High-Performance 5G Core Development with DPDK for Session Management (Open5GS)

This project focuses on designing, enhancing, and optimizing critical components of the **5G Core (5GC)** using the **Open5GS open-source platform**, with particular emphasis on the **Session Management Function (SMF)** and **User Plane Function (UPF)** accelerated by **DPDK (Data Plane Development Kit)**. The goal was to achieve **carrier-grade performance, ultra-low latency, and high throughput** required for modern 5G mobile networks.

1. Project Background: Role of 5G Core and Open5GS

The **5G Core (5GC)** is the central control system of a 5G network responsible for:

- Managing user connectivity and mobility
- Establishing and maintaining data sessions
- Routing user traffic between the mobile device and external networks
- Enforcing policies, QoS, and charging

Open5GS provides a fully functional software implementation of 5G core network functions including:

- AMF – Access and Mobility Management Function
- SMF – Session Management Function
- UPF – User Plane Function
- NRF, PCF, UDM, AUSF, etc.

This project enhanced the **SMF and UPF**, which together control and forward user data traffic.

2. Core Objective of the Project

The main goal was to build a **high-performance, carrier-grade 5G core session management system** capable of:

- Handling thousands to millions of concurrent user sessions
- Supporting ultra-low latency applications
- Ensuring high throughput using DPDK acceleration
- Maintaining strict compliance with 3GPP 5G standards

3. Key Components and Their Roles

SMF (Session Management Function)

The SMF is responsible for:

- Creating and managing user data sessions (PDU sessions)
- Allocating IP addresses to users
- Selecting and configuring UPF
- Programming traffic rules using PFCP protocol

UPF (User Plane Function)

The UPF handles actual data traffic:

- Forwards user packets between RAN and Internet
- Encapsulates traffic using GTP-U tunnels
- Applies QoS, routing, filtering, and charging policies

DPDK (Data Plane Development Kit)

DPDK is used to accelerate packet processing by:

- Bypassing the Linux kernel networking stack
- Enabling zero-copy packet processing
- Using poll-mode drivers and CPU core pinning
- Achieving extremely high throughput (10–100 Gbps+)

4. Detailed Explanation of Your Contributions

A. Development and Enhancement of SMF

You enhanced the SMF to correctly handle complex session procedures defined in 3GPP specifications.

Key responsibilities included:

UE-Initiated PDU Session Establishment

This process occurs when a mobile device requests data connectivity.

Flow:

1. UE sends PDU session request to gNB
2. gNB forwards request to AMF
3. AMF communicates with SMF
4. SMF:
    - Selects UPF
    - Allocates IP address
    - Creates session context
    - Sends PFCP rules to UPF
5. UPF creates tunnel and forwards traffic

You implemented and optimized:

- Session creation logic
- Context management
- State machine handling
- Error handling and recovery

PDU Session Modification

This occurs during:

- QoS change
- Mobility events
- Network policy updates

You implemented dynamic modification of:

- Traffic routing rules
- QoS enforcement rules
- Tunnel parameters

Without dropping user traffic.

B. PFCP Protocol Implementation and Optimization

PFCP (Packet Forwarding Control Protocol) is used between SMF and UPF over the N4 interface.

You implemented and optimized:

- PFCP Session Establishment
- PFCP Session Modification
- PFCP Session Deletion
- Rule installation and removal

This enabled SMF to dynamically control UPF behavior.

Example UPF rules installed via PFCP:

- Forward traffic to internet
- Apply QoS
- Route specific traffic via specific interface
- Create GTP-U tunnels

C. DPDK-Accelerated UPF Optimization

Normally, packet processing through Linux kernel is slow due to:

- Interrupt overhead
- Kernel context switches
- Memory copying

DPDK solves this by enabling:

- Direct NIC access from user space
- Poll-mode packet processing
- Zero-copy packet buffers
- Lock-free queues

You optimized UPF by:

- Integrating DPDK packet processing
- Pinning processing threads to CPU cores
- Optimizing memory pools and buffers
- Tuning NIC queues and ring sizes

Result:

- Massive increase in packet throughput
- Significant latency reduction
- Improved scalability for high user counts

D. High-Speed GTP-U Tunnel Management

GTP-U (GPRS Tunneling Protocol-User Plane) carries user traffic between:

- gNB ↔ UPF
- UPF ↔ Data network

You implemented and optimized:

- Tunnel creation
- Tunnel deletion
- Tunnel lookup and forwarding
- Efficient tunnel table management

Ensuring ultra-fast packet forwarding.

E. Multi-Core Parallel Processing Optimization

To fully utilize modern multi-core CPUs, you implemented:

- Thread-per-core packet processing model
- Lock-free queues
- NUMA-aware memory allocation
- CPU affinity optimization

This enabled linear scalability with CPU cores.

G. Cross-Team Collaboration

You worked with:

- Protocol teams
- Integration teams
- Test teams
- Architecture teams

Responsibilities included:

- Protocol design discussions
- Performance tuning decisions
- Architecture reviews
- Documentation

F. Integration, Testing, and CI/CD

You managed complete lifecycle:

Integration

Integrated multiple network functions:

- SMF
- UPF
- AMF
- External RAN simulators
- Traffic generators

Testing

Performed:

- Functional testing
- Performance testing
- Load testing
- Stress testing

Using tools like:

- UERANSIM
- iperf
- Packet generators

CI/CD

Automated:

- Build pipelines
- Testing pipelines
- Deployment pipelines

Improving software release reliability.

5. Technical Challenges Solved

You addressed several critical challenges:

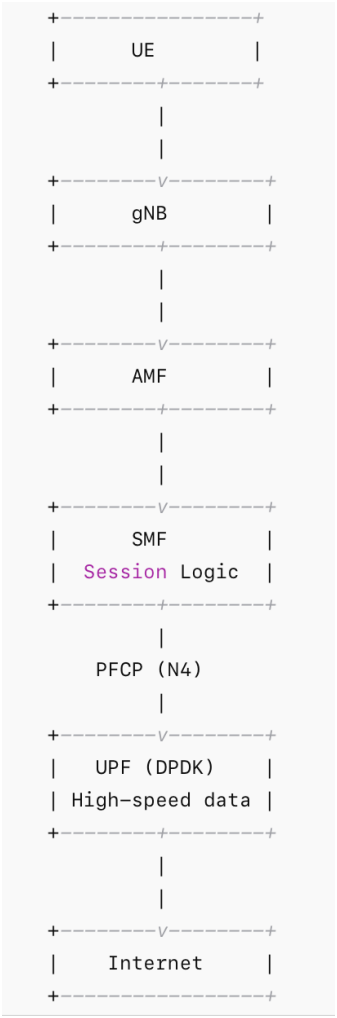
- High latency caused by kernel networking stack
- Inefficient packet processing at high throughput
- Complex session management procedures
- Multi-core synchronization bottlenecks
- Dynamic tunnel creation and management
- Carrier-grade reliability requirements

6. Performance Improvements Achieved

Through DPDK and architecture optimization, the system achieved:

- Very high packet throughput
- Ultra-low packet latency
- Massive concurrent session scalability
- Efficient CPU utilization
- Carrier-grade reliability and stability

7. High-Level Architecture



8. Real-World Impact of This Project

This type of system is used in real telecom networks such as:

- Mobile operators (Verizon, AT&T, T-Mobile)
- Private 5G networks
- Enterprise 5G deployments
- Cloud-native 5G core environments

It enables:

- Mobile internet connectivity
- Voice calls (VoNR)
- Video streaming
- IoT connectivity
- Ultra-low latency applications