4.1.1 Reflected XSS

Az admin felhasználóként bejelentkezve megnézhetjük rendeléseink állapotát, amely a következő formátumú linken elérhető: **http://localhost:3000/#/track-result?id={id}**. Ebből az ID paraméter közvetlenül megjelenítésre kerül, ellenőrzés nélkül. Mivel általában ott szöveg van, ez nem okoz problémát, viszont ha értelmes HTML adatot adunk értékként, támadást indíthatunk. Az <iframe> tag például képes általunk megadott kódot végrehajtani. Egy egyszerű támadás, amellyel csak a konzolra iratnánk ki, a következőképpen nézne ki:

http://localhost:3000/#/track-result?id=<iframe src=”javascript:console.log(‘xss’)”>

Viszont ebből bizonyos karaktereket a browser számára értelmezhetővé kell tenni, például **‘** helyett %60, space helyett pedig %20. A végső link tehát lehet például:

http://localhost:3000/#/track-result?id=%3Ciframe%20src%3D%22javascript:console.log(%60xss%60)%22%3E

4.1.2 DOM XSS

A feladat nagyon hasonlít az előzőhöz, csak annyi a különbség, hogy itt nem közvetlenül lesz beszúrva a felhasználó által megadott input, hanem egy másik komponens innerHTML attribúmaként kerül megjelenítésre, ezt a következő sorban láthatjuk:

<span id="searchValue" [innerHTML]="searchValue">

Így tehát elég rákeresnünk a **<iframe src="javascript:console.log('xss')">** kifejezésre, és kódunk végre is hajtódik.

4.1.3 Persisted XSS

A regisztrációs formnál beírt emailek megjelennek a **/#/administration** oldalon, tehát ha sikerül az **<iframe src=”javascript:console.log(‘xss’)”>** emaillel regisztrálni, a kód végrehajtódik az oldal megnyitásakor. Ehhez mindössze annyit kell tenni, hogy kitöltjük a regisztrációs formot, email helyett a fenti tag-et írjuk be. Ezt önmagában nem fogja engedni a rendszer, mivel nem valid az email, de a beküldő gombot aktívvá tehetjük (kivesszük a „disabled” attribútumot), és így már elfogadja az új felhasználót (szerveroldalon nincs leellenőrizve). Ha ez megtörtént, akkor adminisztrátor fiókkal belépve navigálhatunk az administration oldalra, és meg is látjuk az eredményt: a kódrészlet végrehajtódik.

4.2.1 Admin login

A következő sor lehetőséget nyújt SQL injection típusú támadásra: **query(`SELECT * FROM Users WHERE email = '${req.body.email || ''}' AND password = '${security.hash(req.body.password || '')}' AND deletedAt IS NULL`, { model: UserModel, plain: true })**

Mivel nincs levédve az input (például prepared statementekkel), kikerülhetjük a jelszó megadását, ha az email mező végére egy idézőjelet két kötőjelet teszünk '**--**. Ezzel lezárjuk a stringet, és a query hátralevő részét komentté tesszük. Innen csak annyi dolgunk van, hogy megtaláljuk az admin email címét. Az **About Us** oldalon láthatjuk, hogy az oldal domain-je **juice-sh.op**, szóval kipróbálhatjuk az admin@juice-sh.op címet. A bemenetünk tehát:

email: admin@juice-sh.op' --
password: akármi

4.2.2 Server sleep

Látjuk a **server.ts** file-ban (vagy ha ehhez nincs hozzáférésünk, megnézhetjük a böngészőnkben is), hogy a következő route szolgálja egy bizonyos termékhez tartozó review-kat: **app.get('/rest/products/:id/reviews', showProductReviews())**

Ebben a címben az ID egy módosítható paraméter, amit a handler a következő helyen használ fel: **db.reviewsCollection.find({ $where: 'this.product == ' + id })**. Ez összehasonlítja a termék azonosítóját a megadott ID-val, viszont az id lehet akármi, felhasználó által megadva. Ha valós ID helyett a **sleep(5000)** függvényhívást írjuk, a következő lesz az összehasonlítás: **this.product == sleep(5000).** Ez valószínűleg nem fog találatot visszatéríteni, viszont az összehasonlítás elvégzéséhez végrehajtja a várakozó függvényt. Tehát csak annyit kell tennünk, hogy valamely request küldő alkalmazásból (Postman, Burp Suite, stb) kérést küldünk a **localhost:3000/rest/products/sleep(5000)/reviews** címre. Sikerrel is járunk, a szerver nem válaszol egyéb kérésekre a megadott ideig.

4.2.3 Reset Jim's password

A korábban bemutatott SQL injection módszerrel bejelentkezhetünk Jim fiókjába, és a lementett lakcímek között megtaláljuk a **Room 3F 121, Deck 5, USS Enterprise, 1701** címet, amely elárulja, hogy **James T. Kirk** fiktív szereplőről beszélünk. Ezután kijelentkezünk a fiókjából, és a **Forgot Password** funkcionalitásra irányítjuk figyelmünk. Itt beírjuk Jim email címét, majd meg kell válaszolnunk egy biztonsági kérdést, hogy jelszót tudjunk változtatni. A kérdés Jim legidősebb testvérének középső nevét kéri, amely könnyen

megtalálható az interneten: **Samuel.** Miután ezt beírtuk, tetszőleges új jelszót állíthatunk Jim számára.