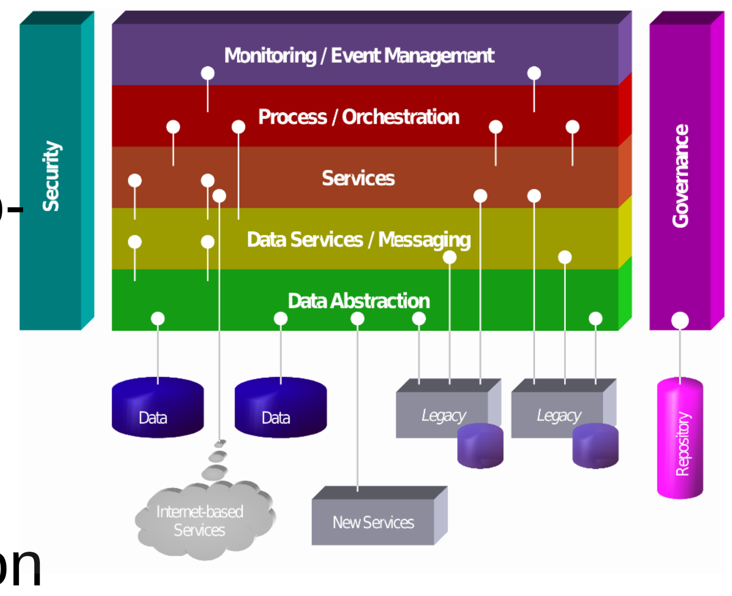
Software-defined Adaptive Resource Management for Cloud-hosted Group Communications Applications

Work supported by US National Science Foundation Ignite Program: CNS #1531079

Prithviraj Patil[1], Akram Hakiri[2] and Aniruddha Gokhale[1]
[1]Vanderbilt University, Nashville, TN USA & [2]ISSAT Mateur, Univ de Carthage, Tunisia

Motivation: Group Communications in the Cloud

- Many cloud-based applications and infrastructure management tasks require group communication semantics
- Illustrate event-based publish subscribe semantics with many-to-many communication patterns



- Examples:
  - Cloud resource monitoring
  - Active/passive data replication
  - Software upgrades and security patching
- *IP Multicast (IPMC) provides one possible solution*

Limitations of IP Multicast

- Stability and scalability issues
- Address structure does not lend itself to hierarchical aggregation => expensive to reconstruct multicast trees
- Use of IGMP requires high CPU and memory resources
- Significant manual efforts needed in addressing these challenges => error-prone and often *ad hoc*

Key Research Challenges Beyond IPMC

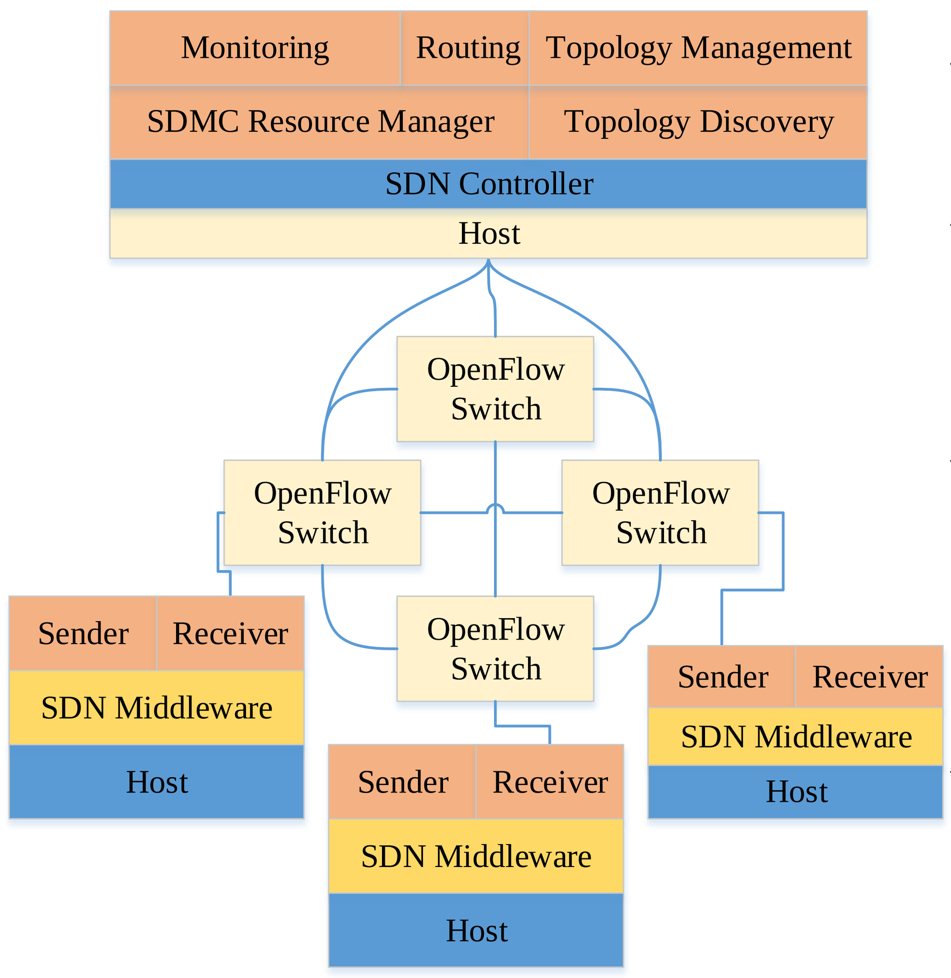
- Estimating the available egress network bandwidth at each switch is critical to ensuring group communication requirements are met
- Network switches hold state for each flow => switch memory utilization must be kept below a threshold to prevent buffer overflows
- However, a cloud data center comprises hundreds of network switches

How to instrument, coordinate and maintain a consistent and global view of the entire system of network switches and link utilization to support competing group communication flows?

Solution Design Considerations

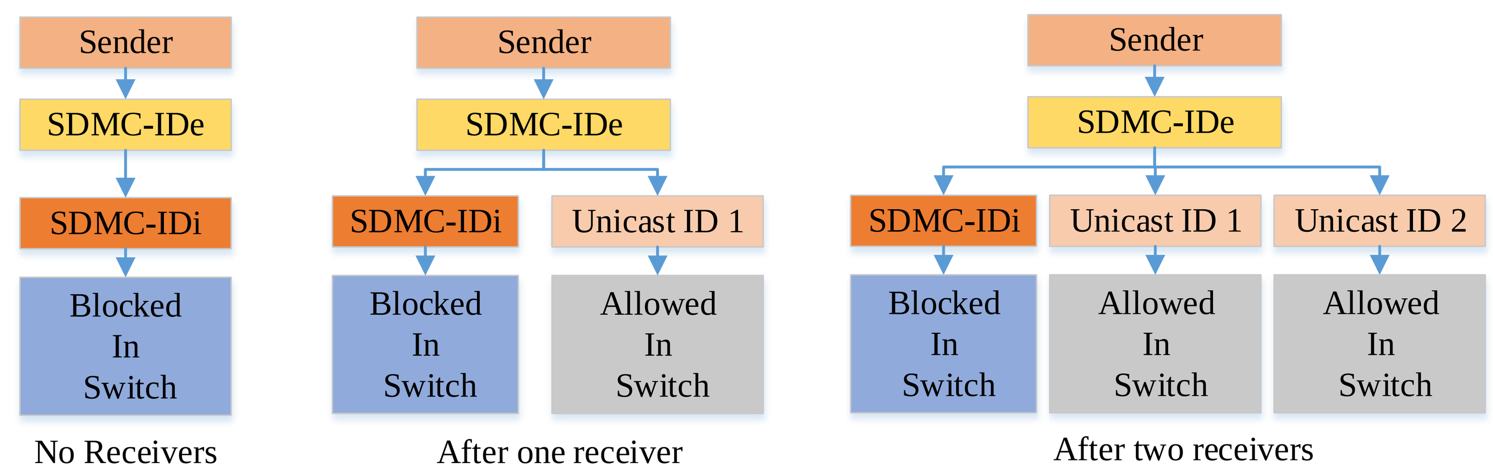
Property	Consideration
Flexible/Dynamic and application agnostic	Allow application to transparently switch between unicast and multicast
Support reuse of overlapping multicast trees	Be able to reuse partial or complete multicast trees, which is not allowed by IPMC
Minimize initial setup latency	Creation of multicast senders and receivers should not create latency issues for receivers due to immediate reconstruction of multicast trees
Adapting to network load and memory consumption	Be able to trade-off between network link and network switch memory utilization

Contributions: Software-defined Multicast (SDMC)



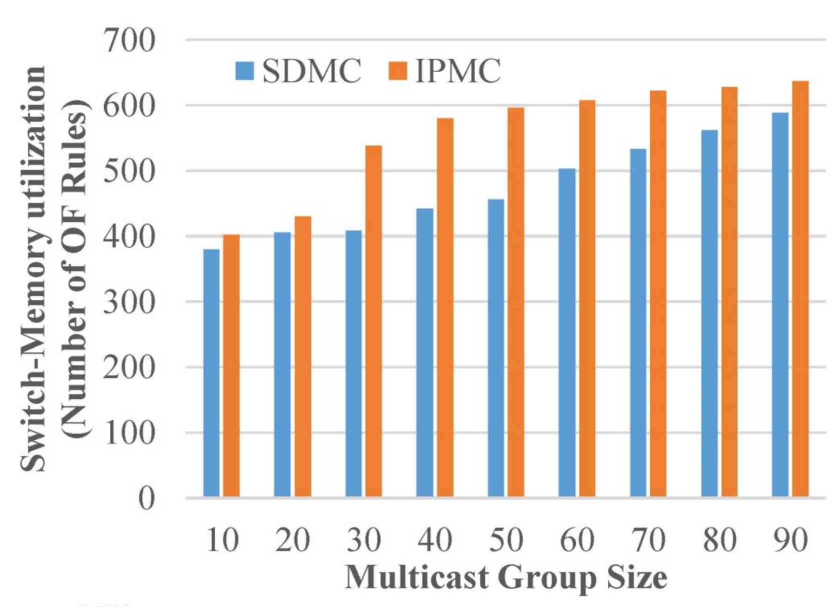
- Control and data plane separation
- Control plane comprises the SDN controller with the adaptive intelligence
- SDMC capabilities provided by five different building blocks within the controller
- Data plane comprises routers connected to hosts communicating via SDMC middleware

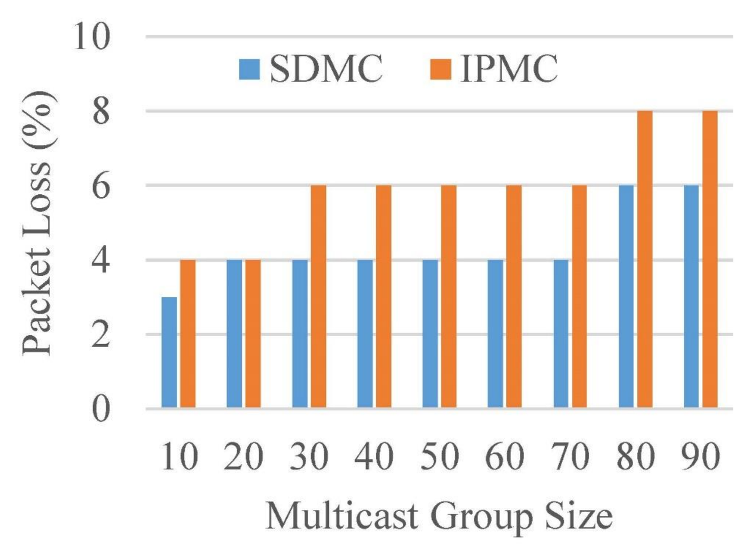
SDMC's 2-Level ID Structure

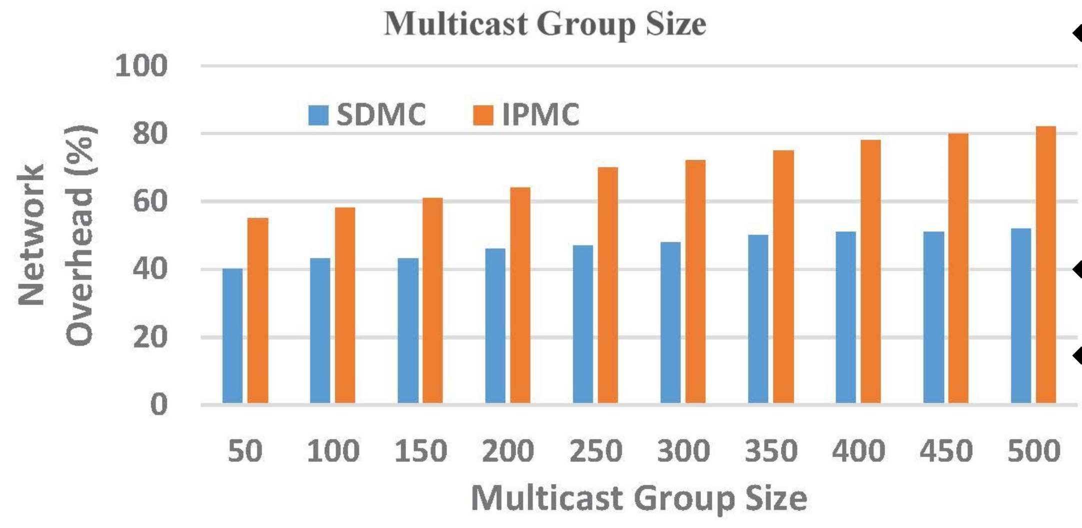


- 2-level SDMC ID structure: application always uses the external ID, which can be mapped to either a unicast or multicast internal ID
- SDMC supports lazy evaluation: initially all flows start as unicast; multicast remains blocked
- SDMC makes a trade-off between network link and switch memory utilization by adaptively switching between unicast and multicast transparently to the application

Experimental Results







- Experiments conducted using Mininet & OpenFlow
- POX controller
- Jelly-fish, Tree, Fat tree and random topologies

Ongoing Work

- Global-scale optimization across competing flows
- Dependability of distributed controllers and coordination
- Extending the capabilities to wireless networks at the fog and edge networks

Source code can be downloaded from: https://github.com/prithviraj6116/sdmc