## Web 2.0

We've grown to love the internet and use it in many aspects of our everyday lives, for studying, connecting with friends and family, shopping, managing our finances, and much more.

We've become dependent on the internet to a point where we really need it to be reliable and safe, but that's not really the case in this day and age. We've all experienced connectivity issues, and we also feel that our identity and data are not safe across the multiple online services that we use daily.

For example, if you want to send a photo to a friend that lives in the same street, the number of service providers the photo passes through is too high: your phone will capture the image and send it through the network until it reaches the application servers (possibly overseas); the server will then store the photo in a database, analyze it for facial recognition to suggest tagging friends in the picture, and send a notification to your friend's device; and finally your friend will check the notification and load the photo from the overseas database.

There are so many things that can go wrong in this scenario, and the truth is, you probably didn't want the photo to be stored in some far away database which you can't delete. You just wanted to share your photo directly with your friend.

**Location Addressing vs Content Addressing**

A lot of the issues that we face come from using a [location addressing](https://proto.school/content-addressing/02) approach to find the content the user is looking for, and serving it from that address. For example, we expect our libp2p logo to be available at <https://proto.school/tutorial-assets/T0009L01-libp2p-logo.svg>, but what if the server of this website is down? What if the DNS server is down? What if the image was converted to a PNG file and so the new location is T0009L01-libp2p-logo.png? What if you are in a country that has blocked the proto.school domain? All of these issues are common ones and are caused by location addressing, where we try to find the content we need by its location.

[IPFS](https://ipfs.io/), however, approaches this problem in a different way by using [content addressing](https://proto.school/content-addressing/03) instead. If we know the [CID](https://proto.school/anatomy-of-a-cid) (Content IDentifier) of our libp2p logo instead of its location, we can simply ask the network who has the image with this CID. The advantages in this approach are very appealing since they solve all the other issues previously mentioned, but to get it right there are other problems that need to be tackled first, specifically within networking.

## Networks are complex

To have a web powered by Content Addressing, we need to redesign and re-imagine how computer networks work. Networks are very complex systems that have their own rules and restrictions, and so when designing these systems, we need to take into account a lot of situations and use cases:

**Firewalls**: You might have a firewall installed in your laptop that blocks or restricts specific connections.

**NAT**: Your home WiFi router with NAT (Network Address Translation), which translates your local IP address of your laptop to a single IP address that networks outside your home can connect to.

**High latency networks**: These networks have very slow connections and leave users waiting a long time to see their content.

**Reliability**: There are many networks scattered around the world, and more often than not, a lot of users encounter slow networks that don't have robust systems in place to offer good connectivity to users. Connections drop frequently and users are left with a subpar network system that doesn't serve the users as it should.

**Roaming**: Mobile addressing is another situation where we need to guarantee that the users' devices remain uniquely discoverable when navigating through different networks across the world. Currently they work in a distributed system that requires a lot of coordination points and connections, but the best solution would be a decentralized one.

**Censorship**: In the current state of the web, blocking a website at a specific website domain is relatively easy to do if you are a governmental entity. This is useful to block illegal activity, but becomes a problem when, for example, an authoritarian regime wants to remove access to resources from its population.

**Runtimes with different properties**: There are numerous types of runtimes around, such as IoT (Internet of Things) devices (Raspberry Pi, Arduino, etc) which are gaining a lot of adoption. Because they're built with limited resources, their runtimes often use different protocols that make a lot of assumptions about their runtime.

**Innovation is very slow**: Even the most successful companies with a lot of resources can take decades to develop and deploy new protocols.

**Data Privacy**: Consumers have become increasingly concerned lately about the growing number of companies that do not respect users' privacy.

IPFS had to solve in order to create successful P2P (Peer-to-Peer) communications.

There had to be a better way. Seeing that the main issue was interoperability, the IPFS team envisioned a better way to integrate all current solutions and provide a platform that facilitated innovation. A new modular system that would enable future solutions to be seamlessly integrated into the networking stack.

This way each project can solely focus on their own objectives:

* IPFS is more focused on **content addressing**, i.e., finding, fetching and authenticating any piece of content in the web.
* libp2p is more focused on **process addressing**, i.e., finding, connecting and authenticating any data transfer processes in the network.

## What is Libp2p:

libp2p, (short for “library peer-to-peer”) is a peer-to-peer (P2P) networking framework that enables the development of P2P applications. It consists of a collection of protocols, specifications, and libraries that facilitate P2P communication between network participants, known as “[peers](https://docs.libp2p.io/concepts/fundamentals/peers/).”

## Peer-to-peer:

P2P networks are decentralized, meaning participants communicate directly with one another on a relative “equal footing.” No central server or authority controls the network.

P2P networks can take many forms, including file-sharing systems like [BitTorrent](https://www.bittorrent.com/), blockchain networks like [Bitcoin](https://bitcoin.org/en/) and [Ethereum](https://ethereum.org/en/), and decentralized communication standards like [Matrix](https://matrix.org/). These systems all have different challenges and tradeoffs, but they share the goal of improving upon the traditional client-server networking model.

## Why libp2p:

**Modularity:** it is designed to be modular, so developers can easily mix and match the different components based on their requirement.

**Extensive transport configurability**: libp2p provides a set of specifications that can be adapted to support various [transport protocols](https://docs.libp2p.io/concepts/transports/overview/), allowing libp2p applications to operate in various runtime and networking environments.

**Versatility:** In addition to supporting a wide range of transports, libp2p offers a range of discovery mechanisms, data storage and retrieval patterns, and is also [implemented in many programming languages](https://libp2p.io/implementations/), providing developers with great flexibility when building P2P applications.

**Security:** ibp2p includes [several security features](https://docs.libp2p.io/concepts/security/security-considerations/), such as peer identity verification using public key cryptography and [encrypted communication](https://docs.libp2p.io/concepts/secure-comm/overview/) between peers using modern cryptographic algorithms.

**Robustness:** libp2p is a robust and reliable networking protocol that is designed to withstand stress, disturbance, and change. Its features and design choices ensure that it is able to function effectively and efficiently in a wide range of environments, and it is able to recover quickly from disruptions or failures. It also offers protection against network attacks through the use of [mitigation techniques](https://docs.libp2p.io/concepts/security/dos-mitigation/).

**Resiliency:** P2P networks are often more resilient than traditional client-server networks, as there is no single point of failure. libp2p includes features such as [peer discovery and content routing](https://docs.libp2p.io/concepts/discovery-routing/overview/) that help to ensure that the network remains available and accessible even if some peers are offline or unreachable.

**Efficiency:** P2P networks can be more efficient in resource utilization, as data is distributed across multiple peers rather than stored on a central server. libp2p includes various storage and retrieval patterns that allow developers to distribute data efficiently across the network, making it possible to store and retrieve data in a cost-effective and scalable way.

**Piercing NAT Barriers:** libp2p is equipped with capabilities for [NAT traversal](https://docs.libp2p.io/concepts/nat/overview/), which allows P2P communication between peers even when they are behind NAT devices or firewalls. This helps to maintain the connectivity of the network and ensure that it remains accessible despite the presence of these obstacles.

**Message Distribution and Dissemination:** One such pattern libp2p uses is [publish/subscribe (pubsub)](https://docs.libp2p.io/concepts/pubsub/overview/), which allows a sender (publisher) to send a message to multiple recipients (subscribers) without the publisher having to know who the subscribers are. libp2p implements pubsub through the use of protocols like [gossipsub](https://docs.libp2p.io/concepts/pubsub/overview/" \l "gossip), providing developers with a flexible and efficient means of exchanging data and messages within their P2P applications.

**Interoperability:** libp2p implementations in different programming languages and libp2p releases across versions are designed to be interoperable with one another. This enables applications /from different language ecosystems to communicate seamlessly. This helps to promote a healthy, interconnected ecosystem of P2P applications.

**Decentralization:** One of the main advantages of P2P networks is their decentralized nature, allowing them to operate without a central authority. libp2p is designed to facilitate decentralized communication between peers, making it possible to build P2P applications resistant to censorship and more resilient in the face of network disruptions.