Passwörter sicher als #HASH gespeichert:

Benutzer vertrauen Ihrer Internetseite Daten an. Verantwortung im Umgang mit Passwörtern ist daher Pflicht.

Kryptographische Hash-funktionen sind Einweg-funktionen. Passwörter lassen sich nicht wieder ableiten.

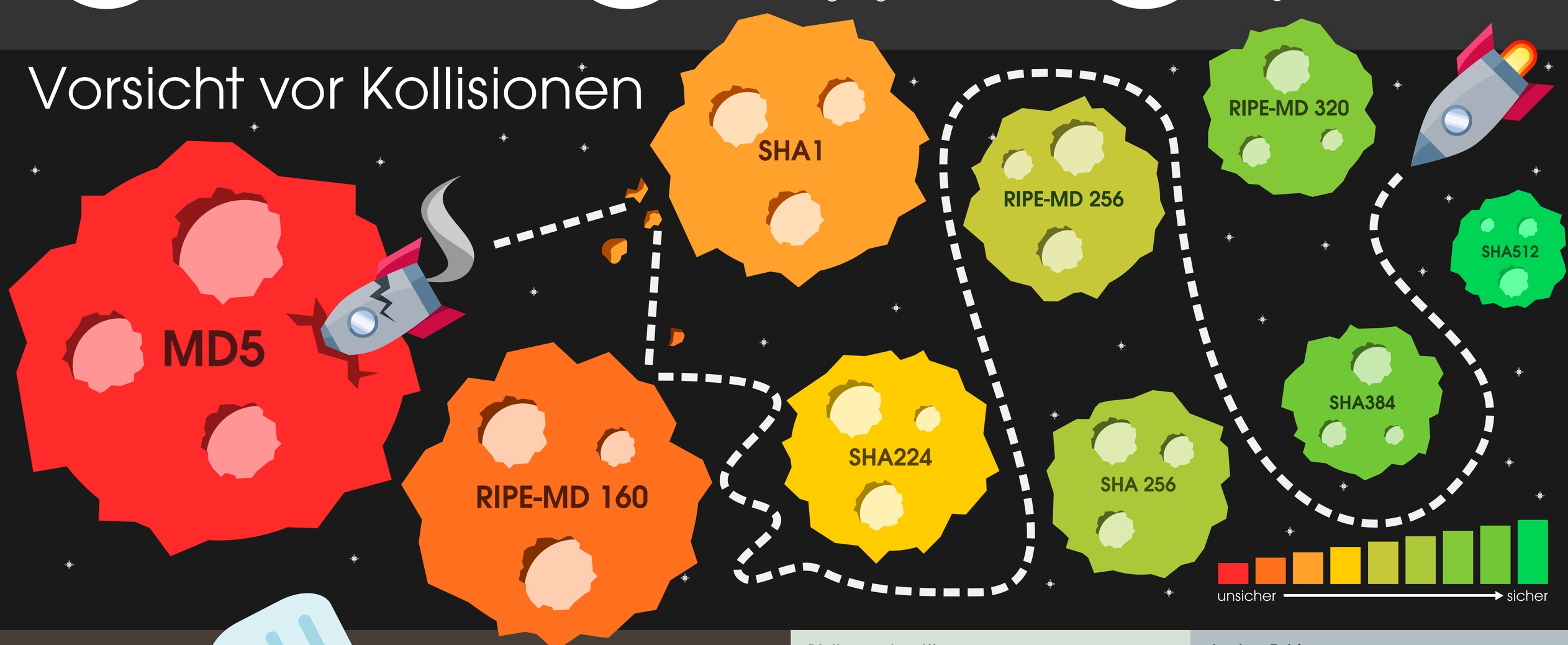
Starke Funktionen generieren einzigartige Hashes. Es sollten nie zwei gleiche Hashes erzeugt werden (Kollision).

Klartext Passwörter dürfen niemals auf der Festplatte gespeichert werden, nur ihre Hashwerte in der Datenbank.

Bei Passwortprüfung werden nur die Hashes verglichen. Sie werden dabei über sichere Verbindungen gesendet.

Selbst die Änderung eines einzelnen Zeichens des Passworts erzeugt einen völlig anderen Hashwert.

Vorsicht vor Kollisionen



SALT

Bevor der Passwort-Hash erzeugt wird, kann dem Passwort eine zufällige Zeichenkette namens Salt hinzugefügt werden.

Jeder Nutzer erhält einen einzigartigen Salt. Das macht sogar Reverse-Lookup-Tables nutzlos.

Die Angreifer kennen Salts einer gestohlenen Datenbank im Voraus nicht, daher sind vorberechnete Lookup- und Rainbow-Tables ineffektiv.

Der Salt muss nicht geheim sein, er erfüllt seinen Zweck, indem er den resultierenden Hash randomisiert.

Nutzernamen sollten nicht direkt als Salt genutzt werden, denn sie sind sehr vorhersehbar.

Salts dürfen nicht zu kurz sein! Ein guter Salt ist mindestens so lang, wie die Ausgabe der Hashfunktion.

Einfache Zufallszahlen reichen nicht. Es sollten nur kryptographisch sichere Zufallszahlen-generatoren genutzt werden.

Ein Salt sollte nie fest in ein Programm eingebaut werden, sonst könnten mehrere Nutzer dieselben Hashes führen.

PEPPER

- Ein Pepper ist eine **geheime** Zeichenkette und wird nicht in der Datenbank abgelegt.
- Der Passwort-Hash wir dann aus Pepper, Salt und Passwort erzeugt.
- Pepper sollten so lang sein, wie ein guter Salt.

Keyed-Hash Message Authentication Code (HMAC)

Um die Integrität und Authentizität einer Nachricht festzustellen erzeugt ein **HMAC** den Hashwert aus der Kombination von kryptografischer Hashfunktion und **geheimem** Schlüssel. Nur jemand mit Kenntnis des Schlüssels kann den Hashwert des Passworts berechnet haben.

Dictionary Angriff

Der Angriff findet mit einer Datei statt, in der Wörter, Phrasen und Zeichenketten gespeichert sind, die sich als Passwort eignen. Die Hashes werden dann mit denen der Passwörter verglichen. Gefüllt werden die Dateien mit Wörtern aus Texten und gestohlenen Passwörtern.

Lookup-Table

Vorberechnete Hashes werden in einem Wörterbuch abgelegt und den tabellarisch sortierten, korrespondierenden Passwörtern zugeordnet. Hashes werden damit effektiv gefunden. Reverse-Lookup-Table ordnen jedem Hash einen Nutzer zu und machen Nutzer mit gleichem Passwort sichtbar.

Brute-Force Angriff

Dieser Angriff testet die Hashes jeder möglichen Zeichenkette bis zu einer bestimmten Länge. Die Methode benötigt zwar die meiste Rechenzeit, mit genug Zeit aber, wird das Passwort gefunden. Das lässt sich erschweren, indem lange Passwörter genutzt werden, sodass sich der Aufwand nicht mehr lohnt.

Rainbow-Table

Rainbow-Table arbeiten wie reguläre Lookup-Table mit Zeit-Speicher-Kompromiss. Sie enthalten vorberechnete Ketten von Hashes und brauchen daher mehr Zeit pro Hash. Sie können jedoch bei kleinerem Speicher mehr Lösungen ablegen, was sie sehr effektiv macht.

Kryptografische Funktionen nie selbst implementieren, sondern auf erprobte Standardbibliotheken zurückgreifen!

Key Stretching

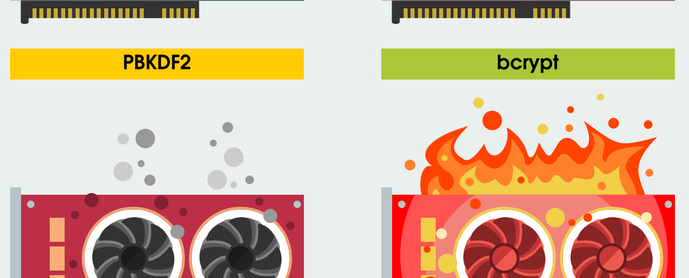
Einzelne Hashes sind trotz Salt anfällig für Brute-Force oder Dictionary Angriffe. Die Nutzung langsamer **Key-Derivation-Functions** wie **PBKDF2, bcrypt, scrypt** und des neueren **argon2** erhöhen den Zeitbedarf und mindern so die Effizienz von Angriffen erheblich.

Custom-Hardware Angriff

Sehr effiziente Angriffe finden mit maßgefertigter Hardware statt, die Milliarden von Hashes pro Sekunde berechnet. Es werden dazu sowohl speziell entwickelte, integrierte Schaltkreise **(application-specific integrated circuits - ASIC)** verwendet, als auch **field-programmable gate arrays (FPGA)**. In letzteren kann sogar eine logische Schaltung programmiert werden.

GPU Resistenz

Hardware Attacken mit High-End Grafikkarten bedienen sich auch des Grafikprozessors **(GPU)** um die Aufgaben zu parallelisieren. Memory-hard Funktionen wie **bcrypt**, **scrypt** und **argon2** sind jedoch speicherabhängig und können auf GPUs nicht so effektiv berechnet werden. Die Parallelisierung der Operationen ist durch ihren hohen Speicherbedarf aufwendiger.



Quellen und Referenzen: https://www.internet-sicherheit.de/passwort-hashing

if(is) internet-sicherheit.

Sicheres Passwort-Hashing

Institut für Internet-Sicherheit - if(is)

www.internet-sicherheit.de

Stand: Februar 2016
Bereich: Web Development
Autor(en): Bothe, David
bothe@internet-sicherheit.de