Datenbanken-Praktikum Sommersemester 2006
Fallstudie 1:
Sicherheit

Um den Service zur Handhabung von Prüfungen und Notenvergaben zu implementieren, wurde zunächst einmal ein Datenbankserver auf dem Rechner SNAREDRUM aufgesetzt. Das Datenbankschema wurde angelegt. Datenbankseitig wurden drei Benutzer angelegt, die die jeweiligen Rollen widerspiegeln:

- Der DB-User 'auth' dient der Authentifizierung von Benutzern. Mittels `GRANT`-Statements wurde festgelegt, dass 'auth' ausschließlich auf die Tabellen *Person*, *Dozent* und *Student* zugreifen kann.
- Der DB-User 'student' hat lesenden Zugriff auf die Tabellen *Kurs* und *prüfung*. Er kann neue Einträge in *Prüfung* anlegen und dabei die Spalten *kursnummer*, *matr_nr* und *datum* ändern.
- Der DB-User 'dozent' darf ebenfalls *Kurs* und *Prüfung* lesen. Ferner kann er zu bestehenden Einträgen die Spalte *note* ändern.

Schadensbegrenzung beim Erraten von Datenbank-Passwörtern

Passwörter werden in der Tabelle *Person* MD5-verschlüsselt in der Spalte *passwort* gehalten. Der Service auf CONGA checkt alle SQL-Statements auf gefährliche Zeichen (wie '-' und ';') um Exploits vorzubeugen.

Die Authentifizierung läuft folgendermassen ab:

(a) Der Benutzer gibt auf dem Client TABLA seine Benutzer-ID und das Passwort ein

(b) Eine Funktion `getPassword` wird auf dem Server CONGA aufgerufen; diese hat der eingegebenen Benutzer-ID als Parameter und ermittelt mittels eines SQL-Statements wie

```
SELECT passwort FROM Person where id='benutzer-id'
```

das eigentlich Passwort des Benutzers. Der Server CONGA nimmt dazu Verbindung mit dem Datenbankserver SNAREDRUM auf, loggt sich dabei als DB-User 'auth' ein und schickt obiges SQL-Statement an die Datenbank. Die Ausgabe des SQL-Statements wird an CONGA zurückgeliefert und dort ausgewertet; wurde ein Eintrag gefunden, wird dieser an den Client TABLA zurückgeschickt; ansonsten wird eine entsprechende Fehlermeldung an TABLA geschickt.

Hier wird das echte Passwort vom Server zum Client geschickt, wo die eigentliche Authentifizierung stattfindet. Da auch kein sicherer Übertragungskanal vorliegt, kann das echte Passwort abgefangen und durch den Hashwert des Passworts ersetzt werden, das der Eindringling auf dem Client eingegeben hat. Dadurch wird der Eindringling authentifiziert.

(c) Nachdem der Client auf TABLA das (MD5-verschlüsselte) Passwort von CONGA bekommen hat, wird das vom Benutzer eingegebene Passwort ebenfalls MD5 verschlüsselt; beide Hashwerte werden miteinander verglichen. Bei Gleichheit war die Authentifizierung erfolgreich; bei Abweichung oder dem vorherigen Empfang einer Fehlermeldung von CONGA schlägt sie fehl.

Nachdem der Benutzer erfolgreich authentifiziert wurde, werden zunächst mal alle Daten, die der Benutzer lesen oder manipulieren kann, vom Server CONGA angefordert. Dazu schickt TABLA die Benutzer-ID an CONGA, der sich daraufhin wieder als DB-User 'auth' auf der Datenbank einloggt und mittels des Statements

```
SELECT id FROM Dozent WHERE id='benutzer-id'
```

herausfindet, ob die Person unter dieser ID als Dozent gelistet ist. Ist dieser Test positiv, nimmt der Service auf CONGA an, dass es sich um einen Dozenten handelt; ist er negativ, geht der Service von einem Studenten aus. Je nachdem, ob es sich um einen Dozenten oder Studenten handelt, loggt sich der Service auf CONGA nun als 'dozent' oder 'student' auf der Datenbank ein. Die jeweiligen Daten werden aus der Datenbank gelesen (Student: Kurse, alle Prüfungen mit seiner/ihrer Matrikelnummer inkl. Note; Dozent: Kurse, die von ihm/ihr gehalten werden; Prüfungen zu diesen Kursen inkl. Note) und an den Client zurückgeschickt (zusammen mit der Information, ob der Benutzer Dozent oder Student ist).

Die Daten werden unverschlüsselt über's Netz geschickt, können also abgehört werden. Ferner können mögliche Funktionsaufrufe abgefangen und simuliert werden, da keine weitere Authentifizierung mehr stattfindet. Wegen der fehlenden Verschlüsselung und Zertifizierung sind jederzeit Man-in-the-Middle-Attacken möglich.

Der Client zeigt die Daten nun entsprechend an und bietet entsprechende Operationen an: Studenten können ihre Noten einsehen und sich für neue Prüfungen anmelden, Dozenten können Noten vergeben. Jede Änderung der Daten verursacht einen neuen Funktionsaufruf vom Client auf dem Server, bei dem die geänderten Daten, die Benutzer-ID und die Information, ob es sich um einen Studenten oder Dozenten handelt, übermittelt werden. Der Server loggt sich dabei wieder als Benutzer 'student' oder 'dozent' auf der Datenbank ein, und schickt entsprechende `INSERT`, `UPDATE` und `DELETE`-Statements an die Datenbank.

Die Betreiber des Systems argumentieren, dass ihr System sicher sei, da Passwörter nur verschlüsselt als Hashwert vorliegen und ein Benutzer erst authentifiziert sein muss, um überhaupt Daten einzusehen. Änderungen sind nur auf Basis der jeweiligen Rolle (Student/Dozent) möglich. Exploits werden verhindert.

- Was haltet ihr von dieser Aussage? Welche Maßnahmen wurden unternommen, um das System sicher zu machen? Wo liegen mögliche Gefahren? Hinweis: Ist die Überprüfung der Authentifizierung im Client, wie wir es hier vorfinden, sinnvoll? Was kann insbesondere passieren, wenn Passwörter von Server zum Client über einen unsicheren Kanal übertragen werden?
- Welche Änderungen würdet ihr vornehmen, um das System sicherer zu machen?