SQL injection tehtävät:

1. **SQL injection vulnerability in WHERE clause allowing retrieval of hidden data:**

Tehtävä oli mielenkiintoinen koska siinä suoraan syöttämällä URL linkin jatkoksi '+OR+1=1—koodin sai tietokannassa kyselyn SELECT * FROM products WHERE category = 'Gifts' AND released = 1 aikaiseksi. Tämä paljasti käyttäjälle piilotettua sisältöä palvelimelta. Opin kuinka SQL injektioita tehdään periaatteessa ja kuinka niitä voi tehdä myös URL kenttään syöttämällä. Olin joskus aiemmin törmännyt credentials kenttään syötettyihin arvoihin SQL injektioissa.

Tehtävä oli haastava ilman apua, koska minulla ei ollut aiempaa kokemusta SQL injektioiden tekemisestä. Käytin ohjetta avuksi, mutta en tarvinnut ohjevideota avuksi.

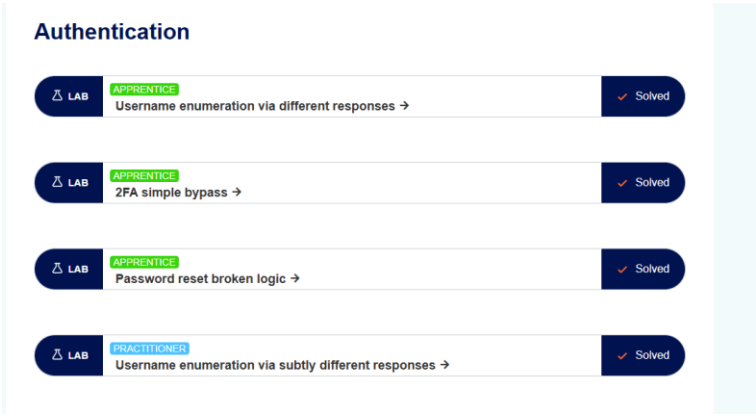
2. **SQL injection vulnerability allowing login bypass:**

Tehtävässä opin Kuinka tietyn komennon syöttämällä login credentialseihin on mahdollista saada tietokannassa kysely, joka ohittaa salasanan varmistuksen. Kun tehtävässä syötti ” ’ ” merkin niin se saa aikaiseksi selaimella SQL kyselyn. Kun antoi usernameksi ”administratorin” niin sai admin oikeudet, joilla voi esim. poistaa käyttäjiä. Kun laittoi ”--” kyselyn loppuun niin se ohittaa salasanan tarkistuksen. Opin siis tehtävässä paljon tuon haavoittuvuuden moniuloitteisuudesta ja logiikasta.

Tehtävä sujui toisena SQL injektiona ihan hyvin ja turvauduin vain ohjeen apuun. Minun ei tarvinnut katsoa apuvideoita.

3. **SQL injection attack, querying the database type and version on Oracle**

Opin kuinka eri vaiheiden kyselyillä voidaan tiedustella tietokannan tyyppiä → Oracle database.



Passwd based login tehtävät:

4. **Username enumeration via different responses:**

Opin käyttämään paljon kurssin työkaluja kuten Burpia. Tämä tehtävä oli ensimmäinen ”hakkerointi” minulle, joten se oli hyvin mielenkiintoista. Termi brute force on minulle hyvin tuttu, mutta nyt oli hauskaa päästä käyttämään ihan työkalua sitä varten. Opin tehtävässä tosiaan Burp käyttöä, proxyn käyttöä, http pyyntöjen tarkastelua Burpin avulla vs. devtools ja sniper attackin toteuttamisen Burpin avulla. Myös periaate ensin username ja sitten salasana selkeni tehtävässä hyvin eri pyyntöjen koodien takia. Tehtävässä tärkeintä oli huomata kuinka oikean usernamen sattuessa kohdalle, virheilmoitus muuttui merkittävästi.

Tehtävässä haastavinta oli siihen pystyminen täysin itse, vaan seurasin opettajan luentotallennetta, kun tein kyseisen tehtävän alusta loppuun.

5. **Username enumeration via subtly different responses**

Opin taas lisää burpin käyttöä ja brute force tekniikkaa. Tehtävässä oli hienoa huomata, kuinka oikean usernamen sattuessa virheilmoituksessa oli vain hyvin pieni virhe ”Invalid username or password.” vs. ”Invalid username or password”, jossa pisteen puuttuminen paljasti usernamen olevan oikea. Pyynnöissä ei ollut muuten eroa niiden numeroinnissa.

Tehtävä oli haastava tehdä, koska minulla ei ole aiempaa kokemusta penetraatiotestauksesta. Jouduin turvautumaan apuvideoihin tehtävää tehdessäni. Pidin kovasti toisesta hakkeroinnistani.

6. **2FA simple bypass**

Opin Kuinka 2FA pystyy kiertämään joissain tapauksissa intruderin avulla, jolloin pääsee viesteihin käsiksi väliin ja voi muokata URL (/login → /my-account) ja jättää 2FA pyynnön sen jälkeen käyttämättä.

7. **Password reset broken logic**

Intrudasin password vaihto viestin, jonka jälkeen muokkasin siitä, tokenit ja usernamen wieneristä → carlokseksi, jonka jälkeen lähetin sen serverille ja pääsin sen jälkeen kirjautumaan niillä tokeneilla sivulle.

Access contol vulnerabilities tehtävät:

8. **User role can be modified in user profile:**

Opin kuinka käyttäjän roolia voi muuttaa, kun on lähettänyt serverille post pyynnön sähköpostiosoitteen tallentamisesta. Samanlaisella, mutta uudella post pyynnöllä kykeni muuttamaan Wiener:peter käyttäjän roleid 1 to roleid 2 =adminiksi ja kykenin poistamaan Carloksen.

Haastavinta oli keksiä miten se email pyynnön avulla saa ujutettua myös roolin muutoksen. Tarvitsin ohjevideon apua. Myös jouduin käynnistämään kaikki prosessini ja tietokoneeni uudestaan, koska en

kyennyt kirjautumaan enää wiener:peter tunnuksilla, vaan hakkasin niissä päätäni seinään pitkän aikaa.

9. **Unprotected admin functionality:**

Opin että, kirjoittamalla domain kenttään tiettyjä parametereja, kuten admin-panel, on mahdollista päästä kirjautumatta muuttamaan käyttäjätietoja palvelusta. Myös robots.txt tekstitiedoston sisältöä pääsi tarkastelemaan kirjautumatta palveluun. Sieltä näki, että polku admin paneeliin loppuu:

User-agent: *

Disallow: /administrator-panel.

Tehtävä ei ollut kovin haastava vaan onnistuin tekemään sen ilman apuvideota. Tehtävässä ei tarvinnut tarkastella ollenkaan koodia.

10. **Unprotected admin functionality with unpredictable URL**

Opin Kuinka DevToolsista saattaa löytyä admin tietoja, joilla voi kirjautua ja poistaa käyttäjiä.

11. **Lab: User role controlled by request parameter**

Opin Kuinka Admin roolin sai käyttöön syöttämällä Cookie tietoon Admin false → Admin trueksi

12. **User ID controlled by request parameter**

Sain API avaimen haltuuni vain vaihtamalla ID wieneristä carlokseksi ja lähettämällä sellaisen muokatun viestin uudelleen serverille.

13. **User ID controlled by request parameter, with unpredictable user IDs**

Sain Carlosin API avaimen etsimällä oikean XML viestin, jossa oli wienerin tekemä pyyntö ja etsimällä oikean postauksen carlosilta ja lähettämällä sen palvelimelle, jolloin sain carlosin host id:n jonka liittämällä wienerin XML viestiin sain hänen API avaimen.

14. **User ID controlled by request parameter with data leakage in redirect**

Sain suoraan carloksen API avaimen XML dokumentissa, jonka sain kirjautumalla wieneriksi ja vaihtamalla URL sinne carlos ja lataamalla sivun uudestaan. Sivu ohjasi minut uudelleen kirjautumiseen, mutta paljasti API keyn.

15. **Lab: User ID controlled by request parameter with password disclosure**

Sain Administrator salasanan vaihtamalla XML viestiin wiener tilalle administrator ja lähettämällä viestin serverille. Kirjauduin tunnuksilla ja poistin carlos käyttäjän admin paneelissa.

16. Insecure direct object references

Sain vanhan keskustelun käsiini muokkaamalla payloadia. Minun keskusteluni ohjattiin 2.txt tiedostoon, joten vaihdoin siihen payloadiin 1.txt ja sain keskustelusta käsiini carlosin salasanan.

XSS tehtävät:

17. Reflected XSS into HTML context with nothing encoded

Opin Kuinka URL hakukenttään syöttämällä <script>alert()</script> saa aikaan virheilmoituksen.

18. Stored XSS into HTML context with nothing encoded

Opin Kuinka blogipostauksen kommenttiin voi syöttää javascriptiä esim. <script>alert()</script>.

19. Lab: DOM XSS in document.write sink using source location.search

Opin Kuinka payloadilla ensin saadaan javascript koodi näyttämään haluttu sana hakutuloksissa esim. ”moi”, jonka jälkeen sinne syötetään moi” onload=”alert(), joka aiheuttaa alert ilmoituksen.

20. DOM XSS in innerHTML sink using source location.search

Opin innerHTML voidaan huijata toteuttamaan javascript käskyjä esim. seuraavalla tavalla

<img src=”0” onerror=”alert()”>