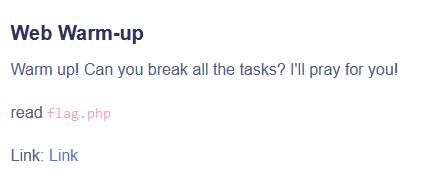
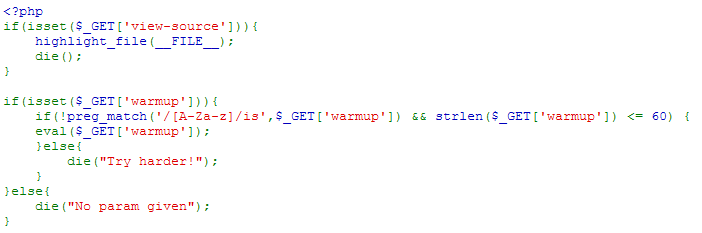
**Task**

****

**Solution**

При переходе по ссылке видим исходный php код:



Нам необходимо через аргумент warmup передать некоторый код, который затем выполнится в eval(). Проблема заключается в том, что мы не можем использовать буквы английского алфавита и, кроме того, длина аргумента ограничена 60-ю символами.

Попробовав найти информация по preg\_match, я наткнулся на данную статью на ctf-wiki:

<https://ctf-wiki.github.io/ctf-wiki/web/php/php/#preg_match-code-execution>

Выходит, что в php мы можем использовать побитовые операции между строками чтобы получать новые строки, символы в которых будут соответствовать новым, полученным в результате операции, кодам.

Например, написав ”@” ^ ”2”, мы получим символ “r”, так как XOR между ASCII кодами @ и 2 – ASCII код символа r.

Таким образом мы можем закодировать некоторую строку. Нам нужно закодировать примерно следующую:

readfile(“flag.php”);

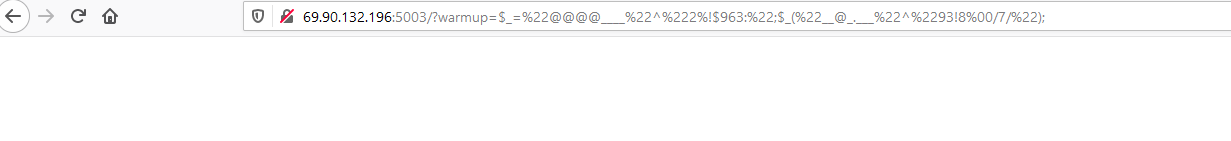
Закодированная строка:

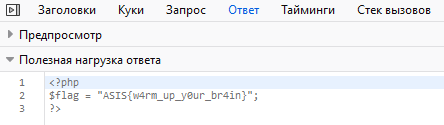
$\_=%22@@@@\_\_\_\_%22^%222%!$963:%22;$\_(%22\_\_@\_.\_\_\_%22^%2293!8%00/7/%22);

В данном случае мы создаём переменную с именем \_, которая хранит строку “readfile” (@^2=r, @^%=e и т.д.). Затем мы обращаемся к этой переменной как к функции и передаём в качестве аргумента “flag.php”, тем самым вызывая readfile(“flag.php”);

Стоит отметить, что подобрать 2 символа, XOR которых давал бы символ точки, не используя символы A-Za-z, не удалось, а при использовании конкатенации строк исчерпывался лимит в 60 доступных символов, но сам символ точки не запрещён и XOR его с %00 дал ожидаемый результат.

В итоге, отправив сформированную строку в качестве аргумента warmup, получаем пустую страницу в браузере, но на самом деле ответ пришёл, просто браузер интерпретировал его как пустую страницу:





**Флаг:**

ASIS{w4rm\_up\_y0ur\_br4in}