accenture

BEZPIECZEŃSTWO APLIKACJI W PHP

Patryk Arsiuta

Jan 28, 2020

CONFIDENTIAL PRESENTATION

accenture

CONFIDENTIAL
FOR COMPANY
INTERNAL USE ONLY

SQL INJECTION

CZYM JEST SQL INJECTION

Jest to podatność aplikacji, dzięki której można wstrzyknąć własny fragment zapytania SQL.

Zazwyczaj wykorzystywana jest do odczytu dowolnych danych z bazy, jednak – w zależności od używanych technologii – atak może pozwolić również na zmianę danych, zapis plików na dysku czy nawet wykonywanie dowolnych poleceń systemu operacyjnego.

Najczęściej związana jest z błędnym podejściem do budowania zapytań do baz danych przez programistów.

- Wstrzykiwanie kodu SQL w adresie przeglądarki, gdy budujemy zapytanie z wykorzystaniem **$_GET**.
- Niezabezpieczony (brak walidacji) formularz, np. logowania.

OCHRONA

**PODATNOŚCI, TO BRAK ZABEZPIECZEŃ
PRZED NIMI MOŻNA SIĘ BRONIĆ**

Zapytania parametryzowane

- Najbardziej podstawową i najbardziej zalecaną metodą ochrony są zapytania parametryzowane.

Walidacja typów danych

- Jednym z typowych zaleceń, jeśli chodzi o zabezpieczenie się, jest walidacja danych przyjmowanych od użytkownika.

Stosowanie systemów klasy ORM

MATERIAŁY

HTTPS://WWW.PHP.NET/MANUAL/EN/SECURITY.DATABASE.SQL-INJECTION.PHP

HTTPS://WWW.W3SCHOOLS.COM/SQL/SQL_INJECTION.ASP

HTTPS://WWW.GURU99.COM/LEARN-SQL-INJECTION-WITH-PRACTICAL-EXAMPLE.HTML

HTTPS://WWW.RAPID7.COM/FUNDAMENTALS/SQL-INJECTION-ATTACKS/

XSS

CROSS-SITE SCRIPTING

Główne skutki podatności to możliwość

- Wykonania dowolnych akcji w kontekście zalogowanego użytkownika
- Odczytu dowolnych danych w kontekście zalogowanego użytkownika

Ochrona przed XSS

- Enkodowanie danych w sposób odpowiedni dla kontekstu, w którym te dane są umieszczane
- Dobrą praktyką jest czyszczenie HTML ze złośliwych elementów
- W praktyce czcęsto używa się silników dablonów, które automatycznie wykrywają kontekst i enkodują dane.

RODZAJE ATAKU

Reflected XSS

- Gdy kod HTML/JS zawarty w dowolnym parametrze zapytania (np. GET, POST czy nawet w ciasteczkach) wyświetlany jest następnie w odpowiedzi.
- Praktyczne wykorzystanie takiego wariantu polega zazwyczaj na wysłaniu ofierze odpowiednio spreparowanego linku zawierającego złośliwy kod.

Persistant/stored XSS

- W tym przypadku złośliwy kod JS zostaje zapisany w bazie (lub innym zewnetrznym systemie) i wykonany automatycznie po przejściu na odpowiednią podstrone.
- Jako przykład takiego typu podatności XSS weźmy system blogowy, na którym użytkownicy mogą zostawiać komentarze. Do jednego z nich można wstrzyknąc zainfekowany kod, który przy odczytywaniu komentarza w przeglądarce się wykonuje.

MATERIAŁY

https://www.veracode.com/security/xss

https://sekurak.pl/kilka-slow-o-podatnosci-xss-oraz-polyglot-xss/

https://lukasz-socha.pl/php/niebezpieczny-kod-xss/

UWIERZYTELNIANIE

AUTORYZACJA

ZARZĄDANIE SESJĄ

ATAKI BRUTE-FORCE

Proces weryfikacji danych uwierzytelniających wprowadzonych w formularzu logowania przez użytkownika polega na sprawdzeniu po stronie serwera, czy użytkownik podał zestaw danych odpowiedający informacji zapisanej w systemie.

Aby zabezpieczyć się przed nadużywaniem tego warto zaimplementować mechanizm CAPTCHA, należy również nałożyć na ten proces odpowiednią liczbą prób, po których użytkownik zostaje zablokowany na jakiś czas.

ZARZĄDZANIE SESJĄ

Analizując zagrożenia związane z mechanizmem zarządzania sesją, szybko dojdziemy do wniosku, że newralgicznym punktem jest identyfikator sesji. Identyfikator sesji przekazywany jest za pomocą ciasteczek HTTP.

Zadanie:

Przekopiować sesję zalogowanego użytkownika z jednej przeglądarki i wykorzystać ją w innej, w ten sposób, że będziemy zalogowani na dwóch przeglądarkach, logując się na jednej.

MATERIAŁY

https://coderwall.com/p/sauviq/brute-force-protection-in-php

https://www.comparitech.com/blog/information-security/brute-force-attack/

https://haker.edu.pl/2016/05/06/xss-i-session-hijacking-cookies-wep-15/