Is large MTU beneficial to cellular core networks?

Youngmin Choi, Junghan Yoon, YoungGyoun Moon, KyoungSoo Park
Increasing Network Bandwidth Requirements

Today

- 4K video streaming: 15Mbps ~ 32Mbps
- 8K 120Hz video streaming: 85Mbps ~ 110Mbps

Future

- 4K cloud VR: 60Mbps ~ 1.6Gbps
- 8K cloud VR: 1.9Gbps ~ 4.3Gbps

*Note: Bandwidth requirements are estimated and can vary depending on content and network conditions.*
The Architecture of 5G Cellular Network

- AMF: Access and Mobility management Function
- SMF: Session Management Function
- UPF: User Plane Function

Data plane
Control plane

Public Data Network (PDN)
How to Enlarge the Effective Bandwidth of a Core Network?

- Simple Approach to improve RX/TX efficiency
  - Adopt a large maximum transmission unit (MTU)
    merit 1) improve DMA efficiency between host CPU and NIC at endpoints

1500B MTU

Sender

Host CPU
1500B packets

DMA Controller

Receiver

Host CPU

DMA Controller

9000B MTU

Sender

Host CPU

9000B packet

DMA Controller

Receiver

Host CPU

DMA Controller
How to Enlarge the Effective Bandwidth of a Core Network?

- Simple Approach to improve RX/TX efficiency
  - Adopt a large maximum transmission unit (MTU)
    1) improve DMA efficiency between host CPU and NIC at endpoints
    2) reduce packet header processing cost

1500B MTU

<table>
<thead>
<tr>
<th>Host CPU</th>
<th>1500B packets</th>
</tr>
</thead>
<tbody>
<tr>
<td>DMA Controller</td>
<td></td>
</tr>
</tbody>
</table>

Sender

Receiver

9000B MTU

<table>
<thead>
<tr>
<th>Host CPU</th>
<th>9000B packet</th>
</tr>
</thead>
<tbody>
<tr>
<td>DMA Controller</td>
<td></td>
</tr>
</tbody>
</table>

- 6x header processing
- 1x header processing
Simple Approach to improve RX/TX efficiency
- Adopt a large maximum transmission unit (MTU)
  - merit 1) improve DMA efficiency between host CPU and NIC at endpoints
  - merit 2) reduce packet header processing cost
- Benefit in UPF: packet header based rule table lookups in UPF

How to Enlarge the Effective Bandwidth of a Core Network?

1500B MTU
- PDR
- FAR
- URR
- GTP Encap/Decap
- UPF
- 6x1500B pkts

9000B MTU
- PDR
- FAR
- URR
- GTP Encap/Decap
- UPF
- 1x9000B pkt
- 6x less processing load with 9000B MTU!
Why Not Using a Large MTU?

- Reason 1) There might be old Ethernet devices that cannot support a large MTU size.
  - IP fragmentation will lower the forwarding performance
    → No, modern Ethernet devices support MTU larger than 1500B!
    → Can avoid IP fragmentation with MTU discovery, TCP MSS option

- Reason 2) There is no benefit in using a large MTU size.
  - NIC offload feature (TSO/LRO) and GRO are enough to reduce DMA overhead
    → LRO/GRO are less effective with concurrent flows
    → UPF can benefit from reducing # packets

We will show you the benefit of a large MTU size!
Impact of Large MTU and LRO/GRO on Endpoints

- 2.1x performance gain with 9000B MTU
  - Measure RX throughput at the receiver
  - Use iPerf to generate TCP traffic with a single TCP flow
  - Endpoint: Intel Xeon Gold 6346 @ 3.10 GHz, Ubuntu 22.04

- Similar performance gain with LRO/GRO (1500B MTU)
Impact of Concurrent Flows with LRO/GRO on Endpoints

Throughput (Gbps)

<table>
<thead>
<tr>
<th>Number of concurrent flows</th>
<th>Throughput</th>
<th>1500B</th>
<th>3000B</th>
<th>4500B</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>54</td>
<td>43</td>
<td>43</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>32</td>
<td>34</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>21</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Measure the aggregate RX throughput at the receiver with enabling LRO+GRO
- Use iPerf to generate TCP traffic with concurrent TCP flows
- Endpoint: Intel Xeon Gold 6346 @ 3.10 GHz, Ubuntu 22.04

Large MTU is still beneficial to endpoints with concurrent flows!

↑ # of flows → less chance of packets being merged → performance drop!
Impact of Large MTU on 5G UPF

- Measure forwarding throughput of 5G UPF (detail of UPF will be explained later)
- Use iPerf to generate TCP traffic with 100 concurrent flows
- UPF: Intel Xeon Gold 6346 @ 3.10 GHz, use a single CPU core
Measure forwarding throughput of 5G UPF (detail of UPF will be explained later)

Use iPerf to generate TCP traffic with 100 concurrent flows

Large MTU is beneficial to the performance of both endpoints and UPF!

How to increase the MTU size at core network without coordinating with PDN?

UPF: Intel Xeon Gold 6346 @ 3.10 GHz, use a single CPU core
Key Idea: Dynamic MTU Translation

- Leverage LRO and GRO to merge TCP packets in UPF
Key Idea: Dynamic MTU Translation

- Leverage LRO and GRO to merge TCP packets in UPF

Challenges:

1) It is difficult to merge packets with multiple concurrent flows
2) GRO consumes CPU cycles if it deals with large # of packets
Solution 1) Increasing # of RX queues

- Receive Side Scaling (RSS)
  - Distribute received packets to RX queues based on 5-tuple
  - Ensure the flow-to-core affinity
  - High # of flows per queue -> less chance of LRO
Solution 1) Increasing # of RX queues

- **Receive Side Scaling (RSS)**
  - Distribute received packets to RX queues based on 5-tuple
  - Ensure the flow-to-core affinity

- **Increasing # of RX queues**
  - reduce # of flows for each RX queue
  - increase the chance of LRO
  - improve efficiency of GRO
Solution 2) Hash Table-based GRO

- Existing GRO implementation in DPDK
  - ) linear search to find the matching flow

\[ O(kn) \] running time

\[ O(n) \] running time

- ) one-sided merging algorithm

\[ n \text{ received packets} \]

- ) merging algorithm considering both sides

Arrive out-of-order from same TCP flow

Arrive out-of-order from same TCP flow

- Our solution
  + ) Hash table to find the matching flow
    - RSS hash value tagged with NIC HW

\[ O(n) \] running time

\[ n \text{ received packets} \]
dUPF Implementation

- UPF implementation
  - Open Mobile Evolved Core (OMEC)
  - Employ BESS, a software-based datapath that runs on DPDK

- Implementation of dynamic MTU translation
  - Enable LRO to merge packets up to 9000B
  - Adjust TCP MSS option field in TCP SYN packet
  - 189 LOC for MTU translation
  - 463 LOC for hashtable-based GRO
  - 281 LOC for BESS script to support multiple NICs and multiple RX queues
Evaluation Setup

- **Hardware setup**
  - UPF machine
    - 16-core Intel Xeon Gold 6346 CPU @ 3.10Ghz, four of NVIDIA Connectx-6 100GbE NIC
  - Eight machines for servers & clients
    - Equipped with one Connectx-6 NIC, handle 200Gbps traffic

- **Workload & baseline**
  - iPerf – generate large TCP traffic from servers to clients
  - Baseline – 1500B-MTU UPF
  - Ours – dUPF
    - 1500B-MTU between clients and UPF
    - 9000B-MTU between UPF and servers

- **UPF rule setup**
  - PDR (Packet Detection Rule) : 4M
  - QER (QoS Enforcement Rule) : 500K
  - FAR (Forwarding Action Rule) : 500K
  - MBR (Maximum Bit Rate) : 2Gbps
  - GBR (Guaranteed Bit Rate) : 100Mbps
**Overall Performance**

- **dUPF outperforms by 3.1x ~ 4.9x and reaches 628 Gbps with only 8 cores**
  - 8.7x increased average packet size
- **No performance improvement after 8 cores due to memory bandwidth bottleneck**
  - 108GB/s average memory utilization at 8 cores
Performance with Multiple RX Queues

- Increasing number of RX queues → fewer flows per queue → increase the effectiveness of LRO/GRO

- 800 concurrent flows and 8 CPU cores for UPF
Performance with HashGRO

- 100 concurrent flows and a single CPU core for UPF

- LRO + HashGRO achieves 4.85x performance gain over baseline!
Packet Size Distribution

- Size of packets processed in UPF
- Size of packets received at clients

- On UPF and client, size of packets are ~63KB and ~9KB, respectively
Future Plan

- *Split-Process-Reassemble* strategy with SmartNIC
  - **Split** the header and payload at SmartNIC, DMA only header
**Future Plan**

- **Split-Process-Reassemble** strategy with SmartNIC
  - **Split** the header and payload at SmartNIC, DMA only header
  - **Process** UPF operation with only packet header

![Diagram](attachment:image.png)
Future Plan

- **Split-Process-Reassemble strategy**
  - **Split** the header and payload at SmartNIC, DMA only header
  - **Process** UPF operation with only packet header
  - **Reassemble** header and payload at SmartNIC

- Reduce DMA overhead for payload!
Conclusion

- Large MTU is beneficial to both endpoints and UPF

- Dynamic MTU translation
  - Leverage LRO and GRO to merge received packets at UPF
  - Increase # of RX queues per core
  - Implement hashtable-based GRO

- dUPF improves the performance of UPF by 3.1x to 4.9x over baseline UPF
  - Average packet size increases by 8.7x
  - Expect to achieve 1.3+Tbps with 2 CPUs in a single UPF

- Future Plan
  - Split-Process-Reassemble strategy
Thank you