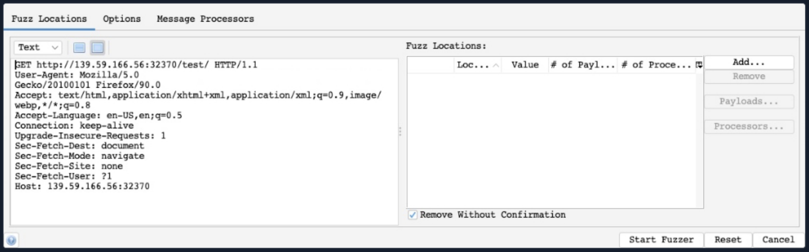
# ZAP Fuzzer

ZAP's Fuzzer is called (`ZAP Fuzzer`). It can be very powerful for fuzzing various web end-points, though it is missing some of the features provided by Burp Intruder. ZAP Fuzzer, however, does not throttle the fuzzing speed, which makes it much more useful than Burp's free Intruder.

In this section, we will try to replicate what we did in the previous section using ZAP Fuzzer to have an "apples to apples" comparison and decide which one we like best.

## Fuzz

To start our fuzzing, we will visit the URL from the exercise at the end of this section to capture a sample request. As we will be fuzzing for directories, let's visit `<http://SERVER_IP:PORT/test/>` to place our fuzzing location on `test` later on. Once we locate our request in the proxy history, we will right-click on it and select (`Attack>Fuzz`), which will open the `Fuzzer` window:



```
GET http://139.59.166.56:32370/test/ HTTP/1.1
User-Agent: Mozilla/5.0
Gecko/20100101 Firefox/90.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/
webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: none
Sec-Fetch-User: ?1
Host: 139.59.166.56:32370
```

Fuzz Locations:

| Loc... | Value | # of Payl... | # of Proce... |
|---|---|---|---|

✓ Remove Without Confirmation

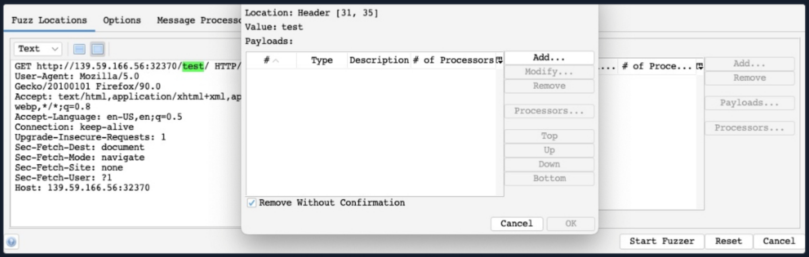
The main options we need to configure for our Fuzzer attack are:

- `Fuzz Location`
- `Payloads`
- `Processors`
- `Options`

Let's try to configure them for our web directory fuzzing attack.

## Locations

The `Fuzz Location` is very similar to `Intruder Payload Position`, where our payloads will be placed. To place our location on a certain word, we can select it and click on the `Add` button on the right pane. So, let's select `test` and click on `Add`:



```
GET http://139.59.166.56:32370/test/ HTTP/
User-Agent: Mozilla/5.0
Gecko/20100101 Firefox/90.0
Accept: text/html,application/xhtml+xml,ap
webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: none
Sec-Fetch-User: ?1
Host: 139.59.166.56:32370
```

Location: Header [31, 35]
Value: test
Payloads:

| # | Type | Description | # of Processors |
|---|---|---|---|

✓ Remove Without Confirmation

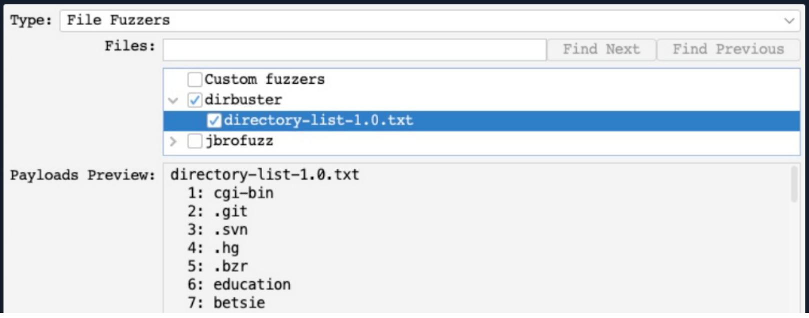
As we can see, this placed a `green` marker on our selected location and opened the `Payloads` window for us to configure our attack payloads.

## Payloads

The attack payloads in ZAP's Fuzzer are similar in concept to Intruder's Payloads, though they are not as advanced as Intruder's. We can click on the `Add` button to add our payloads and select from 8 different payload types. The following are some of them:

- `File`: This allows us to select a payload wordlist from a file.
- `File Fuzzers`: This allows us to select wordlists from built-in databases of wordlists.
- `Numberzz`: Generates sequences of numbers with custom increments.

One of the advantages of ZAP Fuzzer is having built-in wordlists we can choose from so that we do not have to provide our own wordlist. More databases can be installed from the ZAP Marketplace, as we will see in a later section. So, we can select `File Fuzzers` as the `Type`, and then we will select the first wordlist from `dirbuster`:



Type: File Fuzzers

Files:

- ☐ Custom fuzzers
- ☑ dirbuster
  - ☑ directory-list-1.0.txt
- ☐ jbrofuzz

Payloads Preview:

```
directory-list-1.0.txt
  1: cgi-bin
  2: .git
  3: .svn
  4: .hg
  5: .bzr
  6: education
  7: betsie
```

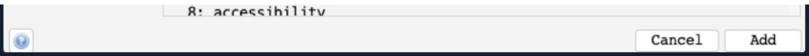
Cheat Sheet
Go to Questions
Table of Contents
Getting Started
Intro to Web Proxies ✓
Setting Up ✓
Web Proxy
Proxy Setup ✓
Intercepting Web Requests ✓
Intercepting Responses ✓
Automatic Modification ✓
Repeating Requests ✓
Encoding/Decoding ✓
Proxying Tools ✓
Web Fuzzer
Burp Intruder ✓
ZAP Fuzzer ✓
Web Scanner
Burp Scanner ✓
ZAP Scanner ✓
Extensions ✓
Skills Assessment
Skills Assessment - Using Web Proxies ✓

My Workstation

OFFLINE

Start Instance

∞ / 1 spawns left



```
8: accessibility
```

Once we click the `Add` button, our payload wordlist will get added, and we can examine it with the `Modify` button.

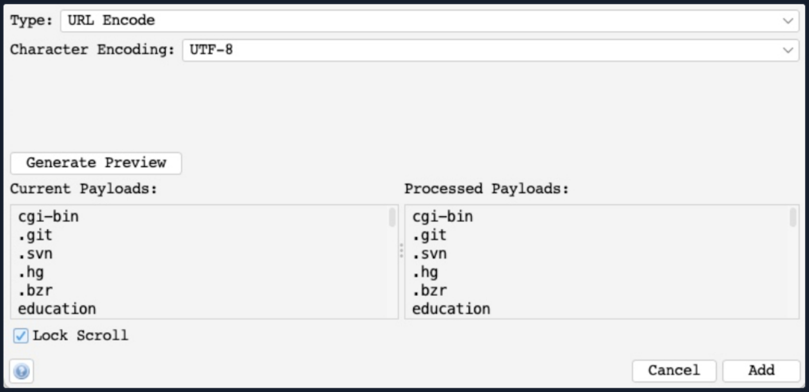
## Processors

We may also want to perform some processing on each word in our payload wordlist. The following are some of the payload processors we can use:

- `Base64 Decode/Encode`
- `MD5 Hash`
- `Postfix String`
- `Prefix String`
- `SHA-1/256/512 Hash`
- `URL Decode/Encode`
- `Script`

As we can see, we have a variety of encoders and hashing algorithms to select from. We can also add a custom string before the payload with `Prefix String` or a custom string with `Postfix String`. Finally, the `Script` type allows us to select a custom script that we built and run on every payload before using it in the attack.

We will select the `URL Encode` processor for our exercise to ensure that our payload gets properly encoded and avoid server errors if our payload contains any special characters. We can click on the `Generate Preview` button to preview how our final payload will look in the request:



Type: URL Encode
Character Encoding: UTF-8

Current Payloads:

```
cgi-bin
.git
.svn
.hg
.bzr
education
```

Processed Payloads:

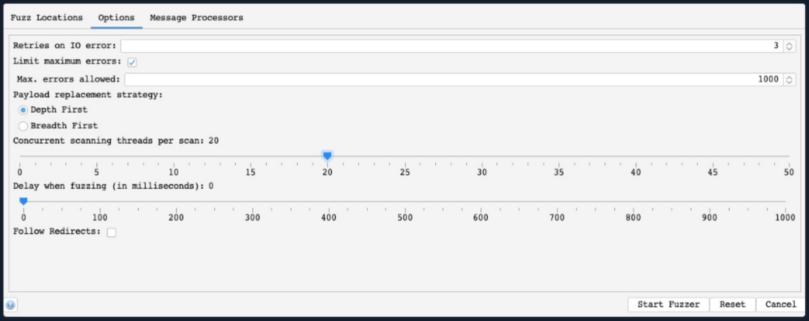
```
cgi-bin
.git
.svn
.hg
.bzr
education
```

Lock Scroll: ✓

Once that's done, we can click on `Add` to add the processor and click on `Ok` in the processors and payloads windows to close them.

## Options

Finally, we can set a few options for our fuzzers, similar to what we did with Burp Intruder. For example, we can set the `Concurrent threads per scan` to `20`, so our scan runs very quickly:



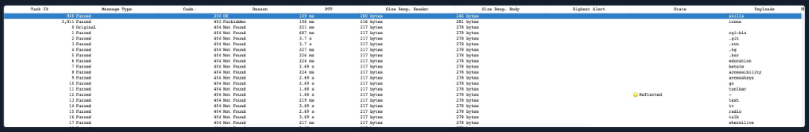
Retries on IO error: 3
Limit maximum errors: ✓
Max. errors allowed: 1000
Payload replacement strategy:
● Depth First
○ Breadth First
Concurrent scanning threads per scan: 20
Delay when fuzzing (in milliseconds): 0
Follow Redirects: ☐

The number of threads we set may be limited by how much computer processing power we want to use or how many connections the server allows us to establish.

We may also choose to run through the payloads `Depth first`, which would attempt all words from the wordlist on a single payload position before moving to the next (e.g., try all passwords for a single user before brute-forcing the following user). We could also use `Breadth first`, which would run every word from the wordlist on all payload positions before moving to the next word (e.g., attempt every password for all users before moving to the following password).

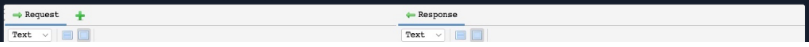
## Start

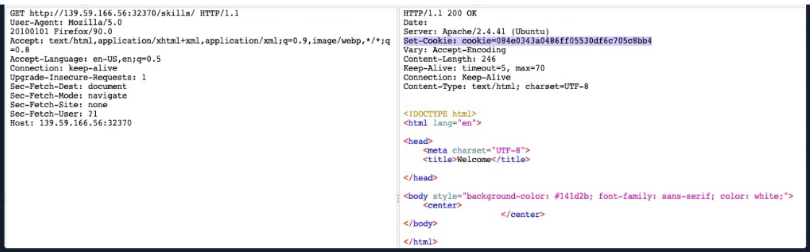
With all of our options configured, we can finally click on the `Start Fuzzer` button to start our attack. Once our attack is started, we can sort the results by the `Response` code, as we are only interested in responses with code `200`:



| Task ID | Message Type | Code | Reason | RTT | Size Resp. Header | Size Resp. Body | Highest Alert | State | Payloads |
|---|---|---|---|---|---|---|---|---|---|
| 938 | Fuzzed | 200 | OK | 109 ms | 280 bytes | 246 bytes | | | skills |
| 3,011 | Fuzzed | 403 | Forbidden | 106 ms | 216 bytes | 281 bytes | | | icons |
| 0 | Original | 404 | Not Found | 221 ms | 217 bytes | 278 bytes | | | |
| 1 | Fuzzed | 404 | Not Found | 687 ms | 217 bytes | 278 bytes | | | cgi-bin |
| 2 | Fuzzed | 404 | Not Found | 3.7 s | 217 bytes | 278 bytes | | | .git |
| 3 | Fuzzed | 404 | Not Found | 3.7 s | 217 bytes | 278 bytes | | | .svn |
| 4 | Fuzzed | 404 | Not Found | 227 ms | 217 bytes | 278 bytes | | | .hg |
| 5 | Fuzzed | 404 | Not Found | 226 ms | 217 bytes | 278 bytes | | | .bzr |
| 6 | Fuzzed | 404 | Not Found | 224 ms | 217 bytes | 278 bytes | | | education |
| 7 | Fuzzed | 404 | Not Found | 2.69 s | 217 bytes | 278 bytes | | | betsie |
| 8 | Fuzzed | 404 | Not Found | 224 ms | 217 bytes | 278 bytes | | | accessibility |
| 9 | Fuzzed | 404 | Not Found | 2.69 s | 217 bytes | 278 bytes | | | accesskeys |
| 10 | Fuzzed | 404 | Not Found | 2.69 s | 217 bytes | 278 bytes | | | go |
| 11 | Fuzzed | 404 | Not Found | 1.68 s | 217 bytes | 278 bytes | | | toolbar |
| 12 | Fuzzed | 404 | Not Found | 1.68 s | 217 bytes | 278 bytes | | 🟡 Reflected | - |
| 13 | Fuzzed | 404 | Not Found | 219 ms | 217 bytes | 278 bytes | | | text |
| 14 | Fuzzed | 404 | Not Found | 3.69 s | 217 bytes | 278 bytes | | | tv |
| 15 | Fuzzed | 404 | Not Found | 2.69 s | 217 bytes | 278 bytes | | | radio |
| 16 | Fuzzed | 404 | Not Found | 3.69 s | 217 bytes | 278 bytes | | | talk |
| 17 | Fuzzed | 404 | Not Found | 217 ms | 217 bytes | 278 bytes | | | whosonline |

As we can see, we got one hit with code `200` with the `skills` payload, meaning that the `/skills/` directory exists on the server and is accessible. We can click on the request in the results window to view its details:





```
GET http://139.59.166.56:32370/skills/ HTTP/1.1
User-Agent: Mozilla/5.0
20100101 Firefox/90.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q
=0.8
Accept-Language: en-US,en;q=0.5
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: none
Sec-Fetch-User: ?1
Host: 139.59.166.56:32370
```

```
HTTP/1.1 200 OK
Date:
Server: Apache/2.4.41 (Ubuntu)
Set-Cookie: cookie=084e0343a0486ff05530df6c705c8bb4
Vary: Accept-Encoding
Content-Length: 246
Keep-Alive: timeout=5, max=70
Connection: Keep-Alive
Content-Type: text/html; charset=UTF-8


<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <title>Welcome</title>

</head>

<body style="background-color: #141d2b; font-family: sans-serif; color: white;">
    <center>
                    </center>
</body>

</html>
```

We can see from the response that this page is indeed accessible by us. There are other fields that may indicate a successful hit depending on the attack scenario, like `Size Resp. Body` which may indicate that we got a different page if its size was different than other responses, or `RTT` for attacks like `time-based SQL injections`, which are detected by a time delay in the server response.

**Connect to Pwnbox**
Your own web-based Parrot Linux instance to play our labs.

Pwnbox Location

UK 137ms

Terminate Pwnbox to switch location

Start Instance

∞ / 1 spawns left

Waiting to start...

Enable step-by-step solutions for all questions

## Questions

Cheat Sheet

Answer the question(s) below to complete this Section and earn cubes!

Target(s): Click here to spawn the target system!

+ 2 The directory we found above sets the cookie to the md5 hash of the username, as we can see the md5 cookie in the request for the (guest) user. Visit '/skills/' to get a request with a cookie, then try to use ZAP Fuzzer to fuzz the cookie for different md5 hashed usernames to get the flag. Use the "top-usernames-shortlist.txt" wordlist from Seclists.

HTB{fuzz1n6_my_f1r57_c00k13}

Submit Hint

Previous Next Mark Complete & Next

Powered by HACKTHEBOX