Sprawozdanie projektu z przedmiotu: Ochrona Danych w Systemach Informatycznych `prosta bezpieczna aplikacja internetowa`

Michał Żdanuk

Informatyka Stosowana
Wydział Elektryczny
Politechnika Warszawska
Styczeń 2023

Spis treści

1 Wstęp **3**

2 Wymagania bezpieczeństwa **3**
2.1 Wymagania podstawowe . . . 3
2.2 Wymagania dodatkowe . . . 3

3 Rozwiązanie **3**
3.1 Technologia . . . 3
3.2 Opis realizacji wymagań . . . 4
3.2.1 Restrykcyjna walidacja danych . . . 4
3.2.2 Przechowywanie hasła chronionego hash'em z dodaną solą . . . 4
3.2.3 Możliwość udostępniania obrazków prywatnie lub wybranym użytkownikom . . . 4
3.2.4 Weryfikacja bezpieczeństwa przechowywanych plików . . . 4
3.2.5 Zabezpieczenie transmisji protokołem https . . . 5
3.2.6 Możliwość zmiany hasła . . . 5
3.2.7 Możliwość odzyskania hasła . . . 5
3.2.8 Monitorowanie liczby nieudanych prób logowania . . . 6
3.2.9 Informowanie użytkownika o jakości jego hasła (jego entropii) . . . 6
3.2.10 Kontrola odporności nowego hasła na ataki słownikowe . . . 6

4 Wnioski **6**

1 Wstęp

Celem przedmiotu było zrealizowanie projektu - napisanie prostej aplikacji spełniającej wysokie standardy bezpieczeństwa. Tworząc oprogramowanie jako programiści bardzo często skupiamy się na funkcjonalności i wydajności systemu, postępując w ten sposób często zapominamy o kwestiach bezpieczeństwa. Dlatego w tym projekcie należało się skupić nie na wyglądzie, czy liczbie funkcjonalności, a na sprawdzaniu danych wejściowych oraz restrykcyjnej walidacji/sanityzacji wykorzystując do tego algorytmy kryptograficzne.

Dobrym przedmiotem do realizacji tego rodzaju standardów bezpieczeństwa jest moduł uwierzytelniania. Jest on kluczowym elementem systemu i musi być maksymalnie bezpieczny, by nie doszło do wycieku danych użytkowników bądź podszywania się pod nich.

Za modułem uwierzytelniania została napisana prosta aplikacja do dzielenia się obrazkami. Aplikacja umożliwia kontrole dostępu do obrazków: publiczne, prywatne oraz udostępniane tylko wybranym użytkownikom. Kluczowym zagadnieniem tutaj jest restrykcyjna weryfikacja załączanych plików, by nie doszło do sytuacji, gdy wstrzyknięty zostanie złośliwy kod.

2 Wymagania bezpieczeństwa

Na zajęciach nauczono mnie wielu technik mających na celu zapewnienie odpowiedniego poziomu bezpieczeństwa. Począwszy od walidacji, kończąc na przechowywaniu danych/sekretów. Napisana przeze mnie aplikacja spełnia wszystkie zadane podstawowe wymagania oraz realizuje większość dodatkowych wymagań. Poniżej prezentuje listę zadanych wymagań.

2.1 Wymagania podstawowe

Zrealizowane wymagania podstawowe:

- restrykcyjna walidacja danych pochodzących z formularza login-hasło,
- przechowywanie hasła chronione funkcją hash z solą,
- możliwość umieszczenia na serwerze obrazków dostępnych prywatnie lub dla określonych użytkowników,
- weryfikacja bezpieczeństwa przechowywanych plików graficznych,
- zabezpieczenie transmisji poprzez wykorzystanie protokołu https,
- możliwość zmiany hasła,
- możliwość odzyskania dostępu w przypadku utraty hasła.

2.2 Wymagania dodatkowe

Zrealizowane wymagania dodatkowe:

- monitorowanie liczby nieudanych prób logowania,
- informowanie użytkownika o jakości jego hasła (jego entropii),
- kontrola odporności nowego hasła na ataki słownikowe.

3 Rozwiązanie

3.1 Technologia

Do realizacji projektu zdecydowałem się na język programowania: **Python**. Frameworkiem, w którym została napisana aplikacja był **Flask**. Flask umożliwia bardzo szybkie pisanie aplikacji webowych. Daje możliwość kontrolowania adresu url poprzez dekoratory (app.route("/sciezka")). Co więcej Flask pozwala na tworzenie jednolitych formularzy danych, które wykorzystałem jako formularze: rejestracji, logowania, dawania

dostępu do zdjęć, czy publikowania obrazkow. Do przechowywania danych wykorzystałem prostą bazę danych SQLITE. W bazie zdefiniowałem cztery tabele: user, permission, picture oraz public_picture. Główna tabela **user** przechowuje dane użytkownika w tym hasło, które jest posolone i hash'owane. Tabele **picture** oraz **public_picture** przechowują w sobie nazwę zapisanego pliku oraz id użytkownika, do którego należy obrazek. Tabela **permission** określa prawa dostepu - jaki użytkownik komu udostępnia swoje obrazki.

Wykorzystane biblioteki Pythonowe:

- bcrypt - do dodawania soli do hasła oraz hashowania
- re - do sprawdzania wyrażeń regularnych, celem nałożenia restrykcyjnej walidacji na pola: login i hasło
- SQLAlchemy - do zarządzania bazą danych
- os, dotenv - do wyciągnięcia z systemu operacyjnego sekretów, tak by nie przechowywać tak wrażliwych danych wewnątrz kodu
- wtforms - do częściowej realizacji nałożenia walidacji na dane w formularzach (m.in. sprawdzenie pattern'u mail'a, czy nałożenia restrykcji na długość wprowadzanych danych i ich typ)
- itsdangerous - do generowania czasowego JSONWebSignature celem wysyłania mail'a z możliwością zmiany hasła
- PIL - do zapisywania zdjęć oraz odpowiedniego przeskalowania

3.2 Opis realizacji wymagań

3.2.1 Restrykcyjna walidacja danych

Do sprawdzenia bezpieczeństwa wprowadzanych danych został napisany walidator, którym objąłem pola: nazwa użytkownika oraz haslo. Walidator sprawdza wzorzec pola - tzn. pole może być złożone tylko z małych/wielkich liter, cyfr oraz znaków '.' , '-' i '_'. Dodatkowo napisalem funkcje sprawdzajace, czy wprowadzone dane nie są próbą ataków typu **XSS** lub **SQL INJECTION**. Poza własnymi walidatorami wykorzystałem oferowane przez Flask do sprawdzenia długości znaków wprowadzanych danych, minimalnej i maksymalnej liczby znaków, sprawdzenia wzorca mail'a, sprawdzenie rozszerzenia .png zamieszczanych zdjęć oraz walidatory do sprawdzenia typu wprowadzanych danych.

3.2.2 Przechowywanie hasła chronionego hash'em z dodaną solą

Wprowadzone przez użytkownika hasło jest wzbogacone o sól oraz hash'owane. Zrealizowałem to za pomocą biblioteki bcrypt. Wykorzystane metody: **hashpw()**, **gensalt()**, **checkpw()**.

3.2.3 Możliwość udostępniania obrazków prywatnie lub wybranym użytkownikom

Udostępnianie zdjęć realizowane jest przez prosty formularz, który ma tylko jedno pole: na umieszczenie pliku. Do wyboru mamy możliwość publikacji publicznej, bądź chronionej. Aby inne osoby widziały nasze chronione obrazki należy przejść do zakładki `Permissions` oraz wpisać nazwę użytkownika, któremu chcemy dać takie uprawnienie. Istnieje możliwość późniejszego odebrania uprawnienia użytkownikowi. Te pola również są restrykcyjnie walidowane.

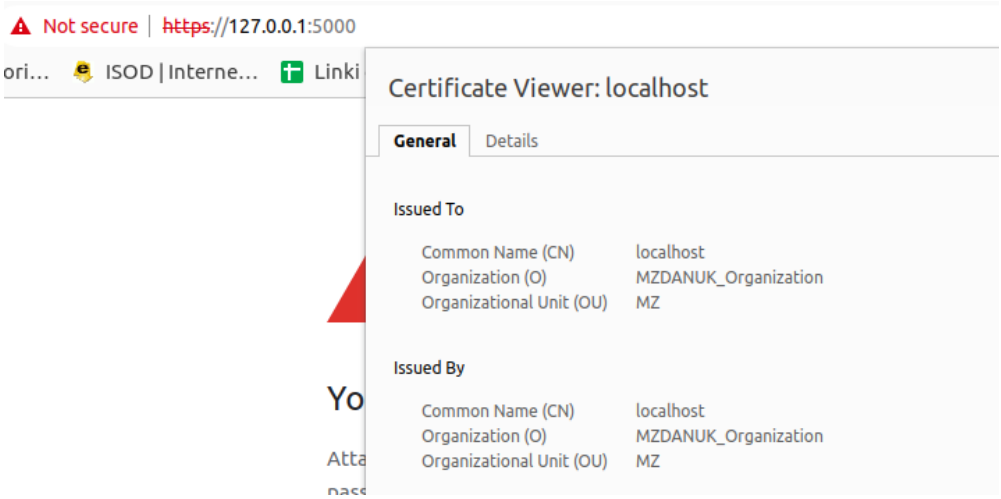
3.2.4 Weryfikacja bezpieczeństwa przechowywanych plików

Weryfikację plików graficznych zrealizowałem poprzez czytanie bajtowe załączanego pliku. Następnie sprawdzane jest kilka początkowych bajtów. Pliki graficzne png muszą zawierać w tych początkowych bajtach znacznik **PNG**. Jeżeli nastąpi próba zamieszczenia złośliwego pliku, w którym zmieniono sztucznie rozszerzenie na .png, to funkcja wykryje to i nie pozwoli na zapisanie pliku po stronie serwera.

3.2.5 Zabezpieczenie transmisji protokołem https

W celu dodania możliwości transmitowania protokołem https należało w funkcji uruchamiającej aplikacje dodać parametr: ssl_context=('cert.pem', 'key.pem'). Samo podpisany certyfikat wygenerowałem poleceniem: `openssl req -x509 -newkey rsa:4096 -nodes -out cert.pem -keyout key.pem -days 365`.

```
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:PL
State or Province Name (full name) [Some-State]:Mazowieckie
Locality Name (eg, city) []:Warsaw
Organization Name (eg, company) [Internet Widgits Pty Ltd]:MZDANUK_Organization
Organizational Unit Name (eg, section) []:MZ
Common Name (e.g. server FQDN or YOUR name) []:localhost
Email Address []:mz@pl
```



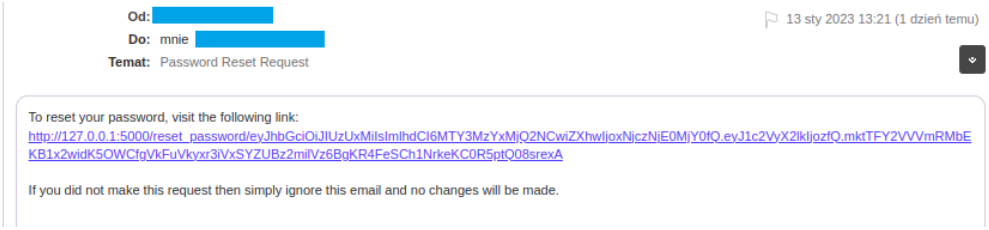
Rysunek 1: zrzut ekranu prezentujący połączenie https

3.2.6 Możliwość zmiany hasła

Możliwość zmiany hasla, jak i innych danych takich jak nazwa użytkownika, czy mail znajduje się w zakładce `Account`, którą widzi każdy zalogowany użytkownik. Aby zmienić hasło należy podać nowe oraz potwierdzić operacje podając aktualne hasło.

3.2.7 Możliwość odzyskania hasła

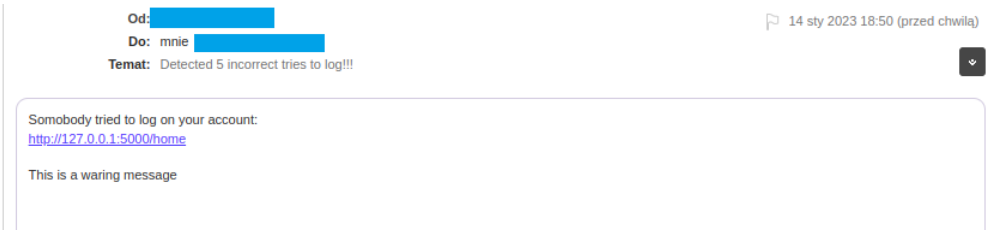
Możliwość odzyskania hasła istnieje po podaniu mail'a związanego z podanym hasłem. Przy użyciu JSON-Tokena, wysyłany jest na mail link do zrestartowania hasła. Link aktywny jest tylko przez 30 minut od wysłania.



Rysunek 2: zrzut ekranu prezentujący maila do odzyskania hasła

3.2.8 Monitorowanie liczby nieudanych prób logowania

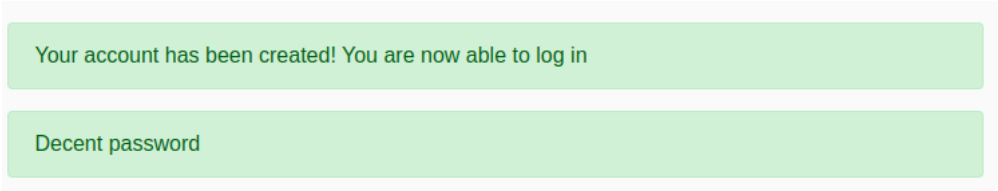
Monitorowanie liczby nieudanych logowań realizowane jest za pomocą ciasteczek. Po każdej nieudanej próbie wyświetlany jest komunikat o stanie nieudanych logowań. Po wykonaniu 5 nieprawidłowych prób zostaniemy przekierowani ze strony logowania i zostanie wysłany mail z komunikatem o podejrzanej aktywności na adres konta, gdzie probowane się zalogować.



Rysunek 3: zrzut ekranu prezentujący maila informującego o próbach logowania

3.2.9 Informowanie użytkownika o jakości jego hasła (jego entropii)

Przy rejestracji w przypadku podania zbyt słabego hasła użytkownikowi wyświetlany jest odpowiedni komunikat. Przy pomyślnej rejestracji wyświetlany jest komunikat o jakości hasła w zależności od poziomu entropii hasła.



Rysunek 4: zrzut ekranu prezentujący maila informujacych o probach logowania

3.2.10 Kontrola odporności nowego hasła na ataki słownikowe

Do kontroli odporności przed atakami słownikowymi przyłożyłem szczególną uwagę. Porównywane jest hasło przy rejestracji z trzema plikami. Są to: 500 najgorszych haseł, 50000 najpopularniejszych polskich slow, 10000 najczęściej używanych słów/haseł angielskich. Jeżeli podane przez użytkownika hasło znajduje się, na którejś z list to następuje odmowa i jest on informowany o zbyt słabym podanym haśle. Co więcej, w przypadku listy polskich slow pokusiłem się również o rozszerzenie tej listy poprzez sprawdzanie, tych slow z dołożonym najczęściej występującymi końcówkami (ang. suffixes) takimi jak '1', '123', '12345', czy 'x'.

4 Wnioski

Zadanie okazało się nie tak banalne, jak mogłoby się z początku wydawać. Trudności sprawiło mi wymyślenie sposobu dzielenia się obrazkami oraz jego realizacja. Początkowo walidacja danych przeze mnie była znacząco zbyt słaba, a dopiero po udaniu się na konsultacje poprawiłem i wzbogaciłem walidacje. Również nieprostym problemem była weryfikacja bezpieczeństwa wstawianych obrazków. Zadowolony jestem z tego, że udało zrealizować się wszystkie podstawowe wymagania oraz znaczną część dodatkowych postawionych mi przy rozdawaniu projektów.