Security Practical 3 Answers

Dr Chris G. Willcocks

Email: christopher.g.willcocks@durham.ac.uk

Practical 3

Here are the answers to the third practical on database security.

Creating the 'Users' table

Listing 1: SQL

```
DROP TABLE IF EXISTS Users;
CREATE TABLE Users (
    id INTEGER PRIMARY KEY NOT NULL,
    username NVARCHAR(80) NOT NULL,
    firstname NVARCHAR(80) NULL,
    lastname NVARCHAR(80) NULL,
    grade FLOAT NULL,
    college NVARCHAR(80) NULL,
    fines FLOAT NULL
);
INSERT INTO Users VALUES(1,'jess', 'Jess', 'Adams', '86.4', 'Castle', 4.21);
INSERT INTO Users VALUES(2,'greg7', 'Greg', 'Courier', '62', 'Chads', 0);
INSERT INTO Users VALUES(3,'alice4', 'Alice', 'Smith', '57.9', 'Castle', 1.21);
INSERT INTO Users VALUES(4,'chrzi', 'Chris', 'Jackson', '73.8', 'Cuths', 5.33);
INSERT INTO Users VALUES(5,'lauran3', 'Laura', 'Walker', '21.2','Ustinov', 0.34);
INSERT INTO Users VALUES(6,'ai219', 'Andrea', 'Ivanov', '84.2', 'Ustinov', 0.92);
INSERT INTO Users VALUES(7,'seb123', 'Seb', 'Elbert', '17.3', 'Chads', 0);
SELECT * FROM Users;
```

Creating the 'Private' table

Listing 2: SQL

```
DROP TABLE IF EXISTS Private;
CREATE TABLE Private (
    id INTEGER PRIMARY KEY NOT NULL,
    userID int NOT NULL,
    wallet NVARCHAR(80) NOT NULL,
    FOREIGN KEY (userID) REFERENCES Users(id)
);
INSERT INTO Private VALUES(1, 3, 'KworuAjAtnxPhZARLzAadg9WTVKjY4kckS8pw38JrD33CeVYUuDm');
INSERT INTO Private VALUES(2, 1, 'Kwi5LPxVehUieD18AXiXTay9UDkRC7wLShe4tR5kzym1k2NhzEQ6');
INSERT INTO Private VALUES(3, 5, 'L4mGG15YacXWPU7LHM8Lj2LboxabRriZGHFZb5eLDN7mPXPPAHQF');
SELECT * FROM Private;
```

Inner join of 'Users' and 'Private' tables

Listing 3: SQL

```
SELECT Users.firstname, Users.lastname, Private.wallet
FROM Private
INNER JOIN Users ON Private.userID = Users.id;
```

SQL injection attacks

The following will download the Users table above. The −− is a comment.

Listing 4: SQL in HTML input field

```
' or 1=1; --
```

Prepared statements (parameterized queries)

You should always write prepared statements (parameterized queries) when handling user inputs.

Listing 5: Python

```
# don't ever write code like this:
cur.execute("select * from users where username='" + username + "';")

# instead use parameterized queries:
cur.execute("select * from users where username=?;", [username])

...

# also change this:
cur.execute("select private.wallet from users left join private on private.userid = users.id
    where username='" + username + "';")

# to use parameterized queries:
cur.execute("select private.wallet from users left join private on private.userid = users.id
    where username=?;", [username])
```

Anonymous grades

Introduce a random selection order:

Listing 6: Python & SQL

```
cur.execute('select grade from users order by random();')

# you may also want to do
cur.execute('select firstname, lastname, college from users order by lastname;')
```

Average grade of users & fines by college

Listing 7: Python & SQL

```
cur.execute('select AVG(grade) from users;')
```

Listing 8: Python & SQL

```
cur.execute('select college,SUM(fines) from users GROUP BY college;')
```

Inference attack on the class statistics

1. You can infer that Chris has a fine of 5.33 as he is the only member of Cuths.
2. You can infer that Greg and Seb have fines of 0 as they are members of Chads and fines can not be negative.
3. You can infer that Jess has a fine of 4.21 as you are logged in as Alice. You know your own fine of 1.21 and your own college is Castle, and you can see that Jess is the only other member of Castle, which has total fines of 5.42. Therefore you can do 5.42-1.21 = 4.21.