Peer-to-peer hole punching without centralized infrastructure

Max Inden (@mxinden)

January 16, 2022
Max Inden

- Software Developer at Protocol Labs.
- Stewarding the libp2p project.
- Maintainer of the libp2p Rust implementation.
- Maintainer of Prometheus and Kubernetes in a past life.
- @mxinden on GitHub, Twitter, ...
- https://max-inden.de
Introduction to libp2p

- A modular peer-to-peer networking stack
- Composable building blocks based on a shared core
- Specified and implemented in 7+ languages
- Runs on many runtimes: browser, mobile, embedded
- Powers the IPFS, Ethereum 2, Filecoin and Polkadot network
- ~ 100 000 libp2p based nodes online at any given time
The Problem with Firewalls and NATs
Hole Punching
Hole Punching

Goal: Full connectivity among all nodes of a libp2p network despite NATs and Firewalls.

Requirements

- No central infrastructure
- IPv4/v6
- QUIC / TCP (/ WebRTC)
- Integrate into libp2p stack
libp2p’s way of decentralized hole punching

Phases

1. B: Preparing to be dialable.
   1.1 Determine whether one is dialable (AutoNAT).
   1.2 If not, find closest public Relay nodes (e.g. Kademlia).
   1.3 Listen for incoming connections via closest Relay (Circuit Relay v2).

2. A: Connecting to B.
   2.1 A: Establish relayed connection to B (Circuit Relay v2).
   2.2 A & B: Coordinate simultaneous dial (DCUtR).
Project Flare / Status

- Works with TCP and UDP/QUIC (~90% success rate).
- Implemented in Go and Rust.
- Released in Go.
- Included in IPFS.
  - Public nodes run limited relay server.
  - Non-public nodes can punch holes (Not yet enabled by default).
Overview

The peer A establishes a direct connection to the non-dialable peer B.

**AutoNAT Protocol**
- B determining whether one is dialable
  - Dial Request
  - For each provided supposedly public address
    - Dial
  - [Dialable. No need for Hole Punching, 
    go to next step]
  - Dial Response "OK"
  - [Not dialable. 
    Hole Punching needed, go to next step]

**Kademlia Protocol**
- B finding closest public Relay nodes
  - Find nodes closest to B's Peer ID
  - List of nodes closest to B's Peer ID

**Circuit Relay v2 Protocol**
- B listening for incoming connections via closest Relay
  - For each closest Relay
    - Establish connection
    - Request reservation
    - Accept reservation
    - Announce oneself as "RELAY_ADDR=<peer ID>"

**A connecting to B**
- Establish connection
- Request connection from B
- Accept connection request
- Relayed connection established

**Direct Connection Upgrade through Relay (DCUR) Protocol**
- A and B coordinate dial

Loop: Until direct connection established
- Via relayed connection
  - Sync message
  - Measure round-trip time
  - Sync message
  - Connect message
  - Simultaneously establish connection
    - A after 1/2 RTT
    - B when receiving Connect
  - Attempt direct connection / Hole Punching
1.1 Determine whether one is dialable (AutoNAT)

For each provided supposedly public address
- Dial

alt
[Dialable. No need for Hole Punching. Don't continue.]
- Dial Response "Ok"

[Not dialable. Hole Punching needed. Continue.]
- Dial Response "Error"
1.2 Find closest public Relay nodes (e.g. Kademlia)
1.3 Listen for connections via Relay (Circuit Relay v2)

For each closest Relay:
- Establish connection
- Request reservation
- Accept reservation

Announce oneself as /<RELAY_ADDR>/p2p-circuit/<B_PEER_ID>
2.1 Establish relayed connection (Circuit Relay v2)

A → R → B

- Establish Connection
- Request connection to B
- Request connection from A
- Accept connection request
- Accept connection request

Relayed Connection established
2.2. Coordinate simultaneous dial (DCUtR)

[Loop]

[Until direct connection established]

Via relayed connection

- Sync message
- Measure round-trip time
- Sync message
- Connect message

Simultaneously establish connection
- A after 1/2 RTT
- B when receiving Connect

[Attempt direct connection / Hole Punch]
Questions?

Next steps

- Documentation - docs.libp2p.io/
- Forum - discuss.libp2p.io/
- Specification & Roadmap - github.com/libp2p/specs/

Contact

- @mxinden on GitHub, Twitter, ...
- https://max-inden.de