WEB: Flag

The website prints the following php:

```php
<?php
  highlight_file(__FILE__);
  $lang = $_SERVER['HTTP_ACCEPT_LANGUAGE'] ?? 'ot';
  $lang = explode(',', $lang)[0];
  $lang = str_replace('../', '', $lang);
  $c = file_get_contents("flags/$lang");
  if (!$c) $c = file_get_contents("flags/ot");
  echo '<img src="' . base64_encode($c) . '">';
```

Warning: file_get_contents(flags/en-GB): failed to open stream: No such file or directory in **/var/www/html/index.php** on line **6**

Warning: file_get_contents(flags/ot): failed to open stream: No such file or directory in **/var/www/html/index.php** on line **7**

The description says that the flag is at ‘./flag’.

Let’s focus on the third line: `$lang = $_SERVER['HTTP_ACCEPT_LANGUAGE'] ?? 'ot';`.
The variable $lang is assigned through the HTML header called Accept-Language. Then, based on the language, the string is splitted with `$lang = explode(',', $lang)[0];` , and the first token is taken. In `$lang = str_replace('../', '', $lang);` there is a string sanctification, where the pattern ‘../’ inside the variable $lang is replaced with ‘’. Then, the code tries to open the flag at that given language.
We need to go back to the main directory in order to find the flag. As on bash, for going in the father we need to digit ‘../’ but, in order to escape the sanctification process we can write the following ‘....//’. How many times? We need to discover it.

I used the ModHeader as Google Chrome extension plug-in: it allows us to modify HTTP headers.

The following path seems right:

....//....//....//....//flag

The file is corrupted, however, we are interested in its name:

MzVjM190aGlzX2ZsYWdfaXNfdGhlX2JlNXRfZmw0Zwo=

It’s a base64, and, once converted it, we reveal the flag.

35c3_this_flag_is_the_be5t_fl4g