Vysoké učení technické v Brně
Fakulta informačních technologií

WAP
Výuková aplikace demonstrující webové útoky

April 29, 2021 Svätopluk Hanzel

1 Úvod

Cieľom tohto projektu je implementovať webovú aplikáciu, ktorá dokáže demonštrovať základné zraniteľnosti webových aplikácií ako XSS, CSRF a Clickjacking a zároveň demonštruuje aj ochranu proti nim.

1.1 XSS

Cross-Site Scripting (XSS) je útok založný na vložení/podstrčení škodlivého kódu do cieľovej webovej stránky, ktorá je za bežných okolností dôveryhodná a nie je nebezpečná. Vložený kód je väčšinou vo forme JavaScriptu, ktorý sa spúšťa až v prehliadači obete, ktorá je už v tom čase autentifikovaná a z pohľadu servera dôveryhodná. Nebezpečenstvo takejto zraniteľnosti sa môže líšiť a pohybuje sa od relatívne neškodného zobrazovania cudzieho obsahu až po ukradnutie *HTTP cookies*[1], čím dokáže útočník napodobniť obeť a vykonávať akcie v jej mene.

XSS útoky sa ďalej rozdeľujú na:

- **lokálne** - útok je založený na úprave parametru stránky, ktorý sa následne použije ako súčasť zdrojového kódu stránky.
- **perzistentné** - škodlivý kód je uložený v DB a načíta sa všetkým návštevníkom stránky.

Ochrana Ochrana proti útokom XSS spočíva hlavne v odstránení potenciálne nebezpečného kódu zo všetkých vstupov, resp. v znemožnení ich injekcie do aplikácie. Toto zahŕňa hlavne sanitáciu vstupných polí na strane klienta aj serveru a sanitáciu kódu vkladaného do stránky.

1.2 CSRF

Cross-Site Request Forgery (CSRF) je útok založený na exekúcií nechcených požiadaviek v mene prihláseného používateľa. Útočník tohto môže docieliť pomocou sociálneho inžinierstva, kde cez napr. podvodný mail navedie prihláseného používateľa na svoju stránku, ktorá následne zneužije toho, že je používateľ prihlásený v cieľovej aplikácií a spustí požiadavku na server, ktorá sa vykoná. V prípade, že je prihlásený administrátor, dokáže tento typ útoku kompromitovať celú aplikáciu.

Ochrana Proti CSRF útokom je možná hlavne použitím tzv. *autorizačného token-u*, ktorý môže byť unikátny pre jedno prihlásenie používateľa alebo (lepšie) pre každý request. Unikátny token pre každý request však môže znamenať zhoršený užívateľský zážitok, pretože sa nebudú dať odoslať formuláre, na ktoré sa používateľ dostane spätným prechádzaním históriou svojho prehliadača.

1.3 Clickjacking

Clickjacking je útok založený na navedení obete na zdanlivo neškodnú stránku, kde je však zavádzaný a svojimi akciami v skutočnosti spustí nechcené akcie, či pošle nechcené požiadavky na server. Clickjacking je väčšinou prevádzaný pomocou viacerých vrstiev, kedy je cieľová stránka zobrazena cez *iframe* v nevidieľnej vrstve (`opacity: 0`) a pod ňou je napr. nejaké tlačidlo, ktoré používateľ síce vidí, ale nemože naň kliknúť.

[1] https://developer.mozilla.org/en-US/docs/Web/HTTP/Cookies

2 Implementácia

Projekt je implementovaný formou demo bankovej webovej aplikácie, ktorá má oddelenú frontend a backend časť. Komunikácia medzi nimi je riešená cez AJAX HTTP requesty s JSON enkódovaním.

2.1 Frontend

Frontend časť aplikácie je implementovaná pomocou JS knižnice React[2], ktorá sa stará o vykresľovanie a aktualizáciu stránky. Ďalšími použitými knižnicami sú react-router[3] sanitize-html[4]. Samotná aplikácia je potom rozdelená na 2 časti - prihlasovanie a samotnú aplikáciu.

Prihlasovanie prebieha AJAX requestom na backend endpoint `login.php`. Následne sa údaje uložia do `window.localStorage`, čo na rozdieľ od `sessionStorage` umožňuje clickjacking útok. Portálová časť frontendu sa ďalej skladá z niekoľkých ciest, ktoré ukazujú základnú prácu so systémom a zraniteľnosti.

/transactions/list/ obsahuje zoznam transakcií, v ktorých sa dá vyhľadávať. Avšak zadaním škodlivého kódu do poľa sa dá uskutočniť XSS útok. Napríklad `<img src="1" onerror="alert('you have been pwnd!');">` zobrazí vyskakovacie okno. Tento útok sa dá zároveň zopakovať pomocou URL `http://localhost:3000/transactions/list/%3Cimg%20src=%221%22%20onerror=%22alert('you%20have%20been%20pwnd!');%22%3E`. takto dokáže útoční získať kontrolu nad prehliadačom obete, získať cookies a pod. Túto zraniteľnosť ďalej opravuje `/transactions/list-secure/`

/transactions/new umožňuje vytvorenie novej transakcie v konte používateľa. Súčasťou tohto formuláru je aj checkbox, ktorý mení použitý endpoint zo zabezpečenej varianty na nezabezpečenú, čím ukazuje funkcionalitu overovania CSRF tokenu.

2.2 Backend

Backend časť aplikácie je založená na už zastarej verzii PHP 5.3 bez použitia akéhokoľvek frameworku. Obsahuje spoločný kód na spracovanie požiadaviek (vytvorenie session, overenie prihlásenia, atď.) a jednotlivých PHP skriptov, ktoré vykonávajú samotné akcie.

login.php Overuje meno a heslo používateľa a v prípade zhody (test:test) nastaví v jeho session príznak prihlásenia. Zároveň preňho vytvorí niektoré údaje potrebné pre funkciu aplikácie a vygeneruje mu CSRF token, na ochranu pre CSRF útokmi.

logout.php Zruší session používateľa, čím príde k odhláseniu

account.php Umožňuje prihlásenému používateľovi získať informácie o jeho účte vrátane aktuálneho zostatku a všetkých transakcií.

[2] https://reactjs.org/
[3] https://reactrouter.com
[4] https://github.com/apostrophecms/sanitize-html

transfer.php Je skript implementujúci ukážkové prevedenie financií z účtu prihláseného používateľa inému používateľovi. **Nezabezpečená verzia**

transfer_secure.php Implemetuje rovnakú funkcionalitu ako `transfer.php`, no zároveň kontroluje CSRF token v hlavičke `X-XSRF-TOKEN`. Pokiaľ sa nezhoduje s tokenom vygenerovaným pre túto session používateľa, považuje používateľa za neprihláseného.

Ďalšie nastavenie backend časti je v pribalenom `.htaccess` súbore, ktorý nastavuje Apache server, aby pridával k jednotlivým requestom CORS[5] hlavičky.

2.3 Aplikácia útočníka

Aplikácia útočníka je len kolekcia jednoduchých HTML stránok, ktoré demonštruujú útoky v ich najzákladnejšej forme.

image.html je stránka demonštruujúca CSRF útok na nezabezpečenej verzii backend endpointu `transaction.php`. Pomocou jednoduchého načítania obrázku s URL odkazujúceho na `transfer.php` sa vykoná prevod financií na účet útočníka.

image-futile.html demonštruuje ochranu proti CSRF útoku pomocou `transfer_secure.php` a GET request-u, ktorý nie je podporovaný.

form-futile.php taktiež demonštruuje ochranu proti CSRF útokom, no tu sa používa POST request. Bez platného CSRF tokenu v hlavičke request je však aj tento request neplatný.

iphone-winner.html je stránka demonštruujúca útok Clickjacking, kde sa v ifram-e načíta frontend našej bankovej aplikácie a používateľ sa kliknutím na tlačidlo *Claim!* odhlási.

3 Záver

Týmto projektom som naimplementoval plne funkčnú aplikáciu, ktorá demonštruuje rôzne spôsoby útokov a ochrany proti nim. Počas práce na nej som zistil, že väčšina moderných nástrojov, frameworkov a prehliadačov aktívne bojuje proti týmto útokom a pomáha tak vývojárom v dodávaní bezpečných aplikácií.

[5] https://fetch.spec.whatwg.org/

References

[1] *Cross Site Request Forgery (CSRF)* URL https://owasp.org/www-community/attacks/csrf.

[2] *Clickjacking* URL https://owasp.org/www-community/attacks/Clickjacking.

[3] *Cross Site Scripting (XSS)* URL https://owasp.org/www-community/attacks/xss/.

[4] Wikipedia contributors *Cross-site scripting — Wikipedia, The Free Encyclopedia* 2021 URL https://en.wikipedia.org/w/index.php?title=Cross-site_scripting&oldid=1019545290 [Online; accessed 29-April-2021].