Assignment 4 Report

Ο/Ε: Μιχαήλ Κρατημένος, ΑΜ: 2018030104

Γεώργιος Πιπεράκης, ΑΜ: 2018030012

Task 1:

Παρατηρήθηκε στη γραμμή 39 του app.py αρχείου ότι το user input εισέρχεται απευθείας στο SQL query χωρίς κανέναν έλεγχο. Οπότε αντί για τον κωδικό της εκφώνησης (“user”), βάλαμε το: “password' OR '7'='7”, το οποίο βγάζει πάντα αληθή password, αφού πάντα ισχύει ότι 7=7 και κάναμε έτσι bypass το login του user (εδώ ουσιαστικά κάναμε sql injection με τον «κωδικό» που βάλαμε).

Task 2:

Παρατηρήθηκε στο αρχείο greet.js ότι το uname γράφεται στο DOM χωρίς κάποιο έλεγχο. Για να εκθέσουμε αυτήν την αδυναμία, επισκεφτήκαμε την εξής διεύθυνση: http://139.91.71.5:11337/dashboard#<script>alert('DOM-XSS')</script>. Όταν επισκεπτόμαστε αυτήν τη διεύθυνση, το <script> εισέρχεται στο DOM και εκτελείται από τον browser, προκαλώντας την ειδοποίηση: “DOM-XSS” στην οθόνη.

Task 3:

Παρατηρήθηκε στη γραμμή 44 του dashboard.html (<h3>No item with name: {{ noitem|safe }}</h3>) ότι η τιμή του noitem εισέρχεται απευθείας στο HTML χωρίς έλεγχο, αφού και το safe φίλτρο δεν αφήνει να αγνοηθεί το input, κάτι το οποίο είναι επικίνδυνο αν το noitem περιέχει κακόβουλο κώδικα. Οπότε για να εκθέσουμε αυτήν την αδυναμία, στο /search βάλαμε σαν είσοδο: “<script>alert("XSS")</script>”, το οποίο επιστρέφει μια ειδοποίηση στην οθόνη: “XSS”, αντί να ψάξει για κάποιο item.

Task 4:

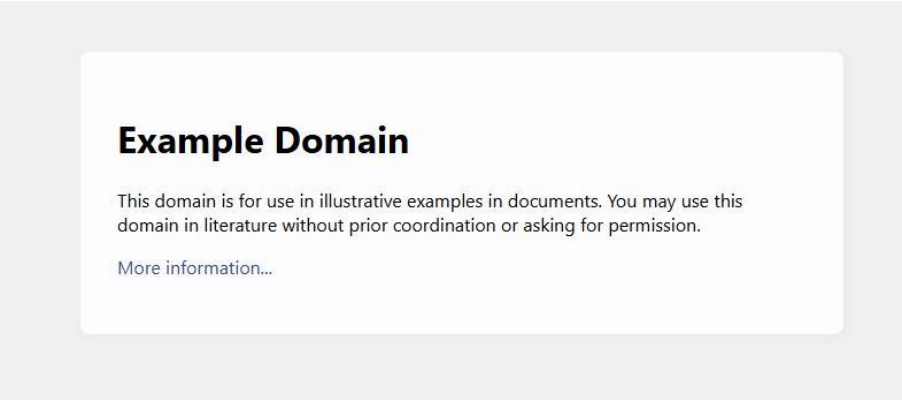
Παρατηρήθηκε στη γραμμή 80 του app.py αρχείου
(query = f"SELECT name,category,price FROM items WHERE name = '{name}'") η παράμετρος “name” χρησιμοποιείται απευθείας στο SQL query. Οπότε για να εκθέσουμε αυτήν την αδυναμία, στο /search βάλαμε σαν είσοδο:
“' UNION SELECT username, password, -1 FROM users--", το οποίο μας επιστρέφει το username, password του administrator, το οποίο είναι:
“$youCantCrackMyPassword$” (και εδώ ουσιαστικά κάναμε sql injection με τον «κωδικό» που βάλαμε).

Task 5:

Για να συνδεθούμε σαν administrator πήγαμε στη διεύθυνση:
http://139.91.71.5:11337/admin και βάλαμε τον κωδικό από το προηγούμενο ερώτημα.

Task 6:

Παρατηρήθηκε ότι στην goto() συνάρτηση στη γραμμή 131 του app.py, η μεταβλητή “to” περνιέται στην επόμενη συνάρτηση χωρίς κανέναν έλεγχο. Οπότε οτιδήποτε URL υπάρχει στην “to” παράμετρο χρησιμοποιείται ως redirection target. Για να εκθέσουμε αυτήν την αδυναμία, επισκεφτήκαμε το url: http://139.91.71.5:11337/go?to=http://example.com, το οποίο μας έστειλε σε αυτό το παράθυρο:



Task 7:

Παρατηρήθηκε ότι στη γραμμή 111 του app.py το “filename” μπαίνει στο file path χωρίς κανέναν έλεγχο, με αποτέλεσμα μπορούμε να δούμε το περιεχόμενο των αρχείων στο app. Για να εκθέσουμε αυτήν την αδυναμία αρχικά επισκεφτήκαμε τη διεύθυνση: http://139.91.71.5:11337/files για να δούμε ποιο αρχείο να ψάξουμε για να πάρουμε το flag. Το αρχείο αυτό είναι το realflag.txt, οπότε για να δούμε το flag πήγαμε στη διεύθυνση: http://139.91.71.5:11337/admin?show=/files/realflag, όπου το real flag είναι το:
TUC{972f02eb8227012f0b9954e95efc4001a28290ef48047a922efc2a4db40954e6}