Nethermind's C# implementation of Libp2p

Rose Jethani

Ethereum Protocol Fellow

Section 1

Introduction to Libp2p

What is Libp2p?

- A modular networking framework to create a distributed p2p network.
- Provides transport protocols, stream multiplexers, secure channels and authentication, peer discovery, messaging, NAT traversal etc.
- Contrary to devp2p, it was not designed to serve Ethereum's needs specifically.
- Implementations in multiple languages with interoperability.
- Used in every consensus client.

Components:-

- Transport
- Handshaking
- Protocol Negotiation
- Multiplexing
- Discovery
- Messaging

Transport

- Handles addressing and delivery of data packets.
- Dialing and listening using multiaddress `(/ip4/192.0.1.0/tcp/1234)`.
- Several communication methods:
  - TCP
  - QUIC
- Publically dialable listening endpoints.

Handshake

- Secure communication channel for message exchanges.
- Why?
  - Peer authentication
  - Confidentiality
  - Integrity
  - Encrypted messages between nodes
- Protocols
  - Noise
  - TLS

Protocol Negotiation

- Two peers in a network agree on the protocol to use for communication.
- Protocol IDs `(/my-app/protocol-name/1.0.1)`
- Dialing peer sends protocol ID.
- Listening peer checks compatibility.
- Accepts or rejects.
- mutlistream-select.

Multiplexing

- Creating multiple virtual connections within a stream.
- Exchange different types of data within the same connection.
- Why?
  - Prevent blocking of applications
  - Reduce resource overhead and latency penalty caused by frequent connection establishment.
  - Fast, efficient
- mplex, yamux

Section 2

TLS Protocol Implementation

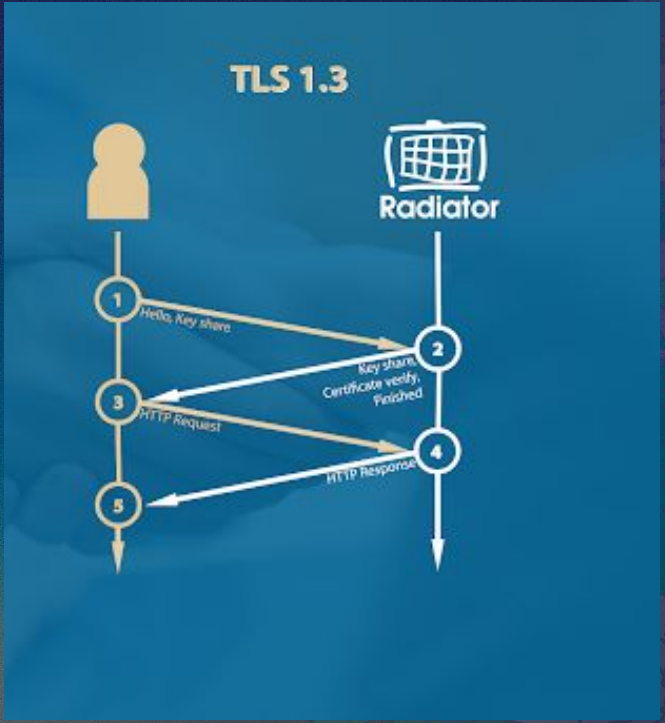
TLS Protocol

What is TLS in libp2p?

- Establishes secure communication between peers.

How it Works:

1. Encryption & Authentication: TLS encrypts data and verifies peer identities.
2. Handshake Process: Peers exchange X.509 certificates with public keys.
3. libp2p Public Key Extension: Peers use their host key in certificates.
4. Signature Verification: Ensures authenticity and aborts connection if invalid.



TLS Protocol

Merged PR's:

https://github.com/NethermindEth/dotnet-libp2p/pull/107

https://github.com/NethermindEth/dotnet-libp2p/pull/106

https://github.com/NethermindEth/dotnet-libp2p/pull/104

Section 3

Perf Protocol

Perf Protocol

What is the Perf Protocol?

- A benchmarking protocol in libp2p based on QUIC Performance draft
- Helps measure upload/download speeds between clients and servers
- Client-driven: Client decides the amount of data to send and receive
- Ideal for testing throughput and connection efficiency

How it Works:

- **Client**: Requests server to send data, uploads its own data, closes the write stream.
- **Server**: Reads the client's request, sends the requested data back, and closes the connection.

Benchmarks:

- **Throughput**:
- **Handshakes per Second**:

PROBLEMS 17 OUTPUT TERMINAL TEST RESULTS PORTS

TERMINAL bash - perf-benchmarks

```
Listener is listening on: /ip4/0.0.0.0/tcp/39961/p2p/12D3KooWPnYR8NEgUs6SJ5nHYSnq8uYY6KbgkoyfPgptn8J98vk7
11:02:03 info: Nethermind.Libp2p.Protocols.YamuxProtocol[0] Dial
11:02:03 info: Nethermind.Libp2p.Protocols.YamuxProtocol[0] Listen
11:02:03 info: Nethermind.Libp2p.Protocols.IdentifyProtocol[0] Listen
11:02:03 info: Nethermind.Libp2p.Protocols.IdentifyProtocol[0] Dial
11:02:03 info: Nethermind.Libp2p.Protocols.IdentifyProtocol[0] Listen
11:02:03 info: Nethermind.Libp2p.Protocols.IdentifyProtocol[0] Dial
11:02:03 info: Nethermind.Libp2p.Protocols.IdentifyProtocol[0] Received peer info: { "publicKey": "CAESIM+MCvQ40UjoDq2jF8tD2Dnuc646jHdQoTmhlF5AOdqw", "listenAddrs": [ "BAAAAAAGAACcGQ==" ], "protocols": [ "
/perf/1.0.0", "/ipfs/id/1.0.0", "/ipfs/ping/1.0.0", "/meshsub/1.0.0", "/floodsub/1.0.0" ], "observedAddr": "BH8AAAEGAADn0A==", "protocolVersion": "ipfs/1.0.0", "agentVersion": "dotnet-libp2p/1.0.0" }
11:02:03 info: Nethermind.Libp2p.Protocols.IdentifyProtocol[0] Received peer info: { "publicKey": "CAESIFDQ85zwB3VqspEIrq5npwpolaFueWcCyP4WIMe8+TAD", "listenAddrs": [ "BH8AAAEGAADn0A==" ], "protocols": [ "
/perf/1.0.0", "/ipfs/id/1.0.0", "/ipfs/ping/1.0.0", "/meshsub/1.0.0", "/floodsub/1.0.0" ], "observedAddr": "BH8AAAEGAACcGQ==", "protocolVersion": "ipfs/1.0.0", "agentVersion": "dotnet-libp2p/1.0.0" }
Successfully connected to the remote peer.
{
  "Type": "final",
  "TimeSeconds": 0.0208858,
  "UploadBytes": 104857600,
  "DownloadBytes": 104857600
}
rose2221@rose2221-Victus-by-HP-Laptop-16-e1xxx:~/workspace/TLS_tests/dotnet-libp2p_test/src/samples/perf-benchmarks$
```

PR(in progress):-

https://github.com/NethermindEth/dotnet-libp2p/pull/109

Section 4

Noise Protocol Feature Addition

Pre selecting muxer during the handshake

- During the handshake, Noise can communicate the preferred muxer to avoid back-and-forth negotiation.
- By pre-selecting a muxer, Noise reduces the time and resources needed for agreement.
- Uses a specific property, **NoiseExtensions**, to carry muxer preferences in the initial exchange.

Merged PR:-
https://github.com/NethermindEth/dotnet-libp2p/pull/90

Thank you!

Special Thank to Alexey Osipov, Josh and Mario

Rose Jethani

rosejethani28@gmail.com

X: @0xrosetteeee

TG: @rose22221