Hole punching in the wild

Learnings from running libp2p hole punching in production, measured from vantage points across the globe.

FOSDEM 2023
Dennis Trautwein and Max Inden
About us

Dennis Trautwein
Research Engineer at Protocol Labs
Github: @dennis-tra
Twitter: @dtrautwein_eu
Web: https://dtrautwein.eu
Email: dennis@protocol.ai

Max Inden
Software Engineer at Protocol Labs
GitHub/Twitter/...: @mxinden
Web: https://max-inden.de
Email: mail@max-inden.de
Agenda

- Quick intro to libp2p
- The Problem – Firewalls and NATs
- The Solution – Hole Punching
- Measurement Campaign
- Next Steps
libp2p

- Peer-to-peer networking library
- One specification, many implementations (Go, JS, Rust, Nim, C++, Java, ...)
- Low level features like encryption, authentication and hole punching
- High level features like DHT or Gossiping
- All you need to build peer-to-peer applications
Motivation

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

- **NAT**
  - Local to public IP address mapping

- **Firewall**
  - Control incoming/outgoing network traffic based on security rules

<table>
<thead>
<tr>
<th>Source IP</th>
<th>Source Port</th>
<th>Dest. IP</th>
<th>Dest. Port</th>
<th>Transport</th>
</tr>
</thead>
<tbody>
<tr>
<td>192.168.0.2</td>
<td>12345</td>
<td>198.51.100.0</td>
<td>54321</td>
<td>TCP</td>
</tr>
<tr>
<td>192.168.0.2</td>
<td>12345</td>
<td>198.51.100.1</td>
<td>54322</td>
<td>UDP</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Hole Punching

Problem

A

Router A

Internet

Router B

B

Packet

<table>
<thead>
<tr>
<th>Src IP</th>
<th>Src Port</th>
<th>Dst IP</th>
<th>Dst Port</th>
<th>Trpt</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Src IP</th>
<th>Src Port</th>
<th>Dst IP</th>
<th>Dst Port</th>
<th>Trpt</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Problem

Hole Punching

<table>
<thead>
<tr>
<th>Src IP</th>
<th>Src Port</th>
<th>Dst IP</th>
<th>Dst Port</th>
<th>Trpt</th>
</tr>
</thead>
<tbody>
<tr>
<td>Internal</td>
<td>Internal</td>
<td>Public IP</td>
<td>Public Port</td>
<td>TCP</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Src IP</th>
<th>Src Port</th>
<th>Dst IP</th>
<th>Dst Port</th>
<th>Trpt</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Router A</td>
<td>Router B</td>
<td></td>
</tr>
</tbody>
</table>
Hole Punching

Problem

<table>
<thead>
<tr>
<th>Src IP</th>
<th>Src Port</th>
<th>Dst IP</th>
<th>Dst Port</th>
<th>Trpt</th>
</tr>
</thead>
<tbody>
<tr>
<td>Internal IP</td>
<td>Internal Port</td>
<td>Public IP</td>
<td>Public Port</td>
<td>TCP</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Src IP</th>
<th>Src Port</th>
<th>Dst IP</th>
<th>Dst Port</th>
<th>Trpt</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Hole Punching
Hole Punching

Process

<table>
<thead>
<tr>
<th>Src IP</th>
<th>Src Port</th>
<th>Dst IP</th>
<th>Dst Port</th>
<th>Trpt</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Src IP</th>
<th>Src Port</th>
<th>Dst IP</th>
<th>Dst Port</th>
<th>Trpt</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Hole Punching

**Process**

<table>
<thead>
<tr>
<th></th>
<th>Src IP</th>
<th>Src Port</th>
<th>Dst IP</th>
<th>Dst Port</th>
<th>Trpt</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Internal IP</td>
<td>Internal</td>
<td>Public IP</td>
<td>Public Port</td>
<td>TCP</td>
</tr>
<tr>
<td></td>
<td>Internal Port</td>
<td></td>
<td>Router A</td>
<td>Router A</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Src IP</th>
<th>Src Port</th>
<th>Dst IP</th>
<th>Dst Port</th>
<th>Trpt</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Internal IP</td>
<td>Internal</td>
<td>Public IP</td>
<td>Public Port</td>
<td>TCP</td>
</tr>
<tr>
<td></td>
<td>Internal Port</td>
<td></td>
<td>Router A</td>
<td>Router A</td>
<td></td>
</tr>
</tbody>
</table>
Hole Punching

DCUtR

A

Router A

CONNECT

CONNECT

CONNECT

CONNECT

Router B

CONNECT

CONNECT

Relay

CONNECT

Start measure RTT

B
Hole Punching

DCUtR

Wait for SYNC

A

Router A

CONNECT

CONNECT

CONNECT

Relay

CONNECT

CONNECT

Router B

CONNECT

CONNECT

B

Start measure RTT
Hole Punching

DCUtR
hole punching relies on the properties of EIM-NATs to allow appropriately designed peer-to-peer applications to “punch holes” through the NAT device(s) enroute and establish direct connectivity with each other, even when both communicating hosts lie behind NAT devices.

**Implementation in libp2p**


**FOSDEM 2022**

Peer-to-peer hole punching without centralized infrastructure

How libp2p can traverse NATs and firewalls without coordination through central STUN and TURN servers.

**Role out on the IPFS network**

Start with relay capabilities on all public nodes, followed by hole punching capabilities on all private nodes.

**Hole punching month**

Measurement campaign to gain insights across networks, routers, endpoints, …
Measurements
Punchr

Architecture

**Honeypot**
- DHT Server
- Announces itself to the network
- Tracks inbound connections

**Server**
- Exposes gRPC API
- Query for recently seen NAT’d DCUTR peers
- Track Results

**Clients**
- Rust and Go implementations
- Periodically queries server
- Reports hole punch outcome

https://github.com/libp2p/punchr
**Measurement Results**

**General**

**Measurement Campaign**
- From 2022-12-01
- To 2023-01-01 (2023-01-10)

**Statistics**
- >6.25M Hole Punch Results Reported
- 154 Clients punched >47k Peers

**Outcomes**
- NO_CONNECTION (~795k)
- NO_STREAM (~369k)
- CONNECTION_REVERSED (~711k)
- SUCCESS (~2.50M)
- FAILED (~1.88M)
Measurement Results

Client Contributions/Outcomes
Network Detection

Grouping by Clients distorts the results
- Hole Punching is dependent on network setup
- A single client can be in multiple networks

How to detect individual networks?
- Group by public IP addresses/ASN
- Group by private IP addresses/ASN

But remote peers are also in different networks!
- Clients are randomly punching remote peers
- Effect applies to all clients equally
- Effect will average out

Example
- HP1:
  - 100.100.100.100
  - 2a02:1000:9999:
- HP2:
  - 100.100.100.100
  - 2a02:1000:7777:
- One network

Results
- 342 unique client networks

Open for other suggestions!
Network Contributions/Outcomes

Measurement Results

Graph: Bar chart showing the distribution of network outcomes. The x-axis represents the top 100 networks based on contributed data points. The y-axis shows the percentage of outcomes. The chart uses different colors to represent SUCCESS, CONNECTION_REVERSED, FAILED, NO_STREAM, and NO_CONNECTION. The data includes a legend for the categories and a color key for the counts.
Measurement Results

Success Rate Distribution
Measurement Results

IP/Transport Dependence

IPv4 + TCP (325,248)

IPv4 + QUIC (273,894)

IPv6 + TCP (75,472)

IPv6 + QUIC (104,107)
Measurement Results

Final Connection Transport

The chart shows the percentage of successful hole punchings for different transport protocols.

- **TCP**: 18.9%
- **QUIC**: 81.1%
Measurement Results

Virtual Private Network

Router A

Router B

A

B

Success Rate in %

Network Type

Non VPN

VPN

0 10 20 30 40 50 60 70 80 90 100

A

Router A

Router B

B
Measurement Results

Attempts If Successful

- Share of All Successful Hole Punches in %
  - 1,200,947
  - 97.6%
  - 21,612
  - 1.8%
  - 7,366
  - 0.6%

Attempts:
1, 2, 3
Next Steps
Measurement Results

Next Steps

Protocol Improvements
- Consider Changing Strategy on Retry
  - e.g. QUIC hole punch from both sides
    - https://github.com/libp2p/specs/issues/487
- Measure RTT between default gateways
  - https://github.com/libp2p/specs/issues/488

Data Analysis
- Look at individual clients/networks that have low success rates
- Identify causes for hole punching problems

Academia
- Craft a follow-up publication
Decentralized Hole Punching Paper

"Decentralized Hole Punching." – Seemann, Marten, Max Inden, and Dimitris Vyzovitis.

Get involved!

- Talk to us here at the venue
- Documentation - docs.libp2p.io
- Forum - discuss.libp2p.io
- Specification & Roadmap - github.com/libp2p/specs
- Implementations - github.com/libp2p/<LANGUAGE>-libp2p
- Join the community call

---

**Get involved!**

- Talk to us here at the venue
- Documentation - docs.libp2p.io
- Forum - discuss.libp2p.io
- Specification & Roadmap - github.com/libp2p/specs
- Implementations - github.com/libp2p/<LANGUAGE>-libp2p
- Join the community call

---

[Image of a person] Github: @mxinden
Twitter: @mxinden
Web: https://max-inden.de
Email: mail@max-inden.de

[Image of another person] Github: @dennis-tra
Twitter: @dtrautwein_eu
Web: https://dtrautwein.eu
Email: dennis@protocol.ai
Measurement Results

Round Trip Time Dependence
Measurement Results

Relay Location Dependence

Client | Relay | Relay | Relay | Remote

0% | 25% | 50% | 75% | 100%

Graph showing Success Rate and Hole Punches over Relay Location to Remote in %.
Measurement Results

Success Rate over Time
Punchr

Monitoring

Dashboards

- Health
- Performance

https://punchr.dtrautwein.eu/grafana/
Repository

https://github.com/libp2p/punchr
Network Detection Results

Statistics

- 153 Clients operated in 372 unique networks