Iframe Injection Follow Up

Posted At : April 21, 2009 11:29 AM | Posted By : Mark Kruger
Related Categories: ColdFusion, Security

For those of you who have been following the Iframe injection attack saga (see **Iframe Insertion on Index.* Home pages**) I have an update. I would like to thank one of my readers named Kumar for referring me to **this excellent article** (a PDF File) on Black Hat. The article seems to pinpoint the origin and nature of the attack. The document describes an attack in depth with multiple steps (just as we had speculated). The first step was an SQLi attempt. But failing that the attacker compromised the server in a rather ingenious fashion.

- Using an image upload capability he uploaded a file to the server that "looked" like an image but was not.
- The file (containing executable code) was then hit with GET and POST requests.
- The payload of the get and post requests was able to set up scheduled tasks to append the JS code to "index.*" files on a timed basis.

This file that was uploaded was a CDX file. On a properly configured IIS server this attack would fail to succeed. Here's why.

All That Useful Stuff

When you install Windows (and now days even some Linux installs have this problem) the installer wants you to feel like you have received your moneys worth. Consequently, the default install of Windows includes a boat load of features that you do not need and should remove. For example, did you know your server has a print spooler service? It's probably running right now. How often do you actually print from your server? If you don't need it, you should disable it - but make sure and turn off the errors in the print folders "server properties" area advanced tab first or your log file will fill up with silly errors (Oh Windows you mischievous thing you...).

One of the "installed by default" features in IIS are the "configuration mappings". You have probably seen these mappings if you have ever had trouble with getting your IIS web site "connected" to ColdFusion. By default IIS contains many mappings that are never used (probably by anyone). On a Windows 2000 server the screen looks like this:

On 2003 it looks like this (very similar):

In the case of our exploit the user was able to use an image uploader to put a file on the server with a CDX extension. The file header indicated that it was of the "type" gif, so the uploader accepted it as a valid image file. I'm not sure if this is unique to CDX files (I'm not sure if you could do the same thing with an "ASP" file for example). IIS has a mapping for CDX that tells it to use the ASP.dll to parse the file. So our too clever hacker uploaded a file that the *uploader took to be a gif* but was in actuality an ASP script. Once on the server the attacker used HTTP to send POST and GET requests to the file delivering payloads that searched for other vulnerabilities (like the ability to open a command shell for example). So once this file is on the server and able to be served by IIS from the web root, all bets are off.

2 Tips

One thing that would have prevented this attack would have been locking down IIS after install and before sites were added. When I do a new windows install I removed everything from the config tab except what I know the server will need explicitly. In particular these extensions serviced by the classic ASP technology need to be removed (cdr, cdx, htr, cer).

Secondly, if you have an application that accepts files from users you should be filtering them to insure that they are appropriate for type and extension (for example using the "accept" attribute of the cffile tag). Probably even more important however is that you not serve these particular files in a way that they can be executed on the server. For example, you can use CFCONTENT to serve the file. You could also use a separate domain (images.mydomain.com etc) for your user content, and configure this new domain to not have permissions to execute scripts. Here's an **article** from security focus giving more information on the topic of securing IIS and Windows. FYI - the use of IIS Lockdown would probably have blocked this extension.