WHITEPAPER

Cloudflare Ethereum Gateway

INDEX

Disclaimer

The architecture and respective behavior of Cloudflare Ethereum Gateway as described in this document is what is intended for general availability (GA). As this document was written prior to GA, it is possible that some of the technical details, architecture, features, or decisions on what is released may change prior to GA.

Overview

Ethereum is a decentralized, open source, programmable blockchain. A blockchain is a shared, distributed database that stores data in blocks that are ultimately linked together using cryptography, making verification easy and transparent.

There is no central authority within an Ethereum network. The nodes in an Ethereum network communicate with each other via a peer-to-peer (P2P) network, and the application is verified and distributed by every node. Ethereum can be used to deploy decentralized applications (DApps) and employs a trustless model in which users are not explicitly required to trust a central authority; thus, Ethereum can be seen as a key enabler of Web3, which incorporates concepts of decentralization for the Web.

Similar to how the Interplanetary File System (IPFS) is seen as a storage layer for Web3 as discussed in the **Cloudflare IPFS Gateway Whitepaper**, Ethereum can be seen as a compute layer for Web3. This whitepaper discusses the what and why of Ethereum and DApps, how Ethereum helps bring the decentralized application model to the decentralized web, and the benefits of Cloudflare Ethereum Gateway.

The need for DApps

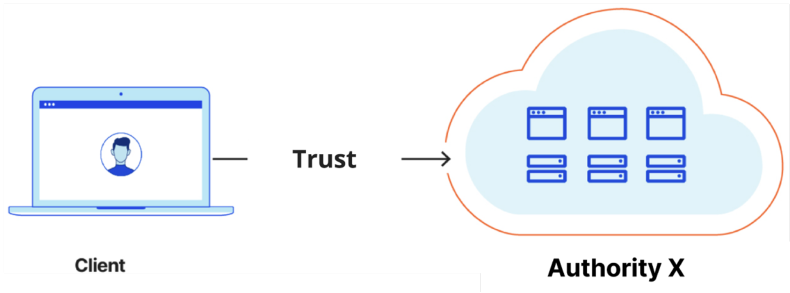

Figure 1: Centralized application model

Applications deployed on the web today are centrally hosted, usually controlled by a single authority. These applications are deployed in private clouds on-premise, in colocation facilities managed by third parties, or on public cloud environments. In this model, the central authority operations could impact all aspects of the application. Users are locked in with the central authority in deciding whether or not to use its respective system or application. This is the fundamental aspect of how the current Web 2.0 model and respective applications work today. Figure 1 above shows a representation of centralized applications.

A few key items to note of the conventional centralized application model:

- End users must explicitly trust the centralized authority and respective application
- The central authority has full control of the the application or system and respective components
- The application's policies for behavior, transactions, results, and disputes are not fully transparent unless the central authority explicitly makes it so, and any dispute involves interaction with and judgment by this central authority

A DApp has backend code running on a decentralized P2P network. The frontend user interface can be written in familiar, popular languages for user interaction and to access the backend.

With DApps, such as those that can be deployed on the Ethereum blockchain, there are some key tenets:

1. No requirement to trust a central authority
2. Independence from any central authority and transparency in application behavior, transactions, and results (the independence and transparency depend on all components of the application avoiding any reliance on a central authority)
3. Immutability and verification are implicitly built in

Decentralized applications, once deployed, do not depend on interaction with a central entity for the purposes of execution, verification, or disputes. The application resides on every node and behavior is governed by a set of rules and policies that are fully transparent. Figure 2 shows a representation of decentralized applications.

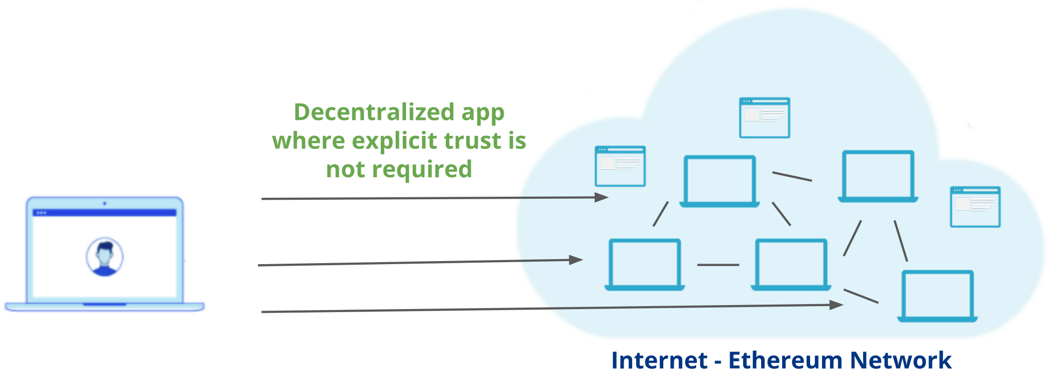

Figure 2: Decentralized application model

Understanding Ethereum

Ethereum is an application platform that can function as a compute layer for Web3. Both IPFS and Ethereum shift away from the centralized model of the web to a decentralized model where explicit trust of a central authority is not required and verification is built into the system. This document covers some foundational aspects of blockchain and Ethereum in order to provide an understanding of the benefits of Cloudflare Ethereum Gateway and how it fits into the broader ecosystem. For additional details and to better understand specific aspects of the Ethereum blockchain, see the **Ethereum developer website**, which has detailed documentation and tutorials

Ethereum can be envisioned as a network of decentralized interconnected nodes that are able to communicate with each other using the same protocols for executing transactions, verification, and coming to consensus on the state of the collective environment.

The foundation of Ethereum is blockchain technology. The basics of blockchain are critical to understanding Ethereum and DApps. In the following section, blockchain is discussed from its initial application for digital currency, also known as cryptocurrency (most often associated with Bitcoin), to its evolution of being used as a programming platform for DApps (most often associated with Ethereum).

Blockchain

Blockchain is a shared, distributed database that stores data in blocks that are ultimately linked together using cryptography, thus making verification easy and transparent. A block consists of data and state stored in consecutive groups known as blocks.

The term "chain" refers to the fact that each block cryptographically references its parent — thus, blocks are cryptographically chained together. Data in a block cannot be changed without changing all the subsequent blocks; this is due to the fact that the blocks are chained together. Changing all subsequent blocks is extremely difficult because of the work involved and because it would require consensus of the network. Consensus is another core aspect of blockchain and will be discussed in more detail later in this section; in brief, it can be seen as an agreement between nodes on the P2P network.

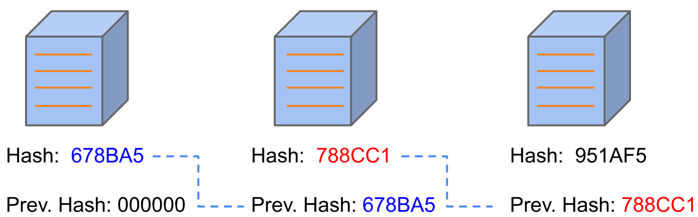

Figure 3: Blockchain: each block references hash of previous block

Blockchain has traditionally been used to record transactions between people or entities. The transactions are recorded in a distributed ledger by every node on the blockchain's P2P network; the ledger entries are immutable and inherently verifiable via cryptographic hashing.

Bitcoin was the first time blockchain was widely introduced and used, with the use case being digital currency. Digital currencies or cryptocurrencies have had a major impact on society and financial markets with a combined market cap of over one trillion dollars as of the time of this writing. Since the introduction of Bitcoin, many different blockchains focused on different use cases and with varying features and capabilities have been introduced, although the foundational aspects of blockchain remain the same.

Introducing the Ethereum blockchain

As mentioned prior, blockchain technology is an enabler of DApps and Web3 due to its characteristics of decentralization, trustlessness, immutability, verification, and transparency.

The focus and application for the Bitcoin blockchain is financial transactions. Ethereum expands upon the foundation and use cases of the Bitcoin blockchain by providing a blockchain that can store and run code — a programmable blockchain.

There are many resources for diving deeper into the inner workings of blockchain and Ethereum, and a deep dive into these subjects is outside the scope of this document. However, there are some important core concepts that should be understood, discussed in the following sections.

Ethereum nodes

In the Ethereum network, all nodes communicate with each other through a P2P network. There are a number of protocols and subprotocols involved in Ethereum node discovery and communication. In short, nodes discover each other over UDP/IP, and the follow-up communication of sending messages and blocks happens over TCP/IP using the Ethereum Wire Protocol (eth/66 is the current version at time of writing). This protocol facilitates the sending and receiving of blocks and respective blockchain information between peers.

Although all nodes communicate with each other, not all nodes perform the same functions. Nodes have the entire Ethereum blockchain downloaded and participate in block validation to verify all blocks and states. A "miner" is a node that is used to create blocks in the blockchain. This process in which new blocks are created is called "mining." Miners process blocks and other nodes verify that the transactions included in a block by a miner are valid. As miners are nodes, miners both validate transactions and add new blocks to the blockchain.

Consensus

Every node in the Ethereum network validates each new block and the resulting final chain. Eventually every node on the network has the same data and agreement on the blockchain state. This agreement between nodes on the network is referred to as consensus. The method of consensus depends on the consensus protocol used. Consensus is critical to the blockchain to ensure the correct information is recorded and to prevent malicious behavior and attacks.

At the time of this writing, the mining method of the consensus protocol used by the Ethereum blockchain is Proof-of-Work (PoW), which is often referred to as the consensus protocol for blockchain. PoW requires miner nodes to find a solution to a cryptographic hash puzzle that involves significant compute resources. Solving the puzzle involves hashing together random numbers via trial and error to discover the solution.

PoW is resource-intensive in regards to both hardware and electricity. To address these resource concerns, there are plans in the near future to move to a different consensus mechanism for Ethereum called Proof-of-Stake (PoS). PoS is not as resource-intensive as PoW because validators are not competing with each other; they are selected at random from a subset of nodes having met certain criteria. As such, mining is not required; the selected node creates the respective block when chosen and, when not selected, validates other blocks.

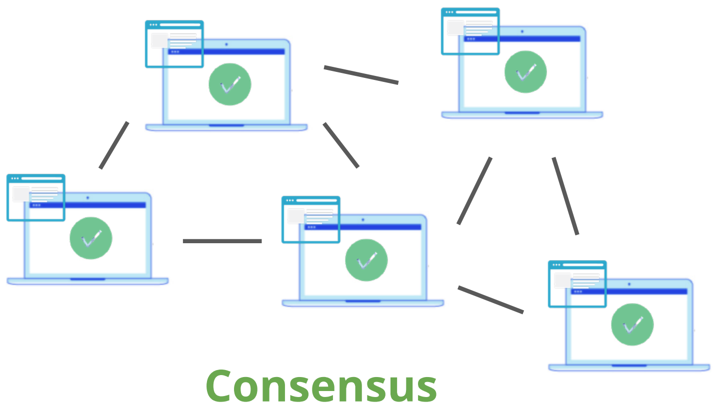

Figure 4: Consensus on Ethereum network

Ethereum Virtual Machine

The Ethereum Virtual Machine (EVM) is a virtual computer whose state is agreed upon and stored by each node of the Ethereum network. Any node on the network can broadcast a request for nodes on the network to perform an operation that will be verified, validated, and executed. The execution of the requested operation results in a state change in the EVM. The new state is then committed and propagated throughout the network.

Smart Contract

”Smart contract” is the term for code that resides on the Ethereum blockchain and can be executed by any node on the network. Smart contracts are used to perform operations like the transfer of digital assets or recording information on the blockchain. There are usually some conditions enforced by the smart contract that have to be met for the operation to be successful; the logic and conditions are fully transparent.

Solidity is the programming language most often used to write Ethereum smart contracts, and it compiles down to EVM bytecode. DApps can consist of a single or multiple smart contracts. The EVM executes smart contracts.

Ether

The work required to add blocks to the blockchain is not free. Miners are paid in Ethereum's native cryptocurrency called Ether (ETH). This provides incentive for the miners to do work and is typically referred to as a “gas fee” paid for by the requestor. Another positive impact of this fee is that it prevents attackers from making unnecessary and malicious requests.

Accounts

Ether is stored in an account. There are two types of accounts: 1.) contract accounts and 2.) externally owned accounts.

Contract accounts are created when a smart contract is deployed to the Ethereum network. Contract accounts are controlled by code. There is no associated public/private key pair, and a smart contract cannot sign transactions or messages.

External accounts are controlled by anyone with the private keys. There is a public and private key associated with an external account; the private key is used to digitally sign transactions. The private key is also used to create the public key and Ethereum address or account to which Ether can be sent.

The EVM state contains all accounts and account balances.

Transaction

A “transaction” refers to code execution on the EVM. A transaction can consist of sending/receiving Ether, publishing a smart contract, and executing code within a smart contract. Accounts initiate and cryptographically sign transactions using their respective private keys, which results in an update to the state of the Ethereum network.

Miners group transactions together into blocks, and these blocks are executed and added to the blockchain.

Gas

Transactions on the Ethereum blockchain require work. Miners have to add new blocks and respective transactions to the blockchain. The cost of the work is paid through Ether and varies based on the complexity and respective computational work required; this cost in Ethereum is referred to as Gas.

Wallet

A wallet is a frontend to a user's Ethereum account and allows users to interact with their account (for example, transferring Ether to or from the user's account).

When a user transfers Ether, deploys a contract, or executes a method in the contract, an Ethereum transaction is actually being created. This Ethereum transaction is signed with the user's private key and contains several properties: the transaction hash, receiver address information, value, etc. The entire process for adding a transaction, from submitting a transaction, mining, and consensus to receiving a reply, can take anywhere from 15 seconds to hours.

To access a DApp on the Ethereum network, users need to have a Web3 client or an Ethereum-enabled browser such as Metamask. Metamask provides a plugin for mainstream browsers like Chrome. Metamask is also a crypto wallet that can be used to store, send, and receive Ether.

DApp benefits

The components of the Ethereum blockchain discussed above allow for customers to develop DApps that have some important characteristics:

Decentralization/availability

Transactions are not processed by a central authority that has full control of the application and respective process. Instead, the transaction is processed by a node on the P2P blockchain network and all participants in the network independently verify it. The architecture shifts from a server host-centric model to a more decentralized distributed model (P2P network). This decentralization provides for 1.) independence from any central authority, and 2.) inherent high availability, as the application data is available across all nodes on the network

Figure 5 and Figure 6 below show the difference between the current centralized model (server-based) and the decentralized model (P2P).

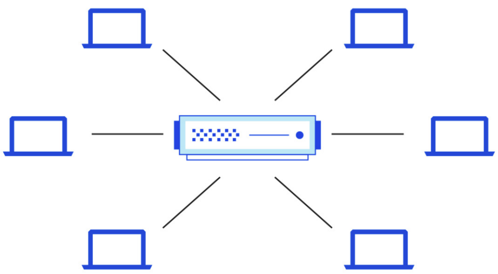

Figure 5: Centralized model (server-based)

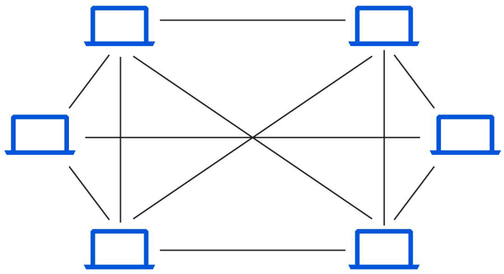

Figure 6: Decentralized model (P2P)

Trustlessness

Unlike a traditional app, there is no requirement to explicitly trust a single authority. The contract and consensus mechanism ensures behavior and conditions. Note that the trustlessness is in respect to the decentralized application on the Ethereum blockchain and deterministic results.

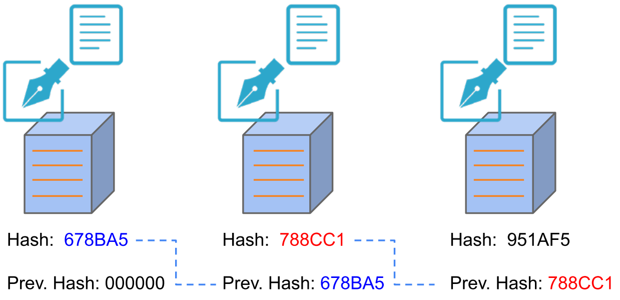

Figure 7: Contract and consensus mechanism ensures behavior and conditions

Verification

Every transaction and block is uniquely identified via cryptographic hash and stored on the blockchain, allowing for easy verification and auditing.

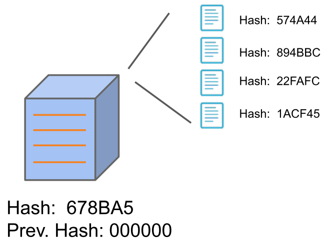

Figure 8: Blocks and respective transactions can be easily verified

Immutability

Transactions are immutable. One of the core aspects of blockchain is that transactions and data cannot be modified retroactively. Any change would require changing all subsequent blocks, which is difficult due to the work and consensus that would be required. Since blocks are verified by every node, if one of the nodes tried to maliciously alter an outcome and propagate the result to other nodes, the other nodes would detect the malicious behavior during their respective validation and reject the malicious transaction.

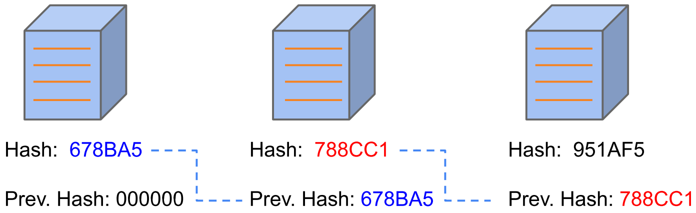

Figure 9: Contract and consensus mechanism ensures behavior and conditions

Transparency

Application logic is transparent; execution and result is solely determined by code.

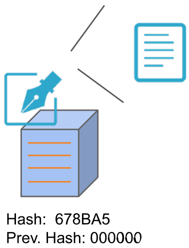

Figure 10: Blocks and respective transactions can be easily verified

Since DApps do not have any central authority processing and intervening in a transaction, a transaction's outcome is solely based on the logic of the smart contract, which is transparent for everyone to see, and the validation by the nodes on the Ethereum network.

Why Cloudflare Ethereum Gateway?

Companies have adopted and deployed DApps for a variety of use cases such as authenticity tracking, lending, gaming, and NFT marketplaces, to name a few. But although there is tremendous interest and momentum, DApps are still in the early phases of adoption with new use cases being discovered.

Challenges of deploying Ethereum nodes

To deploy DApps today using the Ethereum blockchain, users need to have an Ethereum node deployed to make API calls to for accessing the Ethereum blockchain and deploying DApps. There are several challenges users and developers face here:

- Deploying an Ethereum node can be time consuming and requires understanding of infrastructure details to set up effectively
- Configuration of Ethereum nodes is challenging and error-prone
- Deploying full Ethereum nodes is resource-intensive — in addition to certain hardware requirements and the electricity required to keep notes online continuously, the entire Ethereum blockchain must be downloaded (for full nodes this is currently over 1 TB and growing); additionally, bandwidth usage per node can be several terabytes per month
- Users/developers must maintain the Ethereum nodes
- Users/developers are responsible for securing the Ethereum nodes and addressing any security updates

Introducing Cloudflare Ethereum Gateway

Cloudflare Ethereum Gateway makes deploying and accessing DApps and the respective Ethereum blockchain easier. The Ethereum Gateway provides a standard JSON RPC API over HTTP for making requests to the Ethereum network.

Cloudflare Ethereum Gateway can be seen as a bridge between Web 2.0 and Web3: it allows for access to the Ethereum network over HTTP, and users and developers alike are no longer required to maintain and monitor their own nodes.

With Cloudflare Ethereum Gateway, customers can start leveraging the Ethereum blockchain and make the Ethereum network accessible via HTTP over their own domain. Combined with the **Cloudflare IPFS Gateway**, customers have decentralized web and resource hosting with all the benefits of the Cloudflare global network — performance, reliability, and security.

Cloudflare Ethereum Gateway capabilities

Cloudflare Ethereum Gateway provides the following capabilities:

1.) Easily access the Ethereum network
Customers can easily access the Ethereum network for read and write operations to the Ethereum blockchain without having to deploy and secure their own Ethereum nodes. Cloudflare Ethereum Gateway acts as an API endpoint and offers the security, reliability, and performance benefits of the Cloudflare platform. Cloudflare Ethereum Gateway provides an easily accessible API endpoint over HTTP for the Ethereum network.

2.) Interact with the Ethereum blockchain through custom domain names
Cloudflare Ethereum Gateway allows customers to use their own domain names; JSON RPC queries over HTTP can be sent to custom domain names.

3.) Leverage CDN for Ethereum content caching for performance and reliability
Cloudflare Ethereum Gateway acts as a cache in front of the Ethereum network. When using Cloudflare Ethereum Gateway, customers benefit from the Cloudflare CDN, which can cache Ethereum content close to users for low latency and improved overall performance. The results of prior requests will be cached for a short period of time for increased performance.

4.) Single pane of glass for Ethereum Gateway and Cloudflare security, reliability, and performance capabilities
Customers can use Cloudflare Ethereum Gateway and manage their full security model and additional Cloudflare reliability and performance capabilities through a single pane of glass.

Benefits of Cloudflare Ethereum Gateway

1.) Ease of Access: An easy way to access the Ethereum network that does not require installing and running any special software.

2.) Security: Since users are not installing any software but rather leveraging the Ethereum gateway, the burden of security is moved to Cloudflare, which leverages its global Anycast infrastructure to provide **enhanced security**.

3.) No maintenance/monitoring: Since Cloudflare is providing a gateway to the Ethereum network, there is nothing for customers to maintain. Cloudflare maintains and monitors security, reliability, and performance.

4.) Reliability: Cloudflare's global Anycast network provides a high level of **reliability and availability.**

5.) Performance: The Cloudflare Ethereum Gateway benefits from Cloudflare's enhanced security, reliability, availability, and performance capabilities. Cloudflare's network consists of **data centers in over 100+ countries.**

With Ethereum content caching, content can be served from the data center closest to the user.

Cloudflare Ethereum Gateway architecture and design

Cloudflare Ethereum Gateway provides an HTTP-accessible interface to the Ethereum network, thus allowing users to easily deploy applications and read or write to the Ethereum blockchain. The below diagram depicts the Cloudflare Ethereum Gateway architecture.

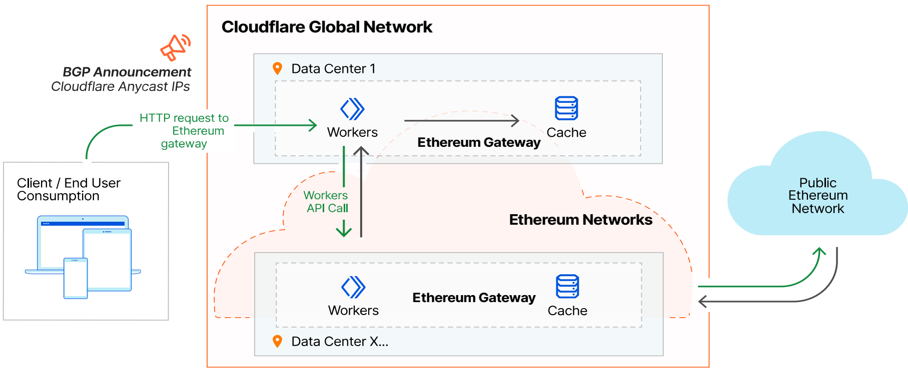

Figure 11: Cloudflare Ethereum Gateway

There are several important items to note:

- Within the DNS for the respective domain, a CNAME will automatically be created that points to ethereum.cloudflare.com.
- The Ethereum Gateway can accept both read and write requests via **JSON RPC API calls.** Each request contains a valid method parameter and respective params for the method. Below is an example to read the block that is at number 0×2424. Find more details and supported methods **here.**

  curl https://cloudflare-eth.com -H ‘Content-Type: application/json’ --data ‘{“jsonrpc”:”2.0”,”method”:”eth_getBlockByNumber”,”params”:[“0×2424”, true],”id”:1}’

- The Ethereum Gateway has a Cloudflare Workers code component that runs on servers globally across Cloudflare data centers. These data centers are located across 100+ countries, reaching 95% of the Internet-connected population globally within **50 ms while providing 100 Tbps of network capacity and DDoS protection.**
- Cloudflare runs Ethereum nodes protected by Cloudflare network and security capabilities. The Cloudflare Workers code makes API calls to the Cloudflare Ethereum nodes, which in turn access their respective Ethereum blockchain database.
- Cloudflare Ethereum nodes peer with Ethereum nodes on the public network; the public Ethereum network can be reached as needed. Cloudflare operates full Ethereum nodes. For any modification to the blockchain, Cloudflare nodes place transactions in the local mempool and propagate them; miners on the public Ethereum network execute transactions from the mempool and add new blocks to the blockchain.
- Since Cloudflare’s network leverages Anycast, users will always access the data center closest to them. Additionally, the Anycast network ensures high availability and automatically reroutes to another data center upon any issue.

Traffic flow

Figure 12 below describes the traffic flow from a DApp perspective from the client to the Ethereum network via Cloudflare Ethereum Gateway. A few important points to note:

- Using a Web3-enabled client, users access a site and perform an operation that makes a JSON RPC API call to a customer's custom domain that is using Cloudflare Ethereum Gateway.
- The CNAME record in the customer's DNS resolves to the Cloudflare Ethereum Gateway. The JSON RPC API call is sent over HTTP to Cloudflare Ethereum Gateway and is received by the closest Cloudflare data center to the user.
- A Cloudflare Workers script receives the API call. If the API call is a read operation and requested content is cached, the Workers script will respond via HTTP to the client with the requested information. If the requested content is not cached, it will first be requested via API call to Cloudflare Ethereum nodes, cached at the edge, and returned via HTTP response to the client.
- If the API call received is a transaction or write operation, the Ethereum Gateway will make an API call to the Cloudflare Ethereum nodes, and the transaction is placed in the local mempool and propagated to peers. A transaction ID is returned to the Ethereum Gateway, which is then returned to the client via HTTP response.
- Miner nodes take transactions from the mempool and place them into a block to execute.
- The new block to add to the blockchain is validated, consensus is reached, and the block is added to the blockchain and propagated to the rest of the network.

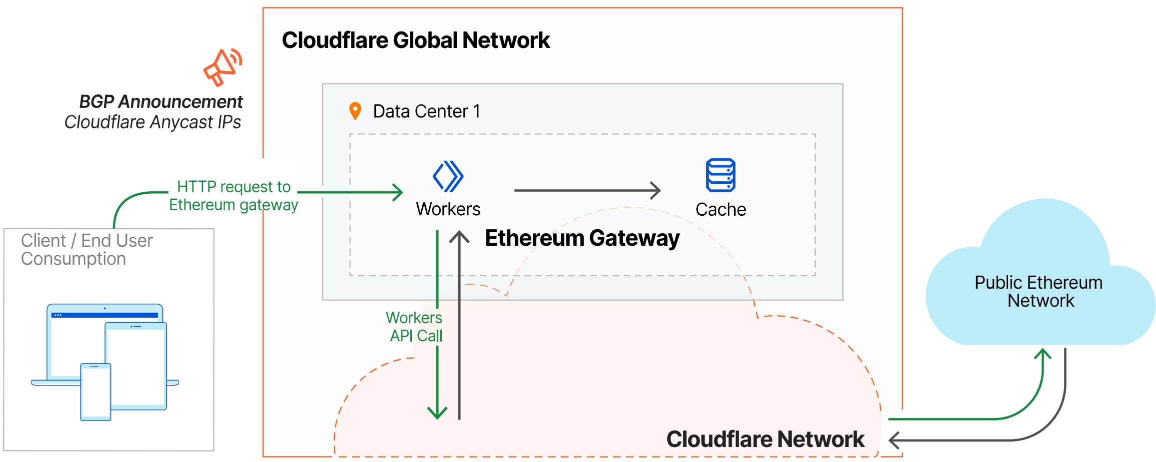

Figure 12: Traffic flow for Cloudflare Ethereum Gateway

Caching DApp content

Cloudflare leverages its global network for caching Ethereum API responses closest to the user, improving performance. As shown in the above diagram, Workers caches the Ethereum API responses, although this is only applicable for read API calls.

Figure 13 below demonstrates the traffic flow when the same API request comes in and the response is cached. Workers does not need to make another API call to the Cloudflare Ethereum nodes, because the response to a prior API call is already cached; this cuts latency and improves overall performance.

Cloudflare Ethereum Gateway leverages the Workers API to write to the Cloudflare cache. Cloudflare caches various API calls differently and has a targeted cache policy to best serve each Ethereum call. The cache is partitioned per block height for RPC calls with a block parameter. The worker leverages multiple Ethereum nodes to get the latest state of the Ethereum blockchain. If the information is not present in cache, the request is routed to a node that is able to fulfill the request.

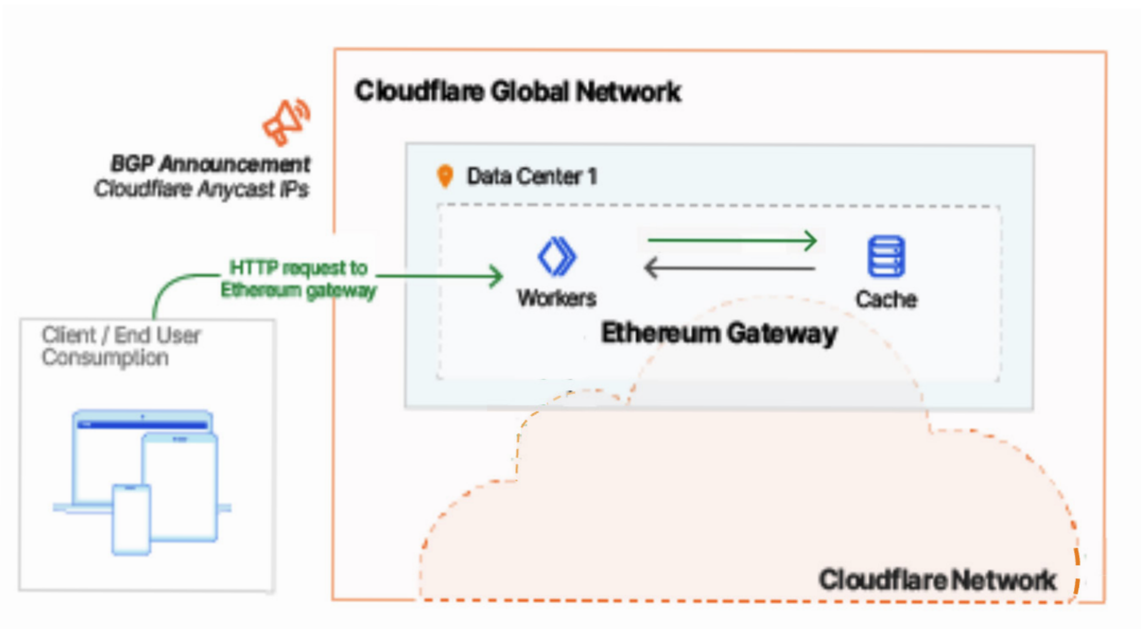

Figure 13: Cloudflare Ethereum Gateway using local cache on second request

Summary

Ethereum provides a compute layer for Web3. Its underlying blockchain technology provides several benefits, enabling decentralized hosting and a trustless model with inherent verification. Cloudflare Ethereum Gateway bridges between Web 2.0 and Web3 by employing an Ethereum Gateway accessible via HTTP interface, allowing users to easily access the Ethereum network over HTTP via their own custom domains. With Cloudflare Ethereum Gateway, customers do not need to deploy and monitor Ethereum nodes in order to start deploying DApps with Ethereum. Additionally, customers benefit from Cloudflare's reliability, security, and performance capabilities.