Blockchain CheatSheet - Hashing

🕓 **Lesezeit: 5 Min**

Inhaltsverzeichnis

§ Grundlagen

Hashing ist keine Verschlüsselung, da du die ursprünglichen Daten nicht aus dem Hash rekonstruieren kannst, wie es bei verschlüsselten Dateien der Fall ist.

Man sollte Hashing wie einen Fingerabdruck betrachten; es liefert eine sichere genetische Referenz auf die Daten, ist aber nicht die "Daten selbst".

Wesentliche Merkmale guter kryptografischer Hashes

1. **Geschwindigkeit**: Der Hash muss in einem gewissen Maß leicht zu berechnen sein, um nicht anfällig für Brute-Force-Angriffe zu werden.
2. **Deterministisch**: Die gleiche Eingabe sollte immer die gleiche Ausgabe erzeugen.
3. **Einweg**: Es muss unmöglich sein, die ursprünglichen Daten aus dem Hash zurückzugewinnen. Insbesondere ist es schwierig, weil beim Hashing Daten verloren gehen können.
4. **Sicherheit**: Wenn du die Daten, die gehasht werden sollen, änderst, erhältst du einen völlig anderen Hash. Wenn du die Änderung rückgängig machst, erhältst du den ursprünglichen Hash zurück.
5. **Kollisionsresistenz**: Es ist unmöglich, dass zwei unterschiedliche Datensätze den gleichen Hash-Wert haben, daher ist Hashing konkurenzsicher*.
6. **Größe**: Es spielt keine Rolle, wie groß die Daten zum Hashen sind. Das Hashing-Verfahren hat allgemein eine große Kapazität.

> - **Kollisionsproblem**: Das Hauptanliegen ist nicht nur die Wahrscheinlichkeit, dass zwei Hashes kollidieren, sondern vielmehr die Wahrscheinlichkeit, dass innerhalb eines Datensatzes mindestens zwei identische Datenpunkte denselben Hash haben. Diese Wahrscheinlichkeit steigt signifikant mit der Größe des Datensatzes, ähnlich wie beim Geburtstagsparadoxon.

Salting

Salting ist die Praxis, einen zufälligen Wert zu dem gehashten Passwort hinzuzufügen. Dies ist der einzige Weg, Passwörter sicher zu hashen.

Miner

Die Aufgabe der Miner besteht darin, Transaktionen oder Daten aus dem Blockchain-Puffer zu nehmen und sie in Blöcke zu gruppieren. Jeder Block-Header ist 80 Byte groß.

Bevor diese Blöcke zur Blockchain hinzugefügt werden, müssen die Miner einen 32-Byte-Hash und einen Nonce einfügen, der den aktuellen Schwierigkeitsgrad erfüllt.

Sie tun dies, indem sie verschiedene Nonce-Werte durchprobieren, bis sie einen finden, der einen Hash produziert, der die Proof-of-Work-Bedingung erfüllt.

---

§ Hashing Mathematik

Überblick

- ***SHA (Secure Hash Algorithm)***: markdown Copia codice `1. SHA-1: 160 Bit 2. SHA-2: - SHA-224: 224 Bit - SHA-256: 256 Bit - SHA-384: 384 Bit - SHA-512: 512 Bit 3. SHA-3: - SHA3-224: 224 Bit - SHA3-256: 256 Bit - SHA3-384: 384 Bit - SHA3-512: 512 Bit`
- ***Technische Begriffe***:
  - **Padding**: Hinzufügen von Bits, um das Ende der Nachricht anzuzeigen.
  - **Padding mit Nullen**: Hinzufügen von '0'-Bits, um eine bestimmte Länge zu erreichen.
  - **Länge hinzufügen**: Hinzufügen der ursprünglichen Länge der Nachricht in Bits.
  - **Komprimierungsfunktion**: Der Prozess des Mischens von Bits, der kryptografische Operationen beinhaltet.
  - **Hash-Wert**: Der resultierende einzigartige geheime Code.

Verfahren

1. Nachricht vorbereiten

Unser Fall: Stell dir vor, du hast einen Satz, zum Beispiel: "Hallo Welt". Dies ist unsere Eingabe. Berechne die Länge der Eingabe in Bits (88 Bits in diesem Fall).

2. Endsignal hinzufügen (Padding)

Um dem Algorithmus zu signalisieren, dass der Satz beendet ist, fügen wir ein spezielles Symbol am Ende hinzu.

Unser Fall: Wir fügen ein '1'-Bit hinzu. Dieses Signal ist das Padding-Bit. Also haben wir jetzt "Hallo Welt1".

3. Blockstrukturen

Der Algorithmus arbeitet am liebsten mit Blöcken einer bestimmten Größe, um die Rechenleistung zu optimieren. Für SHA-256 beträgt die Blockgröße 512 Bit (64 Byte) auf einmal.

Kleine Daten - Fehlende Teile hinzufügen (Padding mit Nullen)

Unser Fall: Wenn der Satz nicht lang genug ist wie "Hallo Welt1", fügen wir Nullen hinzu, um ihn aufzufüllen. Also, wenn "Hallo Welt1" 88 Bit lang ist, fügen wir weitere 424 Nullen hinzu, um 512 Bit zu erreichen.

Große Daten - Portionierung

Wenn die zu hashenden Daten länger als 512 Bit sind, führt der Algorithmus mehrere Durchläufe über die Daten in Stücken durch.

4. Länge hinzufügen (Länge hinzufügen)

Am Ende fügen wir die Länge der ursprünglichen Nachricht in Bits hinzu, wie es die Padding-Regeln von SHA-256 vorschreiben.

Unser Fall: "Hallo Welt" war 88 Bit, also fügen wir eine 64-Bit-Darstellung von "88" hinzu. Nun haben wir insgesamt 512 Bit: 448 Bit Daten und Padding + 64 Bit Länge.

5. Zeichen mischen (Komprimierungsfunktion)

Nun beginnt der Algorithmus, die Zeichen zu mischen. Er nimmt jeden 512-Bit-Block und führt eine Vielzahl von komplexen Operationen durch, die die Bits auf sehr komplizierte Weise ändern, die nur der Algorithmus kennt. Dieser Schritt umfasst Operationen wie XOR, bitweise Verschiebungen und modulare Additionen.

Große Daten - Merkle-Wurzel

Der Algorithmus nimmt alle 512-Bit-Stücke und verknüpft sie paarweise, wobei das Hashing immer wieder durchgeführt wird, bis ein 256-Bit-Hash herauskommt.

Technisch gesehen wird bei langen Gruppen von gehashten Daten eine einzelne Hash-Wert erstellt, der **Merkle-Wurzel** genannt wird.

6. Geheinen Code erhalten (Hash-Wert)

Nachdem der Algorithmus das Mischen abgeschlossen hat, erhalten wir einen einzigartigen geheimen Code, der als Hash oder Digest bezeichnet wird, wie "a7b9c3d2". Dieser Code ist besonders, weil sich der Hash vollständig ändert, selbst wenn du nur einen Buchstaben der ursprünglichen Nachricht änderst.

---

§ Anwendungen

Hashing ist nützlich, um zu überprüfen, ob Daten in einem bestimmten Zeitraum seit ihrer Erstellung beschädigt oder verändert wurden, oder um die Herkunft von Daten zu bestätigen. Dies ist möglich, indem der Hash von T0 mit T1 verglichen wird.

Vorgeschlagene Weiterführung

Blockchain CheatSheet - Kryptographie & Signaturen

Autor: Kenneth Boldrini