P2P CDNs: A hybrid approach to content distribution

By
Gaurav Kumar Chandel (gkc5188),
Parth Vidyadhar Natu(pzn5087)

CDN vs P2P

- **Scalability**: P2P can scale with increasing number of users while CDN requires extra infrastructure to do so.
- **Maintenance Cost**: CDN requires maintenance of servers while P2P generally works off of user resources.
- **Security**: Data transfer happens between CDN servers and clients while in P2P client can serve a malicious file.
- **Copyright Issues**: P2P clients can serve a file with copyright issues while CDN servers will not.

Can we achieve the best of both worlds?

PACDN and CAP2P

PACDN:

- Peer-assisted CDN
- Built over the CDN infrastructure
- CDN will re-route the requests to peers or provide a peer list to the client
- CDN will provide the file if peers can’t satisfy the request
- Not too prone to P2P issues

PACDN and CAP2P

CAP2P:

- CDN assisted P2P
- Built over the P2P infrastructure
- Clients will request the file from peers directly
- If client does not feel QoS satisfied, CDN will provide the file as a backup

Case Studies

Akamai Netsession:

- Peer assisted CDN
- Requires an interface to be installed on clients
- Servers divided into Netsession control plane and data plane
- Interface talks to control plane to get the peer list
- If QoS is not satisfied, control plane provides nearest edge server to satisfy request
- Backoff mechanism to prevent network degradation
- Centrally controlled and vetted files
- 57.4% of data was being served by peers by a survey

Case Studies

Livesky

- Peer assisted CDN for live video streaming
- Management center - content management, configuration system, monitoring system, billing system, DNS based load balance system
- Cache servers (service nodes) which hold the data
- End hosts which requests the data
- Service Nodes can act as regular client, tracker for new clients or seed for peers.
- Use STUN for NAT issues

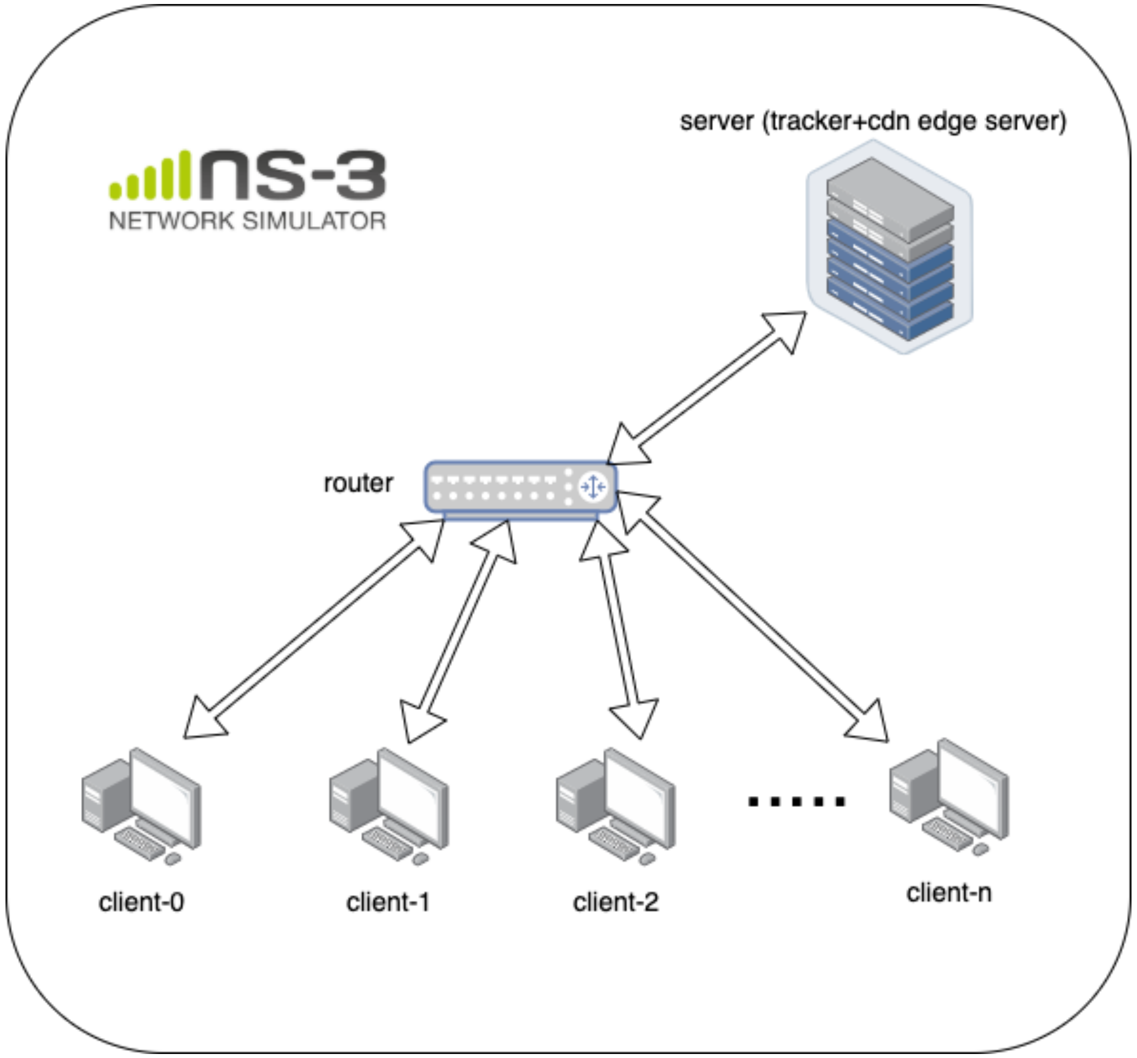
Overcoming P2P shortcomings

So how did they do it?

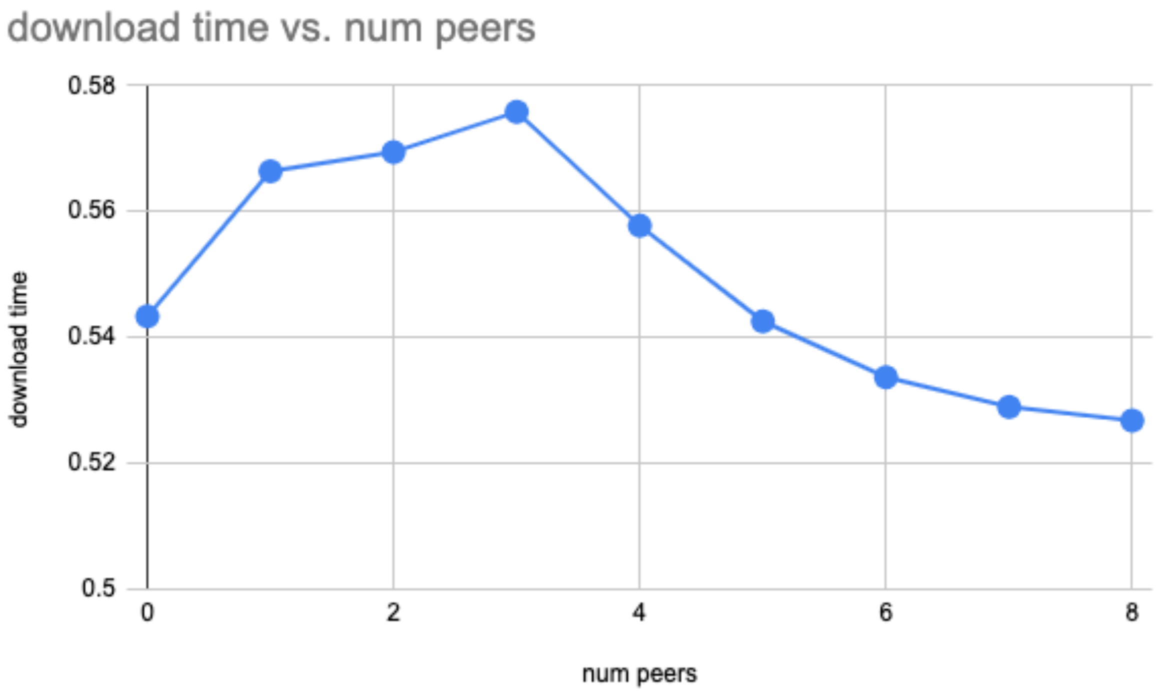
- **Heterogeneity of client resources:** Having a CDN as a backup compensates for the client churn and heterogeneity of client resources
- **Impact on ISPs:** Inter ISP traffic was a major concern in this approach, having the CDN servers providing peers in the peer list that belong to the same ISP covers this issue a bit
- **Security**: CDN servers centrally vet files and authenticate clients
- **Firewalls and NATS:** Some clients might be behind some firewalls or a NAT box, tools such as STUN are used to find the address assigned to the client by NAT

Implementation

Architecture



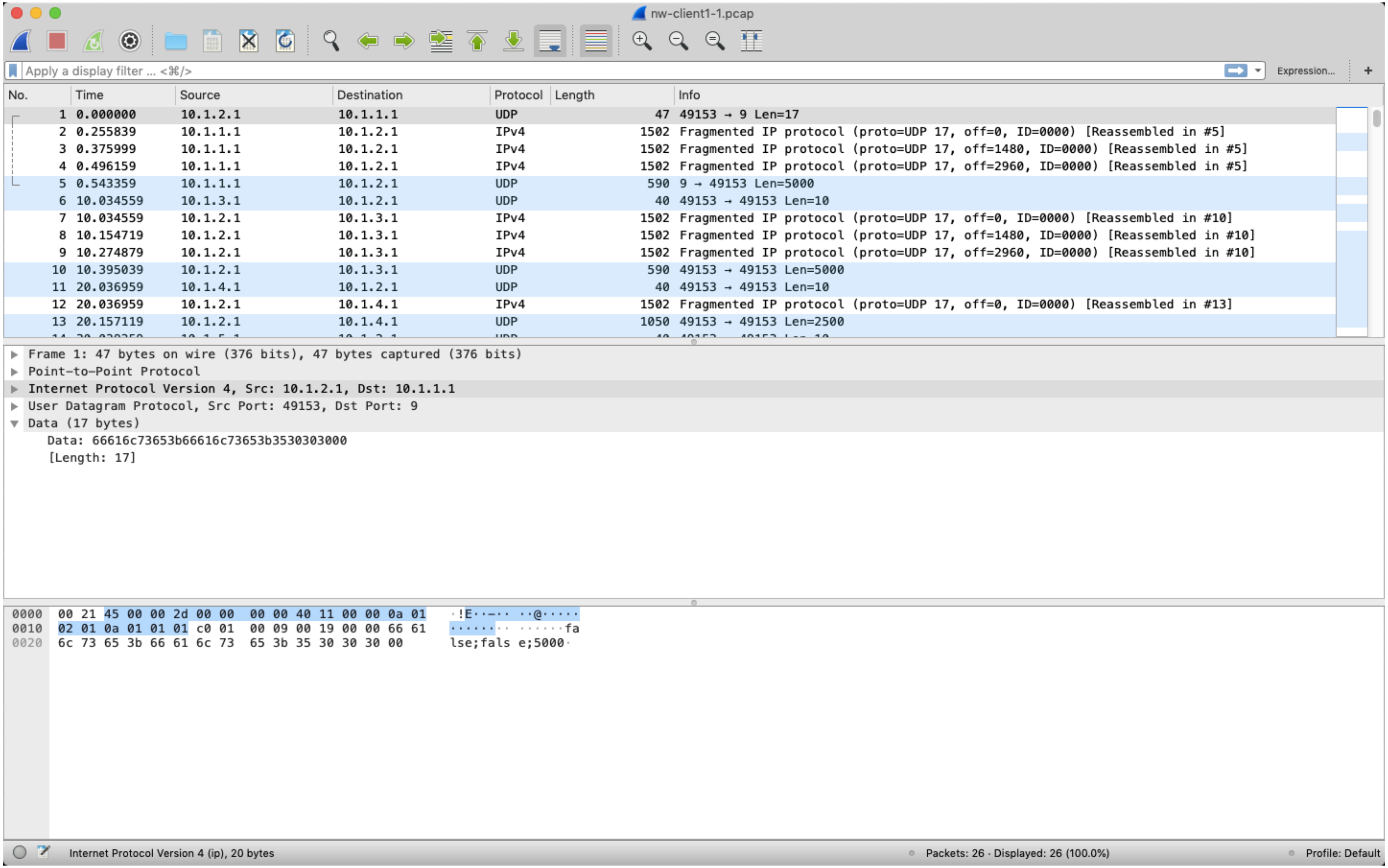
Output



Source Code can be found at:
https://github.com/gauravkc2794/p2p-cdn/

Implementation

Scenario : No peers, only CDN



```
No.   Time       Source      Destination  Protocol  Length  Info
  1   0.000000   10.1.2.1    10.1.1.1     UDP         47  49153 → 9 Len=17
  2   0.255839   10.1.1.1    10.1.2.1     IPv4      1502  Fragmented IP protocol (proto=UDP 17, off=0, ID=0000) [Reassembled in #5]
  3   0.375999   10.1.1.1    10.1.2.1     IPv4      1502  Fragmented IP protocol (proto=UDP 17, off=1480, ID=0000) [Reassembled in #5]
  4   0.496159   10.1.1.1    10.1.2.1     IPv4      1502  Fragmented IP protocol (proto=UDP 17, off=2960, ID=0000) [Reassembled in #5]
  5   0.543359   10.1.1.1    10.1.2.1     UDP        590  9 → 49153 Len=5000
  6   10.034559  10.1.3.1    10.1.2.1     UDP         40  49153 → 49153 Len=10
  7   10.034559  10.1.2.1    10.1.3.1     IPv4      1502  Fragmented IP protocol (proto=UDP 17, off=0, ID=0000) [Reassembled in #10]
  8   10.154719  10.1.2.1    10.1.3.1     IPv4      1502  Fragmented IP protocol (proto=UDP 17, off=1480, ID=0000) [Reassembled in #10]
  9   10.274879  10.1.2.1    10.1.3.1     IPv4      1502  Fragmented IP protocol (proto=UDP 17, off=2960, ID=0000) [Reassembled in #10]
 10   10.395039  10.1.2.1    10.1.3.1     UDP        590  49153 → 49153 Len=5000
 11   20.036959  10.1.4.1    10.1.2.1     UDP         40  49153 → 49153 Len=10
 12   20.036959  10.1.2.1    10.1.4.1     IPv4      1502  Fragmented IP protocol (proto=UDP 17, off=0, ID=0000) [Reassembled in #13]
 13   20.157119  10.1.2.1    10.1.4.1     UDP       1050  49153 → 49153 Len=2500
```

```
▶ Frame 1: 47 bytes on wire (376 bits), 47 bytes captured (376 bits)
▶ Point-to-Point Protocol
▶ Internet Protocol Version 4, Src: 10.1.2.1, Dst: 10.1.1.1
▶ User Datagram Protocol, Src Port: 49153, Dst Port: 9
▼ Data (17 bytes)
    Data: 66616c73653b66616c73653b3530303000
    [Length: 17]
```

```
0000  00 21 45 00 00 2d 00 00  00 00 40 11 00 00 0a 01   ·!E··-·· ··@·····
0010  02 01 0a 01 01 01 c0 01  00 09 00 19 00 00 66 61   ········ ······fa
0020  6c 73 65 3b 66 61 6c 73  65 3b 35 30 30 30 00      lse;fals e;5000·
```

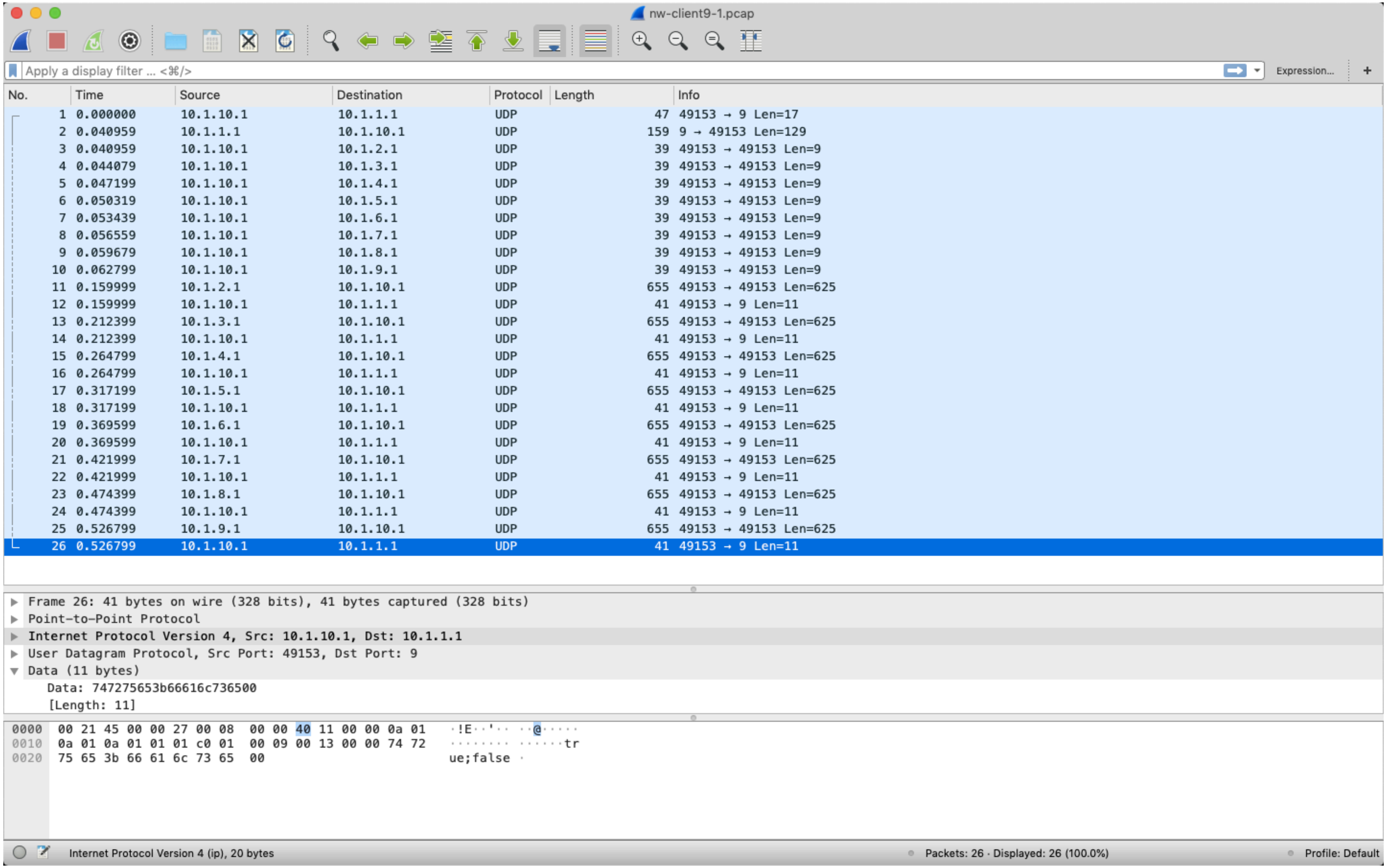
Internet Protocol Version 4 (ip), 20 bytes

Packets: 26 · Displayed: 26 (100.0%)

Profile: Default

Implementation

Scenario : 8 peers



```
No.  Time      Source      Destination  Protocol  Length  Info
  1  0.000000  10.1.10.1   10.1.1.1     UDP           47  49153 → 9 Len=17
  2  0.040959  10.1.1.1    10.1.10.1    UDP          159  9 → 49153 Len=129
  3  0.040959  10.1.10.1   10.1.2.1     UDP           39  49153 → 49153 Len=9
  4  0.044079  10.1.10.1   10.1.3.1     UDP           39  49153 → 49153 Len=9
  5  0.047199  10.1.10.1   10.1.4.1     UDP           39  49153 → 49153 Len=9
  6  0.050319  10.1.10.1   10.1.5.1     UDP           39  49153 → 49153 Len=9
  7  0.053439  10.1.10.1   10.1.6.1     UDP           39  49153 → 49153 Len=9
  8  0.056559  10.1.10.1   10.1.7.1     UDP           39  49153 → 49153 Len=9
  9  0.059679  10.1.10.1   10.1.8.1     UDP           39  49153 → 49153 Len=9
 10  0.062799  10.1.10.1   10.1.9.1     UDP           39  49153 → 49153 Len=9
 11  0.159999  10.1.2.1    10.1.10.1    UDP          655  49153 → 49153 Len=625
 12  0.159999  10.1.10.1   10.1.1.1     UDP           41  49153 → 9 Len=11
 13  0.212399  10.1.3.1    10.1.10.1    UDP          655  49153 → 49153 Len=625
 14  0.212399  10.1.10.1   10.1.1.1     UDP           41  49153 → 9 Len=11
 15  0.264799  10.1.4.1    10.1.10.1    UDP          655  49153 → 49153 Len=625
 16  0.264799  10.1.10.1   10.1.1.1     UDP           41  49153 → 9 Len=11
 17  0.317199  10.1.5.1    10.1.10.1    UDP          655  49153 → 49153 Len=625
 18  0.317199  10.1.10.1   10.1.1.1     UDP           41  49153 → 9 Len=11
 19  0.369599  10.1.6.1    10.1.10.1    UDP          655  49153 → 49153 Len=625
 20  0.369599  10.1.10.1   10.1.1.1     UDP           41  49153 → 9 Len=11
 21  0.421999  10.1.7.1    10.1.10.1    UDP          655  49153 → 49153 Len=625
 22  0.421999  10.1.10.1   10.1.1.1     UDP           41  49153 → 9 Len=11
 23  0.474399  10.1.8.1    10.1.10.1    UDP          655  49153 → 49153 Len=625
 24  0.474399  10.1.10.1   10.1.1.1     UDP           41  49153 → 9 Len=11
 25  0.526799  10.1.9.1    10.1.10.1    UDP          655  49153 → 49153 Len=625
 26  0.526799  10.1.10.1   10.1.1.1     UDP           41  49153 → 9 Len=11
```

```
▶ Frame 26: 41 bytes on wire (328 bits), 41 bytes captured (328 bits)
▶ Point-to-Point Protocol
▶ Internet Protocol Version 4, Src: 10.1.10.1, Dst: 10.1.1.1
▶ User Datagram Protocol, Src Port: 49153, Dst Port: 9
▼ Data (11 bytes)
    Data: 747275653b66616c736500
    [Length: 11]
```

```
0000  00 21 45 00 00 27 00 08  00 00 40 11 00 00 0a 01   ·!E··'··  ··@·····
0010  0a 01 0a 01 01 01 c0 01  00 09 00 13 00 00 74 72   ········  ······tr
0020  75 65 3b 66 61 6c 73 65  00                        ue;false ·
```

Internet Protocol Version 4 (ip), 20 bytes

Packets: 26 · Displayed: 26 (100.0%)

Profile: Default

Conclusion & Future work

1. PACDNs are needed to modernize our content distribution networks

2. QoS can improve further because of using the hybrid CDNs

3. Need easier integration architectures to help migrating current CDNs

4. Our implementation: Need to introduce inter ISP traffic and client churn to have a better sense of real network

Thank You