Proof of work,

eller PoW er et kryptografisk konsept som fungerer som en konsensusmekanisme, oftest brukt i blokkjeder for å validere og sikre transaksjoner. Selv om konseptet til PoW ikke hadde blitt formalisert og hadde en betegnelse gitt enda, var konseptet originalt implementert i 1993 hos Hashcash av Moni Naor og Cynthia Dwork som en metode for å avskrekke «denial-of-service», eller Dos angrep, også kjent som Tjenesteangrep og andre tjenestebaserte angrep. PoW var originalt idé satt grunnet e-post spam, hvor idéen.

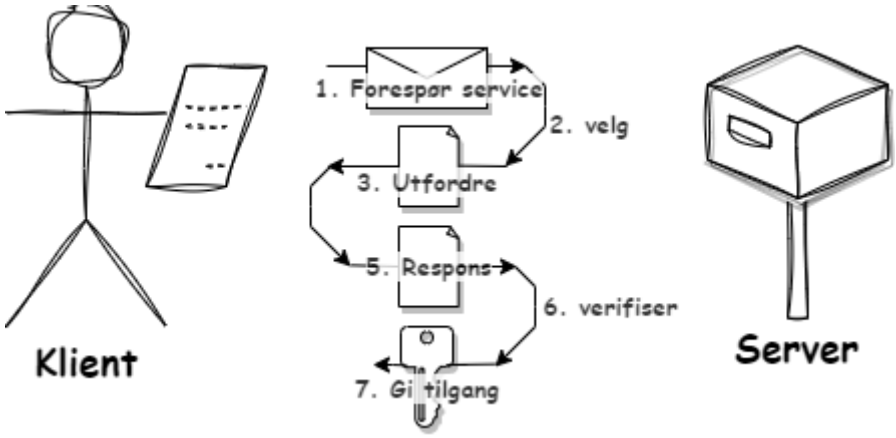
Selve begrepet ble foreslått og formelt blitt tatt i bruk i ett 1999 papir av Markus Jakobbson og Ari Juels, for å senere bli tilpasset som digitale «token» av Hal Finney 2004. Tilpasningen var idegjørt med «gjenbrukbar proof-of-work» i tankene, med bruk av 160-bit secure hash alorithm 1, eller SHA 1.

PoW som en konsensusmekanisme, er basert på ett nettverk av aktører setter avhengighet av hverandre for å bedømme at en viss nødvendig mengde med beregningsinnsats har blitt beregnet, eller gitt. Dette er hvor nettverket er avhengig av to roller, «beviseren» som utgjør beregningsinnsatsen og «verifikatoren» som tar imot arbeidet til «beviseren» for verifikasjon av arbeid. Brukt i mining, så utvinner gruvearbeidere disse blokkjedene, hvor de konkurrerer om å løse komplekse matematiske problemer i kostnad av betydelig beregningskraft.

De som først løser problemet, må vise sin PoW for å få lov til å tilføye en ny blokk som blir da lagret i transaksjonskjeden(blokkjeden) etter verifikasjon av verifikatorene, som regel er det de resterende gruvearbeiderne som verifiserer. I blokkjedeteknologi blir også transaksjonen signert digitalt med asymmetrisk kryptering. Dette gjør at PoW som en konsensusmekanisme er svært kostbar, både i energi og beregningskraft, men som ett resultat er PoW svært sikker, hvor det krever betydelige ressurser å kompromittere nettverket.

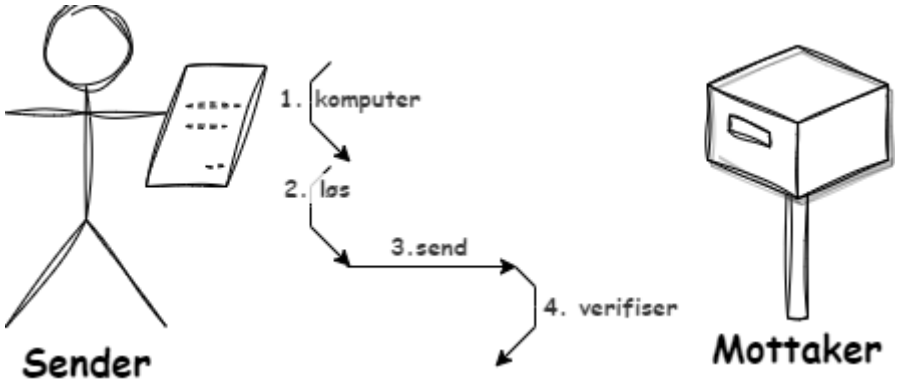
Det finnes to klasser av proof-of-work-protokoller.

Challenge–response-protokoller tar i seg en direkte interaktiv forbindelse mellom forespørreren (klienten) og leverandøren (serveren). Leverandøren velger en utfordring, for eksempel ett element i en liste med en egenskap, og forespørreren finner det relevante svaret i listen, som deretter sendes tilbake og sjekkes av leverandøren. Siden utfordringen velges på stedet av leverandøren, kan vanskelighetsgraden tilpasses basert på nåværende belastning. Arbeidet på forespørrerens side kan være begrenset hvis challenge-response-protokollen har en kjent løsning (valgt av leverandøren), eller er kjent for å eksistere innenfor et begrenset søkeområde.



Figur 1 av Daniel Hao Huynh, basert på figurer fra"Proof of work. (2024). I Wikipedia".

Solution–verification-protokoller antar ikke en slik forbindelse: som et resultat må problemet være selvpålagt før en løsning søkes av forespørreren, og leverandøren må sjekke både problemvalget og den funne løsningen. De fleste slike ordninger er ubegrensede probabilistiske iterative prosedyrer som Hashcash.



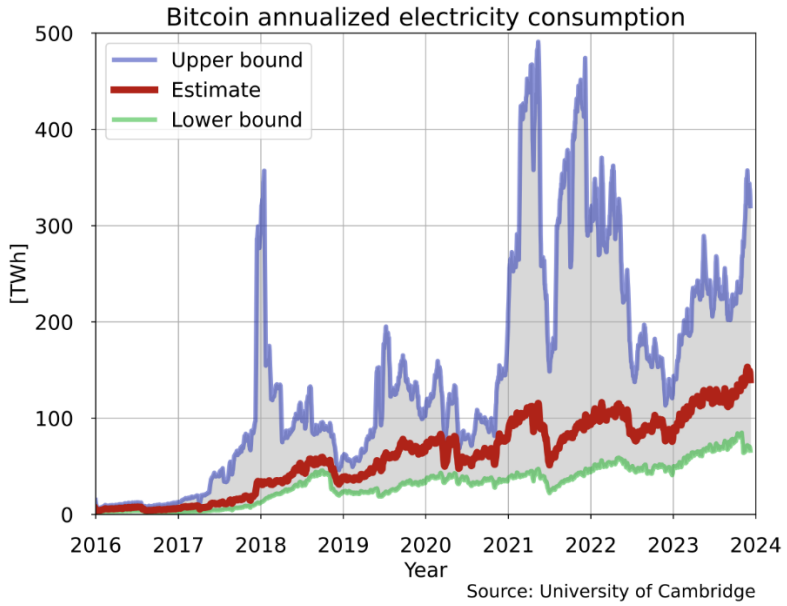
Figur 2 av Daniel Hao Huynh, basert på figurer fra"Proof of work. (2024). I Wikipedia".

Kjente-løsning-protokoller har en tendens til å ha litt lavere varians enn ubegrensede probabilistiske protokoller fordi variansen til en rektangulær fordeling er lavere enn variansen til en Poisson-fordeling (med samme gjennomsnitt). En generell teknikk for å redusere varians er å bruke flere uavhengige underutfordringer, da gjennomsnittet av flere prøver vil ha lavere varians. Det finnes også faste kostnadsfunksjoner som tidslås-puslespillet.

Videre kan de underliggende funksjonene som brukes av disse ordningene være:

- **CPU-bundet** hvor beregningen kjører med prosessorens hastighet, som varierer mye over tid, samt fra høyytelses servere til lavytelses bærbare enheter.
- **Minne-bundet** hvor beregningshastigheten er bundet av hovedminne-tilganger (enten latens eller båndbredde), hvor ytelsen forventes å være mindre følsom for maskinvareutvikling.
- **Nettverks-bundet** hvis klienten må utføre få beregninger, men må samle noen tokens fra eksterne servere før de spør den endelige tjenesteleverandøren. I denne forstand utføres ikke arbeidet faktisk av forespørreren, men det påløper forsinkelser uansett på grunn av latensen for å få de nødvendige tokene.

Til slutt tilbyr noen PoW-systemer snarveiberegninger som lar deltakere som kjenner en hemmelighet, vanligvis en privat nøkkel, generere billige PoWs. Begrunnelsen er at e-postlisteholdere kan generere frimerker for hver mottaker uten å pådra seg høye kostnader. Hvorvidt en slik funksjon er ønskelig, avhenger av bruksscenarioet.



Av Morn – Eget arbeid Datakilder:Cambridge Bitcoin Electricity Consumption Index, CC0, https://commons.wikimedia.org/w/index.php?curid=141942699

Konsensusmekanismer

PoW er en spesielt anerkjent konsensusmekanisme, brukt bl.a. Bitcoin-nettverket. E av deres største fordeler er sikkerhet. Selv om den er veldig sikker, har den som sagt tidligere høyt energi bruk, og er lite effektiv. Dette gjør at PoW har problemer med skalerbare, der det tar lang tid å validere transaksjoner og tilføye nye blokker.

Dermed kan man konkludere at basert på bruk, kan alternativer være vurderbart. Alternativer som mitigerer disse svakhetene. Proof of Stake, eller PoS er et populært alternativ som konsensusmekanisme. I stedet for å sette søkelys på løse matematiske problemer, sikrer PoS nettverket ved å satse en mengde med tokens for å validere, dette vil gjøre denne løsningen mindre energikrevende, men på sikt vil dette gjøre nettverket sentralisert, dette er et problem fordi dette gjør at løsningen er åpent for et såkalt 51% angrep. 51% angrep tilsier at når en part kontrollerer mer enn 50% av gruvearbeiderne (validator) i et nettverk kan flertallet manipulere blokkjeden, majoriteten er som har høyest sjanse for å bli valgt som validator. Selv om PoS har mekanismer mot angrep, er det satt i spørsmål om PoS er sikrere enn PoW.

Andre konsensusmekanismer:

1. **Delegated Proof of Stake (DPoS):**

   PoS-algoritmer insentiverer brukere til å bekrefte nettverksdata og sikre sikkerhet gjennom en prosess med innsats av sikkerhet. En iterasjon av konseptet kjent som Delegated Proof of Stake (DPoS) fungerer på lignende måte, men har en stemme- og delegasjonsmekanisme som gjør prosessen mer demokratisk.

   - Fordeler: Økt effektivitet og raskere transaksjoner.
   - Ulemper: Kan føre til sentralisering, da et lite antall delegater har stor makt.

2. **Proof of Authority (PoA):**

   PoA er energieffektivt og forårsaker minimal forsinkelse; det er mer passende for private nettverk. Sikkerhetstruslene mot nodene er alltid høye, da sikkerhetsangrep kan utføres på validatornoder, noe som resulterer i at de gradvis blir kilden til angrep i nettverket.

   - Fordeler: Høy effektivitet og lav energikostnad.
   - Ulemper: Krever tillit til et lite antall autoriteter, noe som kan redusere desentraliseringen

3. Proof of Useful Work (PoUW)

PoUW forsøker å bruke den samme datakraften til å utføre nyttig arbeid, som vitenskapelige beregninger eller dataanalyse, samtidig som det sikrer blokkjeden.

- Fordeler: Reduserer energisløsing ved å bruke datakraft til nyttige formål. Kan bidra til vitenskapelige og teknologiske fremskritt.
- Ulemper: Mer kompleks å implementere og kan ha utfordringer med å opprettholde samme sikkerhetsnivå som PoW.

Kilder:

Proof of work. (2024). I *Wikipedia*. https://en.wikipedia.org/w/index.php?title=Proof_of_work&oldid=1249618083

Proof-of-Work Alternatives: Analyzing Emerging Consensus Mechanisms. (u.å.). Hentet 13. oktober 2024, fra https://tokenminds.co/blog/knowledge-base/proof-of-work-alternatives

Jenssen, T. B., & Øverby, H. (2024). Bitcoin. I *Store norske leksikon*. https://snl.no/Bitcoin

Lie, K. S., & Øverby, H. (2024). Blokkjede. I *Store norske leksikon*. https://snl.no/blokkjede

Hva er Proof-of-Work (PoW)? (2024, august 30). *Cryptonews Norway*. https://no.cryptonews.com/akademiet/hva-er-proof-of-work/

Proof of Stake - hva er det? Les og lær om kryptomarkedet. (u.å.). *Kryptolisten*. Hentet 13. oktober 2024, fra https://kryptolisten.no/nyhet/hva-er-proof-of-stake/