



IPFS Gateway

Cloudflare's read-only Distributed Web Gateway lets you access content stored on the InterPlanetary File System (IPFS) quickly and easily, without downloading any special software or giving up any storage space on your computer.

Cloudflare's gateway is hosted at <https://cloudflare-ipfs.com/> . You can find the basics on IPFS and [how to serve your website](#) through our gateway there. These docs are going to get further into the weeds.

To improve our service, the IPFS Gateway is now available in Private Beta. You can [register](#)  to get notified when the service opens to a wider audience.



Refresher on IPFS

You're a Client. And a Server.

Every single computer that's running IPFS acts as both a client and a server. In other words, each computer running the IPFS software can serve content to any other computer in the network, as well as request content from anyone in the network. So if you run IPFS on your computer and upload a picture to the IPFS network, that image can be viewed and downloaded by anyone else in the world who is also running IPFS.




Content Identifiers

Every file added to IPFS is given a unique address derived from a hash of the file's content. This address is called a Content Identifier (CID) and it combines the hash of the file and a unique identifier for the hash algorithm used into a single string.


IPFS currently uses [SHA-256](#)  by default, which produces a 256 bit (32 byte) output, and that output is encoded with [Base58](#) . Base58 is a binary-to-text encoding scheme originally developed for Bitcoin and has the advantage that letters that might be mistaken for each other in certain fonts (like zero and the capital letter O) aren't included.

A CID will typically look something like this:

```
QmXoybizjW3WknFiJnKLwHCnL72vedxjQkDDP1mXWo6uco
```

However, the same hash could be encoded with [Base32](#)  and there are a slew of other supported hash algorithms including [SHA-3](#)  and [BLAKE2](#) .

Uploading to IPFS

IPFS is fundamentally a [Distributed Hash Table \(DHT\)](#)  which maps from CIDs to people who have the content addressed by that CID. The hash table is distributed because no single node in the network holds the whole thing. Instead, each node stores a subset of the hash table, as well as information about which nodes are storing other relevant sections.

When someone talks about 'uploading' content to IPFS, what they really mean (usually) is that they're announcing to the network that they have some content by adding an entry to the DHT that maps from CID to their IP address. Somebody else who wants to download their data would lookup the CID in the DHT, find the person's IP address, and download the data directly from them.

The speed and reliability advantages of IPFS come from the fact that many people can upload the same data, and then downloads will be spread between all of them. If any one of them goes offline or decides to stop hosting the data, the others can pick up the slack.

Directories

Note that it's not just individual files that can be uploaded. For example, let's take a folder called `example`, which has exactly one file, `example_text.txt`, containing the string `I'm trying out IPFS`.

If that folder were uploaded with the command `ipfs add -r ./example`, both the folder and the file it contains would have their own CID. In this case, the folder would have the CID `QmdbaSQbGU6Wo9i5LyWWVLuU8g6WrYpWh2K4Li4QuuE8Fr` while the file would have the CID `QmXnnyufdzAWL5CqZ2RnSNgPbvCc1ALT73s6epPrRnZ1Xy`.

You could then access the file in two ways.

1. Requesting the file directly:

<https://cloudflare-ipfs.com/ipfs/QmXnnyufdzAWL5CqZ2RnSNgPbvCc1ALT73s6epPrRnZ1Xy> 

2. Requesting the file by name, from the directory:

[https://cloudflare-](https://cloudflare-ipfs.com/ipfs/QmdbaSQbGU6Wo9i5LyWWVLuU8g6WrYpWh2K4Li4QuuE8Fr/example_text.txt)

[ipfs.com/ipfs/QmdbaSQbGU6Wo9i5LyWWVLuU8g6WrYpWh2K4Li4QuuE8Fr/example_text.txt](https://cloudflare-ipfs.com/ipfs/QmdbaSQbGU6Wo9i5LyWWVLuU8g6WrYpWh2K4Li4QuuE8Fr/example_text.txt)




While the CID of a file will only change if the file itself changes, the CID of a directory changes any time **any** of the files in it change, or if any files are added/removed.

Directories make it possible to address an entire static website with a single CID and access different pages of the website by requesting different files in the directory.

Using DNS to Get Rid of CIDs

Just like DNS saves people browsing the Internet from having to remember the IP address of every website they want to visit, DNS can also save us from having to remember the CIDs of websites on IPFS. The trick is to put the entire website inside a directory, compute the CID of the directory, and put that CID into a special DNS record associated with a domain called a DNSLink.

This way, when a gateway gets a request for <https://example.com/index.html> , the gateway can lookup the CID of the directory from example.com's DNS and then serve the file `index.html` from that directory. When a new version is ready to be published, the site owner updates their DNSLink DNS record to contain the new CID and the gateway will start serving the new version automatically.

[Edit on GitHub](#)  · Updated 1 week ago