Raport z realizacji projektu - Atak phishingowy

1. Informacje ogólne

- **Temat:** Tworzenie fałszywej strony logowania (phishing) w celach edukacyjnych
- **Cel projektu:** Demonstracja ataku phishingowego w bezpiecznym środowisku w celu uświadomienia użytkowników o zagrożeniach
- **Narzędzia:** Python (Flask), HTML, CSS, MSSQL, Git
- **Środowisko testowe:**
    - System operacyjny: Windows
    - Serwer: Flask
    - Baza danych: MSSQL Server (Express)

2. Realizacja projektu

Projekt składa się z trzech głównych elementów:

Stworzenie fałszywej strony logowania

Strona została zaprojektowana w HTML i CSS tak, aby jak najbardziej przypominała prawdziwą stronę logowania banku.
Zawiera dwa etapy logowania:

1. **Pierwszy ekran** – użytkownik wpisuje login.
2. **Drugi ekran** – użytkownik wpisuje hasło.

Kod HTML (login.html i password.html)

```
<form action="{{ url_for('login') }}" method="post">
    <input type="text" name="login" placeholder="Wpisz login" required>
    <button type="submit">Dalej</button>
</form>

<form action="{{ url_for('submit_password') }}" method="post">
    <input type="password" name="password" placeholder="Wpisz hasło" required>
    <button type="submit">Zaloguj</button>
</form>
```

Backend - Obsługa zapisu danych

Backend został napisany w **Python Flask** i łączy się z bazą MSSQL.

- Po wpisaniu loginu użytkownik jest przekierowany na stronę wpisywania hasła.
- Dane są zapisywane w bazie w trzech tabelach:
    - users (loginy)
    - passwords (hasła)
    - logs (IP użytkownika, User-Agent, czas logowania)

Strona ostrzegawcza (Scam)

Po wpisaniu hasła użytkownik zostaje przekierowany na stronę informującą o ataku phishingowym.

Efekty wizualne:

- Migające tło (czerwony, czarny, żółty)
- Duży, wyraźny komunikat w centrum ekranu

3. Baza danych - Struktura

Baza MSSQL została podzielona na **trzy tabele**, które są ze sobą powiązane kluczami obcymi.

```
CREATE TABLE users (
    id INT IDENTITY(1,1) PRIMARY KEY,
    login NVARCHAR(255) NOT NULL UNIQUE
);

CREATE TABLE passwords (
    id INT IDENTITY(1,1) PRIMARY KEY,
    user_id INT NOT NULL,
    password NVARCHAR(255) NOT NULL,
    FOREIGN KEY (user_id) REFERENCES users(id) ON DELETE
CASCADE
);

CREATE TABLE logs (
```

```
    id INT IDENTITY(1,1) PRIMARY KEY,
    user_id INT NOT NULL,
    ip_address NVARCHAR(45) NOT NULL,
    user_agent NVARCHAR(500) NOT NULL,
    login_time DATETIME DEFAULT GETDATE(),
    FOREIGN KEY (user_id) REFERENCES users(id) ON DELETE
CASCADE
);
```

4. Wyniki testów

- Flask działa poprawnie
- Loginy i hasła są zapisywane w bazie MSSQL
- IP i dane przeglądarki są zbierane i logowane
- Użytkownik nie może przejść na stronę hasła bez wpisania loginu
- Strona "Zostałeś oszukany!" działa poprawnie i ostrzega użytkownika

5. Wnioski i podsumowanie

- Projekt pokazał, jak działa atak phishingowy i jakie dane można w ten sposób przechwycić.
- Dzięki logowaniu IP i User-Agent można zobaczyć, skąd użytkownik próbował się zalogować.
- Ważne jest, aby użytkownicy byli świadomi takich zagrożeń i korzystali z uwierzytelniania wieloskładnikowego oraz certyfikatów SSL.

6. Jak uruchomić projekt?

Zainstaluj wymagane pakiety:

```
pip install flask pyodbc
```

Uruchom serwer Flask:

```
python app.py
```