UPB – Zadanie 9

Využité nástroje: OWASP ZAP, SQLMAP

A1 - Injection:

Pomocou **SQLMAP** a príkazu „python sqlmap.py -u http://192.168.56.102/udpb/www-vulnerable/ --forms --crawl=2“

```
[10:50:32] [INFO] GET parameter 'name' appears to be 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)' injectable
```

Test nám ukázal zraniteľnosť SQL Injection, teda máme Injection.

Spôsobuje: Manipulovanie dát v databáze, získavanie citlivých údajov, prístup aj bez prihlasovacích údajov.

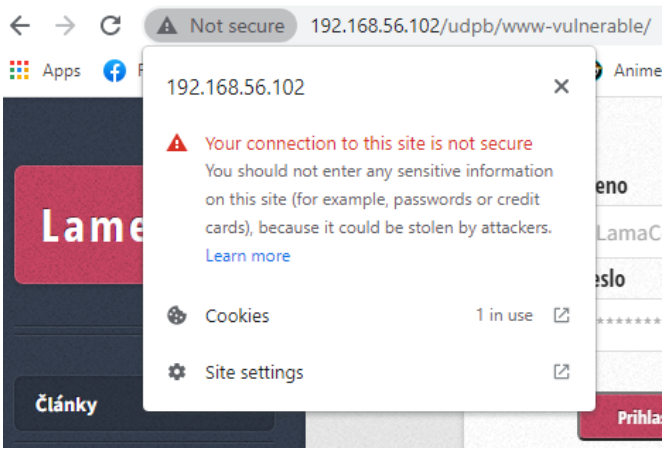
Riešenie: Pridám real_escape_string na našu hodnotu, čím ošetrím možný útok

```
$nasaSearch = $db->real_escape_string($_POST['search']);
$search = $db->query('SELECT * FROM articles WHERE title LIKE "%'.$nasaSearch.'%
" OR content LIKE "%'.$nasaSearch_'%"');
?>
```

A2 - Broken Authentication and Session Management:

Keď otvoríme stránku vyskočí na nás upozornenie že pripojenie na stránku nie je bezpečné a nezadávali tam žiadne citlivé údaje.

Spôsobuje: Stránka nie je zabezpečená voči útoku a môže to spôsobiť únik citlivých údajov ako napríklad heslá, kreditné karty...



Riešenie : Investujem a kúpim si ten certifikát.

A3 - XSS:

Útočník vďaka týmto chybám v zabezpečení webovej aplikácie dokáže do stránok podstrčiť svoj vlastný javascriptový kód, čo môže využiť buď iba k poškodeniu vzhľadu stránky, jej znefunkčneniu alebo dokonca k získavaniu citlivých údajov návštevníkov stránky, obchádzaniu bezpečnostných prvkov aplikácie a phishingu.
Spôsobuje: Útočník môže získať prístup k súborom, skenovať sieť...

Riešenie: Vstup ošetrím pomocou funkcie htmlspecialchars().

```
<!--Co tak dat vysledky vyhladavania a data[title] do htmlspecialchars? -->
<h1> Výsledky vyh♦adavania: <?=htmlspecialchars$_POST['search']?></h1>
```

A4 - Insecure Direct Object References:

Pomocou **OWASP ZAP** dokážeme nájsť odkazy na stránky, ku ktorým by používateľ normálne nemal mať prístup. Napríklad odkaz na index. (Celý výstup je v ZAPReport.txt)

Index of /udpb

Name	Last modified	Size	Description
Parent Directory		-	
README.md	25-Jun-2015 10:24	23	
www-vulnerable/	25-Jun-2015 10:24	-	

Apache/2.2.22 (Debian) Server at 192.168.56.102 Port 80

Spôsobuje: Útočník získava prístup k súborom kde by nemal mať prístup a môže zistiť citlivé informácie, napríklad vidíme že tu je prístup k presnej verzii Apache, kde server beží.

Riešenie: Ošetrím tým že pridám všade .php a následne ho odstránim.

```
                                                    //A4 - LFI najjednoduscha
sie riesenie spociva v dopisani .php za kazdy requirnuty file
                            $pages=$_GET["page"] . ".php";
                            if(!isLogin()) { $pages='login'; }
                            if (!isset($pages) || empty($pages)){
                                require("content/home");
                            }elseif (file_exists("content/$pages")) {
                                require("content/$pages");
                            }else{require ("content/error_page");}

                            /*
                             * A4 - Nuz ked ak to osetris vyssie spomenutym spos
obom tak musis vsetky linky prepisat z whatever.php na whatever nasledne vymazat
 .php
```

```
<?php
   //A4
   $file = $_GET['file'];
   if(isset($file))
   {
       include("$file");
   }
   ?>
```

A5 - Security Misconfiguration:

Veľmi často vyskytujú si sa problém je taktiež nesprávna konfigurácia zabezpečenia. Často to býva dôsledkom nesprávnej alebo neúplnej konfigurácie, nepotrebné vlastnosti aktivované alebo napríklad defaultné účty, ktoré ostali aktívne.

Spôsobuje: Únik informácii, prístup tam kde by nemal byť.

A6 - Sensitive Data Exposure:

Niektoré aplikácie nesprávne chránia citlivé dáta, preto môžu mať k nim útočníci ľahký prístup. Tieto dáta si vyžadujú osobitnú ochranu. Prihlasovacie údaje sa prenášajú cez nezabezpečený HTTP protokol cez GET /udpb/www-vulnerable/index.php?name=student&pass=student&logIN=1.

Výstup z **OWASP ZAP**.

Spôsobuje: Prístup k dátam, kde by útočník nemal mať prístup.

Riešenie: Nahradím GET za POST. Je bezpečnejší.

```
if(@$_POST['logIN']){
    if(verify_login()) {
        header('LOCATION: index.php');
    }else{
        $error = "Wrong name or password!! Pls try it again!!";
    }
}
?>
<?if(!isLogin()){?>
<div style="width:20%;">
    <?=@$error?>
    <form method="post" name="login">
        <label>Meno</label>
        <input name="name" value="" type="text" placeholder="LamaCoder" autofocu
s />
        <label>Heslo</label>
        <input name="pass" value="" type="password" placeholder="********" />
        <br />
        <button class="button" name="logIN" value="1">Prihlasi♦</button>
    </form>
```

A7 - Missing Function Level Access Control:

Nie všetky obmedzenia sú správne nastavené a aj neprihlásený používateľ má prístup k stránke, ktorá by pre neho nemala byť dostupná, napr.

http://192.168.56.102/udpb/www-vulnerable/content/home.php?id=2

Spôsobuje: Prístup by mohol získať aj k súkromným súborom iných používateľov alebo ich informáciám.

A8 - Cross-Site Request Forgery (CSRF):

Spôsobuje: Falšovanie požiadaviek medzi webmi alebo podobne ako XSS.

A9 - Using Components with Known Vulnerabilities:

Súčasťou dnešných webových aplikácii sú často veľké počty komponentov ako napríklad knižnice, frameworky alebo iné moduly, ktoré majú rovnaké práva. V prípade že určité komponenty nie sú najaktuálnejšie, môže sa stať že obsahujú zraniteľnosti, ktoré sú verejne známe a dobre zdokumentované alebo aspoň jednoducho napadnuteľné.

Spôsobuje: Prístup k údajom kde by nemal mať, únik informácii.

A10 - Unvalidated Redirects and Forwards:

Webové aplikácie často presmerujú užívateľa na iné stránky a použijú nedôveryhodné údaje na určenie cieľovej stránky. Bez správneho overenia môže útočník presmerovať obeť na phishing alebo malware stránky.

Spôsobuje: Prístup k súborom kde by nemal útočník mať.

Riešenie: Odstránenie zbytočnosti.

```
// A10 - sessionID sa prenasa v cookine, takze tu je zbytocne, skus to zmazat :)
                                    echo '<li><a href="./?page=logout.php">O
dhlási♦ sa</a></li>';
```

Záver

Zameranie bolo na TOP 10 hrozieb OWASP, tie sme spoznali a odhalili na našom serveri, kde sme ich aj otestovali. Pokiaľ sa nám podarilo tak aj opravili.