Forums](#)

Thread: [Project File Auto Exec Detection](#)

JoshuaSmith - January 4, 2018, 11:35 am

So there was a post a while back that shows you how to backdoor vs project file, even know this was known and can be done with almost any IDE I'm going to show you a simple way to detect the backdoor. Also i use the term backdoor because i can't see why anyone would need to download and run software from the project header file but if you have a valid reason then cool idc.

What Components are used within this example:

1 Richtextbox

1 Textbox

1 button

Openfiledialog(was lazy so i added this as a component but you can call it with programmatically)

Example:

```
Code Me.OpenFileDialog1.InitialDirectory = Application.StartupPath
      Me.OpenFileDialog1.FileName = ""
      Me.OpenFileDialog1.Filter = "C# Project File (*.csproj)|*.csproj|VB Project File (*.vbproj)|*.vbproj|ALL Files (*.*)|*.*"

      If OpenFileDialog1.ShowDialog = Windows.Forms.DialogResult.OK Then
          RichTextBox1.Text = System.IO.File.ReadAllText(OpenFileDialog1.FileName)
          Timer1.Enabled = True
          Label1.Text = "Status: Checking project file for backdoors...."
      Else

      End If

      If RichTextBox1.Text.Contains("Exec") Then
          Label1.Text = "Status: This source contains a backdoor."
          RichTextBox1.Clear()

      Else
          Label1.Text = "Status: This source is clean."
          RichTextBox1.Clear()

      End If
```

Image of it being used:

PS: idc about making this clean or nice its just an example so if you care more about how i call things then you will be talking to yourself.

Reason of making this example:

[VS Code Execution](#)

Unknown - January 4, 2018, 1:20 pm

I've heard of this type of exploit before, never really thought people used it.

It's really cool that you made a project scanner for it though, I would have never thought about that
0.0

I'm amazed at how simplistic it is. Great release!

Ninja - January 4, 2018, 6:02 pm

(Jan 4, 2018 02:20 PM)Cynn. Wrote: [\[CITE\]](#) I've heard of this type of exploit before, never really thought people used it.

It's really cool that you made a project scanner for it though, I would have never thought about that
0.0

I'm amazed at how simplistic it is. Great release!

It's not really an exploit though. It is pretty much a feature in VS. The 'scanner' just looks for the string "Exec", which can be used without ill intend in a VS project. Would be better to have it flag project files that contain "Exec" and then have the user be able to download the files and scan them on an online virus scanner.

JoshuaSmith - January 4, 2018, 8:04 pm

(Jan 4, 2018 07:02 PM)Ninja Wrote: [\[CITE\]](#) It's not really an exploit though. It is pretty much a feature in VS. The 'scanner' just looks for the string "Exec", which can be used without ill intend in a VS project. Would be better to have it flag project files that contain "Exec" and then have the user be able to download the files and scan them on an online virus scanner.

you are right but i don't see what other purpose it would have. it would be simple to make a system to detect new headers and check them.

Ninja - January 4, 2018, 8:07 pm

(Jan 4, 2018 09:04 PM)xmods1337 Wrote: [\[CITE\]](#) you are right but i don't see what other purpose it would have. it would be simple to make a system to detect new headers and check them.

Maybe third party programs that are in executable form are required to work with the project? I can't think of many uses either, but since the Microsoft dev team decided to implement it I am sure there is a use.

Barret - January 5, 2018, 8:29 am

Been a long time ive laughed so hard XD

Washi - January 5, 2018, 10:01 am

PHP Code

```
<?php
```

```
If RichTextBox1.Text.Contains("Exec") Then
```

Whelp, suddenly all my projects consisting of files with a name containing Exec will be flagged.

It's like saying:

PHP Code

```
<?php
```

```
If File.Contains("Virus") Then
```

```
    MessageBox.Show("Virus detected!")
```

```
End If
```

And it would flag all anti virus programs too :/

Ninja - January 5, 2018, 12:13 pm

I kind of agree with the dude above me, and as harsh as it may sound, but this is ridiculous. A tutorial for checking a string for an occurrence of another string?

JoshuaSmith - January 5, 2018, 12:54 pm

(Jan 5, 2018 11:01 AM)Washi Wrote: [\[CITE\]](#) PHP Code

```
<?php
```

```
If RichTextBox1.Text.Contains("Exec") Then
```

Whelp, suddenly all my projects consisting of files with a name containing Exec will be flagged.

It's like saying:

PHP Code

```
<?php
```

```
If File.Contains("Virus") Then
```

```
    MessageBox.Show("Virus detected!")
```

```
End If
```

And it would flag all anti virus programs too :/

well obviously you don't code what so ever because I'm not checking file names. cheers for making yourself the stupidest person on hf :)

(Jan 5, 2018 01:13 PM)Ninja Wrote: [\[CITE\]](#) I kind of agree with the dude above me, and as harsh as it may sound, but this is ridiculous. A tutorial for checking a string for an occurrence of another string?

i can't see how unless you follow in the same ignorant lines he does.

(Jan 5, 2018 09:29 AM)Barret Wrote: [\[CITE\]](#) Been a long time ive laughed so hard XD

i know this shit is garbage but its just an example and i really couldn't careless but not possible as the guy below you just made himself look so fucking stupid come back

Washi - January 5, 2018, 2:35 pm

(Jan 5, 2018 01:54 PM)xmods1337 Wrote: [\[CITE\]](#) well obviously you don't code what so ever because I'm not checking file names. cheers for making yourself the stupidest person on hf :)

Oh shit you got me.

And obviously you don't know the format of the csproj and vbproj files ? .

I wonder what these are in one of my csproj files:

Spoiler [\(Click to View\)](#)

JoshuaSmith - January 6, 2018, 7:43 am

(Jan 5, 2018 03:35 PM)Washi Wrote: [\[CITE\]](#) Oh shit you got me.

And obviously you don't know the format of the csproj and vbproj files ? .

I wonder what these are in one of my csproj files:

Spoiler ([Click to View](#))

Well sir those are the classes you get when importing the name spaces from the .net framework's sdk which allows you to call functions.

at one point yesterday i looked at your github and thought i might have made a mistake or you just didnt read the code before posting. now im 98.7 % sure you are not a programmer.

Vyxn - January 6, 2018, 7:50 am

That Washi was an incredible guy. I've check his work and I must say that his knowledge is so much far ahead from yours. ?

Washi - January 6, 2018, 9:11 am

(Jan 6, 2018 08:43 AM)xmods1337 Wrote: [\[CITE\]](#) Well sir those are the classes you get when importing the name spaces from the .net framework's sdk which allows you to call functions.

at one point yesterday i looked at your github and thought i might have made a mistake or you just didnt read the code before posting. now im 98.7 % sure you are not a programmer.

You are 98.7% sure I am not a programmer. That means there is still a chance that I am! Phew, lucky me.

Unfortunately you are wrong about the namespaces from the .NET Framework's SDK :(First of all these are not namespaces or classes that are defined here, these are source code files. Secondly, these are my own files containing my own classes. Have a look at the reference of the MSBuild project file format. The Compile tag is part of the Item tag class, which instructs the msbuild evaluator to include a file for compilation (source:

<https://msdn.microsoft.com/en-us/library/ms164283.aspx>).

Do I really have to show you my solution explorer? Let us compare, shall we?:

Spoiler ([Click to View](#))

Looks awfully similar, right? If you are still not convinced, have a better look at <https://github.com/Washi1337/AsmResolver/tree/master/AsmResolver> and download the project yourself :D

Seriously though, I am not trying to bash you or anything, I am just stating that your method of detecting backdoors is not so reliable, as you are testing the entire project xml file, and not just the parts that may include the Exec tag.

GhostCode - January 6, 2018, 9:47 am

Your way of detecting a "backdoor" is basically looking for a substring in a text file.

Reading your replies to the previous posts shows that you have literally no idea what you are talking about.

Your code would even flag a .csproj file as dangerous if the project name is Exec because the

project name is stored in the .csproj file as well.

So, even tho there is no real use case for this code you can at least use XML parsing to find the tag

...

JoshuaSmith - January 6, 2018, 11:02 am

(Jan 6, 2018 10:11 AM)Washi Wrote: [\[CITE\]](#) You are 98.7% sure I am not a programmer. That means there is still a chance that I am! Phew, lucky me.

Unfortunately you are wrong about the namespaces from the .NET Framework's SDK :(First of all these are not namespaces or classes that are defined here, these are source code files. Secondly, these are my own files containing my own classes. Have a look at the reference of the MSBuild project file format. The Compile tag is part of the Item tag class, which instructs the msbuild evaluator to include a file for compilation (source:

<https://msdn.microsoft.com/en-us/library/ms164283.aspx>).

Do I really have to show you my solution explorer? Let us compare, shall we?:

Spoiler ([Click to View](#))

Looks awfully similar, right? If you are still not convinced, have a better look at <https://github.com/Washi1337/AsmResolver/tree/master/AsmResolver> and download the project yourself :D

Seriously though, I am not trying to bash you or anything, I am just stating that your method of detecting backdoors is not so reliable, as you are testing the entire project xml file, and not just the parts that may include the Exec tag.

if took the time to read the thread you would see

This with in my thread:

Code PS: idc about making this clean or nice its just an example so if you care more about how i call things then you will be talking to yourself.

yes 98.7% because i did have a look and hoped what you said earlier what misinterpreted by me or miss said by you.

@ [GhostCode](#): xml parsing would be better and yea reading the file is an unreliable way to do it but im just trying to make a simple poc to show people how easy it would be to detect this within the project header.

also some of the those source files have names that look similar to the namespaces or references within the sdk but i did see some odd ones like LazyValue.cs. also they are class files/namespaces which still call with in the sdk so.....

HFDone - January 6, 2018, 11:37 am

looking for "<Exec " will be better

```
Dim f As New IO.DirectoryInfo("folder")
```

```
For Each n In f.GetFiles("*.proj", IO.SearchOption.AllDirectories)
```

```
    If IO.File.ReadAllText(n.FullName).Contains("<Exec ") Then
```

...

Washi - January 6, 2018, 11:54 am

(Jan 6, 2018 12:02 PM)xmods1337 Wrote: [\[CITE\]](#) if took the time to read the thread you would see

This with in my thread:

Code PS: idc about making this clean or nice its just an example so if you care more about how i call things then you will be talking to yourself.

And again you completely missed the point that I was trying to make. I am not talking about how clean your code looks or whatever. If the intention of my reply was to comment on your coding style, then I would have criticised you for things like using a RichTextBox to contain the contents of the file instead of just a simple string variable, not naming your variables and components etc. But that is not what I am criticising your code for. Instead I am criticising your method for not being reliable. I am making the exact same point as GhostCode is making, yet you are somewhat acknowledging his criticism and not mine ?

(Jan 6, 2018 12:02 PM)xmods1337 Wrote: [\[CITE\]](#) yes 98.7% because i did have a look and hoped what you said earlier what misinterpreted by me or miss said by you.

Ok man, whatever. You're taking that a little too serious...

(Jan 6, 2018 12:02 PM)xmods1337 Wrote: [\[CITE\]](#) @ [GhostCode](#): xml parsing would be better and yea reading the file is an unreliable way to do it but im just trying to make a simple poc to show people how easy it would be to detect this within the project header.

But that's exactly our point. Checking the entire file for one occurrence of a string might be easy to implement, but is really not a great way of detecting a backdoor. Try creating a new project with the name "Exec", and test your application. You will see that your method will flag the project as malicious, even though it does not contain any Exec tag whatsoever.

(Jan 6, 2018 12:02 PM)xmods1337 Wrote: [\[CITE\]](#) also some of the those source files have names that look similar to the namespaces or references within the sdk but i did see some odd ones like LazyValue.cs. also they are class files/namespaces which still call with in the sdk so.....

I am not sure what you mean by this. The files in the screenshots, they contain classes and are put in folders that represent the namespaces yes, but they are nowhere to be found in the SDK from what I can tell. And of course these files will call members of the SDK. I mean every VB or C# project will have a reference to members in the SDK. But I really don't see your point.

You are posting this in a coding section, so you can expect to get replies with questions, criticism or pointers for improvement of your code. What other comments did you expect from this thread?

I'm so confused ?

Ninja - January 6, 2018, 12:57 pm

(Jan 6, 2018 12:54 PM)Washi Wrote: [\[CITE\]](#) You are posting this in a coding section, so you can expect to get replies with questions, criticism or pointers for improvement of your code. What other comments did you expect from this thread?

OP expected praise for his ability to use <string>.Contains("Exec").

Sn0wX - January 11, 2018, 1:49 pm

What a clueless prat you are ninja. "who checks a string against a string"

Heaven forbids any One wants to check a file at a certain line in their file.

(Jan 5, 2018 01:13 PM)Ninja Wrote: [\[CITE\]](#) I kind of agree with the dude above me, and as harsh as it may sound, but this is ridiculous. A tutorial for checking a string for an occurrence of another string?

yeh who would ever do as above

OP expected praise for his ability to use <string>.Contains("Exec").