Cutting Packet Fat in Shallow VNF Chain Processing

**Problem**
- Datacenter operators strive to simultaneously provide performance guarantees and optimize resource utilization
- Clients want minimal end-to-end latency for VNF chain processing

**Shallow VNFs Observation**
- Some VNFs process only packet header (shallow VNFs)
- We can save bandwidth by not transmitting payload to such VNFs
- Less Tx/Rx of data will reduce overall packet processing latency

**Proposal: SMP**
*Split-Merge Payload (SMP)* approach will:
- Split packet into header H and payload P [1]
- Forward header H to the VNF chain F₁, F₂, ..., Fₙ
- Merge the output header H' with payload P

**SMP Semantics**
SMP and non-SMP deployments will:
- Preserve VNF topology
- Treat policy-driven and failure-driven packet drops identically

**ToR Deployment**
- SMP operation is transparent to user
- Lowest memory pressure at ToR switch
- Can handle payload processing VNFs
  - Lowest throughput and latency gain
  - Same payload privacy as non-SMP

**Core Switch Deployment**
- SMP operation is transparent to user.
- Higher throughput and latency gain
- Can handle payload processing VNFs
  - Same payload privacy as non-SMP
  - Memory pressure requires external store

**End-host Deployment**
- Least payload overhead on cloud
- Highest throughput and latency gain
- Best privacy for user data
  - SMP operation is not transparent to user
  - Cannot handle payload processing VNFs

**Design Goals**
- Equivalent semantics between SMP and non-SMP deployment
- Design should work within the memory constraints of the SMP switch [2]
- Better latency & throughput of SMP deployment than non-SMP

**Research Questions**
- Can SMP be extended to include payload processing VNFs?
- Can we generalize SMP to process only a subset of the payload [3]?
- At what point will split/merge overhead impede overall packet processing?

---

