



EE 542 Computer Networks

Fall 2026

Administrative Details

- Instructor: Ece Guran Schmidt
- Office: A-402
- Email: eguran@metu.edu.tr
- Schedule: Tuesday 9:40-12:30 @A211
- **Follow**
 - <https://odtuclass.metu.edu.tr/>
for announcements
 - Your e123456@metu.edu.tr email

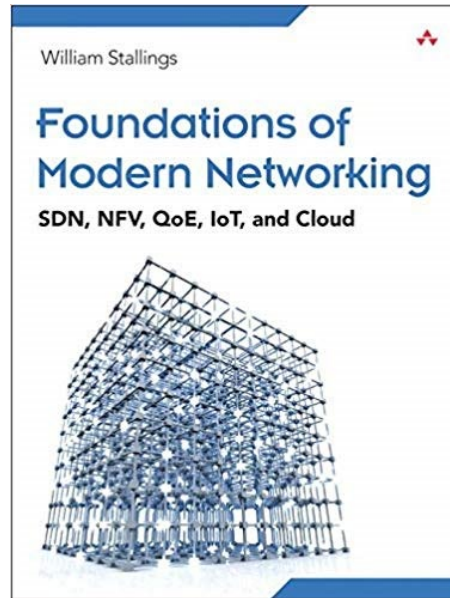
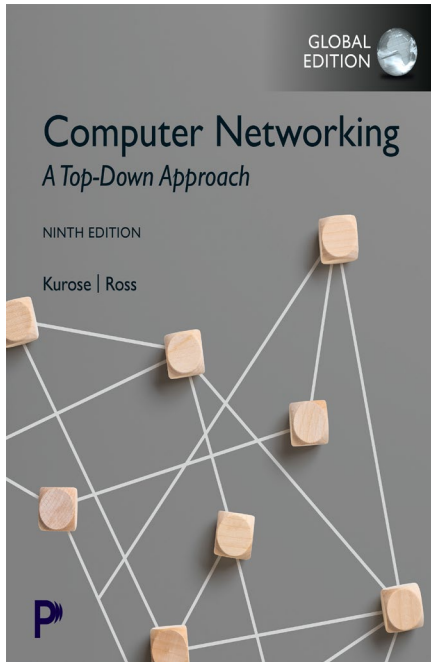


EE542 Syllabus

- Introduction to Modern Computer Networks
- Internet Applications and Delivery Requirements
- QoS I: Classification, Scheduling, Policing and Queue Management
- QoS II: Measurement-Driven Adaptive Routing and Packet Steering
- Programmable Data Planes: P4, eBPF and DPDK
- High-Speed Networking: SmartNICs and DPUs
- Data-Center and Cloud Networking
- High-Performance Data-Center Networking
- 5G Core and Edge Networking
- AI/HPC Networking



Reference Material



Foundations of
Modern Networking
SDN, NFV, QoE, IoT, and
Cloud
William Stallings
Addison-Wesley
Professional 2015

Computer
Networking: A Top
Down Approach
9th edition
Jim Kurose, Keith Ross
Addison-Wesley
2026

- Selected papers for reference and review
- Lecture Note Slides

Copy right notice: Lecture Note Slides are compiled from the teaching material of these books, previous lecture notes of EE542 and additional resources. Part of the slides are entirely created by Ece Schmidt.

Grading

- Midterm(s): 50%
- Final: 30%
- Class Project and Presentation: 20%
- Guideline documents are posted on odtuclass about the paper review, presentation and project.



Class Project

- Track 1: Start with A research paper or multiple papers and build on them with an original idea
- Track 2: Your thesis or other work you do in your professional environment. We need to discuss how to proceed if you choose this.
- Motivation, previous literature, your contribution, evaluations and results
- We can submit to some conference 😊
- There will be a conference style presentation at the end of the semester, with Q&A



Similar Classes

- <https://computer-networks.github.io/15-744-sp23/>
- <http://web.stanford.edu/class/cs244/>
- <https://courses.engr.illinois.edu/cs538/sp2021/>
- <https://faculty.cc.gatech.edu/~lingliu/courses/cs6675/20Spring/cs6675.html>
- <https://cs268.io/>



1

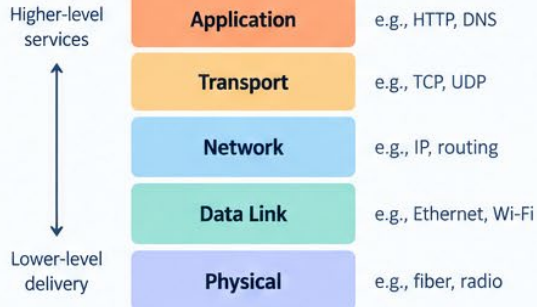
Introduction to Modern Computer Networks

From simple connectivity to a programmable, intelligent infrastructure

CONNECT • SHARE • SCALE • EMPOWER

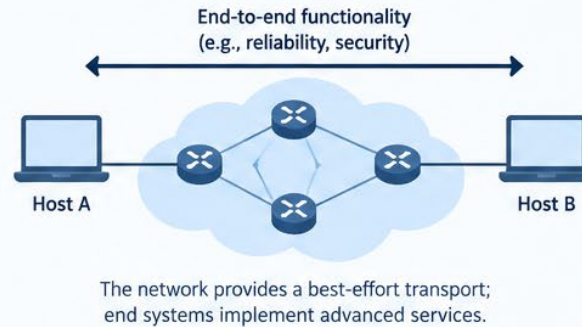
1 Network architecture and layering

Divide complexity, enable interoperability



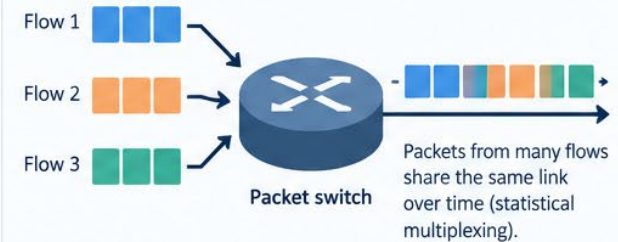
2 End-to-end principle

Keep the network simple, put intelligence at the edges



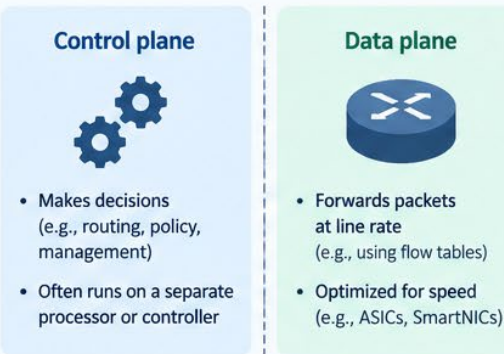
3 Packet switching and statistical multiplexing

Share links efficiently by sending data in packets



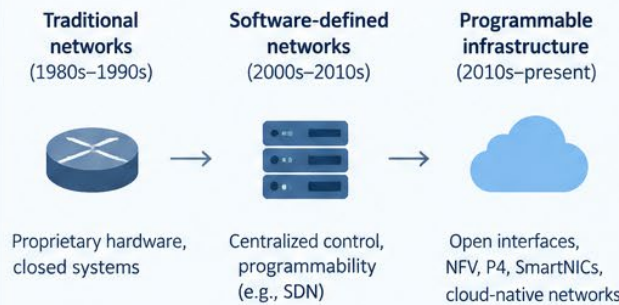
4 Control plane vs data plane

Different roles, often separated



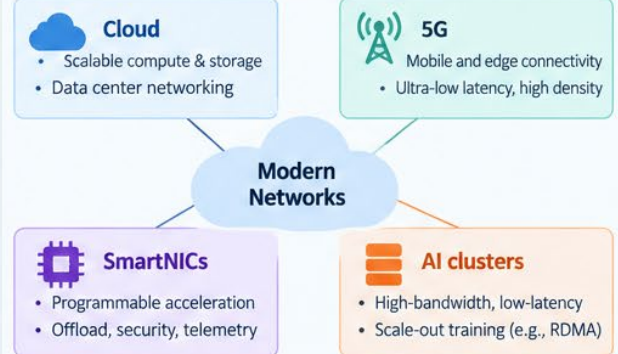
5 Evolution to programmable infrastructure

From fixed-function devices to flexible, software-driven networks



6 Where cloud, 5G, SmartNICs and AI clusters fit

Modern networks connect and accelerate diverse systems



NETWORKS POWER A MORE CONNECTED WORLD

PEOPLE | DEVICES | DATA | OPPORTUNITY

9/22/2026



Ece GURAN SCHMIDT EE542



8

Different applications place different demands on the network

1 Web / cloud applications

Everyday internet and cloud-based services



- Short flows and large transfers
- Responsive page loads
- Cloud services and storage

2 Video streaming

High-quality media to any device



- High sustained throughput
- Buffering tolerance
- Adaptive bitrate delivery

3 Interactive / real-time

Live communication and remote interaction



- Low delay
- Low jitter
- Stable bidirectional exchange

4 Gaming / AR / VR

Ultra-responsive, immersive experiences



- Very low latency
- Smooth updates
- Immersive user experience

5 Industrial / control traffic

Automation, monitoring and critical systems



- Predictable timing
- High reliability
- Tight delay bounds

6 Key delivery metrics

Core network performance characteristics



7 Application needs vs best-effort Internet

Different applications need different performance levels



8 Why QoS?

Meet application needs and deliver better experiences



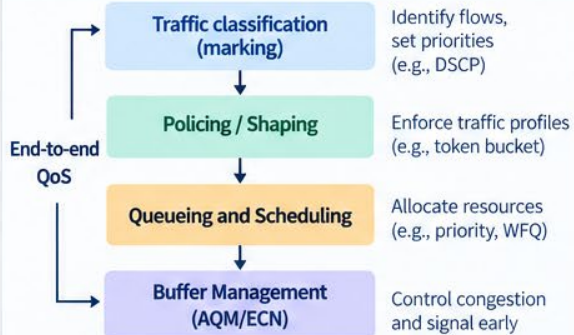
3

QoS I: Classification, Scheduling, Policing and Queue Management

From application requirements to controllable network service

1 QoS architecture

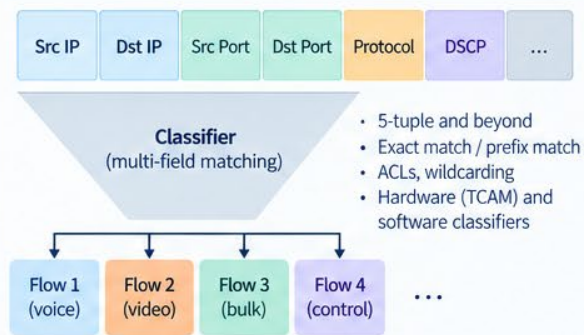
Building blocks for differentiated service



2

Flow and packet classification

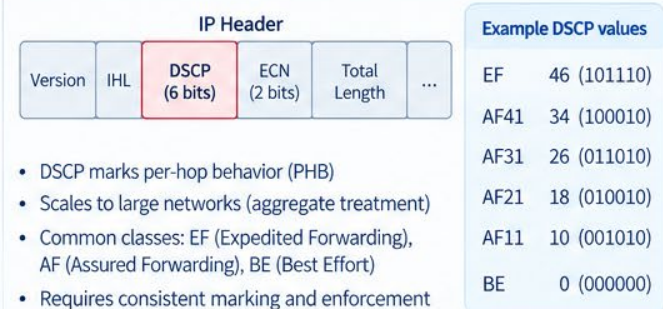
Identify traffic and apply the right treatment



3

DiffServ and DSCP

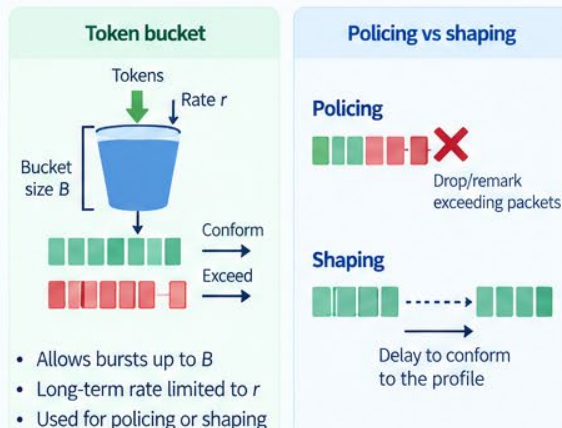
Scalable QoS in the Internet



4

Policing and shaping

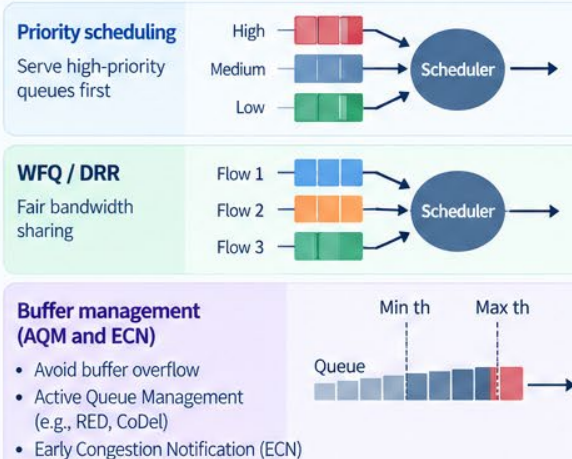
Control traffic to desired profiles



5

Scheduling and queue management

Allocate bandwidth and control delay



6

From application requirements to QoS mechanisms

Match application needs with network mechanisms

Application type	Key requirements	Relevant QoS mechanisms
Voice / real-time communication	- Low delay - Low jitter - High reliability	- Classification (DSCP) - Priority scheduling - Low queuing delay (AQM/ECN)
Video streaming	- High throughput - Moderate delay - Low loss	- Classification (DSCP) - WFQ/DRR - AQM/ECN
Gaming / AR / VR	- Low delay - Low jitter - Moderate throughput	- Priority scheduling - Traffic shaping - Low-latency queue management
Web / cloud applications	- Variable throughput - Tolerant to delay - High reliability	- Classification - Fair scheduling (WFQ/DRR) - AQM/ECN
Industrial / control	- Very low delay - Very high reliability - Bounded jitter	- Strict priority / traffic shaping - Resource reservation (if available) - Low queuing delay

➔ **Motivation for QoS:** best-effort Internet service alone cannot meet the diverse and sometimes stringent requirements of modern applications.

4

QoS II: Measurement-Driven Adaptive Routing and Packet Steering

Using measurements to drive routing and forwarding decisions

1

What network state must be measured

Measure key path and link state to understand performance



Delay

End-to-end or per-hop latency



Loss

Fraction of packets lost



Utilization

Link or path load



Available bandwidth

Spare capacity for new traffic



Metrics describe the state of links or end-to-end paths

2

Active vs passive measurement

Two complementary approaches

Active measurement

Send probe packets to test the network



- Direct estimation of performance
- Control probe rate and target
- Additional overhead on the network
- Useful for path discovery and validation

Passive measurement

Observe existing traffic and telemetry



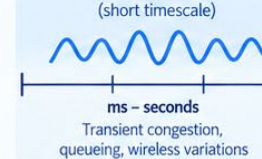
- Use counters, flow records, telemetry
- No additional traffic
- Low overhead
- Indirect inference (may be less accurate)
- Widely used in production networks

3

Measurement accuracy and timescale

Measurements vary over time and must be interpreted carefully

Fast fluctuations (short timescale)



Slower trends (longer timescale)



Measurements contain noise
Use appropriate sampling interval



Tradeoff: responsiveness vs. stability
Faster reacts to change, but can be noisy



Freshness matters
Stale measurements lead to poor decisions



Choose timescale per application
Short for dynamics, long for trends

4

From measurements to routing metrics

Transform raw measurements into actionable metrics

Measurement

Delay, loss, utilization, available bandwidth

Metric construction

Combine and normalize metrics (e.g., weighted cost)

Decision

Select best path(s) subject to constraints

Example path metrics

- Weighted delay: $C = w_d \cdot D$
- Loss penalty: $C = D + w_l \cdot L$
- Available bandwidth: require $B \geq B_{\min}$
- Utilization cost: $C = w_u \cdot U$

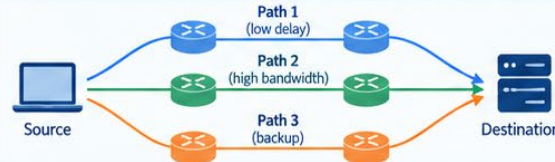
Constraint-based / QoS-aware routing

- Select paths that satisfy delay, loss, and bandwidth requirements
- Optimize metrics (e.g., multi-criteria or weighted sum)

5

Adaptive routing, multipath and steering

Use measurements to adapt paths and distribute traffic



Adaptive routing

- Update routes based on measurements
- React to changing network conditions

Flow steering

- Send different flows on different paths
- Based on application or QoS needs

Multipath routing

- Use multiple paths simultaneously
- Improve performance and reliability

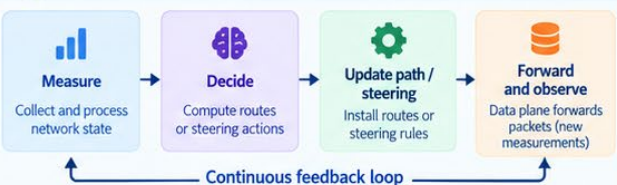
Packet steering

- Direct individual packets dynamically
- Can react to short-term changes

6

Avoiding oscillation and the measurement → decision → forwarding loop

Design a stable control loop from measurement to forwarding



Avoid instability and oscillation

- Use hysteresis and thresholding
- Smooth measurements (e.g., moving average)
- Dampened or rate-limited updates
- Enforce minimum holding time before changing paths again
- Coordinate between control plane and data plane
- Consider multiple metrics and tradeoffs



Key idea: measurements improve QoS decisions only when the control loop is accurate, timely, and stable.



5

5. Programmable Data Planes: P4, eBPF and DPDK

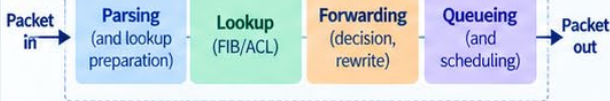
From fixed-function forwarding to flexible packet processing

1

Traditional fixed-function forwarding

Specialized hardware pipeline in routers and switches

Mostly fixed-function hardware

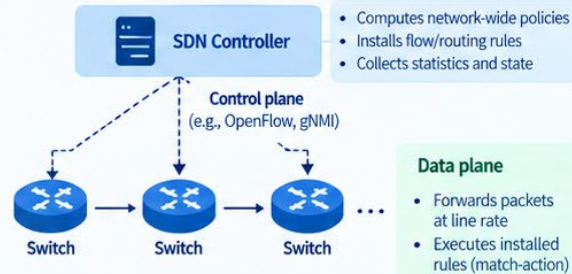


- Optimized for high speed and low latency (ASIC/NPUs)
- Limited flexibility (fixed header formats and predefined features)
- Features are vendor-defined and hard to change

2

SDN and control/data-plane separation

Centralized control, simple fast data plane



3

Match-action processing

Core abstraction for many programmable data planes



Example match fields

- Destination IP (LPM)
- TCP/UDP port (exact)
- DSCP (exact)
- Custom headers/metadata

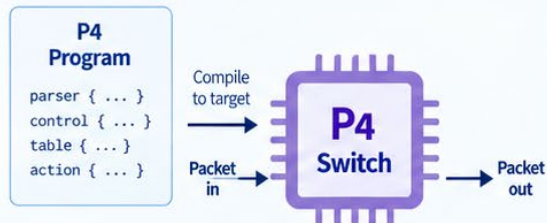
Example actions

- Forward to port/next hop
- Drop
- Modify header fields
- Mirror / clone
- Count / meter

4

P4 and programmable switches

Define custom packet processing pipelines



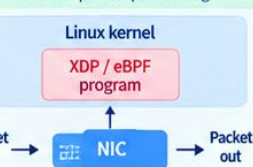
- P4 specifies how packets are parsed and processed
- Enables custom headers, match-action tables, and flexible pipelines
- Ideal for line-rate in-network functions (e.g., telemetry, load balancing)
- Constrained by target hardware resources and supported operations

5

eBPF/XDP and DPDK

Programmable packet processing in software

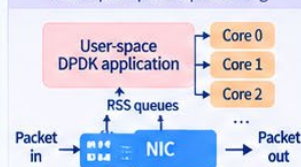
eBPF / XDP



- Runs packet-processing programs in or near the kernel (XDP)
- Useful for filtering, telemetry, load balancing, security
- Tightly integrated with Linux (safe, sandboxed execution)

Key difference: eBPF/XDP is tightly integrated with Linux, while DPDK is a user-space high-speed framework.

DPDK



- Bypasses kernel for high-throughput packet processing
- Direct access to NIC queues (polling)
- Flexible and high performance
- CPU-intensive and less integrated with the OS

6

Software vs hardware, and what the data plane can do

Complementary approaches for different needs

Aspect	Software (eBPF/DPDK)	Hardware (P4 switches)
Flexibility	High	Medium
Throughput	Medium - High	Very High
Latency	Higher	Very Low
Power efficiency	Lower	Higher
Ease of deployment	Easy (software)	Harder (specialized HW)

✓ Can implement (in the data plane)

- Parsing and classification
- Header rewrite
- Forwarding
- Filtering (ACLs)
- Metering / rate limiting
- Telemetry and measurement
- Load balancing / traffic steering

✗ Cannot easily implement

- Arbitrary long-lived per-flow computation
- Complex global optimization
- Heavy control logic
- Tasks needing large mutable state
- Cross-network coordination (e.g., global network state)



Key takeaway: Use the data plane for fast per-packet actions; keep complex decisions in the control plane.



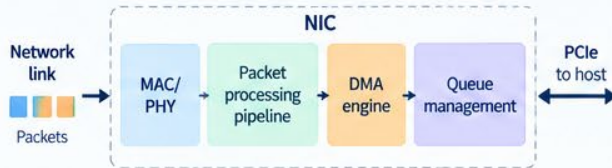
6

High-Speed Networking: SmartNICs and DPUs

Accelerating networking, storage and security at line rate

1 NIC architecture

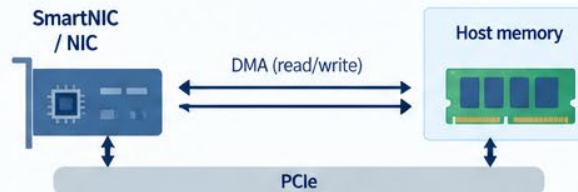
From packets on the wire to host memory



- MAC/PHY: link layer, encoding, line rate
- Packet processing: filtering, RSS, offloads
- DMA: direct memory access to host memory
- Multiple queues for high throughput and parallelism

2 DMA and PCIe

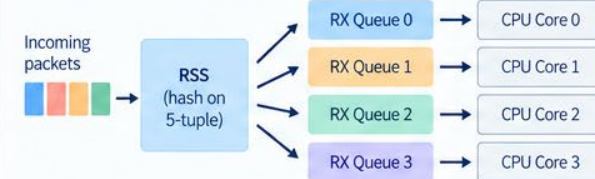
Efficient data transfer between NIC/SmartNIC and host



- DMA moves packets directly to/from host memory (bypasses CPU)
- PCIe provides high-bandwidth, low-latency interconnect
- Modern NICs use PCIe Gen4/Gen5 for multi-100 Gb/s rates

3 RSS and multiqueue NICs

Parallel processing across CPU cores

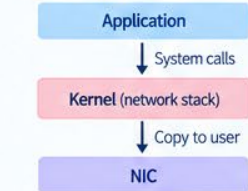


- Receive Side Scaling (RSS) distributes flows across multiple queues
- Enables parallel packet processing on multicore CPUs
- Improves throughput and reduces per-core load

4 Kernel bypass and zero-copy

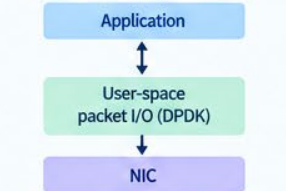
Reduce overhead, increase performance

Traditional (kernel processing)



- Higher CPU overhead
- Multiple memory copies
- Suitable for general-purpose applications

Kernel bypass (e.g., DPDK)



- Bypasses kernel network stack
- Zero-copy packet processing
- Higher throughput and lower latency
- Widely used in high-speed and low-latency applications

5 Hardware offloads

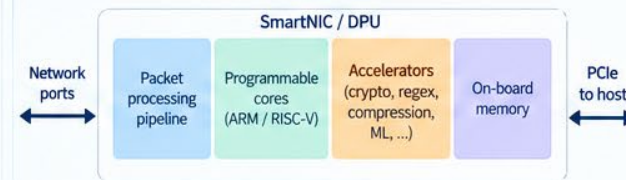
Accelerate common networking and security functions



- Offloads reduce CPU load and increase throughput
- Common offloads available in modern NICs and SmartNICs
- Some functions (e.g., encryption) can be fully offloaded to SmartNICs/DPUs

6 SmartNIC architectures

More than just a NIC: programmable, offload-capable



- Dedicated processors and accelerators on the NIC (SmartNIC / DPU)
- Can execute custom networking, security and storage functions
- Examples: NVIDIA BlueField (ARM cores + accelerators + on-board memory)
- FPGA-, GPU- and DPU-based solutions for different use cases

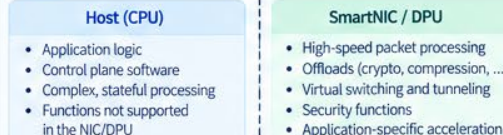
7 Offloading networking and security functions

Moving functions from host to SmartNIC/DPU



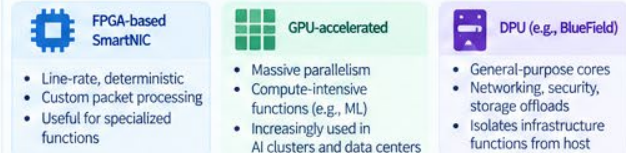
8 Host versus network processing boundary

What runs where?



9 FPGA, GPU and edge approaches

Different platforms, different strengths



SmartNICs and DPUs move networking, security and storage functions closer to the data, enabling higher performance, lower latency and better isolation.

9/22/2026



Ece GURAN SCHMIDT EE542



13

7 Data-Center and Cloud Networking

Scalable, high-performance networks for modern applications

SCALE • VIRTUALIZE • CONNECT • ENABLE

1 Data-center traffic characteristics

Many-to-many, high bandwidth, low latency

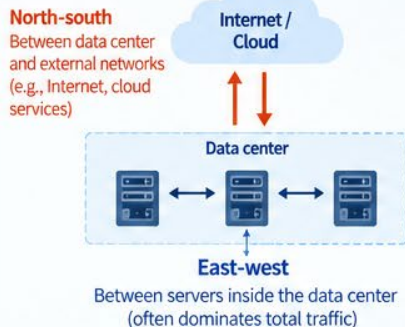
- Large volumes of short flows
- High throughput demands
- Low latency sensitive (e.g., storage, ML training)
- Bursty and dynamic
- Mostly between servers in the same data center

Typical applications

- Distributed storage
- Big data analytics
- Microservices
- ML/AI training
- Real-time data processing

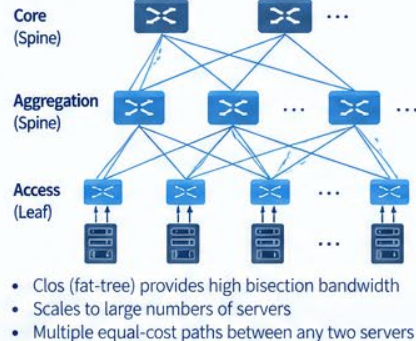
2 East-west versus north-south traffic

Different patterns, different requirements



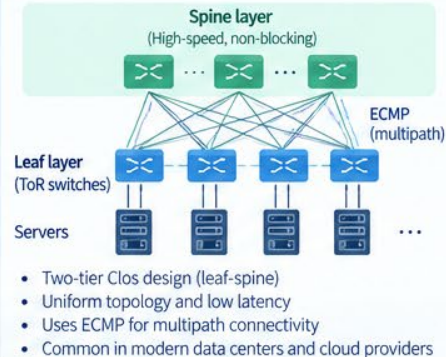
3 Clos / fat-tree

Scalable, high-bandwidth interconnect



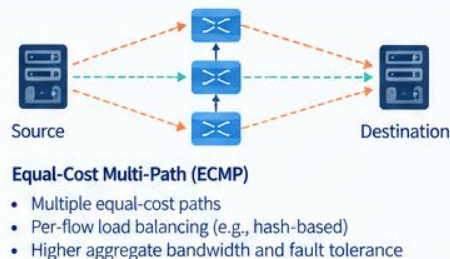
4 Leaf-spine architecture

Simple, scalable, widely deployed



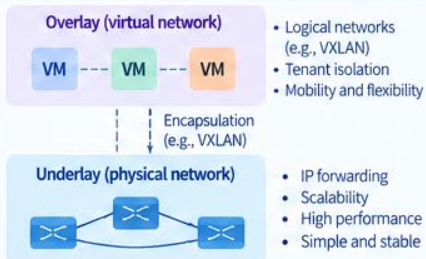
5 Multipath connectivity and ECMP

Use multiple paths for higher bandwidth and resilience



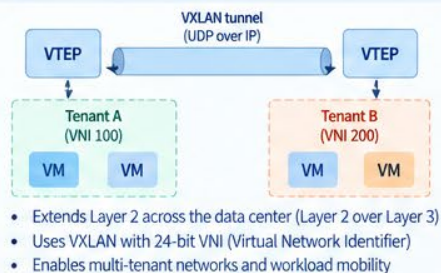
6 Underlay versus overlay

Separation of infrastructure and virtual networks



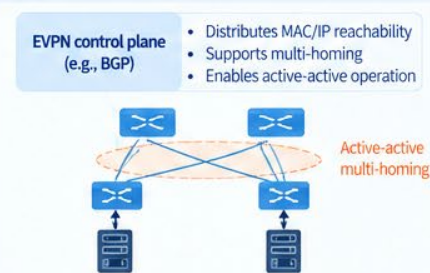
7 Network virtualization: VXLAN

Layer 2 over Layer 3



8 EVPN

Control plane for virtualized networks



9 Virtual networks and tenant isolation

Multiple tenants on shared infrastructure



10 Cloud load balancing

Distribute traffic for scalability and availability



Data-center and cloud networking enable ...

- ✓ Scalable and high-performance infrastructure
- ✓ Efficient use of network resources
- ✓ Isolation and security for multiple tenants
- ✓ Support for modern distributed applications (storage, analytics, AI)
- ✓ Flexible and programmable networks for cloud environments

From bare metal to the cloud and beyond

8

High-Performance Data-Center Networking

Low latency, high throughput and congestion control at scale

SCALE · PERFORMANCE · EFFICIENCY · CONVERGENCE

1

Data-center latency requirements

Tight bounds for modern applications

Typical requirements

- RPC / microservices < 100 μ s
- Storage (e.g., NVMe-oF) < 50–100 μ s
- Distributed training < 50–100 μ s
- Inference (AI/ML) < 1–10 ms

Latency components

Serialization

Propagation

Queueing

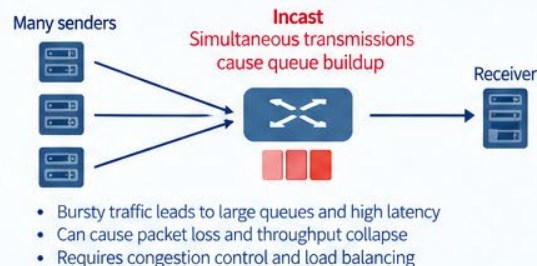
Processing

Queueing delay dominates in data centers

2

Incast and queue buildup

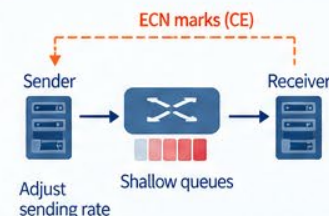
Many-to-one traffic causes congestion



3

DCTCP and ECN

Keep queues short with early congestion notification



Key ideas

- Switch marks packets (ECN) before queues overflow
- DCTCP uses mark fraction to adjust sending rate
- Keeps queues short and maintains high throughput

4

Load balancing beyond ECMP

Finer-grained and adaptive load distribution

Limitations of ECMP

- Per-flow hashing
- Uneven load distribution
- Cannot adapt to short-term congestion

Improved techniques

Flowlets

Switch on flowlet boundaries (idle periods)

Packet spraying

Distribute packets across multiple paths

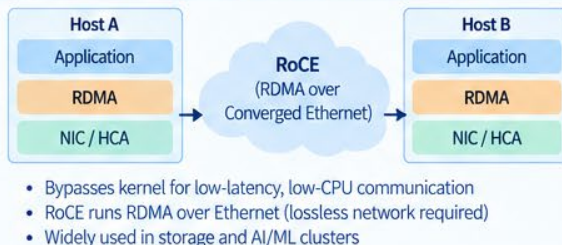
Adaptive load balancing

Use real-time congestion feedback

5

RDMA and RoCE

Remote memory access with low overhead



6

Priority Flow Control (PFC)

Enable lossless Ethernet for RDMA

PFC operation



Key points

- Per-priority pause frames
- Prevents packet loss for high-priority traffic
- Used together with ECN and congestion control
- Misconfiguration can lead to deadlocks

7

Congestion control for RDMA

Maintain high throughput without loss

Challenges

- RDMA requires very low loss rates
- Loss leads to large performance penalties
- Need fast and accurate congestion signals

Approaches

DCQCN

Uses ECN marks to adjust rate

TIMELY

Uses RTT gradient

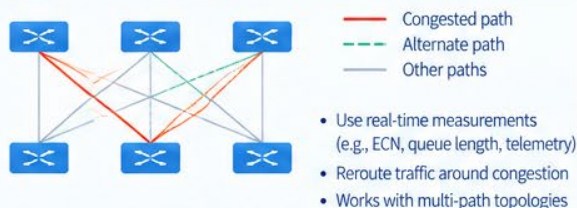
HPCC

In-network telemetry (e.g., INT)

8

Adaptive routing

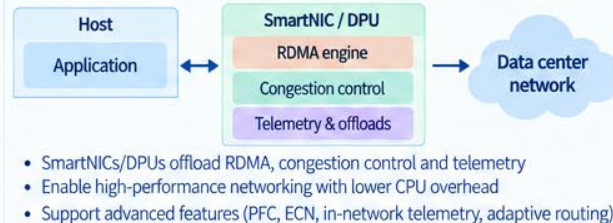
React to congestion in real time



9

Relationship to SmartNICs

Offload, accelerate and provide telemetry



Combining congestion control, adaptive routing and programmable hardware enables low-latency, high-throughput and efficient data-center networks.

FASTER APPLICATIONS
STRONGER INFRASTRUCTURE

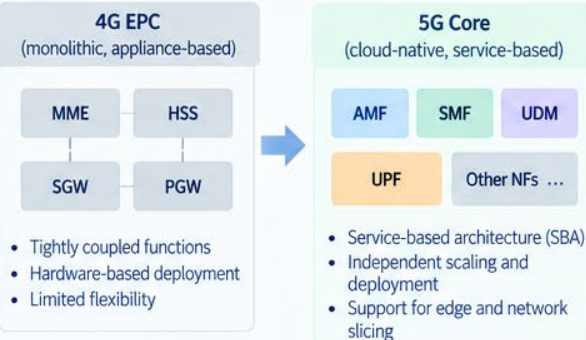
5G Core and Edge Networking

A cloud-native, service-based core with flexible user-plane deployment

MOBILE • CLOUD • EDGE • PROGRAMMABLE

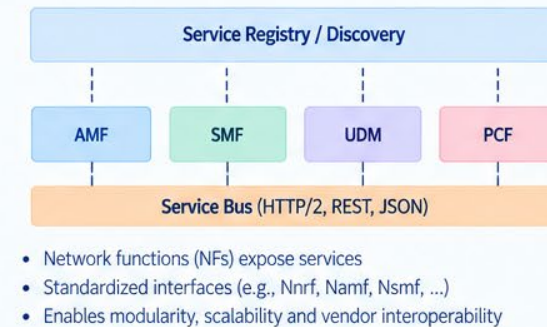
1 From EPC to 5G Core

More flexibility, cloud-native design, service-based architecture



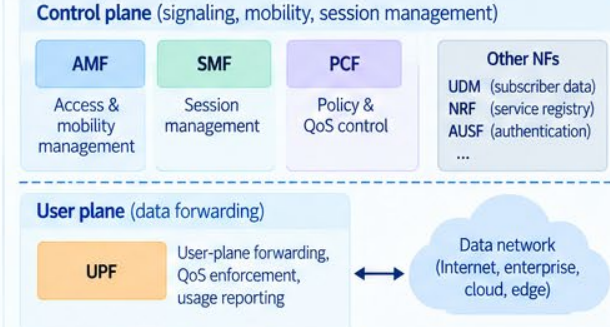
2 Service-Based Architecture (SBA)

Network functions communicate via services



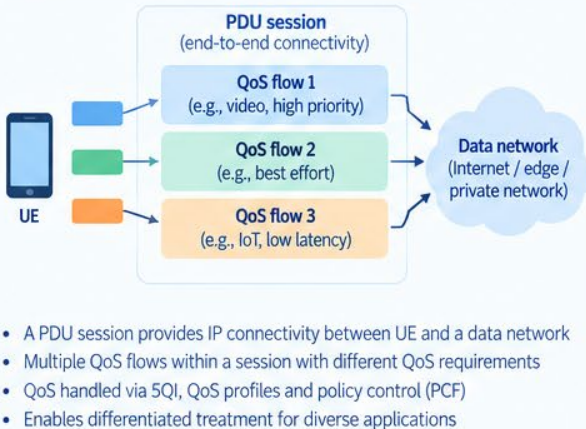
3 Control plane and user plane

Separate functions and different timescales



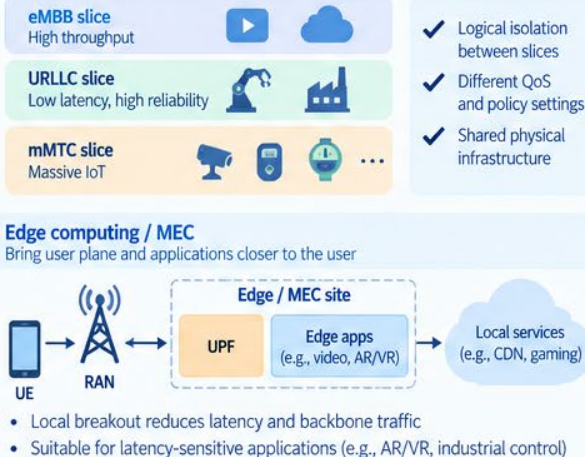
4 PDU sessions and QoS flows

From UE requirements to network forwarding



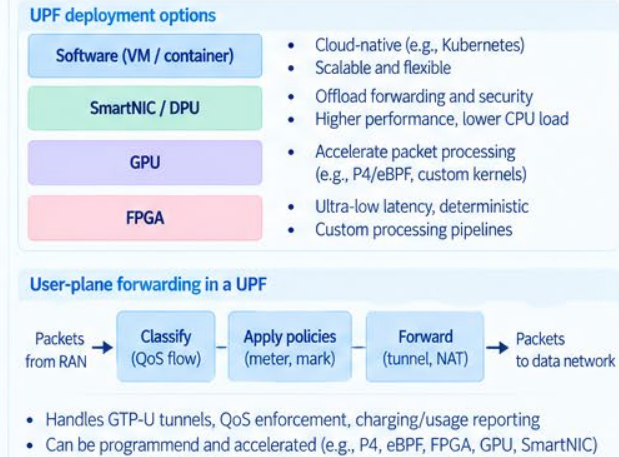
5 Network slicing and edge deployment

Multiple virtual networks on shared infrastructure



6 UPF implementation and acceleration

Flexible deployment from software to hardware



➔ **5G core networking enables flexible, high-performance and service-aware connectivity from the core to the edge.**

CONNECTING PEOPLE, DEVICES AND SERVICES
IN A PROGRAMMABLE 5G NETWORK

1 Why AI workloads stress networks differently

Large-scale, data-parallel training creates massive, synchronized traffic

- Very large models and datasets
- Frequent synchronization between GPUs
- Many-to-many communication (not just client-server)
- High and sustained bandwidth demand
- Sensitive to latency and jitter
- Bursty traffic from collective operations

AI training turns the network into a critical part of the compute system.

4 Communication / computation interaction

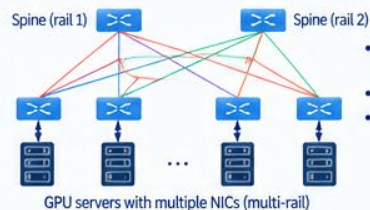
Performance depends on their overlap



- Communication and computation can be overlapped
- Larger models increase the communication fraction
- Synchronization points limit overall performance
- Network jitter and congestion directly affect training time

7 Rail-optimized topologies

High bandwidth and path diversity



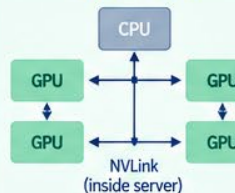
- Multiple parallel networks (rails) for higher bandwidth
- Improved fault tolerance
- Used in large AI clusters (e.g., dual-rail fat-tree)

2 GPU clusters and scale-up vs scale-out

Two levels of interconnection

Scale-up (within a node)

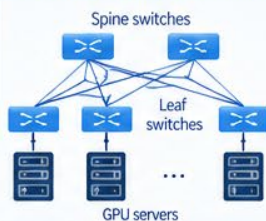
High-bandwidth, low-latency GPU interconnects



High bandwidth (TB/s), very low latency
Shared memory across GPUs

Scale-out (across nodes)

Network interconnect between servers



High throughput, low latency, scalable (e.g., Ethernet or InfiniBand)

5 RDMA / RoCE in GPU clusters

Low-latency, low-CPU data transfer



- RDMA enables direct GPU-to-GPU data transfer (zero-copy)
- RoCE provides RDMA over Ethernet (RoCEv2)
- Reduces CPU overhead and latency
- Widely used in modern AI/HPC clusters

8 SmartNIC / DPU offload

Accelerate communication and data services

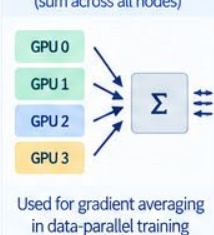


- Reduces CPU overhead
- Accelerates collectives and transport
- Enables in-network processing and security

3 Collective communication

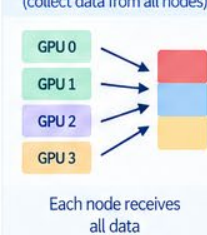
Efficient many-to-many data exchange

AllReduce (sum across all nodes)



Used for gradient averaging in data-parallel training

AllGather (collect data from all nodes)



Each node receives all data

ReduceScatter (reduce and scatter)



Combines reduction with distribution

Collective operations dominate training time and generate large, synchronized traffic.

6 Congestion, synchronization and load balancing

Keep the network stable and GPUs busy

Congestion effects



- Congestion increases latency and reduces throughput
- Can lead to synchronization delays (stragglers)
- Requires congestion control and adaptive routing

Load balancing and adaptive routing

- Distribute flows across multiple paths
- Use topology-aware routing
- Adapt to network load in real time
- Avoid hot spots

Keep all GPUs busy and minimize stragglers.

9 Network requirements of distributed training

- High bisection bandwidth
- Low latency (μs-level)
- High throughput (100+ Gb/s per GPU)
- Low jitter and low packet loss
- Scalability to thousands of GPUs
- Support for collective communication
- Congestion control and adaptive routing
- High availability and fault tolerance

10 Ethernet vs InfiniBand

Two ecosystems for AI/HPC

Feature	Ethernet (RoCE)	InfiniBand
Ecosystem	Broad, commodity	HPC-focused
Performance	Up to 400/800 Gb/s	High (NDR/XDR)
Latency	Low (with RoCE)	Very low
Congestion control	ECN (DCQCN)	Built-in (e.g., CNP)
Scalability	High	Very high
Cost	Lower	Higher
Use cases	Cloud and AI/HPC	Large-scale HPC/AI

High-performance networks are essential for AI and HPC, requiring high bandwidth, low latency and intelligent traffic management, often with hardware acceleration.

NETWORKS POWER
INTELLIGENCE AT SCALE



EE 542 Computer Networks

Fall 2026
