Merkle trær

Blokkjeder

Blokkjeder er en uforanderlig hovedbok som lagrer transaksjoner og sporer eiendeler (assets). En asset eller eiendel kan være fysiske ting som en bil, eiendom og datamaskin eller ikke fysiske ting som patenter, opphavsrett og kryptovaluta. Generelt så kan alt som har en verdi bli solgt og sporet på en blokkjede (IBM, u.å.). Så hva har Merkle trær med blokkjeder å gjøre? Merkle trær er en fundamental byggestein som gjør blokkjeder mulig.

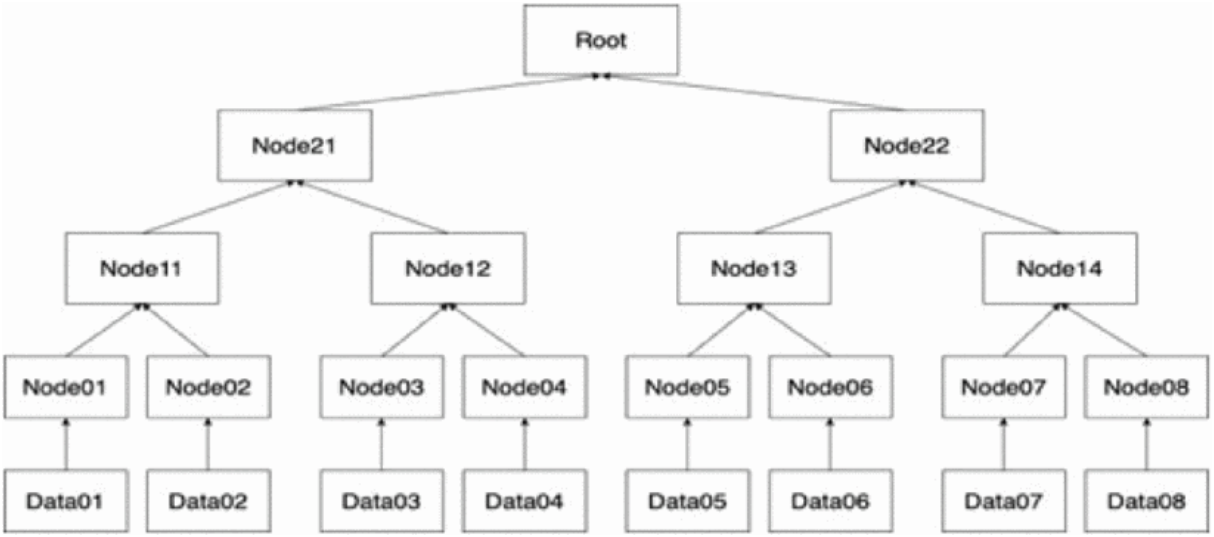
Ralph Merkle

Før vi ser litt nærmere på merkle trær så ser vi litt på mannen bak, Ralph Merkle. Merkle er en Senior Resreach Fellow ved «Institute for Molecular Manufacturing», medgründer i «Nanofactory Collaboration» og direktør i Alcor (Merkle, u.å.). Han er en informatiker og matematiker, kjent for å med-oppfinne offentlig nøkkel kryptografi, kryptografisk hashing (Merkle-Damgård konstruksjon) og Merkle trær (Wikipedia, u.å.). Han har en rekke interesser iblant annet cybersikkerhet med fokus på kryptografi, molekylær produksjon, kvantedatamaskiner og kryonikk (Merkle, u.å.). Merkle er en profilert mann som har fått priser som blant annet IEEE Richard W. Hamming Medal i 2010 og Computer History Museum Fellow i 2011 for sitt arbeid (Wikipedia, u.å.).

Hva er Merkle Trær?

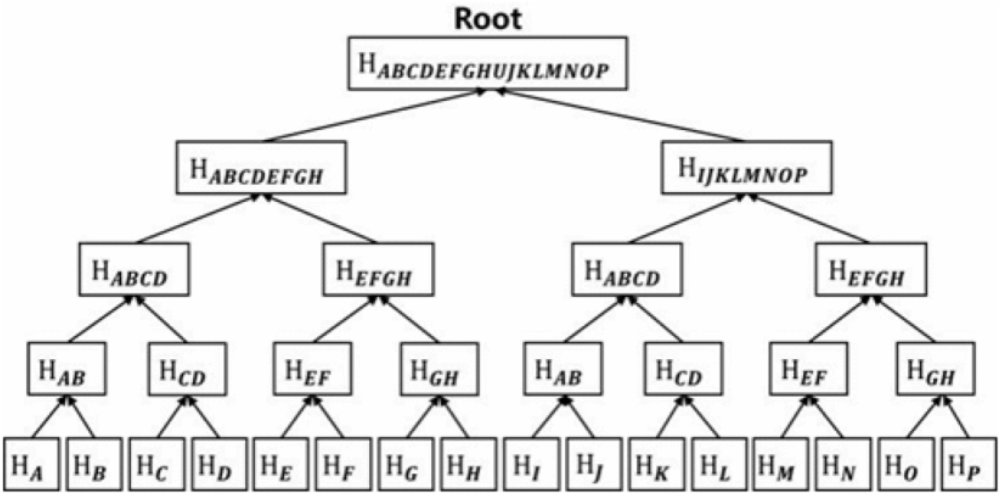
Merkle trær er en datastruktur som er basert på binær trær, men denne datastrukturen tar i bruk en kryptografisk hash funksjon. SHA256 er en kryptografisk primitiv brukt i Merkle trær. En hash funksjon tar inn en input kalt et Preimage som er av n lengde. Deretter blir preimage hash-et og outputer noe som blir kalt for en hash verdi, hash verdien er også ofte kalt et image og fingerprint. Denne hash veriden er i en fastsatt lengde som representerer den originale meldingen eller preimage (H. Liu et al., 2021, s. 556). Hash verdien kan ikke bli reversert med samme funksjon for å få den originale meldingen, dermed kan man verifisere en fil uten å se innholdet for å se om en fil ikke er blitt tuklet med (Investopedia, 2024).

I et Merkle tre så ligger all dataen i bunnen kalt datablokker som inneholder informasjon, i en blokkjede sammenheng er dette transaksjoner. Vær node har ingen eller to barn og alle noder bør ligge på samme nivå. På toppen av et Merkle tre finner du rot noden, denne rot noden inneholder all informasjon om alle datablokkene som en hash verdi. For å kalkulere hash-ene i et Merkle tre så starter man i bunnen ved å beregne hver node fra bunnen også oppover. For eksempel Node01 = H(Data01) «H(x) representerer en hash funksjon» slik gjør man det med hver datablokk. Så for å få noden i neste nivå så regner man Node11 = H(Node01 + Node02) «her er + en betegnelse på konkatenering, til slutt så ender man opp på siste nivå før rot noden hvor man tar Rot = H(Node21 + Node22). Så et Merkle tre er et slags kjede av hasher (H. Liu et al., 2021, s. 557). Figur 1 viser strukturen til et Merkle tre.



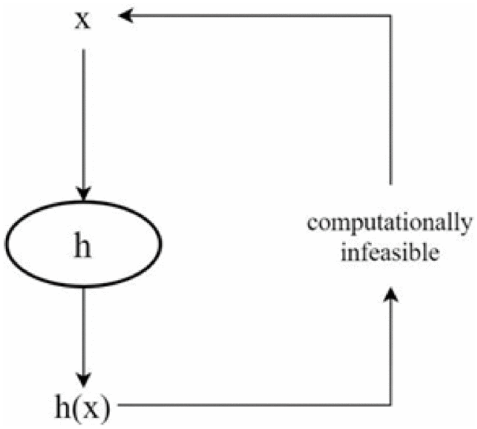
Figur 1 Strukturen til et Merkle tre (H. Liu et al., 2021, s. 557)

Siden Merkle trær bruker hashing så kan man verifisere at dataen i et Merkle tre ikke er tuklet med eller at den eksisterer. Vi vill vite om Data01 ikke er tuklet med så vi trenger hash verdier til noder for å finne for å gjenskape rot noden. Det enkleste er at man lager et nytt Merkle tre basert på informasjon vi vet for eksempel en transaksjon. Så generer vi nye hasher med informasjonen og til slutt vill vi ende opp med en ny rot node som skal være lik til den originale, dette er også kjent som Merkle proof. Det fine med Merkle proof er at man ikke trenger å vite hva dataen i de andre datablokkene er for å verifisere at den er sann (H. Liu et al., 2021, s. 557). Figur 2 viser en Merkle Proof.



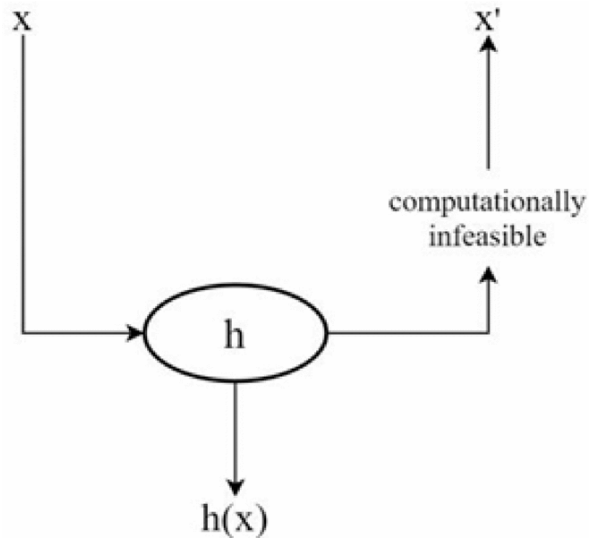
Figur 2 Merkle Proof (H. Liu et al., 2021, s. 558)

Merkle trær er basert på hash funksjoner så er det noen ting vi må tenke på som kan ødelegge integriteten til daten som kollisjoner, preimage og second preimage angrep. Preimage resistance er viktig for det skal være uberegnelig å finne ut av input x til en hash funksjon av hash verdien til x (H. Liu et al., 2021, s. 557). Figur 3 viser preimage resistance.



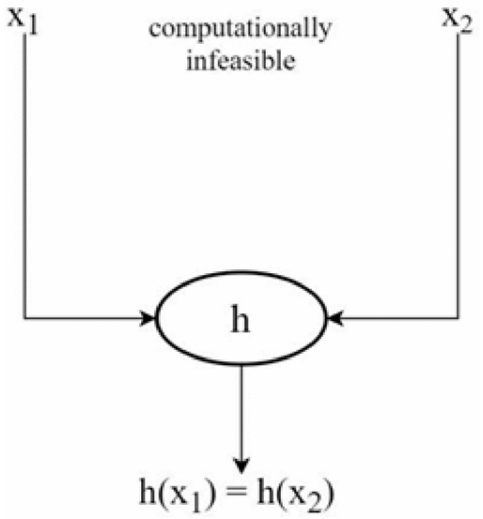
Figur 3 Preimage resistance (H. Liu et al., 2021, s. 557)

Det bør også ha resistans mot second preimage, for det bør være uberegnelig å finne en annen input som produserer den samme outputen (H. Liu et al., 2021, s. 557-558). Figur 4 viser second preimage resistance.



Figur 4 Second preimage resistance (H. Liu et al., 2021, s. 558)

Til slutt er det en egenskap til å tenke på når det kommer til hash funksjoner, collision resistance. For hash funkjsoner er utsatt for kollisjoner og det bør være uberegnelig å finne to forskjellige meldinger som produserer den samme hash verdien slik at H(x1) = H(x2) (H. Liu er al., 2021, s. 558). Figur 5 viser collision resistance.



Figur 5 Collision resistance (H. Liu et al., 2021, s. 558)

Merkle trær bruks områder

Merkle trær er brukt i flere ting enn blokkjeder. For merkle trær kan bli brukt for å verifisere hvilken som helst type data. For eksempel merkle trær er brukt i et par filsystemer som InterPlanetary File System (IPFS), Btrfs og ZFS. De er også brukt i peer to peer nettverk som BitTorrent, Bitcoin, Ethereum og Zeronet. NixOS sin pakkehåndterer bruker Merkle tre og noen NoSQL systemer som blant annet Apache Cassandra, Riak og Dynamo. Versjons håndtering som Git og Mercurial, men de er litt mer på grensen siden de bruker grafer og ikke trær (Wikipedia, u.å.).

Merkle trær fordeler og ulemper

Åpenbare fordeler er at merkle trær forhåpentligvis har resistans mot preimage, second preimage og kollisjons angrep. Dette kan jo variere ut ifra hvilken hash funksjon man velger å bruke, for eksempel så er det ikke anbefalt å bruke hash funkjsonen MD5 lenger fordi den er knekt. En annen fordel er at man kan bruke merkle proof for å verifisere om dataen ikke er tuklet med slik at den beholder integriteten sin. En annen fordel er at i blokkjede sammenheng er at man slipper å lagre på alle transaksjonene for man kan heller lagre de i en merkle rot som inneholder alle transaksjonene for den blokken. Når en ny node blir med i nettverket så krever den bare en merkle rot i stedet for en rask og effektiv synkronisering (Space and Time, 2023).

Som sagt så har sikkerheten til en hash funkjson mye å si for hvor sikker et Merkle tre er. Så det er viktig at man velger en hash funksjon som er sikker som SHA256 og ikke MD5. Av natur så vill hash funksjoner være utsatt for kollisjoner, preimage og second preimage angrep. Så det er viktig at disse er uberegnelige for å unngå dem.

Kilder

H. Liu, X. Luo, H. Liu & X. Xia. (2021). Merkle Tree: A Fundamental Component of Blockchains. *2021 International Conference on Electronic Information Engineering and Computer Science (EIECS)*, 2021, 556-561. https://doi.org/10.1109/EIECS53707.2021.9588047

IBM. (u.å.). *What is blockchain?* Hentet 27. september 2024 fra https://www.ibm.com/topics/blockchain

Investopedia. (2024, 26. juli). *Merkle Tree in Blockchain: What it Is and How It Works* https://www.investopedia.com/terms/m/merkle-tree.asp#toc-how-a-merkle-tree-is-used-in-a-blockchain

Merkle, R. C. (u.å.) *Ralph C. Merkle*. Ralph Merkle's Home Page. Hentet 27. september fra https://ralphmerkle.com/

Space and Time. (2023, 6. april). *Understanding Merkle Trees*. Medium. https://medium.com/@SpaceandTimeDB/understanding-merkle-trees-252f83aa6e51

Wikipedia. (u.å.) Merkle tree. Hentet 8. oktober 2024 fra https://en.wikipedia.org/wiki/Merkle_tree

Wikipedia. (u.å.) *Ralph Merkle*. Hentet 27. september 2024 fra https://en.wikipedia.org/wiki/Ralph_Merkle