Capitolo 1

Blockchain

Introduzione alla blockchain

Prima di partire con lo sviluppo di applicazioni per il Web3, è utile venire a conoscenza di cosa di intende con il termine *blockchain*.
La blockchain ha origine nel 2008, quando un individuo (o un gruppo di persone... a oggi ancora non si conosce chi ci sia dietro la sua invenzione) conosciuto con lo pseudonimo di Satoshi Nakamoto presentò per la prima volta un documento intitolato "Bitcoin: A Peer-to-Peer Electronic Cash System". Il documento è stato pubblicato su una mailing list di crittografia e descriveva un sistema di pagamento elettronico decentralizzato basato sulla tecnologia blockchain.
Come dice il termine stesso, questa tecnologia è basata su un'unica catena di blocchi, dove ogni blocco può contenere una o più transazioni *validate* e *immutabili* nel tempo. Questo perché ogni blocco è legato al precedente tramite un meccanismo di *hashing*, in cui una stringa di caratteri (chiamata appunto *hash*) viene generata a partire dalle informazioni contenute nel blocco precedente. Questo hash viene quindi inserito come parte dei dati del nuovo blocco (Figura 1.1).
Il meccanismo di hashing garantisce l'integrità dei dati contenuti nella blockchain, in quanto qualsiasi modifica apportata ai dati di un blocco causerà un cambiamento nell'hash generato, rendendo immediatamente evidente la modifica stessa. Inoltre, poiché ogni blocco contiene l'hash del blocco precedente, qualsiasi modifica apportata a un blocco renderebbe invalidi anche tutti i blocchi successivi, rendendo impossibile modificare i dati storici della blockchain.

In questo capitolo

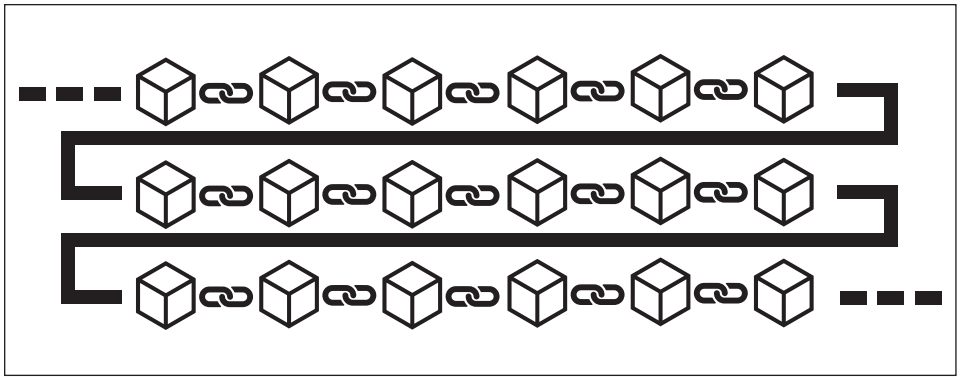

Figura 1.1 Rappresentazione schematica dei dati all'interno della blockchain.

Le entità che contribuiscono al funzionamento della blockchain sono le seguenti.

- Nodo: un computer o dispositivo elettronico che partecipa alla rete blockchain. Ogni nodo ha una copia completa e aggiornata della blockchain.
- Minatore: un nodo che si occupa di confermare le transazioni e aggiungerle alla blockchain. In cambio del suo operato, riceve una ricompensa in forma di criptovaluta. I minatori sono coloro che vengono utilizzati nei meccanismi di consenso di tipo PoW (*Proof of Work*).
- Validatori: hanno la stessa funzionalità dei minatori, in quanto contribuiscono a validare le transazioni e quindi ad aggiungere nuovi blocchi alla blockchain. A differenza dei minatori, però, utilizzano un meccanismo di consenso di tipo PoS (*Proof of Stake*). Anche i validatori vengono premiati attraverso una commissione in forma di criptovaluta.

Il *consenso* è il processo tramite il quale tutti i nodi della rete blockchain concordano sullo stato attuale della blockchain stessa. Esistono diversi tipi di meccanismi di consenso. I più noti e utilizzati sono: PoW, PoS e PoA.

PoW (Proof of Work)

In un sistema PoW, i partecipanti alla rete (noti come "minatori") competono per risolvere un complesso problema matematico utilizzando la potenza di calcolo delle loro macchine. Il primo minatore a risolvere il problema viene premiato con la creazione di un nuovo blocco e la ricezione di una commissione di transazione. PoW è stato utilizzato per la prima volta nella creazione di Bitcoin e attualmente è il sistema di consenso utilizzato da molte delle blockchain più diffuse.

PoS (Proof of Stake)

In un sistema PoS, i partecipanti alla rete (noti come "validatori") sono selezionati in base alla quantità di criptovaluta che detengono e sono disposti a "bloccare" (in gergo: mettere in *stake*) come garanzia per la validazione delle transazioni. I validatori che

effettuano la validazione delle transazioni vengono premiati con la ricezione di una commissione di transazione. PoS è considerato un'alternativa più efficiente a PoW dal punto di vista energetico.

NOTA

Una delle più famose blockchain, Ethereum, nata come PoW, ha iniziato un processo di migrazione nel 2018 denominato Serenity, che si è concluso il 19 settembre 2022 con la realizzazione di Ethereum 2.0 basata su PoS. Tutti i token ETH (il token proprietario di Ethereum) sono stati migrati sulla nuova chain, mentre si è deciso di mantenere attiva la vecchia, sulla quale, però, è stato generato un nuovo token (ETHW), di seguito airdroppato a tutti i possessori di ETH, in quantità pari a quanto detenuto alla data di switch.

PoA (Proof of Authority)

In un sistema PoA, i partecipanti alla rete fanno parte di un gruppo di autorità preselezionate che utilizzano le proprie identità digitali per firmare le transazioni che ritengono valide, garantendone l'inserimento all'interno della blockchain.
Data la sua natura centralizzata, il PoA viene utilizzato soprattutto in blockchain private, dove le autorità sono selezionate da un gruppo di entità che hanno interesse a mantenere il controllo sulla rete.

Funzionamento di una transazione sulla blockchain

Adesso vedremo tutto il processo di creazione di un nuovo blocco, dall'invio di una transazione da parte di un utente fino all'inserimento del blocco validato all'interno della catena.

1. Un utente inizia una transazione utilizzando un portafoglio digitale chiamato *wallet* inserendo le informazioni necessarie, tra le quali l'indirizzo del wallet del destinatario e la quantità di criptovaluta da inviare.
2. La transazione viene quindi inviata alla blockchain per essere verificata. Questo processo include la verifica della disponibilità delle risorse (per esempio, viene verificato il saldo dell'utente proprietario della transazione, per accertarsi che disponga effettivamente della quantità di criptovaluta da inviare al destinatario), la verifica della firma digitale per garantire che la transazione sia autorizzata dal mittente e la verifica dei requisiti di transazione (per esempio, i costi delle transazioni).
3. Se la transazione è valida, viene inserita in una coda insieme ad altre transazioni in attesa di essere elaborate. Tutte le transazioni vengono elaborate in ordine cronologico e confermate da un gruppo selezionato di nodi (minatori o validatori) a seconda del meccanismo di consenso adottato dalla blockchain.
4. Una volta confermata, una transazione viene inserita in un nuovo blocco insieme ad altre transazioni confermate. Il nuovo blocco viene quindi aggiunto alla catena di blocchi esistente, legato all'ultimo blocco attraverso un meccanismo di hashing.
5. A questo punto, il blocco si considera archiviato, la transazione (e tutte le transazioni in esso contenute) si considera completata e le risorse oggetto della transazione stessa risultano disponibili e aggiornate sui wallet dei rispettivi soggetti.

Da Bitcoin a Ethereum

Bitcoin è stato il primo esempio di blockchain della storia. L'idea di partenza che prevedeva la creazione di una moneta completamente decentralizzata di fatto ne limitava le potenzialità, in quanto su tale blockchain sono permesse solamente transazioni peer-to-peer e l'unico utilizzo che si fa della chain è quello dello scambio di moneta digitale tra due portafogli senza l'ausilio di intermediari terzi.
Partendo da questo concetto, un giovane programmatore russo-canadese di nome Vitalik Buterin decise di ampliare le potenzialità offerte da Bitcoin, immaginando una blockchain che permettesse, oltre lo scambio digitale, anche l'esecuzione di contratti intelligenti e su cui potessero essere emessi e potessero girare nuovi token (non soltanto la moneta nativa della chain).
Buterin ha quindi proposto una nuova piattaforma blockchain, chiamata Ethereum, che avrebbe superato le limitazioni di Bitcoin.
Nel 2014, Buterin e il suo team hanno lanciato una raccolta fondi in criptovaluta (ICO) per finanziare lo sviluppo della piattaforma Ethereum. La ICO ha raccolto 18 milioni di dollari in Ether, attirando l'attenzione di investitori e sviluppatori.
Nel luglio 2015, la piattaforma Ethereum è stata lanciata ufficialmente e ha iniziato a consentire ai programmatori di creare e utilizzare contratti intelligenti chiamati *smart contract* sulla sua blockchain. Grazie alla sua programmabilità, la chain di Ethereum è a oggi una delle piattaforme blockchain più famose e utilizzate al mondo. Tra le applicazioni che è possibile sviluppare vi sono le seguenti.

- Pagamenti automatizzati: è possibile creare contratti intelligenti che consentano pagamenti automatici a determinate condizioni.
- Token personalizzati: è possibile creare token personalizzati sulla blockchain di Ethereum che possono rappresentare qualsiasi cosa, come monete, beni, azioni o diritti. Un esempio è la criptovaluta SUSHI oppure la creazione di NFT (*Non Fungible Token*).
- Giochi decentralizzati: è possibile creare giochi decentralizzati che utilizzano contratti intelligenti per gestire le regole e i premi del gioco.
- Finanza decentralizzata (DeFi): Ethereum è diventata una delle principali piattaforme per la costruzione di applicazioni finanziarie decentralizzate, che consentono agli utenti di accedere a servizi finanziari tradizionali, come prestiti e investimenti, senza l'intervento di intermediari terzi.
- Identità digitale: è possibile creare sistemi di identità digitale basati su contratti intelligenti sulla blockchain di Ethereum che consentono agli utenti di controllare e gestire i propri dati personali in modo sicuro e privato.

DEFINIZIONE

Con il termine *smart contract* si intende un programma informatico auto eseguibile e auto applicante che si basa sulla tecnologia blockchain per facilitare, verificare e far rispettare l'esecuzione di accordi e transazioni tra le parti coinvolte, senza la necessità di intermediari centralizzati. Nel Capitolo 2 verrà preso in esame Solidity, il linguaggio maggiormente utilizzato per la programmazione di smart contract, mentre nel Capitolo 4 verrà trattato l'argomento in maniera approfondita, mostrando come realizzare i propri contratti.

NOTA

Mentre sulla blockchain di Bitcoin esiste un'unica criptovaluta chiamata come la chain stessa (BTC), sulla chain di Ethereum, grazie agli smart contract, possono essere creati nuovi token e nuove criptovalute. La moneta proprietaria di Ethereum è chiamata Ether (ETH) e viene utilizzata come carburante per l'esecuzione degli stessi smart contract.

Funzionamento di Ethereum

La blockchain di Ethereum è composta da una serie di blocchi, ognuno dei quali contenente una serie di transazioni e smart contract (Figura 1.2).

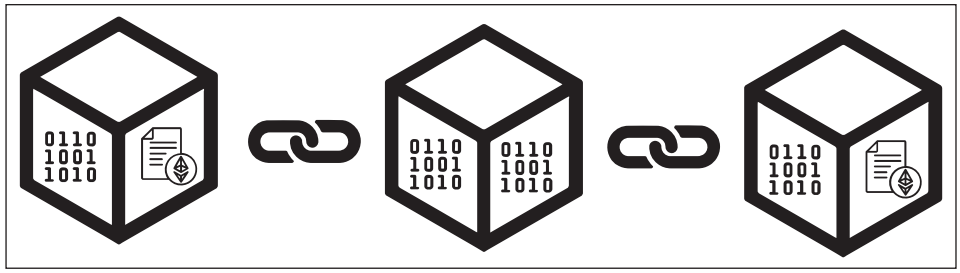

Figura 1.2 A differenza di Bitcoin, un nodo in Ethereum può contenere solo dati, oppure dati e smart contract.

Ogni nodo che partecipa alla blockchain possiede, inoltre, una EVM (*Ethereum Virtual Machine*), un software che simula un computer virtuale che permette di leggere e mandare in esecuzione i contratti presenti sulla chain.
Ogni volta che si interagisce con la funzione di uno smart contract, il codice del contratto viene caricato all'interno della EVM di ciascun nodo ed eseguito, poi il risultato viene restituito al chiamante. Nel caso in cui la funzione chiamata preveda un cambiamento all'interno della blockchain (come per esempio la scrittura di informazioni, oppure il trasferimento di fondi), la EVM si fa carico di calcolare le commissioni di transazione (che in gergo prendono il nome di *gas fee*) che l'utente dovrà pagare alla blockchain per eseguire quel determinato insieme di operazioni.
Le gas fee vengono normalmente pagate attraverso la criptovaluta nativa della blockchain di riferimento: nel caso di Ethereum, vengono pagate in Ether (ETH).
Diversi linguaggi di programmazione consentono di scrivere smart contract, e tra i più noti possiamo annoverare i seguenti.

- Solidity: è il linguaggio di programmazione principale e il più utilizzato. È un linguaggio di programmazione a oggetti simile a JavaScript e C++, progettato specificamente per la creazione di contratti intelligenti su blockchain.
- Vyper: è un linguaggio di programmazione simile a Python, progettato per la creazione di contratti intelligenti su Ethereum.Vyper è meno potente rispetto a Solidity ma offre maggiore sicurezza e semplicità, essendo meno suscettibile a errori.

- Bamboo: è un nuovo linguaggio di programmazione basato su Rust, progettato per la creazione di contratti intelligenti su Ethereum. Bamboo è ancora in fase di sviluppo, ma promette una maggiore sicurezza e più prestazioni rispetto a Solidity e Vyper.

Oltre a questi, è possibile, grazie all'ausilio di librerie di terze parti, sviluppare i propri smart contract utilizzando linguaggi di programmazione generali come JavaScript, Python, C++ e altro.
Una volta scritto il contratto con uno dei linguaggi sopracitati, prima di essere rilasciato sulla blockchain dovrà essere compilato. Il processo di compilazione procede a creare un *bytecode*, un linguaggio fatto di soli codici simile al linguaggio macchina, che può essere letto e interpretato dalla EVM.
Nel prosieguo di questo corso, impareremo a scrivere smart contract utilizzando come linguaggio Solidity. Oltre a questo, vedremo come sia possibile interfacciarsi a esso lato front-end, utilizzando JavaScript e la libreria web3.js, interrogando le funzionalità del contratto e costruendo su di esso una DApp (*Decentralized Application*, ovvero applicazione decentralizzata).

DEFINIZIONE

Con il termine *DApp* si intendono tutte le applicazioni che per il loro funzionamento comunicano e utilizzano le funzionalità di uno smart contract. Approfondiremo il loro significato e vedremo come poterle realizzare nel Capitolo 5.