Εργασία Ασφάλεια Δικτύων SQL Injection

Συντελεστής:Καρλής Κωνσταντίνος(3190077)

Η εργασία έχει υλοποιηθεί με την βάση δεδομένων MySQL και με χρήση της γλώσσας PHP.Ωστόσο,κατα την διαδικασία ενημέρωσης του σερβερ, ενώ τοπικά δουλεύουν όλα μια χαρά με την χρήση της του XAMPP όταν το ανέβασα στο σέρβερ υπάρχει πρόβλημα στην εκτέλεση του PHP κομμάτι.Με μία έρευνα που έκανα στο διαδίκτυο,θεωρώ πως το πρόβλημα βρίσκεται στις εκδόσεις του PHP καθώς έχω ενημερώσει και το configuration file ώστε να αναγνωριζει και να εκτελεί τέτοιου είδους αρχεία.Ωστόσο λύση για αυτό δεν μπόρεσα να βρώ.

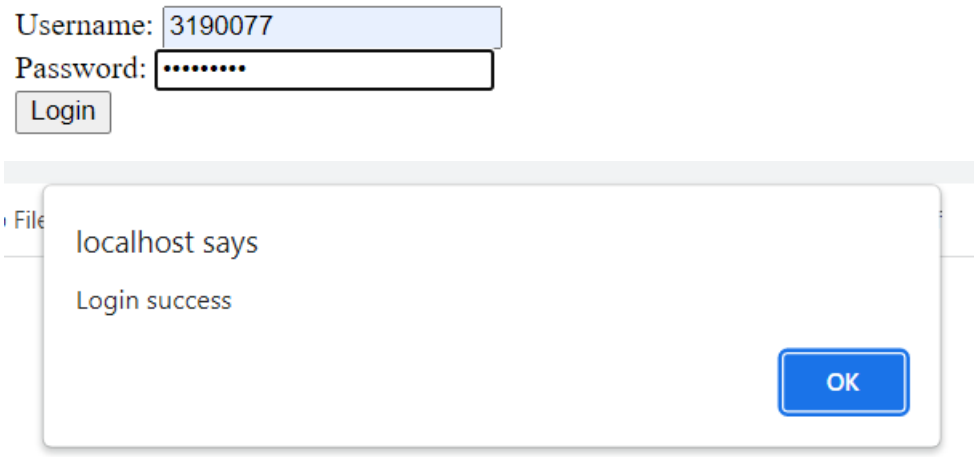
a) Η υλοποίηση σε αυτό το ερώτημα βρίσκεται στο αρχείο “Create table and insert.sql”
b) Οι ενδεδειγμένες λύσεις είναι η εξής:
   i) Κατακερματισμός των κωδικών πρόσβασης: Χρήση ενός ασφαλούς αλγόριθμου κατακερματισμού μονής κατεύθυνσης, όπως το bcrypt ή το Argon2, για να κατακερματίστουν οι κωδικοί πρόσβασης. Αυτό διασφαλίζει ότι ακόμη και αν κλαπούν οι κατακερματισμοί του κωδικού πρόσβασης, δεν μπορούν εύκολα να σπάσουν.
   ii) Χρήση salt: Το salt είναι μια τυχαία συμβολοσειρά χαρακτήρων που προστίθεται στον κωδικό πρόσβασης πριν κατακερματιστεί. Αυτό καθιστά πολύ πιο δύσκολο το σπάσιμο των κατακερματισμών του κωδικού πρόσβασης, ακόμη κι αν ο εισβολέας γνωρίζει τον αλγόριθμο κατακερματισμού που χρησιμοποιείται.
   iii) Χρησιμοποιήστε ένα μοναδικό salt ανά κωδικό πρόσβασης: Η χρήση ενός μοναδικού αλατιού ανά κωδικό πρόσβασης αυξάνει περαιτέρω την ασφάλεια.
   iv) Χρήση συνάρτησης παραγωγής κλειδιού (KDF): Χρησιμοποιήστε μια συνάρτηση παραγωγής κλειδιού (KDF) όπως bcrypt ή scrypt για να εξαγάγετε ένα κλειδί από τον κωδικό πρόσβασης και το salt, το οποίο μπορεί στη συνέχεια να χρησιμοποιηθεί για την κρυπτογράφηση των δεδομένων του χρήστη.
   v) Αποθήκευση του κατακερματισμού του κωδικού πρόσβασης και το salt ξεχωριστά: Αποθηκεύστε τον κατακερματισμό κωδικού

πρόσβασης και το salt σε ξεχωριστά πεδία στη βάση δεδομένων, για να αποτρέψετε έναν εισβολέα από το να μπορεί να χρησιμοποιήσει το αλάτι για να σπάσει τους κατακερματισμούς κωδικού πρόσβασης.

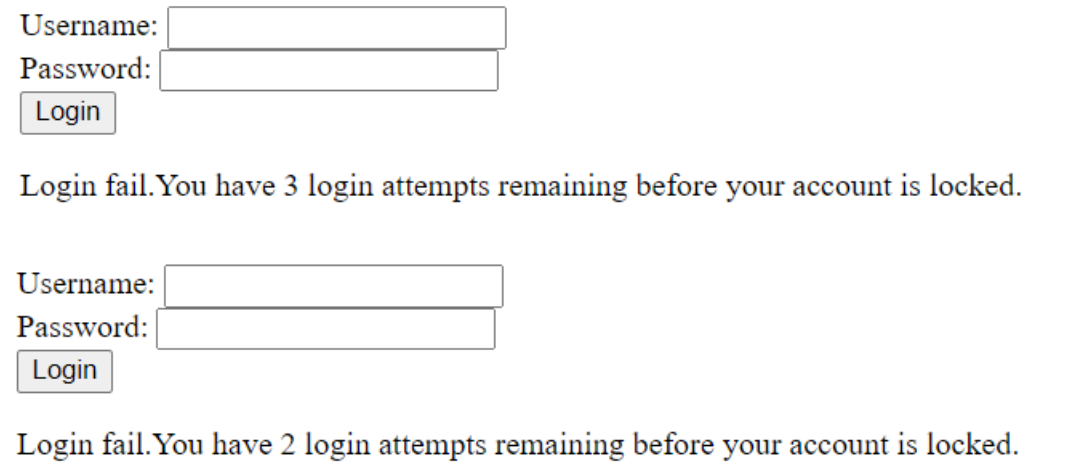
vi) Χρήση ασφαλούς σύνδεσης: Χρήση μια ασφαλή σύνδεση (HTTPS) για να κρυπτογραφηθούν τα δεδομένα κατά τη μεταφορά μεταξύ της συσκευής του χρήστη και του διακομιστή για να αποτραπεί την υποκλοπή.

Στην δική μου υλοποίηση έχω χρησιμοποιήσει το (i) και βρίσκεται στο αρχείο Hashing.sql

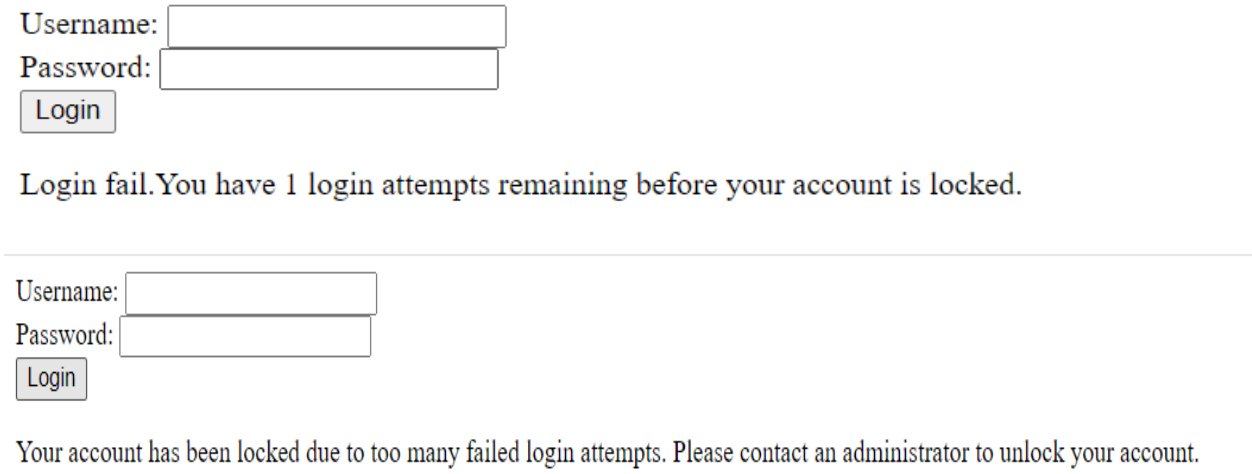
c) Σε αυτό το ερώτημα έχει γίνει η χρήση της γλώσσας PHP.Ο κώδικας βρίσκεται στο αρχείο login.php.Η αποφυγή των sql injections γίνεται μέσα των Connections pools Που έχουν τοποθετηθεί μέσα στο αρχείο. Τα Connection pools μπορούν να βοηθήσουν στην αποτροπή SQL Injection απολυμαίνοντας σωστά την είσοδο του χρήστη πριν χρησιμοποιηθεί σε ένα ερώτημα SQL. Αυτό μπορεί να γίνει χρησιμοποιώντας προετοιμασμένες δηλώσεις ή παραμετροποιημένα ερωτήματα, τα οποία διαφεύγουν αυτόματα από τυχόν επικίνδυνους χαρακτήρες στην είσοδο του χρήστη. Επιπλέον, τα Connection pools μπορούν επίσης να επιβάλουν βέλτιστες πρακτικές ασφάλειας, όπως ο περιορισμός του αριθμού των συνδέσεων που μπορούν να πραγματοποιηθούν και η εφαρμογή κατάλληλων ελέγχων πρόσβασης για τη βάση δεδομένων.
d) Οι αλλαγές που έχουν γίνει για το τελευταίο ερώτημα βρίσκονται στο login_attempt_lock_last_password.sql και στο login.php.Θα ακολουθήσουν αναλυτικά παραδείγματα με screenshots για την υλοποίηση.
   i) Εισαγωγή σώστου κωδικού για τον χρήστη 3190077.



ii) Εισαγωγή λανθασμένου κωδικού για τον χρήστη 3190077 δύο συνεχόμενες φορές.

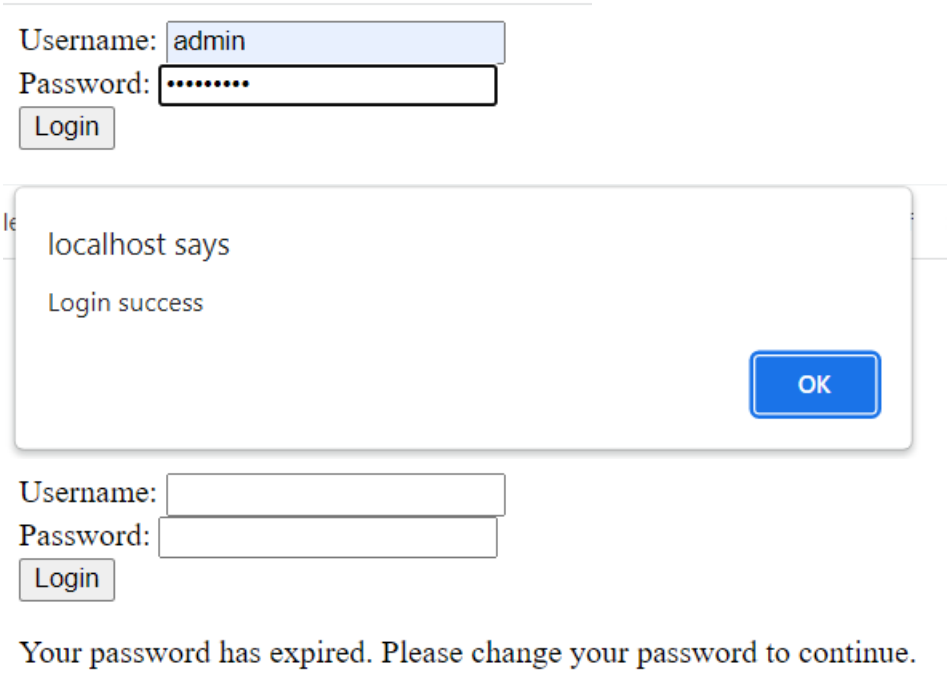


iii) Εισαγωγη λανθασμένου κωδικού πρόσβασης για τον χρήστη 3190077 άλλες 2 φορές ώστε ο χρήστης να κληδωθεί.



iv) Εισαγωγη σωστού κωδικου για τον χρήστη admin.

v) Για το παράδειγμα ορίζω στο Php file ως 1 ημέρα το όριο για να έχεις έναν κωδικό πρόσβασης προκείμενου να δούμε πως λειτουργεί.Στο παραδοτέο είναι 90 μέρες.



Παραθέτω και τα αποτελέσματα απο τον πίνακα Loggings της βάσης δεδομένων.

id	username	login_time	success
1	3190077	2023-01-23 10:03:39	1
2	3190077	2023-01-23 10:04:58	0
3	3190077	2023-01-23 10:06:13	0
4	3190077	2023-01-23 10:07:50	0
5	admin	2023-01-23 10:09:52	1
6	admin	2023-01-23 10:13:17	1
NULL	NULL	NULL	NULL