Szkoła Główna Gospodarstwa Wiejskiego
w Warszawie
Wydział Zastosowań Informatyki i Matematyki

Mateusz Tracz
172391

Implementacja serwisu umożliwiającego dwuetapową weryfikację użytkownika

Implementation of two factor authentication service
at the Warsaw University of Life Sciences – SGGW

Praca dyplomowa inżynierska
na kierunku Informatyka

Praca wykonana pod kierunkiem
dr. hab. Alexandera Prokopenya, prof. SGGW

Wydział Zastosowań Informatyki i Matematyki

Katedra Zastosowań Informatyki
Zakład Modelowania i Analizy Systemów

Warszawa 2017

Oświadczenie promotora pracy

Oświadczam, że niniejsza praca została przygotowana pod moim kierunkiem i stwierdzam, że spełnia ona warunki do przedstawienia tej pracy w postępowaniu o nadanie tytułu zawodowego.

Data ..... Podpis promotora pracy .....

Oświadczenie autora pracy

Świadom odpowiedzialności prawnej, w tym odpowiedzialności karnej za złożenie fałszywego oświadczenia, oświadczam, że niniejsza praca dyplomowa została napisana przeze mnie samodzielnie i nie zawiera treści uzyskanych w sposób niezgodny z obowiązującymi przepisami prawa, w szczególności z ustawą z dnia 4 lutego 1994 r. o prawie autorskim i prawach pokrewnych (Dz. U. Nr 90 poz. 631 z późn. zm.)

Oświadczam, że przedstawiona praca nie była wcześniej podstawą żadnej procedury związanej z nadaniem dyplomu lub uzyskaniem tytułu zawodowego.

Oświadczam, że niniejsza wersja pracy jest identyczna z załączoną wersją elektroniczną. Przyjmuję do wiadomości, że praca dyplomowa poddana zostanie procedurze antyplagiatowej.

Data ..... Podpis autora pracy .....

Spis treści

Streszczenie

TODO: POLSKI TYTUŁ
TODO: POLSKIE STRESZCZENIE

Słowa kluczowe – TODO: POLSKIE TAGI implementacja, SGGW, Szkoła Główna Gospodarstwa Wiejskiego

Summary

TODO: ANGIELSKIE TYTUŁ
TODO: ANGIELSKIE STRESZCZENIE

Keywords – TODO: ANGIELSKIE TAGI thesis, implementation, SGGW, Warsaw University of Life Sciences

1 Wstęp

1.1 Cel pracy

1.2 Pojęcie uwierzytelnienia wielopoziomowego

1.3 Korzyści płynące z używania uwierzytelnienia wielopoziomowego

2 Elementy kryptografii

2.1 Kryptografia symetryczna oraz asymetryczna

2.2 Szyfry blokowe

2.3 Szyfry strumieniowe

2.4 Funkcja skrótu

2.5 Kod uwierzytelnienia wiadomości

2.6 MAC bazujący na funkcji skrótu

2.7 Funkcje typu key stretching

2.8 Pojęcia entropii

3 Kryptografia w praktyce

3.1 Pojęcia pomocnicze

Przed przystąpieniem do opisu praktycznych aspektów kryptografii użytych w projekcie, wymagane jest wyjaśnienie pojęć wykorzystywanych w mechanizmie haseł jednorazowych, lecz które nie są bezpośrednio związane z kryptografią.

3.1.1 Kodowanie transportowe

Kodowanie transportowe wykorzystywane jest w przypadku, gdy zachodzi potrzeba transferu danych w środowiskach, które pozwalają na przesyłanie wyłącznie znaków ASCII.

Użycie kodowania transportowego jest konieczne w celu zachowania kompatybilności przy pracy z protokołami, które przystosowane są do pracy na danych 7-bitowych. W takim przypadku najstarszy bit jest zerowany, co mogłoby uszkodzić przesyłane dane. W przypadku przesyłania wyłącznie znaków ASCII zerowanie najstarszego bitu nie jest problemem, gdyż wszystkie znaki w podstawowej tablicy ASCII mają ten bit wyzerowany.
Bardziej współczesnym przykładem wykorzystania kodowania transportowego jest osadzanie danych graficznych bezpośrednio w kodzie HTML. Konieczne jest wówczas zakodowanie danych w celu wyeliminowania ryzyka pojawienia się znaków ’<’ oraz ’>’, które mogłyby być zinterpretowane jako tagi HTML.

Aby ujednolicić implementacje kodowania transportowego został stworzony dokument RFC 4648 [2], w którym opisany jest prawidłowy sposób implementacji oraz to jaki typ kodowania wybrać w zależności od nałożonych wymagań.

Kodowanie Base64

Najczęściej spotykanym typem kodowania transportowego jest kodowanie Base64. Kodowanie to konwertuje dowolny ciąg bajtów do postaci ciągu złożonego z małych i wielkich liter, cyfr oraz znaków ’+’ i ’/’. Jeżeli po zakodowaniu końcowa część danych jest mniejsza niż 24 bity używany jest także znak ’=’ jako dopełnienie.
Sam proces kodowania polega na pobraniu 24 bitów danych a następnie podzieleniu ich na 4 grupy po 6 bitów. Każda z grup jest interpretowana jako indeks tablicy ustalonego alfabetu Base64. Dla każdej z grup za pomocą indeksu odczytywany jest znak a następnie dopisywany jest on do ciągu zakodowanego.
Istnieje również odmiana kodowania Base64 przystosowana do użycia w przypadku adresów URL czy nazw plików. W alternatywie tej zamiast znaków ’+’, ’/’, które mogłyby zostać błędnie zinterpretowane np w środowisku systemu plików, używane są znaki ’-’ oraz ’_’.

Kodowanie Base32

W porównaniu do kodowania Base64, dane zakodowane w Base32 są dużo bardziej czytelne dla ludzi. Właściwość ta spowodowana jest faktem, że w kodowaniu Base32 nie ma znaczenia wielkość liter, dzięki czemu przykładowo nie ma problemu z rozróżnieniem małej litery 'L' z wielką literą 'I' ('l' oraz 'I').
Alfabet kodowania Base32 składa się z 32 znaków ASCII oraz znaku '=' pełniącego funkcje dopełnienia. Proces kodowania polega na pobraniu 40 bitów danych a następnie ustawienie ich w osiem 5-bitowych grup. Każda z 8 grup interpretowana jest jako jeden ze znaków alfabetu Base32.
Podobnie jak przy kodowaniu Base64, wymagane jest tutaj wstawienie dopełnienia w sytuacji, gdy długość ostatniej z grup jest mniejsza od 40 bitów.

Tabela 3.1. Alfabet w kodowaniu Base32

Indeks	Znak	Indeks	Znak	Indeks	Znak	Indeks	Znak
0	A	9	J	18	S	27	3
1	B	10	K	19	T	28	4
2	C	11	L	20	U	29	5
3	D	12	M	21	V	30	6
4	E	13	N	22	W	31	7
5	F	14	O	23	X		
6	G	15	P	24	Y		
7	H	16	Q	25	Z		
8	I	17	R	26	2		

3.1.2 Czas uniksowy

Czas uniksowy jest sposobem na reprezentację punktu w czasie, polegającym na mierzeniu sekund, które upłynęły od daty 1 stycznia 1970 (UTC). W systemach uniksowych zwykle reprezentowany jest w postaci 32-bitowej liczby całkowitej ze znakiem.
W przypadku architektur typu serwer-klient wskazane jest synchronizowanie czasu wykorzystując czas uniksowy, gdyż nie zależy on od lokalizacji w której jest mierzony. Właściwość ta eliminuje problem synchronizacji czasu pomiędzy strefami czasowymi.

3.1.3 Ujednolicony identyfikator zasobów

Ujednolicony identyfikator zasobów (ang. Uniform Resource Identifier, URI) jest ciągiem znaków jednoznacznie identyfikującym dany zasób.
Składnia identyfikatora jest wyrażana następująco:

schemat ":" ścieżka ["?" zapytanie] ["#" fragment]

Warto zauważyć, że składnia ta determinuje schemat (protokół), jaki wykorzystywany jest przy interakcji z identyfikowanym zasobem.

Przykłady identyfikatorów:

- ftp://randomftp.com/files/file.docx
- https://www.randomwebsite.pl/index.html
- mailto:jan.nowak@wp.pl
- tel:+48-25-123-88

Szczegóły dotyczące standardu URI są opisane w dokumencie RFC 3986 [1].

3.2 Hasło jednorazowe

3.3 Interfejs Windows Data Protection

4 Ataki na mechanizm OTP

4.1 Atak urodzinowy

4.2 Atak przez powtórzenie

4.3 Atak „Man in the middle”

4.4 Phishing

5 PicnicAuth

5.1 Architektura projektu

5.2 Generowanie OTP po stronie użytkownika

5.3 Przechowywanie sekretu użytkownika

5.4 Przykład użycia projektu

5.5 Planowane ulepszenia

6 Zakończenie

6.1 Podsumowanie i wnioski

6.2 Podziękowania

7 Spis literatury

[1] T. Berners-Lee, R. Fielding, L. Masinter. *Uniform Resource Identifier (URI): Generic Syntax.*
`https://tools.ietf.org/pdf/rfc3986.pdf`, 2005

[2] S. Josefsson *The Base16, Base32, and Base64 Data Encodings*
`https://tools.ietf.org/pdf/rfc4648`, SJD, 2006

Wyrażam zgodę na udostępnienie mojej pracy w czytelniach Biblioteki SGGW w tym w Archiwum Prac Dyplomowych SGGW.

. . . . . .

(czytelny podpis autora pracy)