Network Slicing and Programmable Transport

Raouf Boutaba

David R. Cheriton School of Computer Science

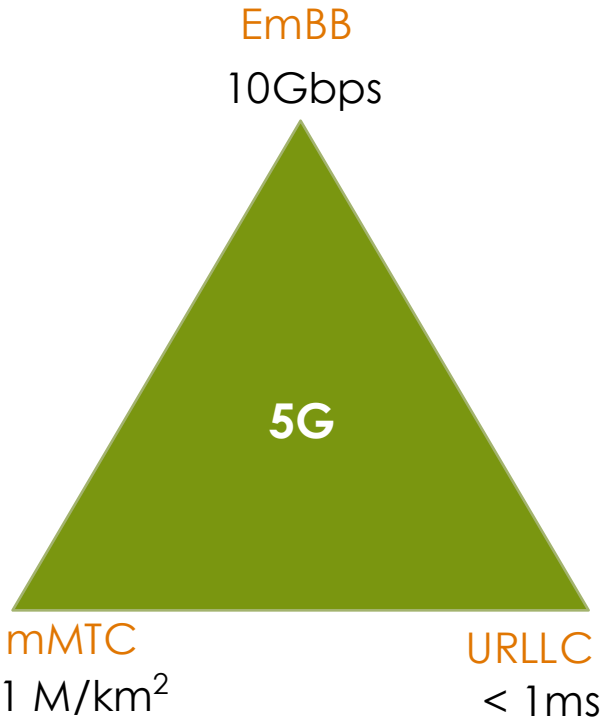
University of Waterloo

Rogers TEP Workshop 3: Transport Networks Concepts #3

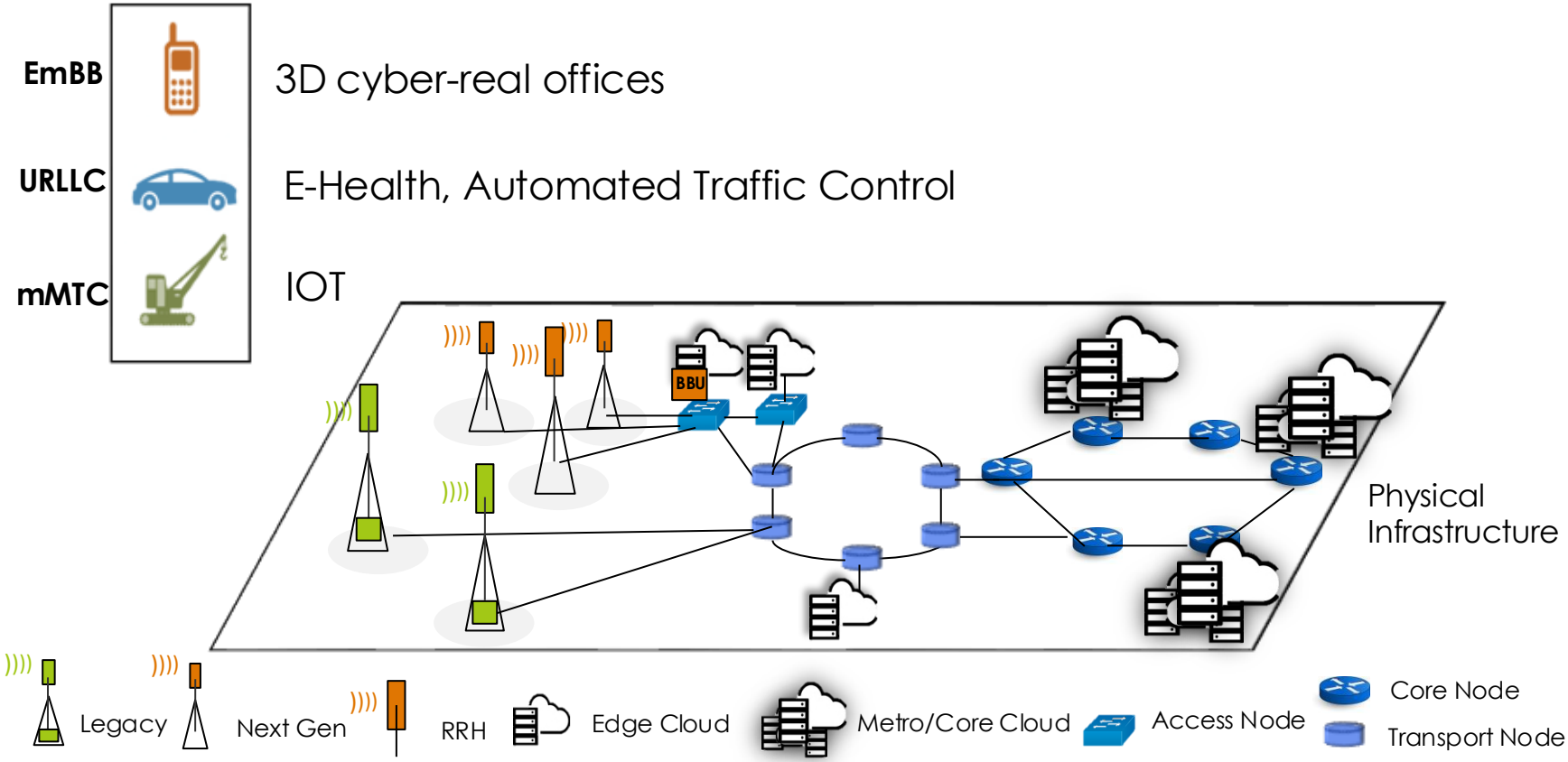
Why Slicing?

Why Transport Slicing? The 5G Challenge

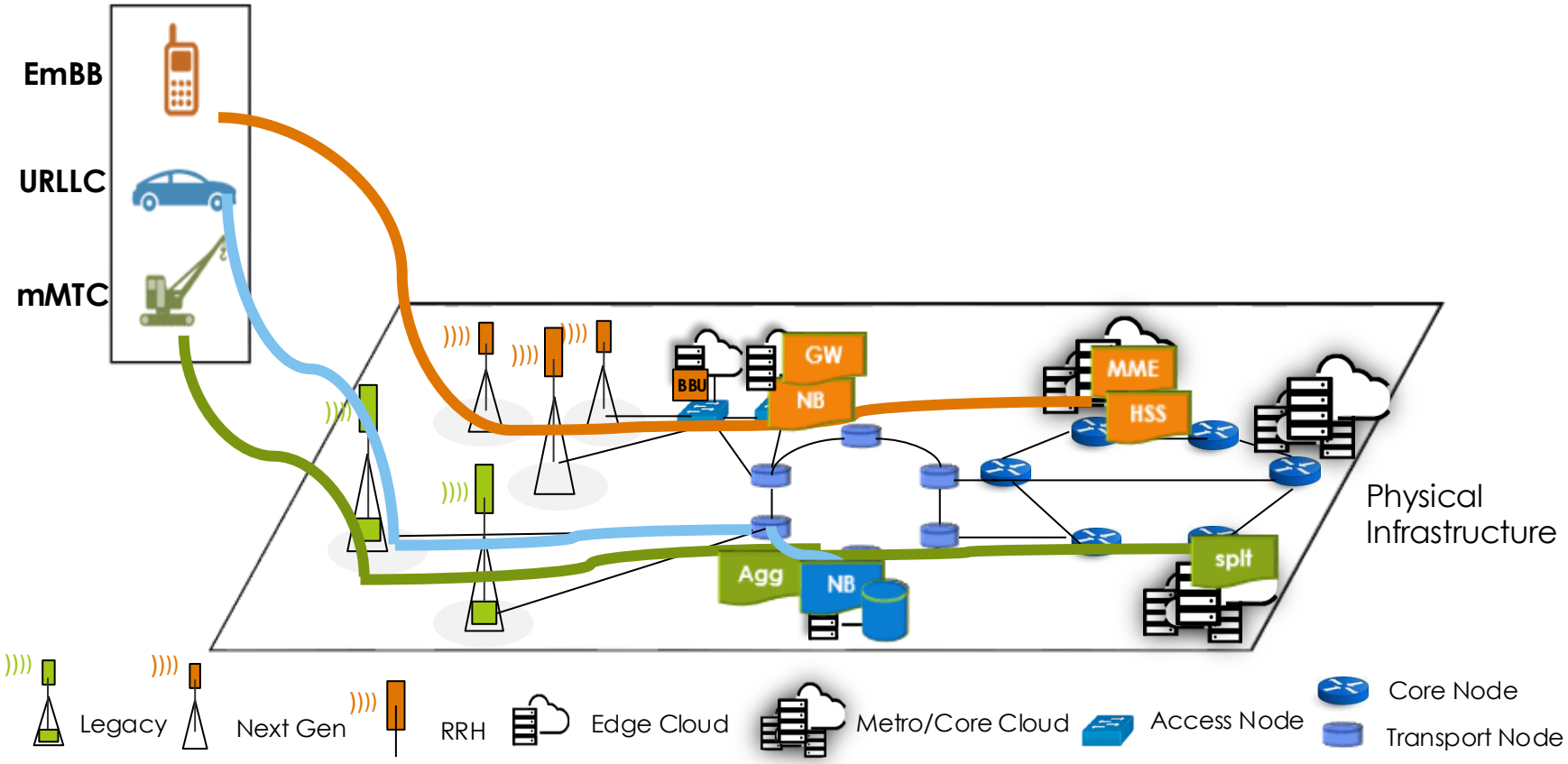
- 3 major service classes :
  - Enhanced mobile BroadBand
    - delivers Gbytes of bandwidth on demand
  - massive Machine-Type Communication
    - connects billions of IoT devices
  - ultra Reliable Low Latency Communication
    - for ultra fast high reliable services
- difficult to support on same network infrastructure
  - Solution: Network slicing (a.k.a. NV)



5G Architecture

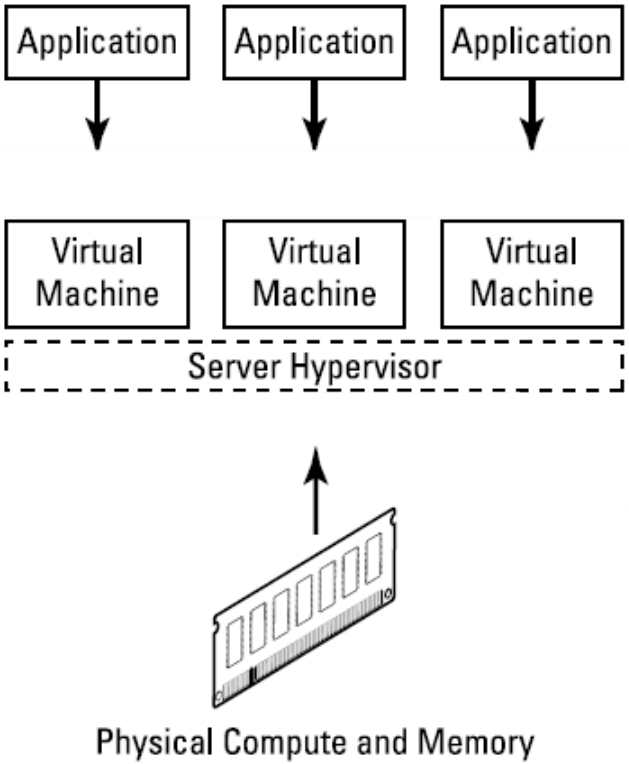


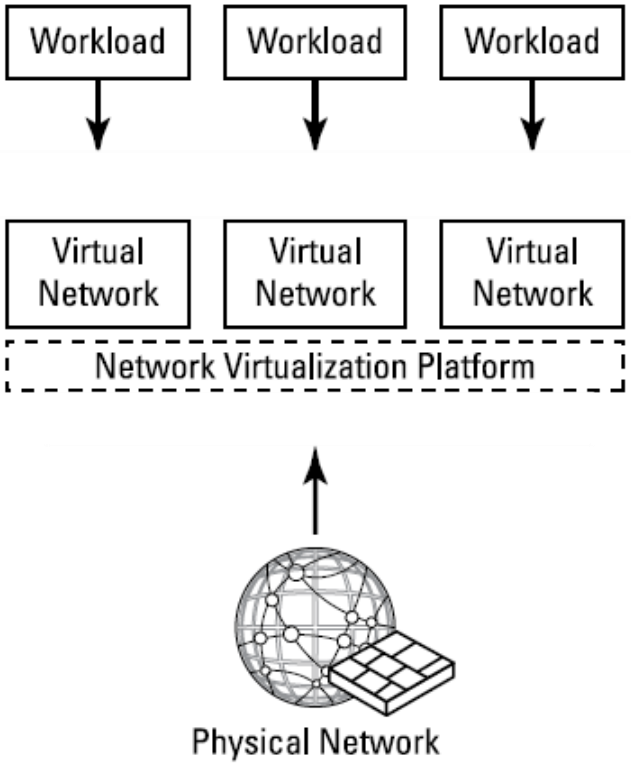
5G Slicing


EmBB
URLLC
mMTC
BBU
GW
NB
MME
HSS
Agg
NB
split
Physical Infrastructure
Legacy
Next Gen
RRH
Edge Cloud
Metro/Core Cloud
Access Node
Core Node
Transport Node

What Network Virtualization means?

Server Virtualization Analogy



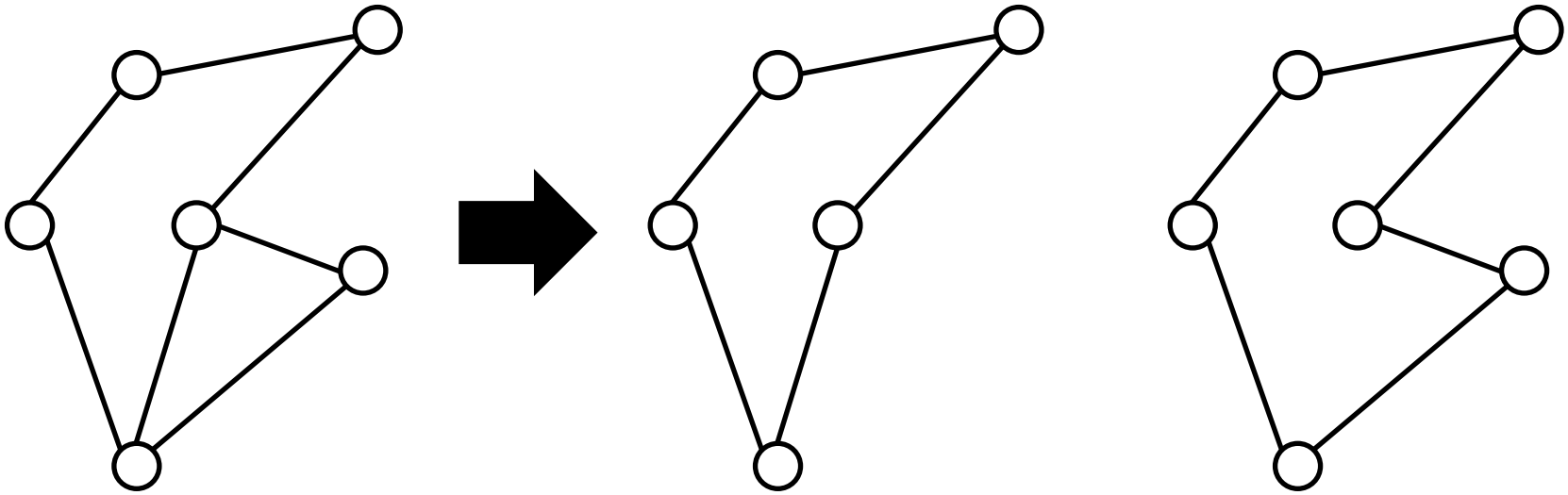


What is Network Virtualization?

Network virtualization (NV) is a networking environment that allows multiple service providers to dynamically compose multiple heterogeneous virtual networks that co-exist together in isolation from each other, and to deploy customized end-to-end services on-the-fly as well as manage them on those virtual networks for the end-users by effectively sharing and utilizing underlying network resources leased from multiple infrastructure providers.

Virtual Networks

- Making a physical network appear as multiple logical ones



- A virtual network is a collection of virtual nodes and links

Resource abstraction

- Hides technology specific details of physical resources
- Represents resources as a uniform set attributes, characteristics, and functionalities
- Gives users the illusion of sole ownership of a resource
- Node and link abstractions
  - Virtual switch/router
  - Virtual Network Interface Card
  - Virtual link
- Network abstraction
  - Collection of virtual nodes connected through virtual links

Resource partitioning & isolation

- Partitions physical resources into multiple independent slices (virtual resources)
- Hard partitioning
  - E.g., dedicated switch ports, dedicated wavelengths
- Soft partitioning
  - E.g., rate limiting, CPU execution capping
- Isolation between slices is critical
  - Hard vs. Soft partitioning

How partitioning/isolation work?

Node virtualization

- Node resource partitioning
  - CPU, Memory
  - Interfaces/ports
  - Optical switching resources, e.g., crossconnects, transponders, ROADMs, etc.
  - Wireless infrastructure, e.g., antennas, radio resource controllers, access points, etc.
- Data plane partitioning
  - Forwarding table
  - Address space
- Control plane partitioning
  - Routing tables
  - Protocols

Link virtualization

- Physical medium partitioning
  - Optical fiber, Copper wire, Radio spectrum, etc.
  - Frequency/Wavelength/time multiplexing
  - Must consider physical layer constraints and physical impairments
- Datapath virtualization
  - Frame labeling and tagging (Enabled by nodes)
  - Tunneling
    - Ensures isolation of traffic from multiple virtual networks transported over a shared network
    - Provides direct connection between non adjacent network devices
    - Uses encapsulation and occasionally encryption
    - E.g., IEEE 802.1Q, L2TP, GRE, IPsec, SRv6

Path Control and Segment Routing

From SDN to Transport Slicing

- Network virtualization predates SDN
  - Can be realized without SDN
- SDN can facilitate network virtualization (InP view)
  - Creating VNs (e.g., through flow space partitioning)
  - Facilitating management (e.g., VN migration)
- Virtual networks can be SDN enabled (SP view)
  - Each VN governed by separate SDN controller
  - Benefit from SDN open interfaces
- Together, SDN + NV give us programmable, isolated networks which is the foundation of transport slicing

Traffic Steering in Transport

- SDN gives us programmable connectivity and flexible policy
  - ONOS manages control plane state across the network automatically
- Transport slicing requires a complementary data plane capability
  - Steer traffic through specific nodes (e.g. a firewall, a low-latency path)
  - Enforce waypoints regardless of shortest-path routing
- Expressing path intent at scale
  - Per-switch flow rules work, but require state at every hop
  - More scalable: embed the path plan directly in the packet
  - Core nodes follow instructions already in the packet header
- This is the key idea behind Segment Routing
  - SDN and Segment Routing work together: control plane + path primitives

Segment Routing

- Source routing paradigm: ingress node determines the complete path
  - Path encoded as an ordered list of *segments*
  - Each segment is an instruction for the network
- Segments can represent
  - A node to visit (e.g. a firewall, a low-latency router)
  - A link to traverse
- No per-hop path state in the core
  - Transit nodes forward based on current segment only
  - Segment list carried in the packet header
- Segment Routing can run over two data planes
  - SR-MPLS: uses MPLS label stack
  - SRv6: uses IPv6 header extensions → what we use today

SRv6: Segments as IPv6 Addresses

- Each segment is an IPv6 address called a Segment ID (SID)
  - Any node, link, or service function can be assigned a SID
  - 128-bit address space enables rich instruction encoding
- The Segment Routing Header (SRH)
  - An IPv6 extension header added to the packet at the ingress node
  - Contains the ordered list of SIDs i.e., the complete path plan
  - The packet carries its own routing instructions
- SRv6 runs natively over IPv6
  - No new protocols required in the core
  - Works alongside existing IPv6 infrastructure

SRv6 for Transport Slicing

- A transport slice requires two properties
  - Path control: traffic must traverse specific nodes
  - Resource isolation: QoS (e.g., bandwidth) guarantees, per slice
- SRv6 provides path control
  - Each slice assigned a distinct SID list
  - Traffic is steered through the correct path automatically
- Resource isolation mechanisms
  - Hardware: dedicated resources per slice
  - Software: bandwidth guarantees through queuing and scheduling
  - In this workshop: OVS HTB queues → one practical realisation
- Lab 3: Path steering with SRv6
- Lab 4: Putting everything together ~ transport slicing