Algorytmy hashujace

Krzysztof Sławek 72902

Kwiecien 2024

1 Wstep

Problem, który postaram sie przybliżyc to temat zwiazany z algorytmami hashowania. Najpierw zaczne od przedstawienia czym jest hashowanie.

Hashowanie to generowanie ciagu znakow o stałej długości na podstawie danych wejściowych o dowolnej wielkości. Wyniekiem takieog hashowania jest losowy ciag znaków, który w przypadku tych samych danych wejsciowych generuje identyczny ciag znaków zwany hashem. Najmniejsza zmiana w danych wejściowych zmienia hash. Hashowanie jest procesem nieodwracalnym co w praktyce znaczy, że nie możemy odtworzyć z hasha danych, które były użyte na wejściu.

2 Przykłady zastosowań algorytmów hashujacych

- Bezpieczne przechowywanie haseł użytkowników w bazach danych.
- Weryfikacja integralności danych w transmisji (np. w protokołach bezpieczeństwa internetowego, takich jak TLS/SSL).
- Generowanie unikalnych identyfikatorów dla plików (np. kontrola wersji, kontrola integralności plików).
- Zabezpieczanie danych w systemach autoryzacji i uwierzytelniania.
- Weryfikacja autentyczności dokumentów (np. cyfrowe podpisy).
- Bezpieczne przechowywanie kluczy kryptograficznych.
- Generowanie skrótów dla danych w celu szybkiego porównywania (np. w bazach danych lub strukturach danych).
- Zabezpieczanie haseł w aplikacjach mobilnych i serwisach internetowych.
- Zapobieganie podstawianiu plików (ang. file spoofing) przez sprawdzanie ich skrótów.
- Utrzymywanie integralności danych w blockchainie.

3 Przykłady algorytmów hashujacych

1. MD5 (Message Digest Algorithm 5)

   Do czego sa najcześciej wykorzystywane: MD5 był kiedyś używany do bezpiecznego przechowywania haseł, weryfikacji integralności plików i innych zastosowań, ale ze wzgledu na swoje słabości kryptograficzne nie jest już zalecany do tych celów.

   Jak działaja: MD5 operuje na wiadomościach o zmiennej długości i zwraca 128-bitowy skrót (32 znaki szesnastkowe). Algorytm ten wykonuje szereg przekształceń na blokach danych wejściowych, w wyniku czego powstaje skrót.

   Złożoność czasowa: $O(n)$, gdzie n to długość danych wejściowych.

2. SHA-256 (Secure Hash Algorithm 256-bit)

   Do czego sa najcześciej wykorzystywane: SHA-256 jest powszechnie stosowany do weryfikacji integralności danych, generowania kluczy kryptograficznych, podpisywania cyfrowego i innych zastosowań wymagajacych bezpiecznej funkcji haszujacej.

   Jak działaja: SHA-256 operuje na blokach danych o długości 512-bitów i zwraca 256-bitowy skrót. Działa poprzez stosowanie serii przekształceń bitowych na danych wejściowych.

   Złożoność czasowa: $O(n)$, gdzie n to długość danych wejściowych.

3. BCrypt

   Do czego sa najcześciej wykorzystywane: BCrypt jest powszechnie używany do bezpiecznego przechowywania haseł w bazach danych.

   Jak działaja: BCrypt jest oparty na funkcji haszujacej Blowfish. Generuje unikalna sól dla każdego hasła i wykonuje szereg iteracji, co sprawia, że jest bardziej odporny na ataki brute force.

   Złożoność czasowa: Zależna od liczby rund algorytmu BCrypt. Typowo $O(2^{cost})$, gdzie *cost* to parametr określajacy liczbe rund.

4. SHA-3 (Secure Hash Algorithm 3)

   Do czego sa najcześciej wykorzystywane: SHA-3 jest używany do weryfikacji integralności danych, generowania skrótów i innych zastosowań, gdzie wymagana jest bezpieczna funkcja haszujaca.

   Jak działaja: SHA-3 operuje na blokach danych i zwraca skrót o określonej długości. Jest oparty na innych zasadach niż starsze wersje SHA, co sprawia, że jest bardziej oporny na niektóre ataki kryptoanalizy.

   Złożoność czasowa: $O(n)$, gdzie n to długość danych wejściowych.

5. Argon2

   Do czego sa najcześciej wykorzystywane: Argon2 jest specjalnie zaprojektowany do przechowywania haseł i jest uznawany za jeden z najbezpieczniejszych algorytmów haszujacych.

   Jak działaja: Argon2 wykorzystuje duża ilość pamieci i czasu obliczeń, co sprawia, że jest bardzo oporny na ataki brute force.

   Złożoność czasowa: Zależna od parametrów, w tym pamieci wykorzystanej przez algorytm. Typowo $O(m \cdot t)$, gdzie m to ilość zużytej pamieci, a t to liczba iteracji.

4 Odnośniki do materiałów

- https://www.okta.com/identity-101/hashing-algorithms/
- https://en.wikipedia.org/wiki/Cryptographic_hash_function
- https://en.wikipedia.org/wiki/MD5
- https://chainkraft.com/pl/co-to-jest-hashowanie/
- https://en.wikipedia.org/wiki/Merkle%E2%80%93Damg%C3%A5rd_construction
- https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.180-4.pdf
- https://en.wikipedia.org/wiki/Bcrypt
- https://datatracker.ietf.org/doc/rfc9106/
- https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.202.pdf