

Cfinclude for Good or Evil

Posted At : May 14, 2009 11:23 AM | Posted By : Mark Kruger

Related Categories: ColdFusion, Coldfusion Security

Yesterday I was doing some searches on a sick server to troubleshoot the **iframe Injection** issue. A user had posted some additional information regarding a file that appeared on his server that had this issue. The file was named "fection.cfm" so we now know the hacker casually removes his prefixes (or I should say 'emoves his 'efixes). I began my search by looking for the file specifically, then moved on to look for the string "cfexecute" in all of the *.cfm files. But that got me thinking. A clever hacker might know some things about ColdFusion. He could in fact, further obscure his code with some knowledge of cfinclude and IIS. Such a technique can be used to secure your code as well. You can create code that is only runnable by ColdFusion using cfinclude. Here's the skinny.

First, understand that IIS 6 *excludes* all files unless it is specifically tasked with serving it. It does this by examining the file extension. It knows, for example, to serve an *.html file as text/html. In the old days you could create any extension you wanted, throw it on the web server and then call it from a browser. IIS would assume (given lack of instruction to the contrary) that it was an HTML file and serve it. So if you created a hello world page, named it "hello.bob" and then called /hello.bob from a browser, the web server would happily serve your file. Try that on IIS 6 and you get "page cannot be found".

On the other hand, cfinclude doesn't know or care about file extensions. You could create hello.bob and then create hello.cfm with this code:

```
<cfinclude template="hello.bob"/>
```

And your hello world would display correctly. In fact, you can put any kind of CFML in hello.bob that you wish and ColdFusion will execute it happily as if it was a cfm page.

Using for Good, Not for Evil

When is this useful? Actually it can be a very useful technique for excluding certain pages from direct execution. Consider a folder structure like this:

```
/
/
/events
/display
/queries
```

Let's suppose I have index.cfm in my root (/) directory and I am including an event template, a query template and a display template:

```
<!--- /index.cfm --->
<!--- some event logic --->
<cfinclude template="events/handler.cfm"/>
```

```
<!--- some lookups --->
<cfinclude template="queries/qry.cfm"/>
<!--- display stuff --->
<Cfinclude template="display/default.cfm"/>
```

This is pretty common stuff. In fact most frameworks have some similar features (although CFCs further abstract the logic blocks). What you may not notice is that it *might* be possible to browse directly to a script. For example, I might be able to do something like `www.mydomain.com/events/handler.cfm`. Sure, this could throw an error, but it could also serve as a new vector for attack. For the sake of argument let's say your `index.cfm` file looked like this:

```
... logic to set up some variables ...
<!--- --->
<Cfif someconditionIsMet>
    <cfset siteID = 5/>
<cfelse>
    <cfset siteID = 10/>
</CFIF>

<cfinclude template="queries/qry.cfm">
```

The included `qry.cfm` file looks like this:

```
<cfquery name="getsiteProps" datasource="#dsn#">
    SELECT    siteProps
    FROM      sites
    WHERE     siteID = #siteId#
</cfquery>
```

Are you with me so far? Good. Now looking at the code above you might think, "Hey Muse, my query code is safe. After all I'm explicitly setting the `siteID` to either 5 or 10 in the preceding code. I don't even need `cfqueryparam`." Yes, and you are free to make faces at the dinner table too - but if your face sticks like that don't come crying to me. Actually, the issue here is that I (the clever hacker) can bypass `index.cfm` and browse to `www.mydomain.com/queries/qry.cfm`. ColdFusion has a convenient (and risky if you don't understand it) order of precedence in evaluating scopes (variables, then url, then form). To hack your site table I could use the following:

```
/queries/qry.cfm?siteID=5%20OR%201%3d1
```

Which translates to:

```
<cfquery name="getsiteProps" datasource="#dsn#">
    SELECT    siteProps
    FROM      sites
    WHERE     siteID = 5 OR 1 = 1
</cfquery>
```

Of course with the clever use of a line break (`%0A`) I could add "truncate table sites" and really cause a aneurysm.

But let's say you are clever too and you decide you don't want to be able to serve `qry.cfm` from the address bar. You have a few choices. You could move the included files *outside of the web root* and use a ColdFusion mapping (this is pretty common). You could add code to the `application.cfc` or `application.cfm` files that exclude any direct access by examining the template path (this is the way some of the frameworks handle it). Or you could change all of your included sub files to be non-CFM files (`qry.inc` for example). Your `index.cfm` page would then look like this:

```
... logic to set up some variables ...
<!--- --->
<Cfif someconditionIsMet>
    <cfset siteID = 5/>
<cfelse>
    <cfset siteID = 10/>
</CFIF>

<cfinclude template="queries/qry.inc">
```

If someone tried to browse to /queries/qry.inc they would get a page not found error. Now if you were to choose this approach I have one very important tip. Do *not choose an extension served by IIS*. For example, if you changed the extension to .txt, someone browsing to /queries/qry.txt would actually be served the file *without any CF involvement*. In other words they would be looking at your source code (Yikes - like sic em to a dog or sending Uncle Harry to Golden Corral).

What About Using it for Evil

My original point had something to do with Iframe injection right? Remember what I was doing when I started? I was looking for cfexecute *in all the *.cfm files*. What if my clever hacker, who obviously has the ability to add files to the server, has put the malicious code into a file with some other extension and is simply using *cfinclude*. My search would never pick it up. So now I'm forced to search *all files* instead - a much slower process.

On a side note (for those of you dealing with this), this attack seems unlike others I have battled. It seems very much *customized* to the server in question - as if I was matching wits with an actual person who continues to tap tap tap away at the server. That would also be a good reason why the signature of the attack is not picked up by common scans and security tools I guess.