eman ta zabal zazu

Universidad del País Vasco
Euskal Herriko Unibertsitatea

BILBOKO INGENIARITZA ESKOLA
ESCUELA DE INGENIERÍA DE BILBAO

4. ENTREGA - ERASOA

Informazio Sistemen Segurtasuna Kudeatzeko Sistemak

Yoel Justel
Xinyan Wang

SARRERA

Lan honetan **Erasoa** atalari dagokion jarduera garatuko da, eta haren helburu nagusia beste talde batek garatutako web sistema batean ahultasun erreal bat identifikatzea eta ustiatzea da. Gure kasuan, “biktima” gisa esleitutako web sistema *Alubias Comunistas* taldearen proiektua izan da. Erasoa sistemaren lehenengo bertsioaren aurka burutu behar da (*entrega_1* adarra), hau da, segurtasun-neurri sendoak gehitu aurreko inplementazioaren aurka.

ERABILITAKO ERASOAK

1. LEHENENGO ERASOA: LOGIN BYPASS-A

SQL Injection erasoa aplikazio batean gertatzen da, erabiltzailearen input-a zuzenean SQL kontsultetan txertatzen denean inolako iragazketa edo prestatutako adierazpenik gabe. Horren ondorioz, erasotzaileak bere kodea txerta dezake datu-baseak exekuta dezan. Kasu honetan, helburua login-prozesua bypass egitea izan da, edozein erabiltzaile gisa autentifikazioa lortuz pasahitzik jakin gabe.

➢ **AHULEZIAK:**

Ahultasuna *app/php/login/index.php* fitxategiko lerro honetan ematen da:

```
$sql = "SELECT * FROM erabiltzaileak WHERE Erabiltzaile = '$user' AND
Pasahitza = '$pas'";
```

- **Inputaren Erabilera Arriskutsua:** Erabiltzailearen sarrera zuzenean txertatzen da kontsultan (*'$user eta '$pas'*), karaktere bereziak iragazi gabe.
- **Parametro-Kontsulten Gabezia:** Ez da *prepared statement*-ik edo parametrorik erabiltzen kontsultak seguruak izateko.
- **Balidazio Ahula Bezero Aldean:** JavaScript balidazioa bezeroan bakarrik dagoenez, erasotzaileek erraz gainditu dezakete (cURL, Burp Suite edo formularioen manipulazioz).
- **Kontsultaren Mugaren Gabezia:** SQL kontsultak ez du LIMIT klausularik, lehen erabiltzailearen hautaketa automatikoa errazten duena.

➢ **PAUSUAK ERASOA GAUZATZEKO:**

Eraso hau oso sinplea da, login-ean hurrengo komandoa idatzi behar da “Erabiltzaile” eremuan, pasahitza edozein izan daiteke:

```
' OR 1=1 LIMIT 1-- -
```

Honela 1. erabiltzailearen kontuan sartzea lortuko da. Komando honen aldakuntzak daude, hala nola:

```
' OR 1=1 LIMIT 1,1-- -
' OR 1=1 LIMIT 2,1-- -
```

1. komandoarekin 2. erabiltzailearen kontuan sartzea lortuko da. 2. komandoarekin 3. erabiltzailearenean...

Komando hau *SELECT* aginduaren jarraipena da. Lehenik, erabiltzailearen balioa hutsik uzten da apostrofea erabiliz. Ondoren, beti egia den baldintza bat gehitzen da, adibidez, 1=1. Gero, *LIMIT* erabiliz eta horren parametroekin jolastuz lehenengo erabiltzailea automatikoki bueltatzen da. Azkenik, kontsultaren gainerakoa komentario bihurtzen da gidoiak erabiliz, eta horrela kendu egiten da.

➢ **ONDORIOAK:**

- **Kautotzea:** Sistema ezin da ziur egon nork egiten duen saioa. Erasotzaileak beste baten kontua erabil dezake pasahitzik gabe. Login-a ez da fidagarria.

- **Osotasuna:** Behin sartuta, erasotzaileak datuak alda ditzake: profila, erregistroak... Datu-baseko informazioa ez da fidagarria izaten jarraitzen.

- **Konfidentzialtasuna:** Erasotzaileak beste erabiltzaileen datuak ikus ditzake. Bai saioa hastean agertzen direnak, bai aplikazio barruko informazioa.

- **Zapuztezintasuna:** Sistemak pentsatzen du erabiltzaile batek egin dituela ekintza batzuk, baina benetan erasotzaileak egin ditu haren izenean. Horrek arazoak sortzen ditu nor den erantzule jakiteko.

- **Prestasuna:** Erasotzaile batek kontu desberdinetan sartuz edo datu-baseko erregistroak manipulatuz, sistemaren funtzionamendu normala oztopatu dezake. Hala ere, gure kasuan, prestasunaren kaltea ez da nabarmena.

2. BIGARREN ERASOA: DATUAK ALDATZEKO WEB GUNE MALTZURRA

CSRF (Cross-Site Request Forgery) eraso mota bat da, non erasotzaileak biktima erabiltzaileari bere nahiaren kontrako ekintzak egitera bultzatzen duen web aplikazio batean. Erasoa burutzeko, biktimaren saioa hasita egon behar da aplikazioan eta erasotzaileak biktimari bere webgune maltzurra bisitatzeko engainatu behar du.

Gure CSRF erasoaren helburua da erabiltzaile baten datu pertsonalak automatikoki eta biktimak jakin gabe aldatzea. Eraso-orria bisitatzen duenean, erabiltzaileak ez du ezer sakatu behar, sistema automatikoki exekutatuko da eta helburu-erabiltzailearen datu guztiak aldatuko ditu.

➢ **AHULEZIAK:**

Ahultasuna *app/php/modify_user/index.php* fitxategian ematen da.

- **CSRF Babesik Eza**: Webguneak ez du egiaztatzen eskaera legitimo batetik datorren ala ez. Edozein jatorritako POST eskaerak onartzen ditu.

```
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
        $erabiltzaile = $_POST['Erabiltzaile'] ?? '';
        $nombre = $_POST['Izen_Abizen'] ?? '';
        $telefono = $_POST['Telefonoa'] ?? '';
        $fecha = $_POST['Jaio_Data'] ?? '';
        $email = $_POST['Email'] ?? '';
```

- **SQL Injection Ahultasuna**: Erabiltzailearen sarrerak zuzenki kontsultan sartzen dira, SQL injection erasoak posible eginez.

```
$sql = "UPDATE erabiltzaileak SET Erabiltzaile = '$erabiltzaile',
Izen_Abizen = '$nombre', Telefonoa = '$telefono', Jaio_Data = '$fecha',
Email = '$email' WHERE NAN = '$nan'";
```

- **Autentikazio Logika Akastuna**: Soilik egiaztatzen du session NAN eta URL parametroa berdinak diren, baina ez du eskaeraren jatorria egiaztatzen.

```
if (!isset($_SESSION['nan']) || $_SESSION['nan'] !== ($_GET['user'] ??
null)) {
    header("Location: /login");
    exit();
}
```

➢ **PAUSUAK ERASOA GAUZATZEKO:**

Hasteko, lehenengo erasoaz baliatuz, erabiltzaile baten DNI-a lortu. Ondoren, errepositorioaren csfr_erasoa.html artxiboa deskargatu behar da eta hurrengo lerroa aldatu behar da:

```
<form id="csrfErasoa"
action="http://localhost:81/modify_user?user=XXXXXXXXX" method="POST">
```

Non, “user=XXXXXXXXX” atalean erasotu nahi den erabiltzailearen DNI-a idatzi behar da. Ondoren, artxiboa exekutatu behar da.

➢ **ONDORIOAK:**

- **Kautotzea**: Erasotzaileak biktimaren sesioaz baliatzen da, eta sistema ez da gai eskaeraren jatorria egiaztatzeko. Horrela, erabiltzailearen izenean ekintzak egiten dira, erabiltzaileak jakin gabe.
- **Osotasuna**: Erabiltzailearen datu pertsonalak automatikoki aldatzen dira. Datuen integritatea galtzen da, eta informazioa ez da fidagarria bihurtzen.
- **Zapuztezintasuna**: Ekintzak erabiltzailearen session bidez egiten direnez, ezin da erraz zehaztu benetan nork egin dituen aldaketak.
- **Konfidentzialtasuna/Prestasuna**: CSRF honek ez du informazio sentikorra filtratzen, ezta sistemaren erabilgarritasuna murrizten ere. Beraz, arlo hauetan eragina minimoa da.

3. HIRUGARREN ERASOA: ITEM-AK ATZITZEKO WEB GUNE MALTZURRA

Clickjacking eraso mota bat da, non erasotzaileak biktima erabiltzaileari elementu ezkutu bat gainean klik egitera engainatzen duen. Erabiltzaileak uste du botoi batean klik egiten ari dela, baina berez beste webgune bateko ekintza bat exekutatzen ari da.

Gure clickjacking erasoaren helburua da erabiltzaileari "Saria jasotzeko" botoian klik egitera engainatzea, baina berez babarrun bat ezabatzen duen URL batean klik egiten ari dela. Erabiltzaileak inoiz ez du jakiten zer benetan gertatzen ari den.

➢ **AHULEZIAK:**

Ahultasuna *app/php/delete_item/index.php* fitxategian ematen da.

- **X-Frame-Options Babesik Eza:** Webguneak iframe batean kargatzea galarazten duen babesik ez du.
- **GET Metodoa Erabiltzea Ekintza Kritikoetarako:** Ekintza kritikoak (ezabatzea) GET metodoarekin egiten dira, erraz exploita daitezkeena.

```
if (isset($_GET['item'])) {
    $item_raw = $_GET['item'];
    $item = $conn->real_escape_string($item_raw);

    if ($item === '') {
        echo "<p style='color:red;'>❌ Izena hutsik.</p>";
    } else {
        $check_sql = "SELECT COUNT(*) AS cnt FROM babarrunak WHERE Izena
= '$item'";
        $check_res = $conn->query($check_sql);
        if ($check_res) {
            $row = $check_res->fetch_assoc();
            $check_res->free();

            if ((int)$row['cnt'] === 0) {
                echo "<p style='color:orange;'>ℹ️ \"" .
htmlspecialchars($item_raw) . "\" ez da existitzen.</p>";
            } else {
                $del_sql = "DELETE FROM babarrunak WHERE Izena = '$item'
LIMIT 1";
```

- **Erabiltzaile Autentikazio Egiaztapenik Eza:** Ez du egiaztatzen erabiltzailea saioa hasita dagoen edo baimenik duen.

➢ **PAUSUAK ERASOA GAUZATZEKO:**

Hasteko, item baten izena lortu behar da. Ondoren, errepositorioaren clickjacking.html artxiboa deskargatu behar da eta hurrengo lerroa aldatu behar da:

```
<iframe src="http://localhost:81/delete_item?item=XXXXXXXXX"></iframe>
```

Non, “item=XXXXXXXXX” atalean atzitu nahi den item-aren izena idatzi behar da. Ondoren, artxiboa exekutatu behar da.

➢ **ONDORIOAK:**

- **Osotasuna**: Clickjacking-aren bidez erabiltzaileak nahi gabe *babarruna ezabatzeko ekintza* exekutatzen du. Datuak ustekabean aldatzen dira, eta horrek osotasun galera dakar.

- **Kautotzea**: Erabiltzaileak uste du botoi arrunt batean klik egiten ari dela, baina webguneak beste ekintza bat exekutatzen du. Erabiltzailearen ekintza benetakoa ez denez, haren borondatea engainatzen da.

- **Zapuztezintasuna/Konfidentzialtasuna/Prestasuna**: Sistemak ez du informazio sentikorra erabiltzen, ez du autentikaziorik eskatzen, eta item bat ezabatzeak ez du zerbitzua eten. Beraz, hiru arlo hauek ia ez dira kaltetzen.

ERREFERENTZIAK

OWASP: SQL Injection (2025-11-20 atzitua)
https://owasp.org/www-community/attacks/SQL_Injection

OWASP Cheat Sheet: SQL Injection Prevention (2025-11-20 atzitua)
https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html

OWASP: Cross-Site Scripting (XSS) (2025-11-20 atzitua)
https://owasp.org/www-community/attacks/xss/

OWASP Cheat Sheet: XSS Prevention (2025-11-20 atzitua)
https://cheatsheetseries.owasp.org/cheatsheets/Cross_Site_Scripting_Prevention_Cheat_Sheet.html

OWASP: Cross-Site Request Forgery (CSRF) (2025-11-20 atzitua)
https://owasp.org/www-community/attacks/csrf

OWASP Cheat Sheet: CSRF Prevention (2025-11-20 atzitua)
https://cheatsheetseries.owasp.org/cheatsheets/Cross-Site_Request_Forgery_Prevention_Cheat_Sheet.html

OWASP: Clickjacking (2025-11-20 atzitua)
https://owasp.org/www-community/attacks/Clickjacking

OWASP: Clickjacking Defense Cheat Sheet (2025-11-20 atzitua)
https://cheatsheetseries.owasp.org/cheatsheets/Clickjacking_Defense_Cheat_Sheet.html

OWASP: Path Traversal (2025-11-20 atzitua)
https://owasp.org/www-community/attacks/Path_Traversal

OWASP: File System Vulnerabilities (2025-11-20 atzitua)
https://owasp.org/www-community/vulnerabilities/Path_Traversal

OWASP: Unrestricted File Upload (2025-11-20 atzitua)
https://owasp.org/www-community/vulnerabilities/Unrestricted_File_Upload

OWASP Cheat Sheet: File Upload Security (2025-11-20 atzitua)
https://cheatsheetseries.owasp.org/cheatsheets/File_Upload_Cheat_Sheet.html

OWASP Top 10: Broken Authentication (2025-11-20 atzitua)
https://owasp.org/www-project-top-ten/2017/A2_2017-Broken_Authentication

OWASP: Authentication Cheat Sheet (2025-11-20 atzitua)
https://cheatsheetseries.owasp.org/cheatsheets/Authentication_Cheat_Sheet.html

OWASP: Insecure Direct Object Reference (2025-11-20 atzitua)
https://owasp.org/www-community/attacks/Insecure_Direct_Object_Reference

OWASP: Access Control Vulnerabilities (2025-11-20 atzitua)
https://owasp.org/www-community/Access_Control