Ασφάλεια Δικτύων - Άσκηση 4

Αργυρόπουλος Χρήστος - 3170010
Τσολάκου Διογένης - 3170164

Δημιουργούμε τον πίνακα με τους χρήστες, στον οποίο αποθηκεύουμε το όνομα, τον κωδικό, την περιγραφή και την τελευταία φορά που άλλαξε ο κωδικός του κάθε χρήστη.

```
create table users (
    username varchar(50) primary key,
    password varchar(128),
    description varchar(120),
    password_last_changed datetime
);
```

Δημιουργούμε τα δύο παρακάτω triggers ώστε κατα την δημιουργία των κωδικών, καθώς και στην αλλαγή τους, να αποθηκεύονται hasharισμένοι μαζί με salt. Το salt επιλέγεται τυχαία και αποθηκεύονται μαζί με τον κωδικό οι 12 πρώτοι χαρακτήρες του. Ο τρόπος αποθήκευσης του password είναι ο εξής:
sha256(sha256(password) + salt) + ‘:’ + salt

```
delimiter $$
create trigger hash_password_insert
before insert on users
for each row begin
    set @salt = substring(sha2(rand(), 256), 1, 12);
    set new.password = concat(sha2(concat(sha2(new.password, 256), @salt), 256), concat(':', @salt));
    set new.password_last_changed = now();
end $$
delimiter ;

delimiter $$
create trigger hash_password_update
before update on users
for each row begin
    if old.password <> new.password then
        set @salt = substring(sha2(rand(), 256), 1, 12);
        set new.password = concat(sha2(concat(sha2(new.password, 256), @salt), 256), concat(':', @salt));
        set new.password_last_changed = now();
    end if;
end $$
delimiter ;
```

Εισάγουμε τους δύο ζητούμενους χρήστες.

```
insert into users (username, password, description) values ("p3170010-p3170164", "team28", "database owners");
insert into users (username, password, description) values ("admin", "admin", "examiners");
```

Έπειτα, τρέχουμε μερικά select και updates ώστε να δείξουμε τις αλλαγές των κωδικών πρόσβασης.

```
mysql> select * from users;
+-------------------+------------------------------------------------------------------+----------------+----------------------+
| username          | password                                                         | description    | password_last_changed |
+-------------------+------------------------------------------------------------------+----------------+----------------------+
| admin             | ff092f0452f4f85e4855ca154340094bccd4a3e6c1e2072c9285746a3653afd2:50aba14c118a | examiners      | 2021-06-12 14:44:38   |
| p3170010-p3170164 | 18a9cf9c06bb3e6d5f0adc583657a1b57165154ed2062fcccabe164f025211df:a9b2d3431748 | database owners | 2021-06-12 14:27:34   |
+-------------------+------------------------------------------------------------------+----------------+----------------------+
2 rows in set (0.00 sec)
```

```
mysql> update users set password = "admin2" where username = "admin";
Query OK, 1 row affected (0.02 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from users;
+-------------------+------------------------------------------------------------------+----------------+----------------------+
| username          | password                                                         | description    | password_last_changed |
+-------------------+------------------------------------------------------------------+----------------+----------------------+
| admin             | dbf1f3689131f5ad8c7c0f528fab7ded216b5b82ea0eb5077778f3367db09e5f:de5b8403d9b0 | examiners      | 2021-06-13 13:44:13   |
| p3170010-p3170164 | 18a9cf9c06bb3e6d5f0adc583657a1b57165154ed2062fcccabe164f025211df:a9b2d3431748 | database owners | 2021-06-12 14:27:34   |
+-------------------+------------------------------------------------------------------+----------------+----------------------+
```

```
mysql> update users set password = "admin" where username = "admin";
Query OK, 1 row affected (0.05 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from users;
+-------------------+------------------------------------------------------------------+----------------+----------------------+
| username          | password                                                         | description    | password_last_changed |
+-------------------+------------------------------------------------------------------+----------------+----------------------+
| admin             | 70d0f20d718ad4d0125fc64444600c696be4056d5055d9814c0060ded52f1c7b:4581d5c5b682 | examiners      | 2021-06-13 13:45:20   |
| p3170010-p3170164 | 18a9cf9c06bb3e6d5f0adc583657a1b57165154ed2062fcccabe164f025211df:a9b2d3431748 | database owners | 2021-06-12 14:27:34   |
+-------------------+------------------------------------------------------------------+----------------+----------------------+
```

Παρατηρούμε πως μετά τις αλλαγές κωδικών αλλάζουν τα passwords και η ημερομηνία τελευταίας αλλαγής. (Επισημαίνεται πως παρόλο που στην αρχή και στο τέλος ο κωδικός πρόσβασης του admin ήταν “admin” και τις δύο φορές, το αποθηκευμένο password είναι διαφορετικό λόγω του τυχαία επιλεγμένου salt την εκάστοτε φορά).

Δημιουργούμε τον πίνακα logging που αποθηκεύει το όνομα χρήστη, τη στιγμή που έκανε απόπειρα σύνδεσης και το αποτέλεσμα της προσπάθειάς του.

```
create table logging (
    username varchar(50),
    time datetime,
    attempt boolean
);
```

Στη συνέχεια δημιουργούμε την διαδικασία login, η οποία δέχεται ως όρισμα το όνομα και τον κωδικό που έδωσε ο χρήστης κι επιστρέφει το αποτέλεσμα της σύνδεσης.
Αρχικά, η μέθοδος βρίσκει πόσες φορές ο χρήστης έχει αποτύχει να συνδεθεί. Στην περίπτωση που αυτός ο αριθμός είναι μεγαλύτερος ή ίσος του 3, η διαδικασία επιστρέφει -1, δηλαδή απαγορεύει την σύνδεση ανεξαρτήτως αν τα στοιχεία που έδωσε ο χρήστης είναι ορθά ή οχι.
Σε αντίθετη περίπτωση, υπολογίζεται το hash του password που εισήγαγε σε συνδυασμό με το αποθηκευμένο salt σύμφωνα με τον προαναφερθέντα τρόπο [sha256(sha256(password) + salt)] και ελέγχεται αν το αποτέλεσμα ισοδυναμεί με τους πρώτους 64 χαρακτήρες του αποθηκευμένου password, δηλαδή το αποτέλεσμα της ίδιας “συνάρτησης” κατά την δημιουργία του ορθού κωδικού. Αν ο χρήστης έχει δώσει τον σωστό κωδικό η συνάρτηση επιστρέφει 1, δηλαδή επιτυχία, αλλιώς επιστρέφει 0, δηλαδή αποτυχία. Σε κάθε περίπτωση, εισάγεται στον πίνακα logging το όνομα του χρήστη, ο χρόνος απόπειρας, και το αποτέλεσμα αυτής.

```
delimiter $$
create procedure login (in un varchar(50), in pw varchar(128), out success smallint)
begin
    select count(*) into @failed_attempts
    from logging
    where username = un and attempt = false;

    if @failed_attempts < 3 then
        select password into @pw_row
        from users
        where username = un;

        set @salt = substring(@pw_row, 66, 12);
        set @hashed_pw = sha2(concat(sha2(pw, 256), @salt), 256);
        set @hashed_password = substring(@pw_row, 1, 64);

        if @hashed_pw = @hashed_password then
            set success = 1;
            insert into logging values (un, now(), true);
        else
            set success = 0;
            insert into logging values (un, now(), false);
        end if;
    else
        set success = -1;
    end if;
    select success;
end $$
delimiter ;
```

Παρακάτω κάνουμε μερικές απόπειρες σύνδεσης με διαφορετικά στοιχεία, όπως φαίνεται στην πρώτη εικόνα. Παρατηρούμε πως στις πρώτες περιπτώσεις, τα ορθά στοιχεία επιστρέφουν θετικό αποτέλεσμα σύνδεσης, ενώ σε αντίθετη περίπτωση επιστρέφεται 0.

```
call login ("admin", "admin", @result);                /* success = 1 */
call login ("admin", "asdfghjkl", @result);            /* success = 0 */
call login ("p3170010-p3170164", "team28", @result);   /* success = 1 */
call login ("admin", "asdfghjkl", @result);            /* success = 0 */
call login ("admin", "asdfghjkl", @result);            /* success = 0 */
call login ("admin", "admin", @result);                /* success = -1 (disallowed although correct) */
```

```
mysql> select * from logging;
+-------------------+---------------------+---------+
| username          | time                | attempt |
+-------------------+---------------------+---------+
| admin             | 2021-06-13 14:01:43 |       1 |
| admin             | 2021-06-13 14:01:43 |       0 |
| p3170010-p3170164 | 2021-06-13 14:01:43 |       1 |
| admin             | 2021-06-13 14:01:43 |       0 |
| admin             | 2021-06-13 14:01:43 |       0 |
+-------------------+---------------------+---------+
```

Μετά τις 3 αποτυχημένες προσπάθειες σύνδεσης του χρήστη admin, η κλήση της μεθόδου με σωστά στοιχεία επιστρέφει -1, επειδή ο χρήστης έχει αποκλειστεί από τη σύνδεση.

```
mysql> call login ("admin", "admin", @result);
+---------+
| success |
+---------+
|      -1 |
+---------+
```

Ο παρακάτω κώδικας σε html εμφανίζεται κατά την εισαγωγή των στοιχείων του χρήστη για σύνδεση.

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="utf-8">
        <title> NetSec Team28 </title>
        <script>
            var username = document.getElementById('username').value
            var password = document.getElementById('password').value
            // sendToServer(username, password)
        </script>
    </head>

    <body style="font-family: sans-serif">
        <form>
            <label for="username"> username: </label><br>
            <input type="text" id="username" name="username" required>
            <br>
            <label for="password"> password: </label><br>
            <input type="password" id="password" name="password" required><br><br>
            <button type="submit" id="login"> Log In </button>
        </form>
    </body>
</html>
```

Στην django, προσθέτουμε τις παρακάτω γραμμές στο αρχείο settings.py ώστε να είναι δυνατή η σύνδεση στη βάση.

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql',
        'NAME': 'GDPR',
        'USER': 'root',
        'PASSWORD': 'X2PayXfxWM#',
        'HOST': 'localhost',
        'PORT': ''
    }
}
```

Στην μεριά του server, αποτρέπουμε το SQL Injection με τον εξής τρόπο:

```
from django.db import connections

#username, password = get_input_from_client()

with connections['GDPR'].cursor() as cursor:
    cursor.execute("call login(%s', %s')", (username, password, ));
    result = cursor.fetchone()
    print(result)
```

Εδώ φαίνονται τα paths των default αρχείων στην django:

```
[root@snf-883459 ~]# ls
anaconda-ks.cfg  mysql80-community-release-el7-3.noarch.rpm  sql_injection  team28.txt
[root@snf-883459 ~]# cd sql_injection/
[root@snf-883459 sql_injection]# ls
manage.py  sql_injection_env  sql_injection_project  static
[root@snf-883459 sql_injection]# cd sql_injection_env/
[root@snf-883459 sql_injection_env]# ls
bin  lib  lib64  pyvenv.cfg
[root@snf-883459 sql_injection_env]# cd ../
[root@snf-883459 sql_injection]# cd sql_injection_project/
[root@snf-883459 sql_injection_project]# ls
asgi.py  __init__.py  __pycache__  settings.py  templates  urls.py  wsgi.py
[root@snf-883459 sql_injection_project]# cd ~
[root@snf-883459 ~]# ls sql_injection/
manage.py  sql_injection_env  sql_injection_project  static
[root@snf-883459 ~]# ls sql_injection/sql_injection_env/
bin  lib  lib64  pyvenv.cfg
[root@snf-883459 ~]# ls sql_injection/sql_injection_project/
asgi.py  __init__.py  __pycache__  settings.py  templates  urls.py  wsgi.py
[root@snf-883459 ~]# ls sql_injection/sql_injection_project/templates/
index.html
[root@snf-883459 ~]#
```