# L2\_2.4 Kennwörter in Datenbank speichern

Wir haben gelernt, dass es recht einfach ist, einen Angriff auf ein DBMS in Form einer SQL-Injection durchzuführen. Die konsequente Verwendung von „prepared statements“ wirkt hier entgegen. Sollte ein Angreifer allerdings mehr Zugriff auf das DBMS erlangen, als ihm zusteht, kann dies fatale Auswirkungen haben. Beispielsweise können Zugangsdaten und wichtige Informationen eines Unternehmens in die falschen Hände geraten.

Üblicherweise werden daher Kennwörter nicht im Klartext, also in lesbarer Form, gespeichert. In der Praxis werden nicht die Kennwörter gespeichert, sondern ein Hashwert, der sich aus diesen Kennwörtern errechnen lässt. Man spricht auch von kryptografischen Hashfunktionen. Eine Eigenschaft einer solchen Funktion ist es, dass der Hashwert sehr schnell aus einer Nachricht (hier: Kennwort) generiert werden kann, der umgekehrte Weg jedoch immens aufwendig und rechenintensiv ist.



Hashfunktion

ee1c6181c49456afa7bc053e95a9c0e4

Hase

Hashwert

Nachricht: Kennwort im Klartext

Moderne DBMS bieten solche Hashfunktionen an, die wir nutzen können, um nicht das Kennwort, sondern den mit der Hashfunktion gebildete Hashwert zu speichern. Sollte ein Angreifer diesen Hashwert ermitteln können, besteht keine praktikable Möglichkeit, wieder das Kennwort im Klartext herzustellen.

Meldet sich der Anwender nun mit seinem Kennwort an, so wird aus dem übermittelten Kennwort erneut der Hashwert berechnet und das Ergebnis mit dem in der Datenbank gespeicherten Hashwert verglichen. Sind beide Hashwerte identisch, ist die Wahrscheinlichkeit, dass das Kennwort korrekt ist sehr, sehr, sehr hoch, der Anwender ist authentifiziert.

**Problem:**

Wir haben erfahren, dass die Ermittlung der Nachricht aus deren Hashwert sehr aufwendig und in der Praxis nicht möglich ist. Man sollte denken, dass ein Angreifer mit einem erbeuteten Hashwert nichts anfangen kann. Dies ist nur dann der Fall, wenn das Kennwort einigermaßen komplex ist. Im Internet finden sich zahlreiche Tabellen (Rainbow Tables), die den Hashwert für verschiedene Hashfunktionen zu weit verbreiteten Kennwörtern liefern. Findet man den erbeuteten Hashwert in diesen Tabellen, so kennt man auch das Kennwort.

**Lösung:**

Man kann kaum sicherstellen, dass ein Anwender ein ausreichend sicheres Kennwort wählt. Daher erzeugt man in der Praxis den Hashwert nicht direkt aus dem Kennwort, sondern ergänzt das Kennwort zuvor beispielsweise mit einem Pre- oder Postfix, einer zufällig gewählten Zeichenfolge. (salt). Diese Zeichenfolge kann für jeden Anwender unterschiedlich oder für alle Anwender gleich gewählt werden. Damit können auch einfache, weit verbreitete Kennwörter sicherer gemacht werden.



X\_

Hase

Hashfunktion

61f4f26f3cecf7a41b3711c2d3ca0883

Hashwert

Nachricht mit „salt“ im Klartext: X\_HASE

Erlangt ein Angreifer nur Zugriff auf das DBMS, so erfährt er zwar immer noch die Hash-Werte, diese stammen aber nun von Kombinationen von Passwort und einem unbekannten Salt. Ein Wörterbuchangriff ist sinnlos, weil kein Wörterbuch zufällig eine der Passwort-Salt-Kombinationen enthalten wird. Auch ein Brute-Force-Angriff wird drastisch erschwert, weil man nicht nur die Passwörter durchprobieren muss, sondern die Kombinationen aus Passwort und Salt.