UPB Zadanie 6

Injection

1. **Login as admin**

Na prihlásenie ako admin user, sme použili príkaz ‘ OR true –, kde ‘ je ukončenie reťazca email, čiže nájde buď prázdny email alebo niečo čo je vždy TRUE, a to nás prihlásilo ako prvého používateľa, v tomto prípade používateľa admina, ktorý bol ako prvý v databáze. Znaky – sú ako zakomentovanie vloženie hesla do query.

2. **Login as user Jim**

Našli sme si v recenziách produktu email používateľa Jim, a aplikovali sme rovnaký postup ako v úlohe 1.

XSS

1. **DOM XSS**

Vložili sme do search baru owasp, aby sme vyhladali owasp produkty. Všimili sme si, že keď do search boxu niečo vyhľadáme, tak sa nám to zobrazí v nejakom div bloku nad produktami. Následne skúsime vložiť <script> tagy s funkciou alert, ale to sa nevykonalo. Namiesto <script> tagu sme skúsili vložiť <iframe> tag, ktorý už nebol nijako obmedzený nejakým bezpečnostným mechanizmom webu. Po vložení iframe tagu sa funkcia vykonala.

2. **XSS**

Pridali sme do košíka nejaké produkty, ktoré sme následne zaplatili. V sekcii track order sme parameter id v URL vložili iframe tag, ktorý spustil skript. Takýto typ útoku dokáže napríklad získať cookies obete a podobné malichernosti.

Broken Access Control

1. **Admin Section**

Na základe inšpekcii stránky, sme vedeli usúdiť na akej technológii daná stránka beží. Z poznatkov týchto technológii sme skúšali rôzne podstránky napríklad do podstránky administrácie. Napríklad pre wordpress je dobrá konvencia podstránka /wp-admin, keďže sme si všimli, že táto stránka beží na Angulari, tak sme skúšali konvencie napr. pre /admin, čo nefungovalo, ale podstránka /administration už áno. Takto sme sa dostali na podstránku s citlivými údajmi.

2. **View Basket**

Ako prihlásený používateľ sme si pozreli svoj košík. V session storage sme si pozreli existujúce premenné, a skúšali sme ich prepísať. Keď sme prepísali bid na iné číslo, pri prepnutí sa na inú podstránku a naspäť sme mali zobrazený košík iného používateľa. Teraz by sme mohli manipulovať s košíkom iného používateľa.

Vulnerable Components

1. **Legacy Typosquatting**

Už z jedného z predošlích zadaní sa vieme dostať na citlivú podstránku ftp, kde vidíme rôzne súbory, medzi ne vidíme aj súbor package.json.bak, čo je backup súbor nejakého package.json súboru, ktorý pravdepodobne obsahuje zoznam závislostí potrebných pre tento web. Po otvorení tohoto súboru dostaneme Error: Only .md and .pdf files are allowed!
Na to, aby sme vedeli spôsobiť nejakú malichernosť, využijeme takzvaný NullByte Injection trick, s tým, že pridáme potrebnú príponu .md alebo .pdf. Pre podstránku ftp/package.json.bak teda konkrétne pridáme /%2500.md alebo .pdf, a týmto sme obišli validáciu vstupu. Server nám vráti súbor toho packagu v podobne .md alebo .pdf, a stiahnutý súbor zobrazíme v textovom editore, následne vidíme zoznam všetkých závislostí. Na webstránke npmjs.com si vieme prezrieť informácie ohľadom všetkých týchto knižníc a po chvíľke študovania narazíme, že knižnica epilogue.js typosquatting (preklep v url)
Cez customer feedback nahlásime danú chybu.

2. **Vulnerable Libraries**

Využijeme poznatky z predchádzajúcej úlohy, pozeráme na vulnerabilities daných knižníc. Keď dáme napríklad do googlu "grunt": "~1.0" vulnerability, tak nám vyskočí niekoľko linkov ohľadom zraniteľností tejto knižice pre nejaké verzie. Napríklad sme sa dohľadali, že pri verziach menších ako 1.3.0 nastáva riziko vysokej zraniteľnosti pre Arbitrary Code Execution, čiže je odporúčané použiť novšiu knižnicu. Keby prechádzame cez ďalšie knižnice, mohli by sme nájsť ďalšie zraniteľnosti.

Broken Authentification

1. **Password strength**

Z recenzí sme si zistili email admina, a skúšali sme cez “brute-force” metódu jeho heslo, uhádli sme ho na pár pokusov, keďže jeho heslo bolo veľmi jednoduché a primitívne (admin123)

2. **Bjoern’s Favorite Pet**

Našli sme si medzi recenziami mail Bjorna, nevedeli sme čo ďalej, skúsili sme hodiť jeho mail do služby google, kde sme našli používateľa Bjorn Kimmnich, jeho popisok na jeho twitter účte nám potvrdil, že to je tá istá osoba. Pri SQL injection, aby sme sa dostali do jeho konta, sme videli profilovku mačky, pri prehliadaní jeho twitter účtu sme videli, že jeho mačka

sa volá Zaya, pri resetovaní hesla sme využili túto informáciu a týmto spôsobom sme nastavili nové heslo k tomuto účtu.

Sensitive Data Exposure

1. **Confidential document**

Na stránke About us, sme si všimli, že tam je odkaz na súbor na podstránke ftp/legal.md. Skúsili sme odstrániť legal.md z linku, a dostali sme sa na podstránku ftp, kde sme videli tajné dokumenty, ako napríklad acquisitions.md

2. **Exposed metrics**

V hinte sme zistili, že stránka používa prometeus metrics, tak sme rozklikli podstránku /metrics, a takto sme sa dostali na podstránku metriky

Improper input validation

1. **Missing encoding**

V podstránke Photo Wall sa nezobrazovala jedna fotka, po inspect element sme videli, že sa tam používa znak #, ktorý sme prepísali na správne enkódovanie a to %23, aby vedela stránka prečítať správnu cestu k súboru.

2. **Repetitive registration**

Pri registrácii používateľa sme si nastavili rovnaké heslo, prvé heslo sme zmenili a už nedalo na výber, že nie sú rovnaké a zaregistrovali sme sa na to upravené heslo, ktoré nebolo rovnaké ako to na overenie. Týmto spôsobom sme sa vedeli znovu zaregistrovať užívateľa na rovnaký mail, ktorý už bol v systéme, a takto sme sa vedeli dostať na jeho konto.