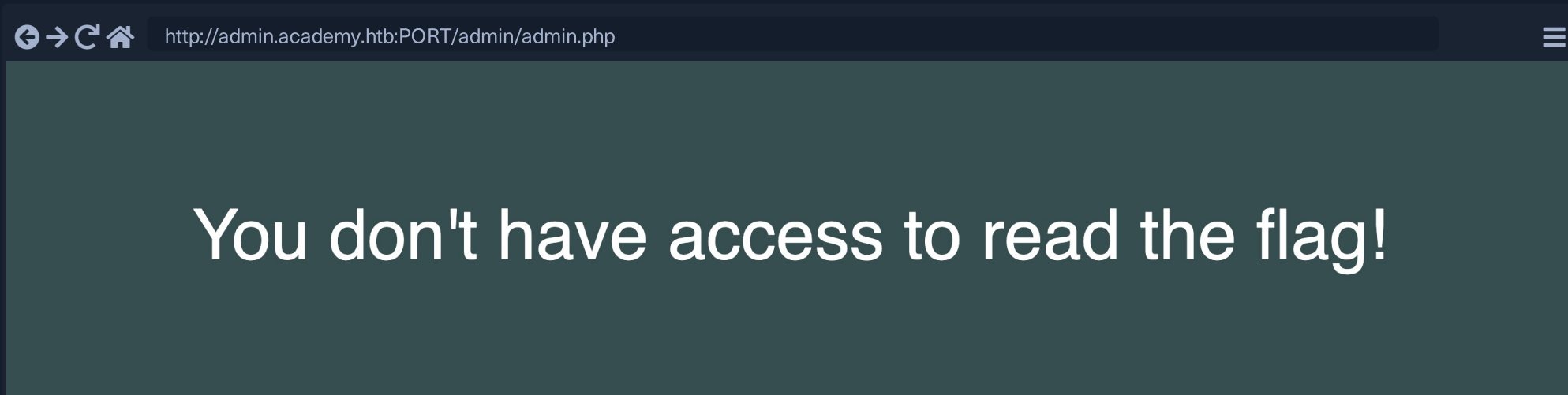


Parameter Fuzzing - GET

If we run a recursive `ffuf` scan on `admin.academy.htb`, we should find `http://admin.academy.htb:PORT/admin/admin.php`. If we try accessing this page, we see the following:



That indicates that there must be something that identifies users to verify whether they have access to read the `flag`. We did not login, nor do we have any cookie that can be verified at the backend. So, perhaps there is a key that we can pass to the page to read the `flag`. Such keys would usually be passed as a `parameter`, using either a `GET` or a `POST` HTTP request. This section will discuss how to fuzz for such parameters until we identify a parameter that can be accepted by the page.

Tip: Fuzzing parameters may expose unpublished parameters that are publicly accessible. Such parameters tend to be less tested and less secured, so it is important to test such parameters for the web vulnerabilities we discuss in other modules.

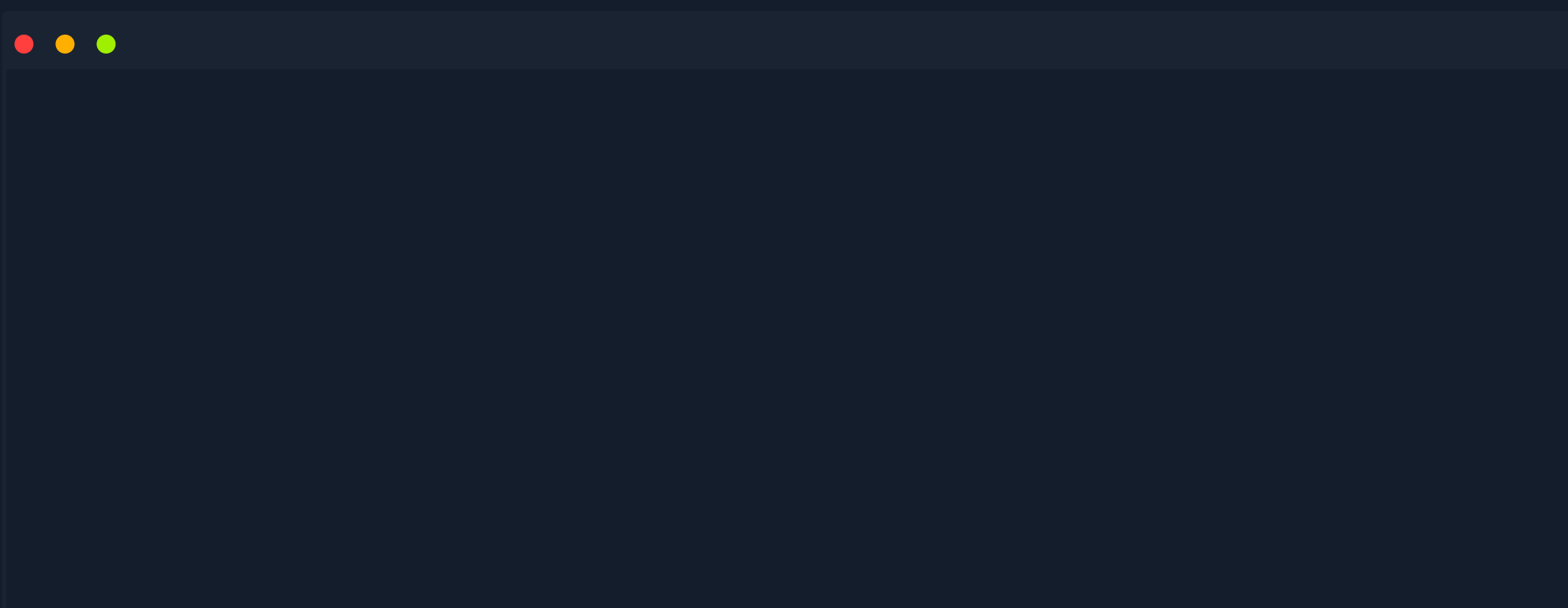
GET Request Fuzzing

Similarly to how we have been fuzzing various parts of a website, we will use `ffuf` to enumerate parameters. Let us first start with fuzzing for `GET` requests, which are usually passed right after the URL, with a `?` symbol, like:

- `http://admin.academy.htb:PORT/admin/admin.php?param1=key`.

So, all we have to do is replace `param1` in the example above with `FUZZ` and rerun our scan. Before we can start, however, we must pick an appropriate wordlist. Once again, `SecLists` has just that in `/opt/useful/SecLists/Discovery/Web-Content/burp-parameter-names.txt`. With that, we can run our scan.

Once again, we will get many results back, so we will filter out the default response size we are getting.



MichaelLuka@htb[/htb]\$ ffuf -w /opt/useful/SecLists/Discovery/Web-Content/burp-parameter-names.txt:FUZZ -u http://admin.ac

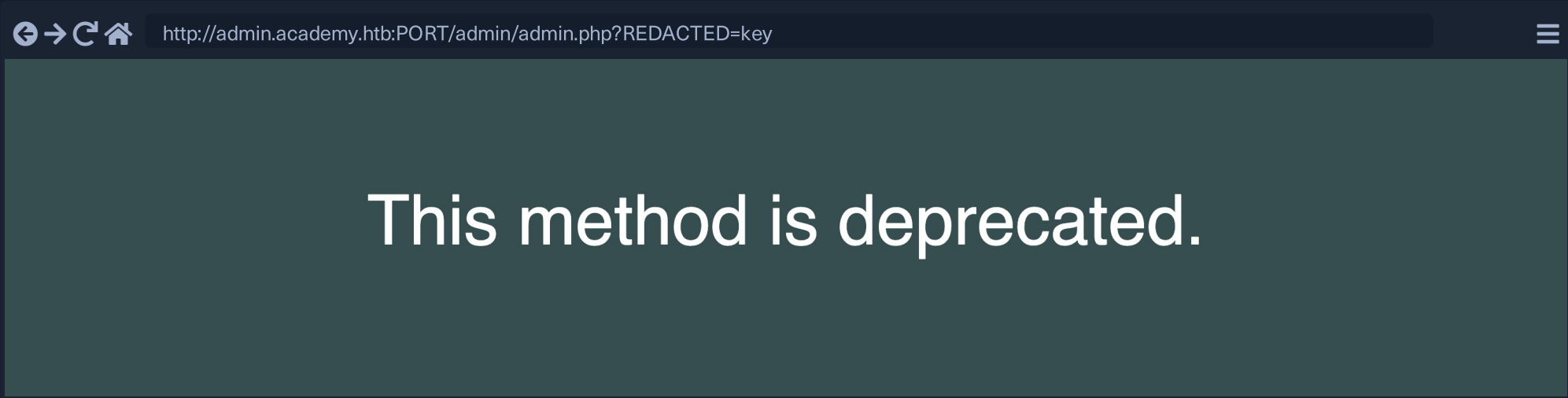


v1.1.0-git

```
-----
:: Method      : GET
:: URL         : http://admin.academy.htb:PORT/admin/admin.php?FUZZ=key
:: Wordlist    : FUZZ: /opt/useful/SecLists/Discovery/Web-Content/burp-parameter-names.txt
:: Follow redirects : false
:: Calibration : false
:: Timeout     : 10
:: Threads    : 40
:: Matcher     : Response status: 200,204,301,302,307,401,403
:: Filter      : Response size: xxx
-----
```

<...SNIP...> [Status: xxx, Size: xxx, Words: xxx, Lines: xxx]

We do get a hit back. Let us try to visit the page and add this **GET** parameter, and see whether we can read the flag now:



As we can see, the only hit we got back has been **deprecated** and appears to be no longer in use.

Start Instance

1 / 1 spawns left


Waiting to start...

Questions

 Cheat Sheet

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

Target: [Click here to spawn the target system!](#)

+ 0 

 Using what you learned in this section, run a parameter fuzzing scan on this page. what is the parameter accepted by this webpage?


user

 Submit

 Previous

Next 

 Mark Complete & Next

 Cheat Sheet









 Go to Questions

Table of Contents







Introduction

Introduction	
Web Fuzzing	


Basic Fuzzing

 Directory Fuzzing	
 Page Fuzzing	
 Recursive Fuzzing	

Domain Fuzzing

DNS Records	
 Sub-domain Fuzzing	
Vhost Fuzzing	
 Filtering Results	

Parameter Fuzzing

 Parameter Fuzzing - GET	
Parameter Fuzzing - POST	
 Value Fuzzing	

Skills Assessment

 Skills Assessment - Web Fuzzing	
---	--

OFFLINE

▶

Start Instance

1 / 1 spawns left