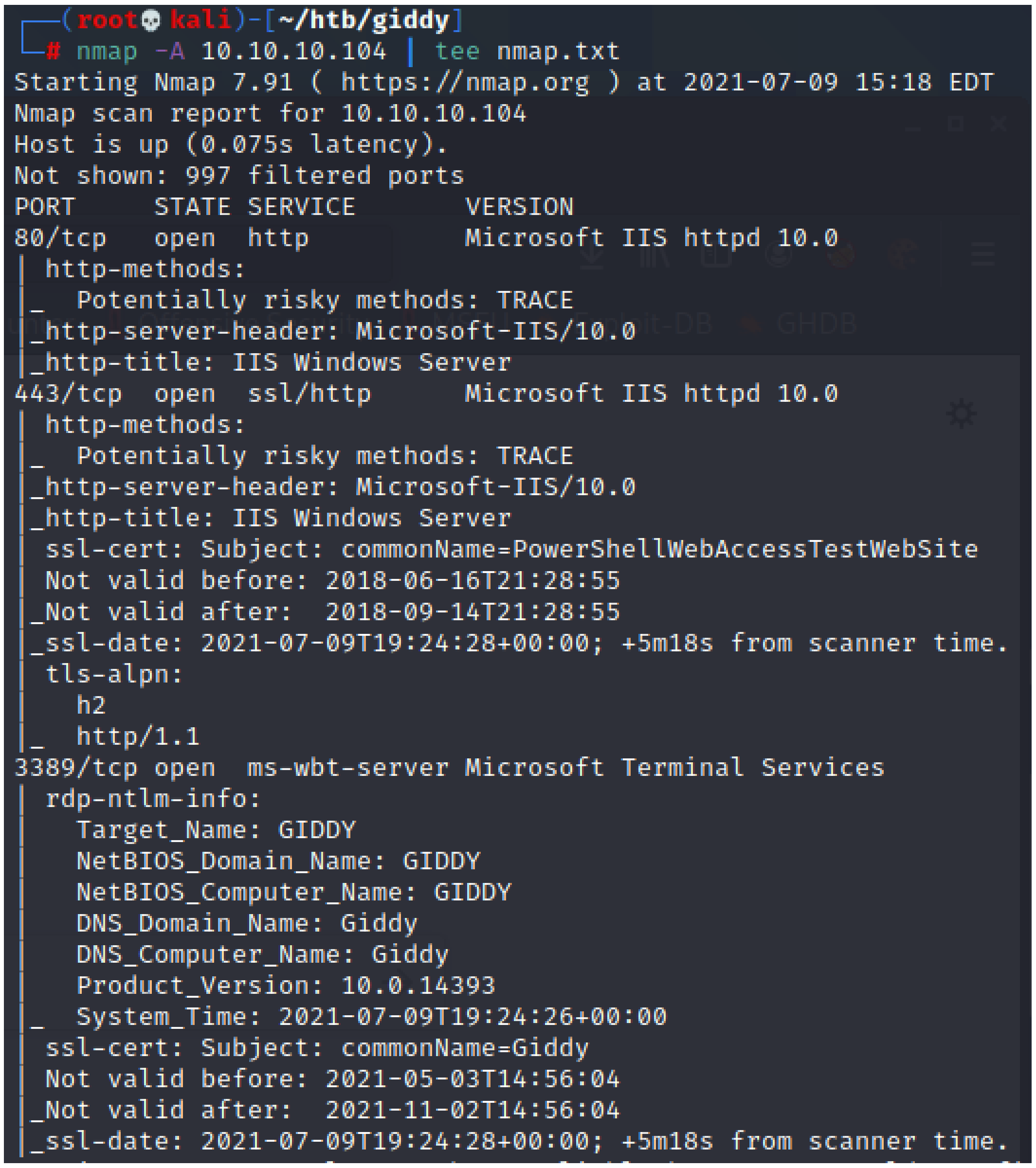
Giddy

Difficulty: Medium

Type: Windows

Nmap

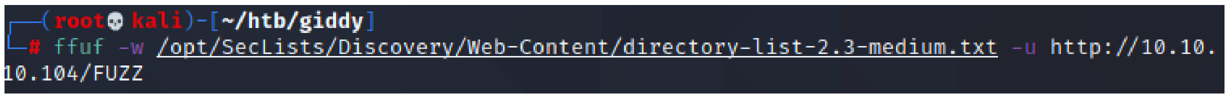
Starting with our aggressive nmap scan, we find ports 80, 443, and 3389 are open.



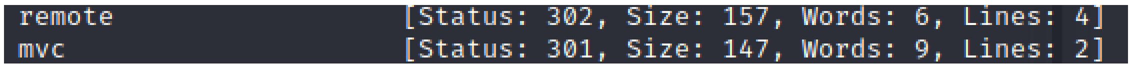
```
┌──(root💀kali)-[~/htb/giddy]
└─# nmap -A 10.10.10.104 | tee nmap.txt
Starting Nmap 7.91 ( https://nmap.org ) at 2021-07-09 15:18 EDT
Nmap scan report for 10.10.10.104
Host is up (0.075s latency).
Not shown: 997 filtered ports
PORT     STATE SERVICE       VERSION
80/tcp   open  http          Microsoft IIS httpd 10.0
| http-methods:
|_  Potentially risky methods: TRACE
|_http-server-header: Microsoft-IIS/10.0
|_http-title: IIS Windows Server
443/tcp  open  ssl/http      Microsoft IIS httpd 10.0
| http-methods:
|_  Potentially risky methods: TRACE
|_http-server-header: Microsoft-IIS/10.0
|_http-title: IIS Windows Server
| ssl-cert: Subject: commonName=PowerShellWebAccessTestWebSite
| Not valid before: 2018-06-16T21:28:55
|_Not valid after:  2018-09-14T21:28:55
|_ssl-date: 2021-07-09T19:24:28+00:00; +5m18s from scanner time.
| tls-alpn:
|   h2
|_  http/1.1
3389/tcp open  ms-wbt-server Microsoft Terminal Services
| rdp-ntlm-info:
|   Target_Name: GIDDY
|   NetBIOS_Domain_Name: GIDDY
|   NetBIOS_Computer_Name: GIDDY
|   DNS_Domain_Name: Giddy
|   DNS_Computer_Name: Giddy
|   Product_Version: 10.0.14393
|_  System_Time: 2021-07-09T19:24:26+00:00
| ssl-cert: Subject: commonName=Giddy
| Not valid before: 2021-05-03T14:56:04
|_Not valid after:  2021-11-02T14:56:04
|_ssl-date: 2021-07-09T19:24:28+00:00; +5m18s from scanner time.
```

Enumeration

Looking at the web server, we find a picture of a happy looking dog. There is nothing else on the site, so we start a web fuzzer looking for directories.



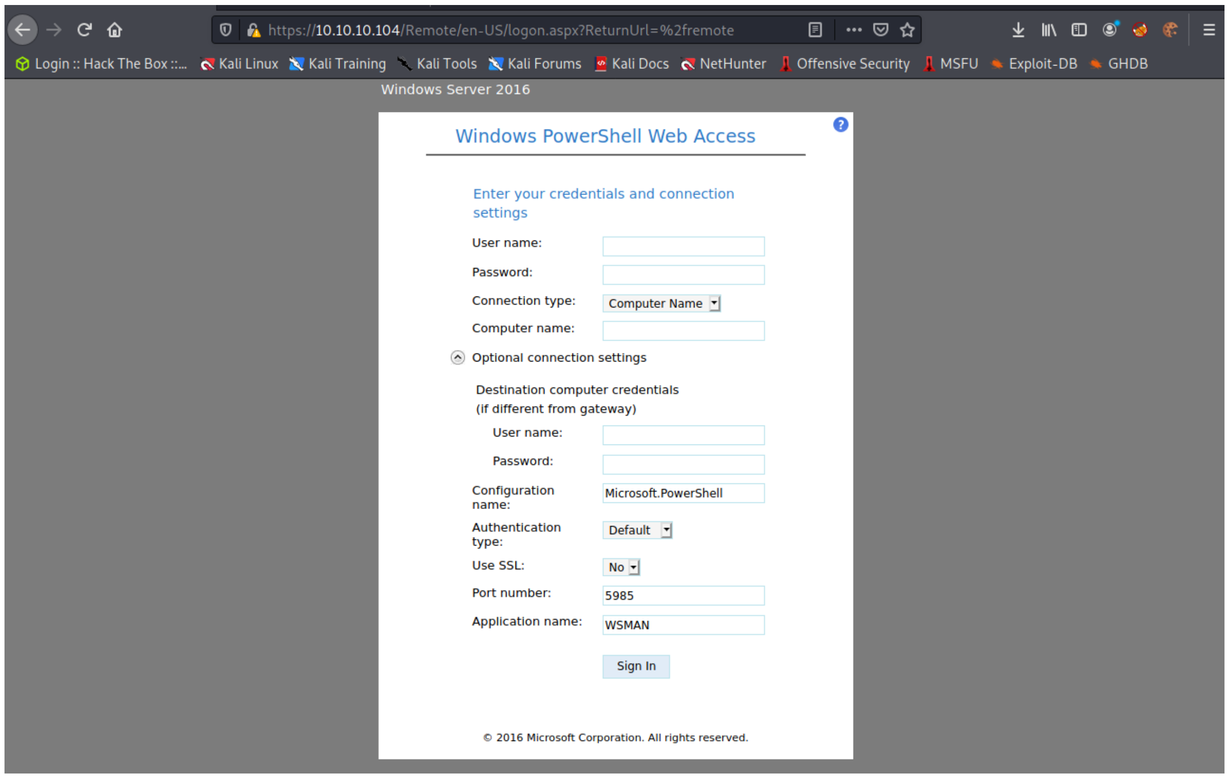
```
┌──(root💀kali)-[~/htb/giddy]
└─# ffuf -w /opt/SecLists/Discovery/Web-Content/directory-list-2.3-medium.txt -u http://10.10.10.104/FUZZ
```



```
remote                  [Status: 302, Size: 157, Words: 6, Lines: 4]
mvc                     [Status: 301, Size: 147, Words: 9, Lines: 2]
```

ffuf -w /opt/SecLists/Discovery/Web-Content/directory-list-2.3-medium.txt -u http://10.10.10.104/FUZZ

Going to the "remote" web directory, we get a prompt for windows powershell through a web interface. The site tells us it needs "https" in the header, so we do that and are able to properly look at the page. Exploring the page, we find the web shell requires a username, password, and computer name. There is another option to use computer credentials instead of gateway credentials.



https://10.10.10.104/Remote/en-US/logon.aspx?ReturnUrl=%2fremote

Windows Server 2016

Windows PowerShell Web Access

Enter your credentials and connection settings

User name:	
Password:	
Connection type:	Computer Name
Computer name:	

Optional connection settings

Destination computer credentials (if different from gateway)

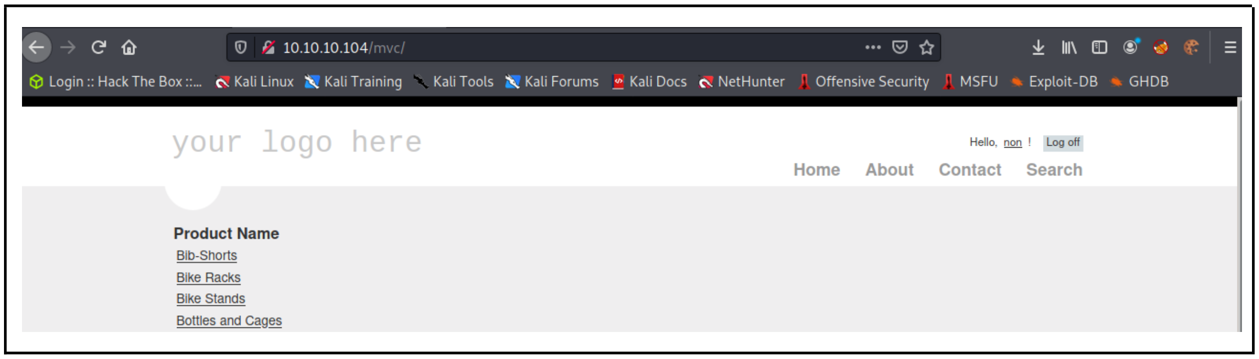
User name:	
Password:	
Configuration name:	Microsoft.PowerShell
Authentication type:	Default
Use SSL:	No
Port number:	5985
Application name:	WSMAN

Sign In

© 2016 Microsoft Corporation. All rights reserved.

Since we do not have credentials, we move on.

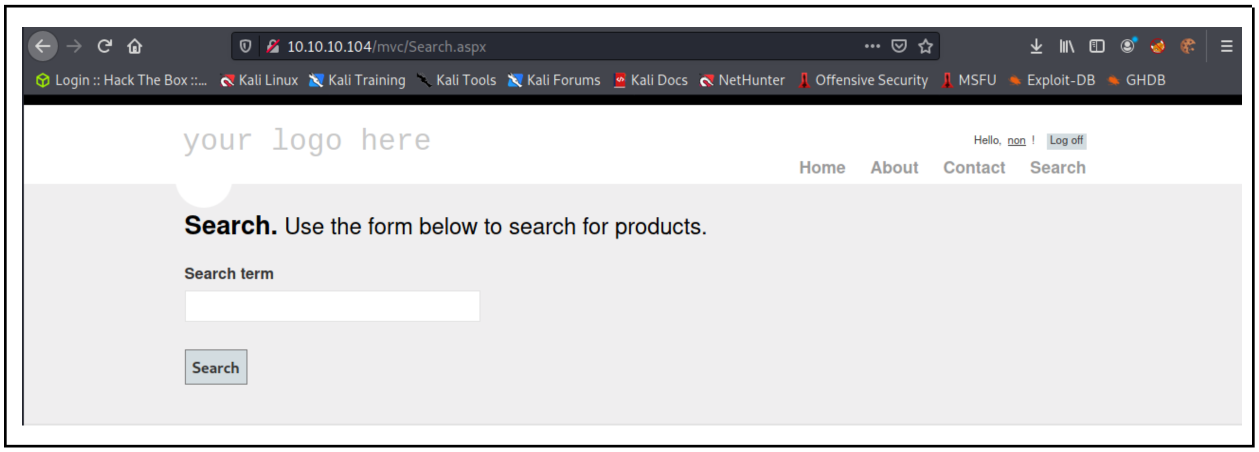
Our fuzzer also found a “mvc” web directory. This one leads to a page containing inventory along with a login



Product Name
Bib-Shorts
Bike Racks
Bike Stands
Bottles and Cages

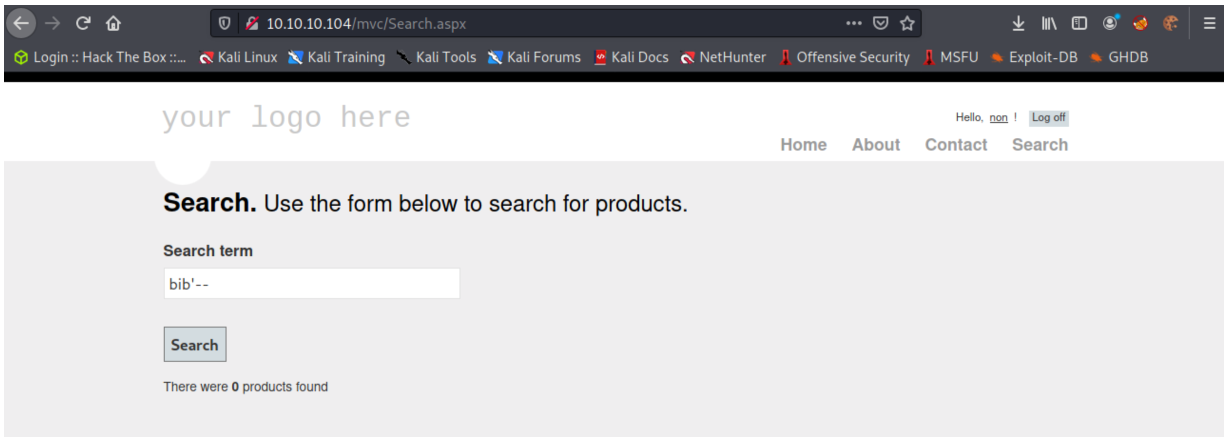
Testing the login, I register and am able to create an account without having to confirm an email address.

Searching here, we come across a search bar.



Search. Use the form below to search for products.

Whenever there is a place for user input, we always have to test it out for vulnerabilities. Doing this, we find the server is running this search bar through sql. To prevent an error, we put in sql comment marks.



Search. Use the form below to search for products.

Search term

bib'--

Search

There were **0** products found

To speed up the process of finding a sql injection, we are goint to capture the request with burp suite, then copy the request to a file. After this we are going to modify the request and send it to sqlmap.

```
4SearchTerm=*&ctl00%24MainContent%24>
```

(modified request with a star "*" so sqlmap knows what to replace)

```
┌──(root💀 kali)-[~/htb/giddy]
└─# sqlmap -r sql.req
```

While that runs, we notice the URL also contains search information concerning the inventory items. Attempting SQL injection here leads to a success.