Attacking Web App

Introduction

Welcome to the `Attacking Web Applications with Ffuf` module!

There are many tools and methods to utilize for directory and parameter fuzzing/brute-forcing. In this module we will mainly focus on the ffuf tool for web fuzzing, as it is one of the most common and reliable tools available for web fuzzing.

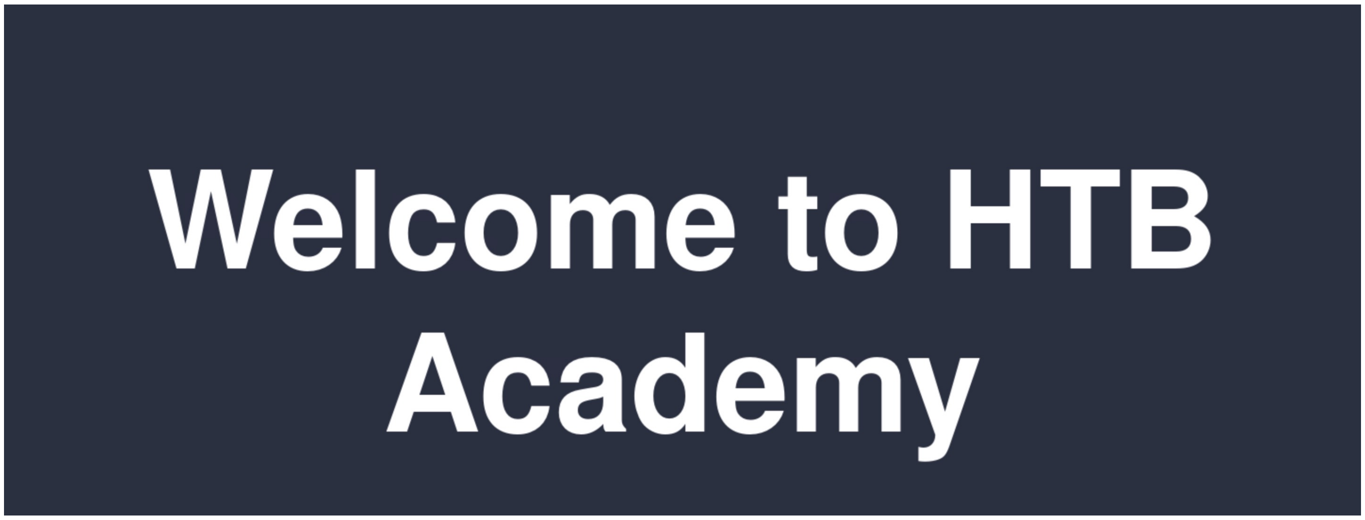
The following topics will be discussed:

- Fuzzing for directories
- Fuzzing for files and extensions
- Identifying hidden vhosts
- Fuzzing for PHP parameters
- Fuzzing for parameter values

Tools such as `ffuf` provide us with a handy automated way to fuzz the web application's individual components or a web page. This means, for example, that we use a list that is used to send requests to the webserver if the page with the name from our list exists on the webserver. If we get a response code 200, then we know that this page exists on the webserver, and we can look at it manually.

Web Fuzzing

We will start by learning the basics of using `ffuf` to fuzz websites for directories. We run the exercise in the question below, and visit the URL it gives us, and we see the following website:

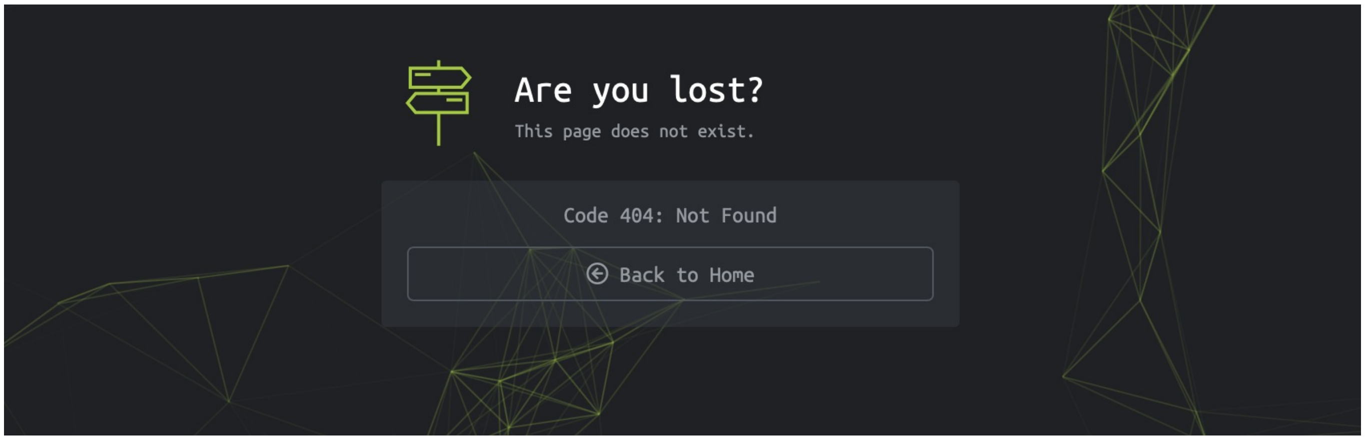

Welcome to HTB Academy

The website has no links to anything else, nor does it give us any information that can lead us to more pages. So, it looks like our only option is to '`fuzz`' the website.

Fuzzing

The term `fuzzing` refers to a testing technique that sends various types of user input to a certain interface to study how it would react. If we were fuzzing for SQL injection vulnerabilities, we would be sending random special characters and seeing how the server would react. If we were fuzzing for a buffer overflow, we would be sending long strings and incrementing their length to see if and when the binary would break.

We usually utilize pre-defined wordlists of commonly used terms for each type of test for web fuzzing to see if the webserver would accept them. This is done because web servers do not usually provide a directory of all available links and domains (unless terribly configured), and so we would have to check for various links and see which ones return pages. For example, if we visit https://www.hackthebox.eu/doesnotexist, we would get an HTTP code `404 Page Not Found`, and see the below page:


Are you lost?

This page does not exist.

Code 404: Not Found

However, if we visit a page that exists, like `/login`, we would get the login page and get an HTTP code `200 OK`, and see the below page:

This is the basic idea behind web fuzzing for pages and directories. Still, we cannot do this manually, as it will take forever. This is why we have tools that do this automatically, efficiently, and very quickly. Such tools send hundreds of requests every second, study the response HTTP code, and determine whether the page exists or not. Thus, we can quickly determine what pages exist and then manually examine them to see their content.

Wordlists

To determine which pages exist, we should have a wordlist containing commonly used words for web directories and pages, very similar to a `Password Dictionary Attack`, which we will discuss later in the module. Though this will not reveal all pages under a specific website, as some pages are randomly named or use unique names, in general, this returns the majority of pages, reaching up to 90% success rate on some websites.

We will not have to reinvent the wheel by manually creating these wordlists, as great efforts have been made to search the web and determine the most commonly used words for each type of fuzzing. Some of the most commonly used wordlists can be found under the GitHub [SecLists] repository, which categorizes wordlists under various types of fuzzing, even including commonly used passwords, which we'll later utilize for Password Brute Forcing.

Within our PwnBox, we can find the entire `SecLists` repo available under `/opt/useful/SecLists`. The specific wordlist we will be utilizing for pages and directory fuzzing is another commonly used wordlist called `directory-list-2.3`, and it is available in various forms and sizes. We can find the one we will be using under:

```
mayala@htb[/htb]$ locate directory-list-2.3-small.txt
/opt/useful/SecLists/Discovery/Web-Content/directory-list-2.3-small.txt
```

Tip: Taking a look at this wordlist we will notice that it contains copyright comments at the beginning, which can be considered as part of the wordlist and clutter the results. We can use the following in `ffuf` to get rid of these lines with the `-ic` flag.