



# Introduzione alla blockchain Ethereum, oracoli e applicazioni

Y2Y: BLOCKCHAIN E SMART CONTRACT

---

Organizzatori: Massimiliano Sala, Andrea Gangemi e Christopher Spennato

**Speaker: Andrea Gangemi**

9 e 16 Novembre, 2023

- **Concetti base di Ethereum**
  - Ethereum Virtual Machine (EVM)
  - Smart contracts
- **Address**
- **Transazioni**
- **Token**
  - Token fungibili: lo standard ERC-20
  - Token non fungibili: lo standard ERC-721
- **Oracoli**
- **Decentralized Finance (DeFi)**
- **DApps e DAO**
- **Protocollo di consenso**
- **Similitudini e differenze tra Bitcoin e Ethereum**

## Approfondimenti

- **Crittografia di Ethereum**
- **I rollup come soluzione di scalabilità**

# Ethereum

---

# Ethereum I

**Ethereum** è un'infrastruttura informatica *open source* decentralizzata, in grado di eseguire in modo rapido ed efficiente programmi chiamati **smart contracts**.

Ethereum è stata fondata da più persone nel 2015, ma l'ideatore principale è stato Vitalik Buterin (31 gennaio 1994).



Ethereum utilizza una blockchain per salvare i cambiamenti di stato del sistema (rappresentati dalle transazioni), insieme ad una crittomoneta chiamata **Ether**, utilizzata principalmente per comprare il **gas**. Il *gas* vincola i costi di esecuzione di una transazione.

A differenza di Bitcoin, quindi, la crittomoneta non viene principalmente usata per svolgere pagamenti, ma svolge una funzione di utilità per la blockchain stessa.

Esattamente come Bitcoin, Ethereum è una blockchain *permissionless*, con una sua criptomoneta e un suo protocollo di consenso, in questo caso basato sul paradigma della Proof of Stake.

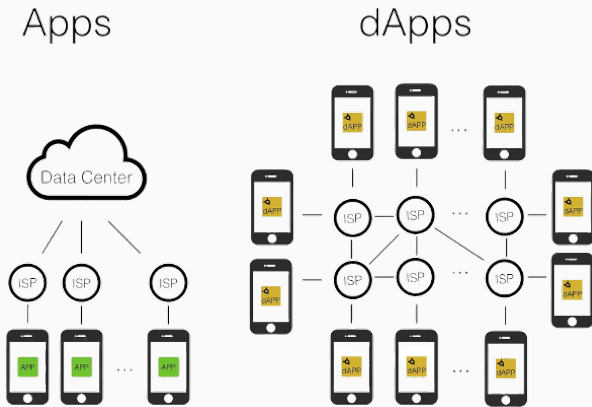
Ha una capitalizzazione di 193 miliardi di dollari, seconda criptomoneta dopo Bitcoin che sta a 511 miliardi di dollari (settembre 2023).

Tuttavia, come dicevamo, lo scopo principale di Ethereum non è gestire pagamenti tramite la criptomoneta, ma creare un ambiente decentralizzato all'interno del quale possono girare delle applicazioni che possono avere scopi molto vari.

L'ether è infatti soprattutto uno strumento per gestire quello che è sostanzialmente un **enorme computer globale**, le cui applicazioni una volta lanciate non possono essere bloccate.

# Ethereum IV

Ethereum permette, tra le altre cose, lo sviluppo di **Decentralized Applications** (DApps).



La piattaforma Ethereum fornisce una macchina virtuale decentralizzata, la **Ethereum Virtual Machine** (EVM), che esegue script utilizzando una rete di nodi pubblici.

EVM è una macchina virtuale, cioè un emulatore che può girare sul computer di qualsiasi nodo Ethereum, qualsiasi sistema operativo abbia, e permette di far girare delle applicazioni disegnate appositamente per l'ambiente EVM.



Bitcoin permette di eseguire semplici script, ma per motivi di sicurezza le operazioni eseguibili sono molto poche: per esempio, i *cicli for* non sono ammissibili.

La *Ethereum Virtual Machine* (EVM) è invece un ambiente di programmazione più avanzato, dove l'unico limite è la fantasia del programmatore.

Questo perché la EVM ha la proprietà di essere **Turing completa**. Detto in parole semplici, sono consentite tutte le operazioni di un linguaggio di programmazione tradizionale, quindi anche i *loops*.

**Come fa Ethereum a evitare l'intasamento della rete?**

# Il ruolo del gas I

Per evitare che la rete possa rimanere bloccata a eseguire un loop infinito, Ethereum adotta una soluzione molto semplice: uno smart contract, o più in generale una transazione, possono essere eseguiti fino a quando non viene consumato tutto il **gas**.

Ogni volta che si effettua una transazione, l'utente deve specificare l'ammontare massimo di *gas* che è disposto a spendere. Ogni operazione computazionale costa una certa quantità di *gas*. La transazione viene eseguita se e solo se l'ammontare di *gas* specificato è superiore a quello richiesto per l'esecuzione della transazione.

## Il ruolo del gas II

Il *gas* non può essere comprato 'in anticipo': esso viene convertito sul momento, a partire dall'ammontare di Ether specificato dall'utente come fee per la transazione.

La EVM non può quindi essere impegnata per troppo tempo mentre esegue uno smart contract, perché ad un certo punto finisce il *gas*, e la quantità massima di *gas* a disposizione dello smart contract deve essere specificata a priori dall'utente che lo attiva.

In questo modo, Ethereum evita attacchi di tipo Denial-of-Service.

# Smart contracts

---

# Smart contracts I

Uno **smart contract** è un programma informatico che facilita, verifica, o fa rispettare l'esecuzione di un contratto, che viene eseguito dall'*Ethereum Virtual Machine*.

A differenza di quello che indica il nome, essi non sono né *smart* (eseguono soltanto operazioni predefinite), né *contract* (non hanno alcun valore legale). Tuttavia, essi sono incredibilmente personalizzabili ed automatizzano moltissime funzioni che normalmente richiedono un input umano.



**vitalik.eth** ✓  
@VitalikButerin



To be clear, at this point I quite regret adopting the term "smart contracts". I should have called them something more boring and technical, perhaps something like "persistent scripts".

7:21 PM · Oct 13, 2018

Le caratteristiche fondamentali di uno smart contract sono:

- **Immutabilità:** dopo che è stato inserito nella EVM, il codice di uno smart contract non può essere modificato.
- **Determinismo:** l'esecuzione di uno smart contract è la stessa per ogni utente che lo utilizza, a parità di input.

L'idea degli smart contract, come abbiamo visto, risale a Szabo nel 1997. Tuttavia, è solo con l'avvento della blockchain che gli smart contracts sono diventati realtà.

L'idea è la seguente: un utente può inviare una transazione in cui chiede a uno smart contract di eseguire una certa operazione. Quest'ultimo la eseguirà, aggiornando lo stato della blockchain stessa.

Tutti i validatori, per verificare la correttezza della transazione, rieseguiranno l'operazione dello smart contract.

La blockchain è utilizzata proprio per garantire che tutti i nodi eseguano la medesima applicazione, e che i dati di input siano gli stessi, assicurandosi quindi indirettamente che anche l'output sarà il medesimo su tutti i nodi.

Inizialmente, esistevano vari linguaggi per programmare all'interno dell'ambiente EVM (ad esempio, LLL, Mutan, Serpent, oramai dismessi) per scrivere smart contract. Attualmente, i più popolari sono **Solidity** e **Vyper**.

Vedremo nelle prossime settimane esempi di programmazione Solidity.

Esempi di queste funzioni sono:

- Pagamento di rate;
- Clausole assicurative;
- Pagamento automatico di stipendi;
- e molto altro!



## **Address e Account**

---

Esattamente come per Bitcoin, non si utilizza direttamente la chiave pubblica come indirizzo a cui si trasferiscono i bitcoin, ma si usa il cosiddetto **indirizzo Ethereum**, che è più breve e viene derivato dalla chiave pubblica.

In questo caso, la chiave pubblica viene hashata con *Keccak256* (una funzione di hash, variante di SHA3). L'address consiste nei 20 bytes meno significativi del digest ottenuto. Infine, si ottiene un indirizzo Ethereum utilizzando la codifica proposta dallo standard [EIP55](#), che contiene al suo interno anche un *checksum*.

- A differenza di Bitcoin, che utilizza il modello UTXO, Ethereum è una blockchain *Account-based*.
- Un *account* Ethereum è paragonabile ad un account bancario: esso avrà un saldo, e se dobbiamo spostare degli ETH semplicemente spostiamo la porzione di crittomoneta necessaria. Non dobbiamo quindi "darci il resto", dato che la parte rimanente del nostro saldo banalmente rimane sull'account.
- Ogni account è associato a un indirizzo, che viene utilizzato per lo scambio di crittomoneta nelle transazioni che avvengono sulla blockchain.
- Una transazione Ethereum "base" coinvolgerà sempre un solo indirizzo (account) come input di una transazione, e sempre un solo indirizzo come output. Per svolgere transazioni più elaborate, bisogna scrivere degli smart contracts appositi.

# Externally Owned Account e Contract Account

- In realtà, su Ethereum esistono due tipi di account. Quelli più classici sono chiamati *Externally Owned Account* (EOA) e sono gli account controllati dagli utenti. Ad ogni account corrisponde un indirizzo che viene ottenuto a partire da una chiave privata, in modo analogo a Bitcoin.
- Esiste anche un secondo tipo di account, chiamato *Contract Account* (CA): come suggerisce il nome, questo account rappresenta uno smart contract e quindi presenta del codice. Un account EOA non può invece avere codice al suo interno.
- Ad un CA viene associato un indirizzo, ma questo indirizzo non è derivato da una chiave privata (vedere Slide 19 per la generazione di un *contract address*). **Questo significa che un CA non può mai iniziare una transazione, in quanto non è in grado di firmarla digitalmente.**

- Esattamente come per Bitcoin, gli ETH possono essere gestiti tramite un wallet. Esistono tanti tipi di wallet, che, come abbiamo già visto, possono essere suddivisi in due macrocategorie, cold wallet e hot wallet.
- L'Ether può essere suddiviso in unità di misura più piccole, e la più piccola possibile è il *wei*, che corrisponde a  $10^{-18}$  ETH.
- Dal punto di vista tecnico, la maggior parte dei wallet utilizzati da Ethereum utilizza gli stessi standard dei wallet Bitcoin.

Successivamente, vedremo più nel dettaglio come funziona il web wallet **MetaMask**.

# Transazioni

---

- Come abbiamo detto, una transazione Ethereum deve sempre essere iniziata da un EOA. Un CA può essere però usato come destinatario della transazione, cioè è in grado di *reagire* ad una transazione.
- Quando un CA viene chiamato in questo modo, esso può avviare una funzione presente al suo interno oppure può richiamare a sua volta un altro contratto. In ognuno di questi casi, si parla di **messaggio** (alcuni usano anche il termine **internal transaction**).
- Attenzione a non fare confusione tra transazione e messaggio: i due termini rappresentano concetti simili ma non sono utilizzabili come sinonimi quando si lavora su Ethereum.

- Una transazione Ethereum funziona in modo leggermente diverso rispetto ad una transazione Bitcoin. Siccome un account può inviare più di una transazione, ad ognuna viene associato un **nonce**.
- Esso è fondamentale in quanto serve per tenere traccia del numero di transazioni inviate sulla blockchain da un particolare indirizzo.
- Inoltre, il *nonce* tiene traccia dell'ordine delle transazioni e le rende, in un certo senso, "uniche".
- L'indirizzo di un contract account è calcolato in modo deterministico, sempre utilizzando la funzione di hash *Keccak256*, a partire dall'indirizzo dell'EOA che ha inviato la transazione, e dal *nonce* di quella stessa transazione.



# Creazione di uno smart contract

- Per creare uno smart contract, il mittente invia una transazione all'indirizzo "nullo", cioè l'indirizzo 0x000... . Esso riceverà quindi il suo indirizzo univoco.
- Per inviare ether a questo contratto, possiamo utilizzare un wallet qualsiasi, inserendo come destinatario il suo *contract address*.
- Gli smart contract possiedono però (di solito) anche delle funzioni. Per interagire con esse, bisogna creare un *frontend* apposito che permetta a un utente di interfacciarsi con il contratto, o utilizzare degli IDE specializzati.

- Ogni volta che si invia una transazione, vanno specificati due campi: *gasPrice* e *gasLimit*.
- Nel campo *gasPrice* specifichiamo quanto siamo intenzionati a spendere in Ether per ogni unità di *gas*. Il prezzo viene misurato in wei per unità di *gas*.
- Il prezzo del *gas* può essere cambiato a piacimento dell'utente: più è alto, più è probabile che la sua transazione venga inserita prima in un blocco.
- Nel campo *gasLimit* specifichiamo invece il massimo numero di unità di *gas* che siamo disposti a spendere per questa transazione.

# Token

---

Un **token** (gettone) rappresenta un valore o un diritto ed è nella sostanza come una moneta virtuale con un utilizzo specifico.

I token esistono anche nella vita di tutti i giorni: i buoni spesa, i bollini per i premi del supermercato, le fiches, i gettoni dell'autolavaggio.

I token in ambito crittografico sono il loro analogo digitale.

La parola token viene usata per significati diversi, che è bene chiarire. I token possono essere in effetti classificati in 3 macrocategorie:

- **Utility token:** danno diritto all'utilizzo di alcune features su una piattaforma, oppure a servizi premium;
- **Security token:** danno diritto di partecipazione al profitto della piattaforma che li ha emessi. Sono quindi l'analogo di una stock share per un'azienda tradizionale;
- **Payment token (coin):** rappresentano le crittomonete.

Facciamo un esempio di *utility token*.

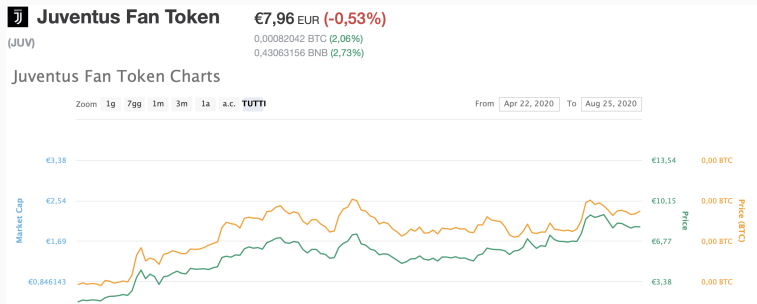
La Juventus è stata la prima squadra di calcio al mondo ad essere stata tokenizzata, creando un suo token (**JUV**), che può essere acquistato dai tifosi, che da diritto a sconti sul biglietto delle partite (carta fedeltà), o la partecipazione a particolari eventi.

Inoltre, cosa più interessante e innovativa permette di partecipare a delle votazioni.

Per esempio, i possessori di JUV hanno votato per il design della nuova maglia, o per decidere la canzone da utilizzare nello stadio quando la Juventus segna un gol.

# Utility token II

JUV non è una vera e propria criptovaluta, dato l'utilizzo limitato al mondo Juventus. E' comunque valutato su CoinMarketCap:



**ERC-20** (ERC sta per *Ethereum Request Comment*) è uno standard stabilito da Ethereum per scrivere gli smart contract relativi ai token fungibili, cioè token intercambiabili (ognuno ha lo stesso valore e la stessa funzione).

Prima della creazione dello standard, gli utenti dovevano riscrivere da capo il codice per la creazione di ogni token, con il conseguente carico di lavoro e rischi per la funzionalità e la sicurezza.



## Lo standard ERC-20 II

Lo standard ERC-20 presenta sei parametri obbligatori per ogni smart contract:

1. **totalSupply**: massimo numero di token che possono essere creati;
2. **balanceOf**: assegna un numero iniziale di token a qualsiasi indirizzo specificato, solitamente i creatori del token;
3. **transfer**: trasferisce i token a chi li acquista;
4. **transferFrom**: invia token da una persona all'altra;
5. **approve**: verifica che uno smart contract possa distribuire token, in base alla fornitura restante;
6. **allowance**: verifica che un indirizzo abbia un saldo sufficiente per inviare token ad un altro indirizzo.

## Lo standard ERC-20 III

La creazione dello standard ERC-20 ha accelerato la crescita delle applicazioni della blockchain all'economia e alla finanza.

La grande espansione e facilità a produrre token anche da parte di persone non molto esperte ha però ampliato anche il problema della vulnerabilità dei token, e per questo motivo sono stati ideati dei nuovi standard (ad esempio, **ERC-223** che ne migliora la sicurezza, **ERC-777** che elimina un bug...) che migliorassero il già esistente standard ERC-20.

Sono inoltre stati proposti alcuni nuovi standard che permettono di aumentare la funzionalità dello standard ERC-20.

## Lo standard ERC-721

L'esempio più importante è sicuramente dato dallo standard **ERC-721**, utilizzato per generare token *non fungibili* (NFT).

Questo nuovo standard è stato introdotto alla fine del 2017 da Dieter Shirley e permette di creare token che non sono tutti uguali e scambiabili l'uno con l'altro, come invece sono normalmente i token, o le normali monete fiat.

I token così costruiti non hanno quindi tutti lo stesso valore, ma il valore è deciso dal mercato dei singoli token.

Sono l'analogo nel mondo crypto del collezionismo del mondo reale.

Alcuni progetti di successo di tale tipo di token sono [CryptoKitties](#), [Decentraland](#) e [CryptoPunks](#).

# Cryptopunks I

CryptoPunks è un insieme di 10.000 personaggi digitali unici da collezione. Ogni elemento della collezione è un ritratto di  $24 \times 24$  pixel, con una serie di attributi e caratteristiche uniche.

Lanciato nel giugno 2017, CryptoPunks non costava nulla a chiunque avesse un wallet Ethereum (a parte le spese di commissione). Quattro anni dopo, durante l'esplosione degli NFT del 2021, i CryptoPunks hanno visto un significativo aumento del valore di mercato, e alcuni sono stati venduti per milioni di dollari.



# Cryptopunks II

I CryptoPunks sono stati inizialmente concepiti per essere utilizzati nei giochi e nelle app, ma il loro approccio alle immagini di ritratto ha ispirato molte altre collezioni di NFT come la famosa collezione **Bored Ape Yacht Club**.



I motivi per acquistare un NFT sono molteplici: dal collezionismo, all'appartenenza alla comunità Cryptopunk, o anche semplicemente per moda. L'idea è la stessa a chi colleziona qualche oggetto nella vita di tutti i giorni.

La seguente tabella confronta come cambierebbe l'internet con o senza gli NFT:

Internet con NFT	Internet senza NFT
I NFT sono digitalmente univoci	Una copia di un file è sempre uguale all'originale
Ogni NFT ha un proprietario (e tutti sanno chi è)	Gli articoli digitali sono memorizzati solo su server controllati da istituzioni
I NFT sono compatibili con qualsiasi applicazione basata su Ethereum	Un articolo digitale può essere venduto solo sulla propria piattaforma
I creatori possono conservare il diritto di proprietà sul proprio lavoro	Le piattaforme trattengono gran parte dei profitti delle vendite

Il mondo dei NFT è quindi relativamente nuovo, ma un token di questo tipo può essere utilizzato per qualsiasi cosa che necessiti di una proprietà dimostrabile.

Di solito, i NFT vengono creati nei seguenti contesti:

- opera d'arte digitale;
- videogiochi;
- collezioni digitali;
- biglietti per eventi;
- certificati o diritti d'autore.

# Oracoli

---



Gli oracoli sono servizi di terze parti che forniscono informazioni dall'esterno agli smart contract. Sono il **ponte tra le blockchain e il mondo esterno**.

Le blockchain e gli smart contract non possono accedere autonomamente a dati che si trovano al di fuori del network. È però fondamentale per molte applicazioni della blockchain avere informazioni dal mondo esterno, ed è qui che operano gli oracoli.

Un oracolo non è la fonte dei dati, è solo un servizio che recupera, verifica e autentica i dati da fonti esterne.

Gli oracoli blockchain possono essere classificati in base a:

- **Fonte:** i dati possono provenire sia da **software** (database, server, siti web, etc), sia da **hardware** (sensori elettronici, codici a barre, etc);
- **Direzione:** la maggior parte delle volte gli oracoli comunicano informazioni dal mondo esterno al mondo blockchain, ma anche il viceversa è tecnicamente possibile;
- **Fiducia:** gli oracoli possono ottenere le loro informazioni da fonti centralizzate, o da più complessi sistemi decentralizzati.

Un oracolo può rientrare in più categorie. Per esempio, un oracolo che fornisce informazioni alla blockchain provenienti dal sito di un'azienda è un oracolo software, in entrata e centralizzato.

Come abbiamo visto, gli smart contract sono dei software che non possono essere spenti né alterati, grazie alla blockchain, e quindi danno la certezza dell'esecuzione e del risultato.

Se però il risultato dello smart contract dipende da dati provenienti dall'esterno della blockchain, come ad esempio da un oracolo, c'è la possibilità di introdurre informazioni false o che possono compromettere l'output di uno smart contract.

Anche supponendo che la fonte esterna dalla quale provengono i dati non sia compromessa, bisogna comunque garantire una comunicazione sicura tra l'oracolo e la blockchain (questa tipologia di attacchi viene spesso denominata *man-in-the-middle*).

Per essere considerato sicuro, le caratteristiche che deve avere un oracolo blockchain sono:

- **Attendibilità:** i dati che fornisce non devono essere manipolabili;
- **Affidabilità:** un oracolo è considerato più affidabile se recupera le informazioni da più parti, invece che utilizzare una sola fonte;
- **Trasparenza:** tutti devono poter verificare la correttezza delle informazioni fornite;
- **Sicurezza:** l'oracolo deve soddisfare certe ipotesi di sicurezza per essere considerato sicuro rispetto agli attacchi sovramenzionati.

Alcuni utilizzi tipici di un oracolo blockchain sono:

- **Valore di un titolo:** ad esempio, prezzo di mercato delle criptovalute, obbligazioni, derivati, tassi di interesse, etc;
- **Rilevazioni per assicurazioni:** feed relativi ad un evento assicurabile, rilevazione su orario di treni/aerei, etc;
- **Supply chain:** rilevazioni da GPS su spedizioni, dati doganali, etc;
- **Sicurezza alimentare:** rilevazioni su temperatura di conservazione, tempi di trasporto, etc;
- **Sharing mobility:** rilevazioni sull'utilizzo dei mezzi di trasporto.

Gli oracoli sono uno dei pochi modi per introdurre **fonti di randomicità deterministiche** all'interno di una blockchain.

Uno dei progetti più interessanti e più di successo nell'ambito degli oracoli è **Chainlink**.

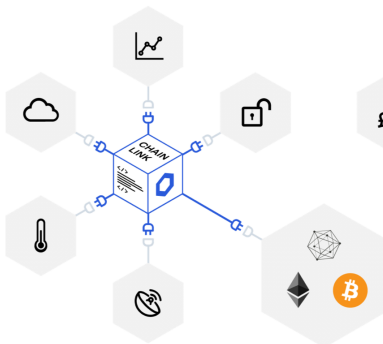
Chainlink è una piattaforma che lavora sulla rete Ethereum. Rileva dati da una rete di oracoli, che vengono incentivati e penalizzati con un sistema reputazionale. Si pone come uno dei più interessanti sistemi per creare un collegamento sicuro tra le blockchain e i dati che risiedono off chain.

Chainlink permette anche di collegare una blockchain a dei servizi di pagamento, grazie al collegamento con il servizio di pagamenti bancari **SWIFT**.

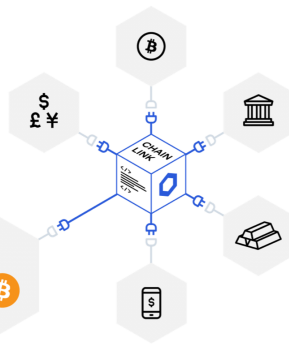
# Chainlink II




Connect to Any API



Send Payments Anywhere



Chainlink ha avuto un grande successo negli ultimi anni, e la sua criptovaluta (**Link**), con più di 3 miliardi di euro di capitalizzazione e un volume giornaliero di quasi 6 miliardi di euro, è oramai stabilmente in 20° posizione per capitalizzazione:

☆ 20	 Chainlink LINK	\$6.20	▲0.08%	▲2.75%	▼1.78%	\$3,335,591,491	\$128,241,743 20,713,594 LINK	538,099,970 LINK
------	--	--------	--------	--------	--------	-----------------	----------------------------------	------------------

Una delle ragioni del grande successo di Chainlink è la **Decentralized Finance**, spesso abbreviata in DeFi.



# DeFi e ICO

---

La **DeFi** è l'insieme dei nuovi strumenti finanziari basati su strumenti e reti decentralizzate (blockchain e criptomonete, soprattutto Ethereum), che ne sfruttano le varie opportunità (ad esempio, gli smart contract e i token).

La DeFi permette di ricreare tutti i tipici servizi finanziari del passato (chiedere un prestito, investire i propri capitali, fare una raccolta fondi per un investimento, etc), ma **senza il bisogno di un intermediario o di una autorità centrale**.

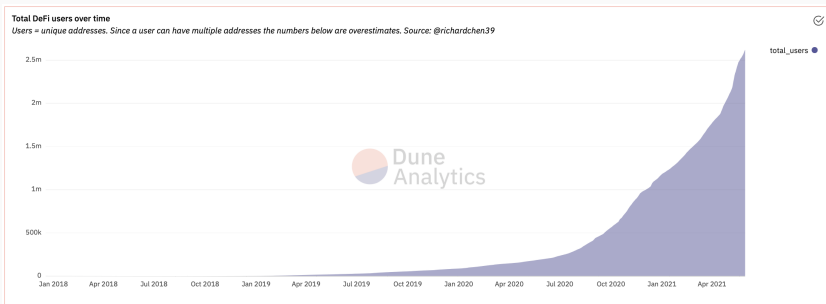
In questo nuovo paradigma non serve più la fiducia dell'utente nei confronti della banca o dell'operatore finanziario, e viceversa nel cliente da parte della banca, e la sicurezza dell'operazione è garantita dalla blockchain.

La blockchain viene utilizzata per creare un sistema sicuro che permetta di fare a meno della fiducia nell'operatore bancario o finanziario.

Nel mondo finanziario classico, chi ha i capitali li fornisce alla banca in cambio di un interesse, e la banca li presta a chi ne ha bisogno, a fronte di garanzie. La banca si fa pagare interessi e costo di intermediazione.

Nella DeFi, invece, chi ha i capitali e chi ha bisogno di finanziamenti **si accordano direttamente, saltando l'intermediario bancario o finanziario**, con il relativo risparmio (rimangono gli interessi, ma non c'è il costo dell'intermediazione). La certezza dell'operazione è garantita da uno smart contract su una blockchain.

Il numero dei progetti DeFi, e degli utenti che la utilizzano, è in continua espansione. Per esempio (dati fino al 2021), gli utenti sono passati da un numero trascurabile nel 2019, a oltre **due milioni e mezzo di utenti** nel maggio 2021.



La DeFi presenta anche dei rischi, dovuti soprattutto alla poca conoscenza delle caratteristiche della blockchain da parte degli utenti.

Uno dei fraintendimenti più ricorrenti, e gravi, riguardo la tecnologia blockchain riguarda la sua immutabilità. Non si possono modificare i dati inseriti in una blockchain, ma questo non dà **nessuna certezza sulla correttezza dei dati!**

Se in una blockchain vengono inseriti dati falsi, rimangono falsi e lo rimangono per sempre.

Affidare i propri risparmi a un sito di DeFi solo perché utilizza la blockchain è sbagliato e pericoloso. E' sempre necessario saper **valutare la serietà e l'affidabilità di un progetto** in cui si intende investire.

## DeFi e stable coins

Forse la più importante applicazione della DeFi sono le **stable coins**, ovvero crittomonete il cui valore rimane stabile rispetto alle monete fiat (come l'euro o il dollaro).

Esempi di stable coins sono Tether e USD Coin. Anche Facebook stava pensando di introdurre una stable coin, Diem (nota come Libra fino al 2020), ma il progetto è poi terminato con un nulla di fatto.

Nessuna di queste può essere considerata completamente decentralizzata, in quanto tutte prevedono il controllo parziale da parte di una struttura centrale (banche o industrie).

[MakerDAO](#) è il primo progetto nato sulla blockchain Ethereum ad avere ideato una crittomoneta stabile rispetto al dollaro americano, senza che nessuno entri mai in possesso del denaro fiat.

Il progetto presenta una stable coin, nota come Dai, ed un token, abbreviato in MKR.

I protagonisti principali della piattaforma sono i keepers e gli oracoli.

- Un **Keeper** è un nodo (quasi sempre un bot) che fornisce liquidità alla piattaforma in cambio di opportunità di profitto. Tra le varie cose, essi possono vendere o comprare Dai, oppure partecipare alle *aste di liquidazione*.

Chiunque può essere un keeper, ma è necessario essere un full node sulla blockchain Ethereum.

- Un **Price Oracle** è un'infrastruttura esterna che aggiorna la DAO continuamente rispetto alle variazioni dei prezzi delle crittomonete, come ad esempio il valore di mercato del Dai.

Gli oracoli sono scelti dagli utenti che possiedono MKR. Per evitare inconvenienti, vengono selezionati 13 oracoli e i prezzi sulla piattaforma Maker saranno uguali alla mediana dei prezzi proposti da tutti gli oracoli.

Per i paesi più poveri, MakerDAO risulta essere una alternativa molto utile per combattere vari tipi di problemi, ad esempio:

- Il Dai è una risorsa per le persone che vivono in paesi con gravi crisi inflazionistiche, come l'Argentina o la Bolivia.
- Il Dai è la soluzione per i paesi piccoli e spesso dimenticati, come le isole Vanuatu. Gli abitanti di questi posti sperduti devono pagare enormi commissioni per convertire la loro moneta in un'altra moneta (come il dollaro americano).

Il Dai è stato anche utilizzato con successo per avviare alcune startup.



La seguente tabella confronta la DeFi con la finanza tradizionale:

DeFi	Finanza tradizionale
L'utente gestisce il proprio denaro	Una compagnia gestisce il nostro denaro
L'utente controlla dove e come spendere il proprio denaro	L'utente deve fidarsi della compagnia su come essa gestisce il denaro
Il trasferimento dei fondi avviene in pochi minuti	Il trasferimento dei fondi può richiedere fino ad alcuni giorni
Le transazioni sono pseudoanonime	L'attività finanziaria è strettamente collegata all'identità della persona
Chiunque può entrare nel mondo DeFi	Bisogna fare richiesta per usare i servizi finanziari
Il mercato è sempre aperto	Il mercato non è sempre aperto

# Initial Coin Offering I

ICO è l'acronimo di **Initial Coin Offering**, ed è un modo per fare raccolta fondi saltando la banca o gli operatori finanziari, andando direttamente a raccogliere liquidità (in criptomoneta) dalle persone.

Per alcuni aspetti è simile alla **IPO (Initial Public Offering)**, con la quale le società fanno ingresso in Borsa ed aumentano il loro capitale, quotandosi nel mercato regolamentato, aprendosi ad un pubblico più ampio di investitori.

In generale, un progetto che intende finanziarsi tramite ICO segue il seguente modello:

- Scrittura del whitepaper;
- Soft cap: specifica dell'importo minimo per partire;
- Hard cap: specifica dell'importo massimo necessario;
- Emissione dei token, che devono necessariamente essere security token.

# Initial Coin Offering II

Nel caso delle ICO, ciò che viene offerto al mercato è un token (**security token**), che può rappresentare solo un diritto ad avere una remunerazione proporzionale al successo della start-up, oppure essere a tutti gli effetti una stock share, rappresentando quindi una quota di proprietà con l'annesso diritto di voto sulle decisioni gestionali.

Un altro classico utilizzo delle ICO è **il lancio sul mercato di una nuova criptovaluta**.

In tal caso, si configura come una sorta di vendita di una nuova criptovaluta a un determinato prezzo, che viene acquistata da chi ritiene che il progetto che sta dietro sia di prospettiva, aspettandosi quindi **un aumento di valore nel tempo**.

# Security Token Offering I

Le **STO (Security Token Offering)** sono simili alle ICO, ma non è detto che siano un'offerta iniziale di una start-up. Infatti, possono essere effettuate anche da aziende già esistenti e, soprattutto, riguardano esclusivamente i **security token**, cioè quei token che danno diritto di partecipazione al profitto della piattaforma che li ha emessi.

Il rapporto tra le parti è regolato da un contratto che disciplina le regole ed i diritti dell'investitore. Naturalmente le STO sono **soggette alle normative relative all'emissione di strumenti finanziari** e, pertanto, vi è il controllo da parte delle autorità centrali.

Di solito, per garantire l'investimento e trovare acquirenti, si collegano i token a asset di beni di proprietà dell'azienda (immobili o beni, azioni, obbligazioni, diritti, etc). Si parla in questo caso di **tokenizzazione**.

Gli asset dell'azienda vengono 'inseriti' in uno smart contract e digitalizzati attraverso un token che ne garantisce autenticità mediante la blockchain.

Con tale procedura, la società **rende liquidi i propri asset favorendo l'investimento da parte di soggetti interessati**, senza dover ricorrere al sistema bancario.

# Dapps e DAOs

---

# Decentralized applications I

Una **DApp (Decentralized Application)** è una applicazione decentralizzata, cioè una applicazione in esecuzione su una rete peer-to-peer decentralizzata, non necessariamente appoggiata alla tecnologia blockchain.

Al contrario, una classica web-application è utilizzata da tanti utenti in luoghi remoti, ma l'applicazione gira sul server centrale. Di conseguenza, tutti i dati sono salvati sul server.

In un sistema decentralizzato, invece, ogni nodo ha la stessa importanza e non esiste un'entità centrale con poteri di decisione.

# Decentralized applications II

L'architettura di una DApp che si appoggia ad una blockchain, esattamente come ogni tipo di App, può essere suddivisa in *backend* e *frontend*.

- La parte di *backend* è rappresentata dagli smart contracts. Essi vengono validati all'interno della *Ethereum Virtual Machine* (EVM), da qua l'aggettivo "decentralized".
- Gli smart contracts vengono di solito programmati in Solidity, testati in *JavaScript* e infine *deployati* sulla EVM.
- La parte di *frontend* è più classica ed è sostanzialmente identica al *frontend* di una qualsiasi applicazione.
- Il *framework* più utilizzato è *React*, una libreria *open source* sempre basata su *JavaScript*.



# Decentralized Autonomous Organization I

Una **DAO - Decentralized Autonomous Organization** - è una organizzazione la cui attività ed il cui potere esecutivo sono ottenuti e gestiti attraverso regole codificate in una serie di smart contract.

Attraverso l'uso di smart contract, una DAO può lavorare con informazioni esterne ed eseguire comandi basati su di esse, tutto ciò senza alcun intervento umano.

Diversamente dalle organizzazioni tradizionali che funzionano secondo una struttura gerarchica e diversi livelli di burocrazia, le DAO non hanno alcuna gerarchia.

**Le DAO utilizzano meccanismi economici per allineare gli interessi dell'organizzazione con gli interessi dei suoi membri.**

**Bitcoin potrebbe essere considerato il primo esempio di DAO ante litteram.**

In effetti, Bitcoin opera in modo decentralizzato ed è coordinato da un protocollo di consenso senza gerarchia tra i partecipanti.

Il protocollo di Bitcoin definisce le regole dell'organizzazione, mentre i bitcoin come moneta forniscono agli utenti un incentivo per proteggere il network.

Questo garantisce che i vari partecipanti possano lavorare insieme per mantenere Bitcoin in funzione come una sorta di organizzazione autonoma decentralizzata.

Uno dei primi veri esempi di DAO è stato **“The DAO”** (2016).

The DAO era costituito da smart contract della blockchain di Ethereum che avrebbero dovuto agire come un venture fund autonomo. Ha avuto un grande successo e **raccolse oltre 150 milioni di dollari**.

I token DAO sono stati venduti in una ICO e fornivano una quota di partecipazione e diritti di voto in questo fondo decentralizzato. Tuttavia, poco dopo il lancio, circa un terzo dei fondi è stato “rubato” in uno dei **più grandi hack nella storia delle criptovalute**.

# Decentralized Autonomous Organization IV

Questo incidente ha acceso un intenso dibattito: si doveva annullare l'operazione di chi aveva sfruttato una debolezza degli smart contract di The DAO, oppure si doveva andare avanti come se nulla fosse (“**code is law**”)?

Non si trovò un accordo, e questo portò a una *hard fork* di Ethereum.

Nell'attuale Ethereum, l'*hack* di The DAO non è mai avvenuto. Invece, il ramo che ha mantenuto l'immutabilità e considera valida tale *hack* ha preso il nome di **Ethereum Classic**.

# Decentralized Autonomous Organization V

La seguente tabella confronta una DAO con una organizzazione tradizionale:

DAO	Organizzazione tradizionale
Solitamente orizzontale	Solitamente gerarchica
L'implementazione di qualsiasi modifica è soggetta a una votazione dei membri	Spesso le modifiche sono decise dalle persone con cariche più importanti
Il conteggio dei voti si effettua senza intermediari	I voti sono conteggiati internamente e manualmente
Tutti i servizi offerti sono decentralizzati e automatizzati	I servizi offerti richiedono la gestione umana
Tutte le attività sono trasparenti e pubbliche	L'attività è di solito privata e quindi limitata al pubblico.

# Protocollo di consenso

---

Ethereum, diversamente da Bitcoin, utilizza la Proof-of-Stake (PoS) come protocollo di consenso. Questa modifica è stata effettuata in data 15 Settembre 2022.

Il cambio di protocollo di consenso ha sicuramente alcuni benefici rispetto alla Proof of Work, tra i quali:

- La velocità di inserimento di un blocco è costante ed è fissata a 12 secondi;
- Il consumo energetico è decisamente minore: si stima che sia ridotto del 99.95% rispetto a quello di un protocollo PoW.

L'algoritmo utilizzato da Ethereum è chiamato [Gasper protocol](#) ed è un protocollo "ibrido", nel senso che utilizza un meccanismo simile a quello della PoW per gestire le *fork*.

# Proof of Stake (PoS)

Ricordiamo velocemente come funziona il protocollo Proof of Stake.

Il principio è che i blocchi vengano inseriti tenendo conto della quantità di criptovaluta posseduta dal **validatore del blocco**, con l'idea che coloro che detengono grandi quantità di valuta abbiano interesse a far funzionare il sistema.

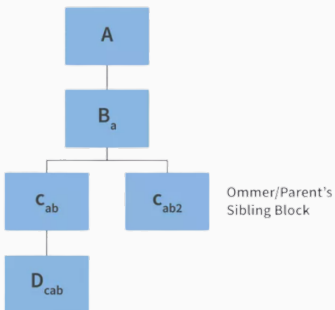
Per diventare un validatore Ethereum, è necessario bloccare nello stake almeno 32 ETH. A questo punto, i validatori vengono divisi in gruppi (chiamati *comitati*), e ogni comitato si occupa di inserire i blocchi in una certa *epoca* (uno slot temporale predefinito).



Su Ethereum, i blocchi validi che però non fanno parte della catena principale vengono chiamati *uncle blocks* oppure *ommer blocks*.

- A differenza di Bitcoin, su Ethereum gli ommer blocks non vengono scartati. I validatori Ethereum sono parzialmente ricompensati per la creazione di questi blocchi.
- Questi blocchi sono quindi in un certo senso parte della blockchain, ma le transazioni contenute al loro interno vengono ignorate, per evitare conflitti con quelle inserite nei blocchi della catena principale.
- Dopo ogni epoca, si conta il numeri di ommer blocks trovati e si assegna una ricompensa al miner che lo ha prodotto.

# Blocco ommer: esempio



## Fork e PoS: come?

La domanda sorge spontanea: se un validatore viene sorteggiato per inserire il blocco, e solo lui ha quel potere, come fanno a nascere *fork* all'interno di un protocollo PoS?

In effetti, se tutti seguono il protocollo non accadrà mai alcuna *fork*. Le *fork* accadono in questi due casi:

- Un validatore viene scelto ma è offline e quindi non inserisce il blocco;
- Un validatore inserisce il blocco ma per qualche motivo si propaga lentamente nella rete e non arriva a tutti in 12 secondi.

# Confronto tra Bitcoin ed Ethereum

---

# Similitudini e differenze tra Bitcoin ed Ethereum I

Per quanto riguarda l'aspetto più finanziario, alcune tra le analogie e le differenze tra Bitcoin ed Ethereum sono:

- Entrambe hanno una crittomoneta, che ha tuttavia uno scopo differente;
- Bitcoin metterà a regime in circolazione un tetto massimo di 21 milioni di BTC, mentre Ethereum non ha invece per ora un tetto massimo;
- Bitcoin utilizza la Proof of Work, mentre Ethereum, a partire dal 15 settembre 2022, si affida alla Proof of Stake;

# Similitudini e differenze tra Bitcoin ed Ethereum II

Per quanto riguarda l'aspetto più tecnico, alcune tra le analogie e le differenze tra Bitcoin ed Ethereum sono:

- Entrambe utilizzano la crittografia su curve ellittiche;
- Entrambe sono basate su una blockchain *permissionless*;
- Bitcoin processa un blocco in media ogni 10 minuti, mentre Ethereum ogni esattamente 12 secondi;
- Il linguaggio di programmazione di Ethereum è *Turing completo*, a differenza di quello di Bitcoin.

# Difetti e limiti di Ethereum

Esattamente come Bitcoin, anche Ethereum ha delle caratteristiche contestate di cui tenere conto:

- La blockchain cresce di dimensioni più in fretta di quella di Bitcoin, visto l'inserimento più assiduo dei blocchi. Al momento (Settembre 2023) occupa più di 1.2 Terabyte di spazio;
- Il network è spesso congestionato, e per questo motivo le commissioni di transazione sono spesso molto alte (in certi momenti si parla di più di 20 euro per transazione);
- Il linguaggio di programmazione per gli smart contracts è in costante evoluzione, ed è quindi possibile imbattersi in *bugs*.

Una soluzione per migliorare la scalabilità di Ethereum è data dai **rollups**.

# **Approfondimento crittografico**

---



# Crittografia di Ethereum I

Esattamente come Bitcoin, Ethereum utilizza la curva ellittica [Secp256k1](#) per la generazione delle chiavi pubbliche.

Ricordiamo che:

- **Curva** :  $y^2 = x^3 + 7$ .
- **Campo** :  $\mathbb{Z}_q$  con  $q = 2^{256} - 2^{32} - 2^9 - 2^8 - 2^7 - 2^6 - 2^4 - 1$ .
- **Punto base** :  $G = (x, y)$ ,  $x =$   
55066263022277343669578718895168534326250603453777594175500187360389116729240  
 $y =$   
32670510020758816978083085130507043184471273380659243275938904335757337482424
- **Numero di punti della curva** :  $n =$   
115792089237316195423570985008687907852837564279074904382605163141518161494337

$q, x, y$  e  $n$  sono a 256 bit, circa 78 cifre decimali.

Quando Bob vuole creare una coppia chiave privata/chiave pubblica, sceglie semplicemente un valore intero molto grande  $d_B$  (chiave privata) in modo casuale, e quindi calcola la chiave pubblica corrispondente  $P_B = d_B G = G + G + \dots + G$  ( $d_B$  volte).

La chiave pubblica non è un numero intero, ma un punto sulla curva, cioè una coppia di valori  $(x, y)$  che verifica l'equazione  $y^2 = x^3 + 7$  modulo  $q$ .

Teniamo a mente che è possibile calcolare rapidamente la chiave pubblica a partire dalla chiave privata, ma il contrario è un problema molto difficile, noto come problema del logaritmo discreto sulle curve ellittiche (ECDLP).

## **Approfondimento: i rollup come soluzione di scalabilità**

---

Un **rollup** è un protocollo *layer 2* (esattamente come il *Lightning Network* per Bitcoin) che si pone come obiettivo l'aumento del numero di transazioni procettate dalla blockchain di Ethereum.

L'idea consiste nel processare le transazioni off-chain, riducendo quindi la computazione svolta on-chain. La sicurezza deriva dal fatto che i risultati finali di queste transazioni vengono comunque memorizzati on-chain.

Esistono due tipi di rollups: gli **zero-knowledge** rollups e gli **optimistic rollups**.

# Rollups II

Più in dettaglio, un rollup funziona nel seguente modo:

- L'utente deposita in uno smart contract la commissione, che verrà convertita in token utilizzabili sul *layer 2 protocol*. Questi token sono utilizzati per eseguire transazioni sul *layer 2*;
- L'utente invia la propria transazione ad un **aggregatore**, nuovamente utilizzando uno smart contract;
- L'aggregatore processa un insieme di transazioni che ha ricevuto, le esegue, e pubblica on chain i **rollup data**, cioè una rappresentazione dei dati in forma compressa.
- I rollup data sono pubblici e immutabili.

Questo approccio permette in primis una riduzione delle commissioni per gli utenti che inviano le transazioni singole.

In un *optimistic rollup*, le transazioni sono raggruppate in grandi lotti.

Il rollup è *ottimistico* in quanto utilizza l'assunzione che tutte le transazioni inserite in un lotto siano transazioni valide. Non viene quindi pubblicata on-chain una *prova di validità* del lotto.

Un rollup di questo tipo migliora la scalabilità dell'intera Blockchain fino a 100 volte.

# Optimistic Rollups II

I rollup ottimistici si basano su uno **schema di verifica delle frodi** per individuare i casi in cui le transazioni non sono calcolate correttamente. Dopo che un lotto di rollup è stato inviato su Ethereum, c'è una finestra temporale durante la quale chiunque può contestare i risultati di una transazione rollup (**prova di frode**).

Se la prova di frode ha successo, il protocollo rollup riesegue la transazione (o le transazioni) e aggiorna di conseguenza lo stato del rollup. Il sequenziatore responsabile dell'inclusione della transazione non correttamente eseguita riceve una penalità.

Se il lotto non è invece contestato, è considerato valido e accettato su Ethereum.

Ogni rollup è gestito da degli smart contracts della blockchain Ethereum. Le transazioni sono processate off-chain, ma il lotto viene poi inviato a uno smart contract, che memorizza on-chain lo stato del rollup.

La fase di processing off-chain avviene su una macchina virtuale che non è la EVM, ma che funziona in modo molto simile.

I rollup ottimistici sono considerati una soluzione di scaling ibrida perchè derivano la loro sicurezza da quella della blockchain (per esempio, lo stato finale dopo il *processing* di ogni lotto è sempre memorizzato on-chain).



Gli *zero-knowledge rollups* (spesso abbreviati in ZK rollups) funzionano in maniera molto simile ai rollups ottimistici, ma con una differenza.

Infatti, gli aggregatori di un ZK rollup inviano anche una proof che lo stato sia stato calcolato correttamente. Questa proof si chiama **validity proof** e viene creata utilizzando una famiglia di protocolli crittografici nota come **zk-SNARKs** (acronimo di *Zero-Knowledge Succinct Non-interactive ARguments of Knowledge*).

Una zk-SNARK consente di generare una proof "concisa" che contiene al suo interno l'informazione su un segreto (*witness*). Inoltre, tale proof può essere verificata molto più rapidamente del tempo richiesto per la sua generazione.

La parte "ZK" aggiunge un'ulteriore caratteristica: la proof non rivela al verifier nessun dettaglio sul witness, ma solo che esso è stato utilizzato per generare la proof.

In poche parole, **Zero-knowledge** significa che, se l'affermazione è vera, l'unica informazione che un verifier disonesto (nel nostro caso, un verificatore dello smart contract) può carpire è che l'affermazione sia vera.

Forse la proprietà più importante, che rende le SNARKs utilizzabili con una blockchain, è la **non interattività**. L'aggregatore può generare una proof e inviarla ad uno smart contract senza avere bisogno di comunicare con i verificatori.

Viceversa, il difetto più grande di una SNARK è che, molto spesso (ma non sempre), è necessario un **trusted setup** prima di potere utilizzare l'algoritmo.

Questo va un po' contro la filosofia di una blockchain, che è stata progettata per funzionare senza avere la necessità di sapere che ogni partecipante sia onesto.

# Applicazioni per la blockchain

Più in generale, in ambito blockchain, questi protocolli hanno due applicazioni molto utili:

- **Scalabilità:** per esempio, se un blocco richiede molto tempo per la verifica, un solo utente può verificarlo e generare una proof, che tutto il resto della rete può verificare in modo molto più rapido.
- **Privacy:** per esempio, si può dimostrare di avere il diritto di trasferire un bene (lo si è ricevuto e non lo si è già trasferito, quindi è ancora presente nell'UTXO set) senza rivelare di quale UTXO si sta parlando. In questo modo, non si rivelano informazioni su chi sta effettuando transazioni con chi.

**I rollup utilizzano le SNARKs per migliorare la scalabilità della blockchain.**

Infatti, l'aspetto cruciale di un rollup è la compressione dei dati da inviare alla blockchain, in modo tale da memorizzare meno informazioni. Questo è reso possibile da una SNARK grazie alla sua proprietà di succinctness.

L'adozione della tecnologia rollup è ancora bassa. Alcuni motivi potrebbero essere i seguenti:

- La tecnologia è molto moderna (l'idea è solo del 2019) e non è quindi stata studiata o testata a sufficienza;
- I rollup non sono al momento decentralizzati, almeno non completamente.