



Übung zur Vorlesung *Einsatz und Realisierung von Datenbanksystemen* im SoSe17

Maximilian E. Schüle (schuele@in.tum.de)
<http://db.in.tum.de/teaching/ss17/impldb/>

Blatt Nr. 03

Hausaufgabe 1

Eine statistische Datenbank ist eine Datenbank, die sensitive Einträge enthält, die aber nicht einzeln betrachtet werden dürfen, sondern nur über statistische Operationen. Legale Operationen sind beispielsweise Summe, Durchschnitt von Spalten und Anzahl der Tupel in einem Ergebnis (**count**, **sum**, **avg**, ...).

Nehmen wir an, Sie haben die Erlaubnis, im **select**-Teil einer Anfrage ausschließlich die Operationen **sum** und **count** zu verwenden. Weiterhin werden alle Anfragen, die nur ein Tupel oder alle Tupel einer Relation betreffen, abgewiesen. Sie möchten nun das Gehalt eines bestimmten Professors herausfinden, von dem Sie wissen, dass sein Rang „C4“ ist und er den höchsten Verdienst aller C4-Professoren hat. Beschreiben Sie Ihre Vorgehensweise.

Lösung: Vgl. Übungsbuch.

```
with Professoren (persnr, name, rang, raum, gehalt) as (  
  values (2125, 'Sokrates', 'C4', 226, 120000) UNION  
  values (2126, 'Russel', 'C4', 232, 110000) UNION  
  values (2127, 'Kopernikus', 'C3', 310, 10000) UNION  
  values (2133, 'Popper', 'C3', 52, 20000) UNION  
  values (2134, 'Augustinus', 'C3', 309, 30000) UNION  
  values (2136, 'Curie', 'C4', 36, 130000) UNION  
  values (2137, 'Kant', 'C4', 7, 100000)  
)  
alle as ( -- alle C4 Profs  
  select count(*) as AnzahlProfsC4, sum(Gehalt) as SumGehaltC4  
  from Professoren  
  where Rang = 'C4'  
)  
einerweniger as ( -- der reichste Prof weniger  
  select count(*) as AnzahlProfs, sum(Gehalt) as SumGehalt  
  from Professoren  
  where Rang = 'C4'  
    and Gehalt < 1.1*( -- Faktor dass AnzahlProfs = AnzahlProfsC4-1  
      select sum(Gehalt)/count(*) -- avg(Gehalt)  
      from Professoren  
      where Rang='C4'  
    )  
)
```

```

-- => MaxGehalt = SumGehaltC4 - SumGehalt
SELECT AnzahlProfcsC4-AnzahlProfcs, SumGehaltC4 - SumGehalt
FROM alle, einerweniger

-- fuer alle weiteren Gehaelter: Faktor weiter erhoehen, dann
-- NaechstesGehalt = SumGehaltAlt - SumGehaltNeu
-- fuer kleinstes Gehalt: analog nur in die andere Richtung

```

Hausaufgabe 2

Skizzieren Sie die Funktionsweise von SSL. Erläutern Sie hierzu, wie der einfache TLS Handshake funktioniert. Eine Lösungsmöglichkeit wäre das Zeichnen eines passenden Message Sequence Charts.

Alles ohne Resumed Handshake:

- **Client:** ClientHello, höchste Version, Zufallszahl1, Ciphers
D.h. der Client übermittelt seine Parameter für die Verbindung, beispielsweise, was er für Techniken unterstützt etc.
- **Server:** ServerHello, genutzte Version, Zufallszahl2, genutztes Cipher
Hierbei sollte gelten: $Cipher \in Ciphers$, $genutzte\ Version = \max(VersionenClient, VersionenServer)$, $genutztes\ Cipher = \max(CipherClient, CipherServer)$
Der Server wählt hier also bereits die geltenden Parameter für die Verbindung aus. Ein gängiges Problem ist, dass der Server sich aufgrund der vom Client übermittelten Informationen nicht auf zu unsichere Protokolle festlegen darf und potentiell Verbindungen abweisen muss, um effektiven Schutz vor Angriffen zu bieten.
- **Server:** Certificate
Im Allgemeinen übermittelt der Server ein Zertifikat an den Client, welches es dem Client erlaubt, die Identität des Servers zu verifizieren. Im Internet, d.h. bei https Verbindungen, dienen hierzu typischerweise Zertifikate, die von einer Autorität signiert wurden, der der Client vertraut. Diese sog. root Zertifikate werden vom Browser Hersteller mit dem Browser an den Client ausgeliefert und typischerweise nicht durch den Anwender geprüft.
- **Server:** ServerHelloDone
Handshake beendet, d.h. alle Informationen für die Einleitung der Verschlüsselung sind ausgetauscht.
- **Client:** ClientKeyExchange, PreMasterSecret, evtl. Public Key
PreMastersecret wird nun i.A. mit dem Public Key des Servers verschlüsselt. Hierdurch ist diese dem Client bekannt, kann vom Server entschlüsselt werden, jedoch niemand, der die Verbindung abhört kann es einfach rekonstruieren.
- **Client:** ChangeCipherSpec: Ab hier verschlüsselt, Finished: Verschlüsselte Nachricht mit Hashes über alles vorherige.
- **Server** ChangeCipherSpec: Ab hier verschlüsselt, Finished siehe oben.
- Ab jetzt wird die Verbindung von der Applikation verwendet.

Alles streng nach http://en.wikipedia.org/wiki/Transport_Layer_Security

Hausaufgabe 3

Bob hat ein Vorlesungsverzeichnis für die Universität programmiert und unter `http://db.in.tum.de/~schuele/sql_verzeichnis.html` online gestellt.

Um die Suche zu erleichtern, kann die Anzahl der SWS durch ein Parameter eingeschränkt werden. Finden sie einen speziell präparierten Parameter, bei dessen Eingabe statt der Vorlesungen die Liste der Studenten ausgegeben wird. Die Datenbank folgt dem bekannten Universitätsschema.

Bob erfährt von der Sicherheitslücke und schlägt vor die bekannten Tabellen einmalig mit zufälligen Namen umzubennen, so seien sie nicht zu finden. Würde diese *Sicherheitsmaßname* helfen?

- Injection: `0 union all select name, matrnr, semester from studenten`
- Nein, da z.B. mit `0 union select *,1,1 from pg_tables` eine Liste der Datenbanken ausgegeben werden kann.

Hausaufgabe 4

Sie haben die User-Tabelle zweier Pizzalieferanten ausgelesen, jedoch scheinen die Passwörter uncharakteristisch kompliziert zu sein. Das von Ihnen erhaltene Resultat ist das Folgende:

id	name	password
1	luigi	4d75e8db6a4b6205d0a95854d634c27a
2	mario	fe78ea401158dd5847c4090b8bb22477e510febf

- Was könnte der Grund für diese hexadezimalen, 32 bzw. 40 Stellen lange Passwörter sein?
- Können Sie trotzdem den Klartext finden?
- Wie können Sie das Passwort sicherer Speichern?
- Wie können Sie für diese Art von Passwortspeicherung Bruteforce-Attacken erschweren?
- Das erste Passwort wurde MD5 gehasht, das zweite SHA-1.
- Webrecherche nach dem Hash, es gibt sog. Rainbow-Tables in denen zahlreiche MD5- und SHA-1 Hashes vorberechnet sind.
- MD5 mehrfach anwenden, besser: Einen Salt verwenden, beispielsweise das Passwort zusammen mit dem Erstellungsdatum des Accounts oder dem Accountnamen hashen.
- Eine bessere Hashfunktion verwenden (MD5 und SHA1 werden nicht mehr empfohlen) die mehr Rechenzeit benötigt. Genauso wichtig, den User zwingen komplexere Passwörter zu benutzen damit Wörterbuchangriffe ineffizient werden.
- Alternativ hilft eine Key-Derivation-Funktion (KDF) ein Passwort mittels einer Pseudozufallsfunktion zu strecken.

Gruppenaufgabe 5

Sie fangen die folgende, mit RSA verschlüsselte Nachricht ab: 13. Sie kennen den öffentlichen Schlüssel (3,15). Wie lautet die Nachricht im Klartext? Geben Sie die komplette Herleitung an.

Alles laut Wikipedia <https://de.wikipedia.org/wiki/RSA-Kryptosystem>:

- Öffentlicher Schlüssel (e,N)
- Privater Schlüssel: (d,N)

$$N = p * q$$

mit p und q sehr großen Primzahlen. e , der sog. Verschlüsselungsexponent wird als Teilerfremde zahl zu $\phi(N)$ gewählt, wobei gilt $1 < e < \phi(N)$. $\phi(N)$ ist hierbei definiert als $\phi(N) = (p - 1) * (q - 1)$. d , der sog. Entschlüsselungsexponent, ist gerade das multiplikative inverse von e bezüglich des Moduls $\phi(N)$. Die Berechnung erfolgt mittels erweiterten euklidischen Algorithmus.

Die Entschlüsselung einer verschlüsselten Nachricht C zu ihrem Klartext K erfolgt mittels der Formel

$$K = C^d \pmod N.$$

Aus der Angabe wissen wir:

- $N = 15$
- $e = 3$
- $C = 13$

Wir müssen also zunächst d berechnen. Dies wäre einfach, wenn wir $\phi(N)$ wüssten. Hierzu ist die Primfaktorzerlegung von N nötig. Dies ist für die Zahl 15 äußerst einfach, es gilt $N = 5 * 3$. Damit ist $\phi(N) = 4 * 2 = 8$. Die Lösung der Kongruenz

$$e * d \equiv 1 \pmod{\phi(N)}$$

bzw. im konkreten Fall

$$3 * d \equiv 1 \pmod 8$$

können wir raten, indem wir alle im Bezug auf 8 teilerfremden Zahlen z betrachten, für die gilt: $1 < z < 8$.

Für $z = 3$ gilt $3 * 3 \equiv 1 \pmod 8$, womit $d = 3$ ist.

Wir entschlüsseln nun die Nachricht:

$$K = 13^3 \pmod{15} = 7$$

Der Klartext K ist also 7.

Hausaufgabe 6

Ein bekannter Anwendungsbereich des RSA-Kryptosystems ist das Verschlüsseln und Signieren von E-Mails. Die beiden Standards sind S/MIME und OpenPGP. Für ersteres

benötigen Sie ein Zertifikat, ausgestellt von einer Zertifizierungsstelle, wie sie die TUM für alle E-Mail-Adressen ausgibt: https://wiki.rbg.tum.de/Informatik/Helpdesk/Mail#A_2.2_Zertifikat_f_195_188r_nicht_in.tum.de_Adresse

Ein Schlüsselpaar für OpenPGP können Sie sich jederzeit selbst und für jede E-Mail-Adresse zulegen. Beschäftigen Sie sich näher mit OpenPGP, damit Sie Ihre E-Mails verschlüsseln können und schicken Sie Ihrem Tutor eine verschlüsselte und signierte E-Mail. Ihr Tutor erklärt Ihnen während der Übungsstunde, an welche Adresse Sie Ihre E-Mail schicken sollen und wo sie den entsprechenden öffentlichen Schlüssel erhalten. Dafür erhalten Sie einen Bonuspunkt.

OpenPGP: <https://de.wikipedia.org/wiki/OpenPGP>

Enigmail für Thunderbird: <https://www.enigmail.net>