MC-SDN: Supporting Mixed-Criticality Scheduling on Switched-Ethernet Using Software-Defined Networking

Jaeyoung Lee

Yonsei University

January 30, 2026

Part I

Introduction

Research Motivation: The Issue

The Shift to Switched Ethernet

- Recent embedded systems, communication technologies and CPS achieves its function using real-time sensing and dynamic control.
- Such system requires increased demands on bandwidth and latency requirements.
- Switched Ethernet is now used to solve such network issue (which goes from point to point), replacing previously used bus based network.

Mixed-Criticality (MC) Systems

- Mixed-criticality systems are being implemented to reduce cost.
- Conflict: Logical separation between applications on different criticality levels vs. efficient scheduling of shared resources.
- **Solution**: Mode-based MC scheduling, providing different level of schedulability guarantee for different system modes.
- Mode-based MC scheduling on switched ethernet network is novel .

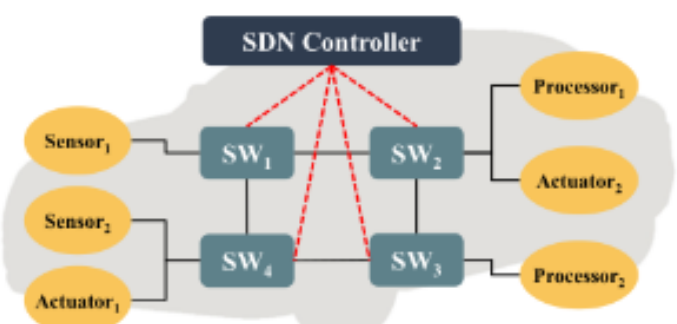
PROPOSED APPROACH: UTILIZING SDN

Problem: Static Nature of Traditional Switches

- Traditional switches have static nature that can not support dynamic behavior of mode-based MC scheduling.
- Cannot change scheduling behavior at runtime.

Solution: Software-Defined Networking (SDN)

- This paper support MC flows on event-triggered switched Ethernet networks using SDN .
- SDN allows updating forwarding policy dynamically in a programmatic way.



(a) MC Networking system overview

ADDRESSING SDN CHALLENGES: MC-SDN

Problem 2: SDN for MC Scheduling

- Pre-existing SDN switches can not detect release interval and size of periodic message from the flow.
- Vanilla OpenFlow protocol imposes long and unpredictable delay while conducting mode change.

Solution 2: MC-SDN Framework

- This paper presents novel SDN-Based network system named MC-SDN .
- Effectively supports flow monitoring and mode change for MC scheduling.

Contribution of the Paper

Key Contributions

- **MC-SDN Support**: Supports MC scheduling in switched Ethernet.
- **Limit Analysis**: Analyzes limitations of standard SDN interface and identifies major delay factors on mode change.
- **Analytic Bound**: Derives a worst-case mode-change delay bound under MC-SDN.
- **Evaluation**: Evaluates MC-SDN to show the improvement.
- **Case Study**: Demonstrates the effectiveness by a case study of scaled autonomous car.

Part II

System Model and Background

System Model: Flow and Properties

Flow Model

- Flow : A set of potentially unbounded series of periodic messages.
- Unlike CPU tasks, messages are divided into multiple packets based on the MTU.
- **Properties**: ⟨period, size, deadline, criticality, source, destination, route⟩.

Key Definitions

- **Period**: Minimum separation time between consecutive messages.
- **Size**: Maximum byte size of each message.
- **Deadline**: Relative end-to-end deadline.

Priority-based Scheduling and MC Logic

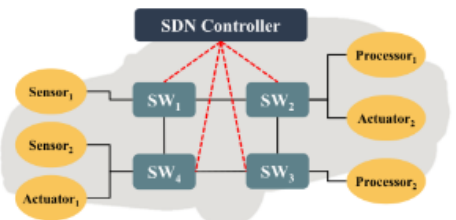
Flow Scheduling

- Switches use priority-based scheduling to forward high-priority packets first.
- Priority of each flow is specified by the forwarding rules of the switch.
- **Assignment**: Uses Rate Monotonic (RM) assignment (supports any fixed-priority policy).

Mixed-Criticality (MC) Mechanism

- Uses dual-criticality systems with HI (high) and LO (low) levels.
- **Logic**:
  1. The system starts on LO mode.
  2. And runs on LO mode until message are generated more frequently or larger than LO requirement.
  3. If so, the system changes its mode into HI mode.
  4. Then each switch update its forwarding table with HI mode rule.
  5. May drop the LO flows or promote the priority of HI flows

SDN Architecture and OpenFlow



(a) MC Networking system overview

Decoupled Control and Data Planes

- Control Plane : External software that determines and handles network policies.
- Data Plane : Forwards packets based on policies received from the control plane.
- Enables dynamic and flexible network control at runtime.

OpenFlow Interface

- The most popular interface between control and data planes.
- **Forwarding Table**: Stores rules (flow entry) as Match-Action tuples.
- **Match**: Identifies packets via fields (e.g., source/destination IP).
- **Action**: Instructions on how to handle the matched packets.

Part III

Challenges of MC Scheduling on SDN

Challenges in Vanilla SDN

Lack of Real-Time Mechanisms

- Vanilla SDN lacks proper mechanisms for:
  - Mode violation detection.
  - Mode change protocols.
  - Real-time support.
- Centralized control paradigm may yield long and unpredictable delays.

Significant Delay Factors

- Mode change delays can compromise the timing guarantees of high-criticality flows.
- Necessary to identify and analyze bottlenecks in the mode-switching process.

Mode Change Delay: Measurement and Definition

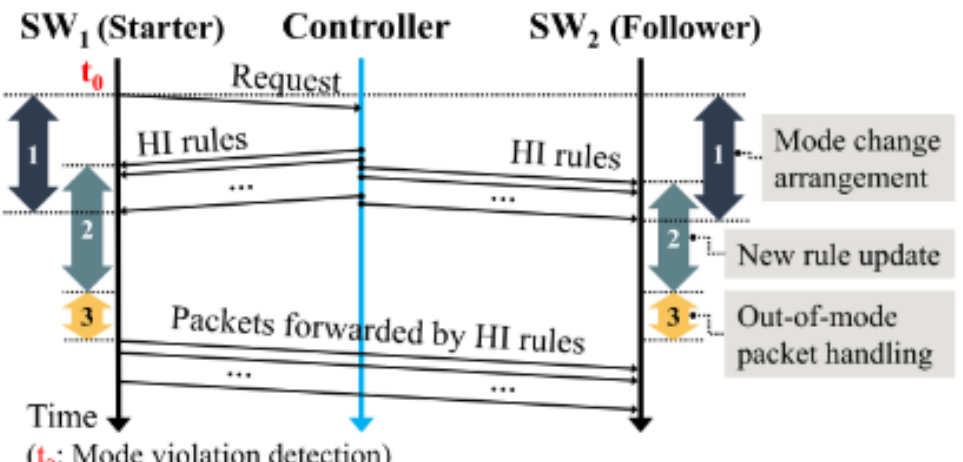
Experimental Setup

- **Approach**: Basic controller-driven mode change.
- **Trigger**: Used a predefined flag to trigger mode switch.

Definition of Mode Change Delay

- The elapsed time between:
  1. The time to send a mode change request.
  2. The time all switches are ready to handle packets based on new HI mode rules.

Significant Delay Factors



(b) Std-SDN: Controller-driven mode change

Delay Factor 1: Mode Change Arrangement

Controller-Switch Communication

- Involves OpenFlow communication between the controller and switches.
- **Process**:
  1. Detect HI mode behavior (at Switch A).
  2. Request mode change to the controller.
  3. Controller sends HI mode rules to every switch.
  4. Every switch recognizes the change and receives new rules.
- Challenge: Difficult to bound delay as the controller consists of complicated software layers.

Delay Factor 2: New Rule Update

Intra-Switch Communication

- Communication between internal modules of the switch.
- **Process**:
  1. **Switch Manager Module**: Receives rules from the SDN controller.
  2. **Datapath Module**: Rules are transferred to the forwarding logic.
- Overlap : Can partially overlap with the arrangement step, but not significantly.
- Challenge: Internal communication channels are highly complicated and hard to bound.

Delay Factor 3: Out-of-Mode Packet Handling

Physical Link Constraints

- **The Issue**: After rules are updated, packets enqueued under LO mode rules may remain in the queue.
- **Impact**: Stale LO packets delay the transmission of new HI flows.
- Challenge: Hard to reduce as it depends on link speed .

	Avg.(Stdev.) (ms)
Overall	50.65 ($\pm$17.94)
Mode change arrangement	31.83 ($\pm$10.54)
New rule update	12.06 ($\pm$0.52)
Out-of-mode packet handling	9.12 ($\pm$8.06)

TABLE I: Breakdown of mode change delay

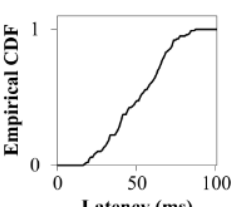


Fig. 2: Empirical CDF of mode change delay

Part IV

MC-SDN: System Design

MC-SDN: Switch-Driven Paradigm

Solving Vanilla SDN Challenges

- MC-SDN supports real-time MC scheduling by shifting the control logic.
- Switch-driven Mode Change : Switches now detect violations and enable mode changes independently, bypassing the controller-driven bottleneck.

Predictability and Performance

- Significantly reduces mode change delay.
- Strictly limits delays to predictable upper-bounds.

FOUR ADDITIONAL COMPONENTS

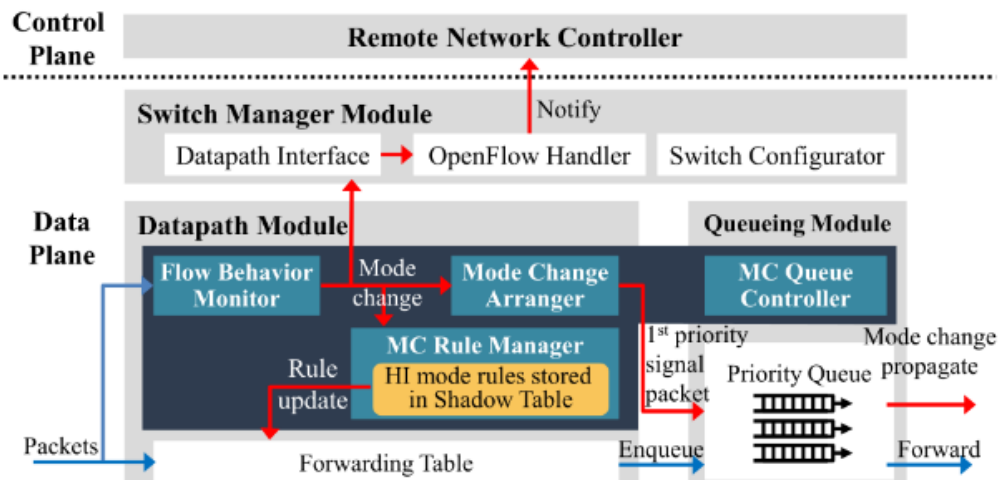


Fig. 3: MC-SDN system architecture

Component 1: Flow Behavior Monitor

Monitoring function added to detect flow that violate mode-specific requirements

Notation (for each message k in flow τ)

- $C_{\tau,k}$: size of message k
- $E_{\tau,k}$: arrival time of message k (absolute value)
- $G_{\tau,k}$: guide time of message k (absolute value)
- $T_{\tau}(LO)$: LO period (relative value)
- J_{τ} : release jitter (relative value)

Component 1: Flow Behavior Monitor

Monitoring Mode Violations

- Detects flows that violate mode-specific requirements (message size and arrival interval).
- HI-mode Detection Logic :
  - $E_{\tau,k} < G_{\tau,k}(\mathrm{LO}) \vee C_{\tau,k} > C_{\tau}(\mathrm{LO})$

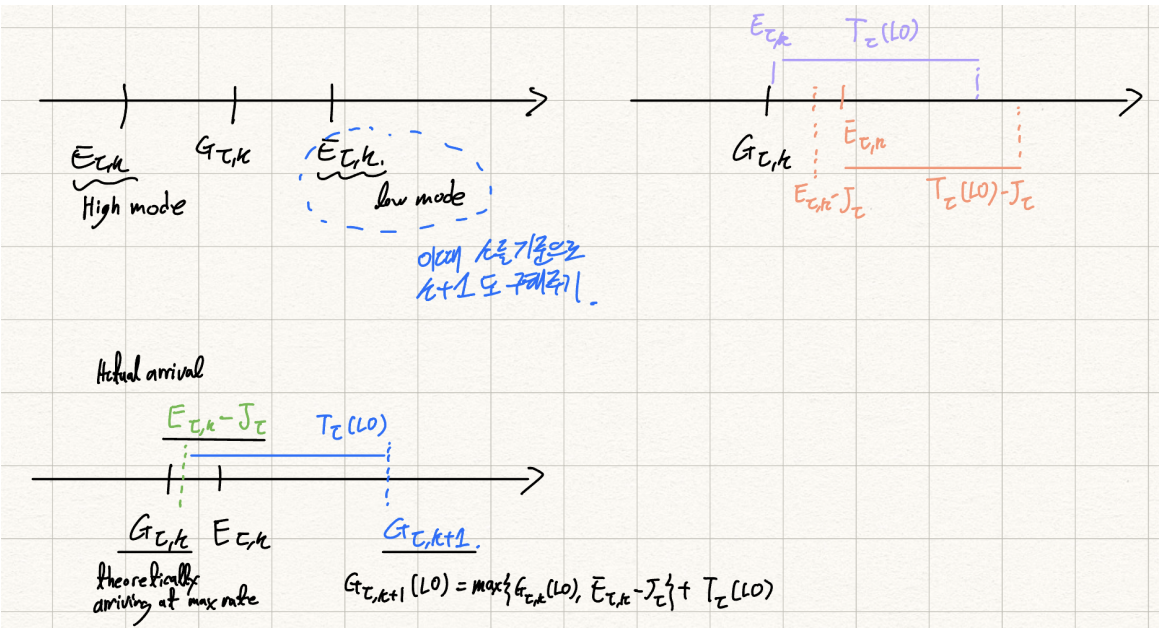
Guide Time ($G_{\tau,k}$)

- A dynamic threshold to distinguish LO and HI modes.
- **Formula**:

$$G_{\tau,k}(\mathrm{LO}) = \max\{G_{\tau,k-1}(\mathrm{LO}),\ E_{\tau,k-1} - J_{\tau}\} + T_{\tau}(\mathrm{LO})$$

- Two consecutive messages can be as close as $T_{\tau}(\mathrm{LO}) - J_{\tau}$ or $T_{\tau}(\mathrm{LO})$.

Guide Time Explanation



Component 2 & 3: Mode Change Arranger and Rule Manager

Distributed Mode Change Triggering

- The switch triggers the mode change and notifies other switches via signal flooding.
- Eliminates the need for slow communication with the SDN controller.

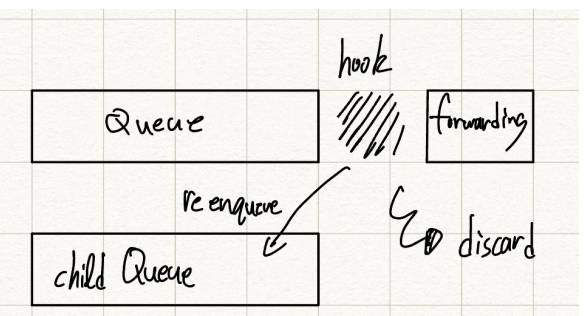
MC Rule Manager

- **Forwarding Table**: Stores default LO mode rules .
- **Shadow Table**: Pre-stores (caches) HI mode rules .
- Eliminates the rule download delay from the controller during mode changes.
- Updates the forwarding table using locally cached rules from the shadow table.
- Optimization : Places the shadow table directly within the packet forwarding module to minimize intra-switch communication.

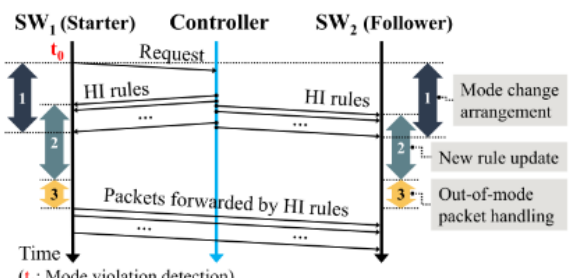
COMPONENTS 4: QUEUE CONTROLLER

MC Queue Controller

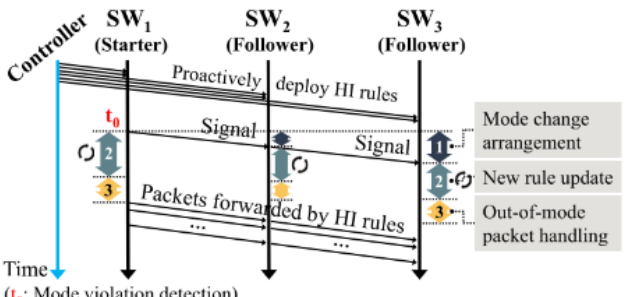
- Rearranges the priority queue based on the new HI rules.
- Hooking Mechanism: Hooks enqueued packets before transmission to discard or re-queue LO flow packets according to HI mode requirements.



MC-SDN: System Design



(b) Std-SDN: Controller-driven mode change



(c) MC-SDN: Switch-driven mode change

Part V

IMPLEMENTATION

Target System and OVS Overview

Target System Environment

- **Open vSwitch (OVS)**: Version 2.4.90.
- **Network Controller**: POX network controller.
- **Operating System**: Linux version 3.10.107.

OVS Components

- **vswitchd**: User process for switch management.
  - Contains forwarding table, OpenFlow Handler, and Datapath Interface.
- **Kernel Datapath**: Linux kernel module for packet forwarding.
  - Contains the cached forwarding table .

OVS Overview

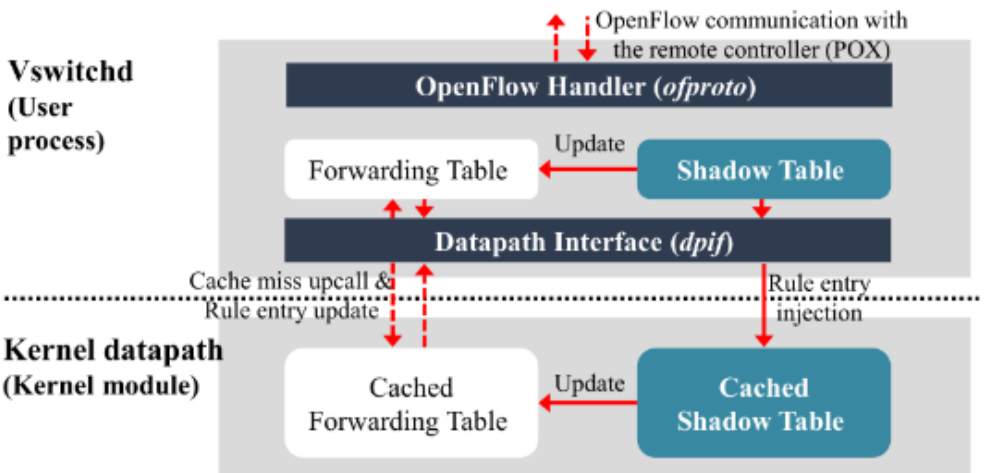


Fig. 4: MC-SDN implementation on Open vSwitch

Shadow Table Implementation

Forwarding Logic and Cache Miss

- Kernel datapath looks up the cached forwarding table.
- On a cache miss, it notifies vswitchd to bring the matching rule.

Proactive Shadow Table Update

- Naive Way: Invalidating rules and updating upon cache miss causes non-negligible delay.
- MC-SDN Approach : Updates the cached shadow table proactively via the datapath interface dpif.
- Injects/removes rules into the cached shadow table without the standard cache miss/update protocol.

OpenFlow Extension and Monitoring

Kernel-level Implementation

- Flow behavior monitoring and mode change arrangement are implemented as new functions in the kernel datapath .
- Implementation is realized through OpenFlow extensions.

Required System Extensions

- New APIs for POX applications and extended encoder/decoder in POX library.
- Extended message handlers and interfaces in ofproto and dpif.
- New data structures in each OVS module.

MC Queue Controller

Queue Management using TC-PRIO

- Utilizes the PRIO Linux queuing discipline .
- Does not modify the queuing logic itself to avoid side effects on forwarding performance.

Hooking Routine

- Places a hooking routine between dequeuing and transmitting packets.
- **Requeue/Discard**: Uses pointer copy to requeue LO flows into child queues or memory free operations to discard them according to HI mode rules.

Part VI

Mode Change Delay Analysis

Worst-Case Mode Change Delay (D_{mc})

Analytic Bound Definition

- The total delay is the sum of three distinct components:

$$D_{mc} = D_{arrange} + D_{update} + D_{q-handle}$$

Application to Schedulability

- By adding D_{mc} to the transmission time of HI flows, it can be incorporated into schedulability tests .
- Ensures the system meets all deadlines even during the mode transition period.

Delay Factor 1: Mode Change Arrangement ($D_{arrange}$)

Signal Propagation Delay

- Time to propagate signal packets to all switches in the network.
- **Formula**:

$$D_{\text{arrange}} = (d_{\text{trans}} + d_{\text{prop}} + d_{\text{queue}} + d_{\text{proc}} + d_{\text{flood}}) \cdot N_{\text{link}}$$

Variable Determinants

- Physical Properties : $d_{\text{trans}}, d_{\text{prop}}, d_{\text{queue}}$ (Link bandwidth, length, and driver buffers).
- Switch Architecture : $d_{\text{proc}}, d_{\text{flood}}$ (Processing and flooding overhead).
- N_{link}: Maximum hop distance between any two switch nodes.

Delay Factors 2 & 3: Rule Update and Queue Handling

Iteration of Simple Operations

- Both components are functions of the number of items (N_{rule} and N_{packet}).

Formulas

- **Rule Update**: $D_{\text{update}} = d_{\text{copy}} \cdot N_{\text{rule}} + d_{\text{u-misc}}$
- **Queue Handling**: $D_{\text{q-handle}} = d_{\text{q-handle}} \cdot N_{\text{packet}} + d_{\text{q-misc}}$

Parameters

- d_{copy}: Max time to copy from shadow table to forwarding table.
- $d_{\text{q-handle}}$: Max time to handle (discard/re-queue) one out-of-mode packet.
- $d_{\text{u-misc}}, d_{\text{q-misc}}$: Fixed overheads for initialization and finalization.

Obtaining Physical and Empirical Parameters

Physical Measurements

- d_{prop}: Based on 100-meter Ethernet cable properties.
- d_{trans}: Calculated as $\frac{58\ bytes}{100 Mbps}$ (Fixed signal packet length).
- d_{queue}: Calculated from combined storage of device driver $(5,460\ bytes)$ and NIC hardware $(2,048\ bytes)$ over $100 Mbps$.

Empirical Sampling

- d_{proc} and d_{flood}: Estimated using maximum values within 99.5% confidence intervals.

Component	Bound (μs)	Average (μs)	Stdev (μs)	Number of samples
Overall	1453.45	-	-	-
d_{prop}	0.53	-	-	-
d_{trans}	4.6	-	-	-
d_{queue}	600.64	-	-	-
d_{proc}	766.90	764	41	1300
d_{flood}	80.78	80.02	1.61	30

TABLE II: Upper-bounds of mode change arrangement delay components

Also Empirical Sampling

- Used `getnstimeofday()` kernel function for high-precision timestamps.
- Measured directly within the kernel's datapath module .
- $d_{\text{q-handle}}$ is bounded by the discard operation (memory free), which is more expensive than pointer copying.

System Capacity Upper Bounds

- N_{rule}: Maximum rules the table can hold.
- N_{packet}: Maximum packets the queue can store.

Component	Bound (μs)	Average (μs)	Stdev (μs)	Number of samples
d_{copy}	3.95	3.80	0.94	280
d_{u-misc}	50.04	49.72	0.68	30
$d_{q-handle}$	1.06	1.04	0.44	5200
d_{q-misc}	2.07	1.92	0.32	30

TABLE III: Upper-bounds of new rule update and out-of-mode packet handling delay components

Advanced Assurance Techniques

Further Tightening the Bounds

- **Static WCET Analysis**: Can be applied for higher theoretical assurance.
- **Architectural Optimization**:
  - Tighten worst-case bounds by considering system cache structures .
  - Rearranging rule update orders to improve cache hit rates.

Part VII

EVALUATION

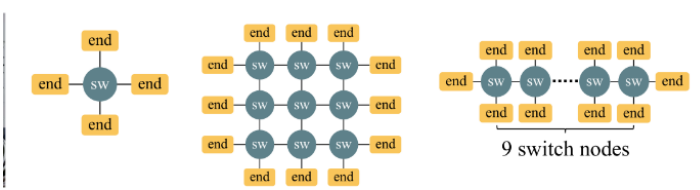
Experimental Setup

Network Testbed

- **Components**: 20 end nodes, 9 software switches, and a SDN controller.
- **Hardware**: Switch nodes with 4 additional USB Ethernet interfaces via USB 2.0 hubs.
- **Connectivity**: 100Mbps Ethernet for switch and end nodes.
- **Management**: Dedicated Ethernet interface for the remote SDN controller.

Network Topology

- Evaluated across various topologies (Star, Grid, Linear).



(b) Star topology (c) Grid topology (d) Linear topology

Evaluation Metrics

Performance Measures

- **Mode Change Delay**: Time from violation detection until all switches update forwarding tables and penalize OOM packets.
- **End-to-End Transmission Time**: Time required to transmit a message from source to destination.

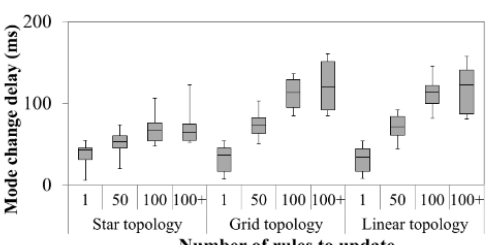
Mode-based Scheduling

- RM Assignment : Switches prioritize packets according to Rate Monotonic policy.

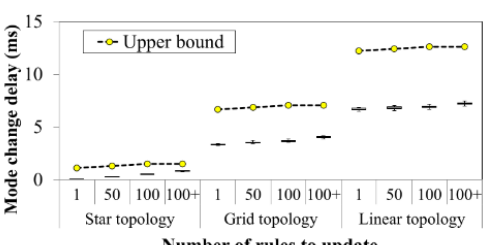
Mode Change Delay Results

Observations

- MC-SDN significantly reduces mode change delay and provides a strict upper bound.
- Minimal fluctuation minimizes uncertainty compared to standard SDN.
- **100+ Scenario**: Includes Out-of-Mode (OOM) packet handling.
- **Linear Topology**: Shows higher upper bounds and fluctuation due to maximum hop counts.



(a) Std-SDN



(b) MC-SDN

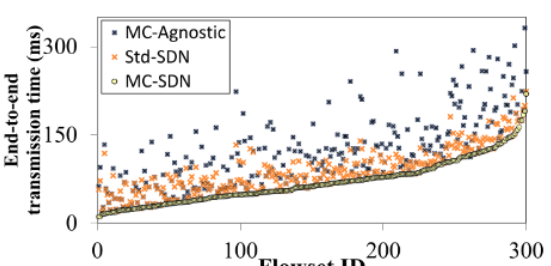
End-to-End Transmission and Overhead

Timing Performance

- Realistic flows generated with random HI and LO flows.
- Plots show transmission times for messages triggering HI behavior for the first time.

Packet Forwarding Overhead

- MC-SDN shows nearly no extra overhead compared to vanilla OVS.
- Monitoring delay is minimal and effectively hidden between packet transmissions.



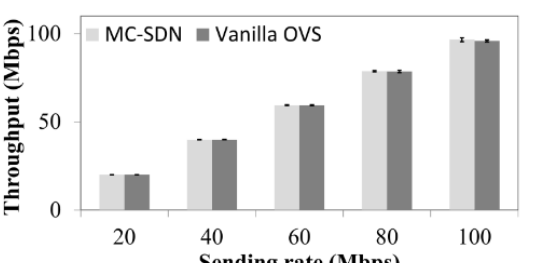


Fig. 8: Packet forwarding throughput

Part VIII

Case Study: Autonomous Vehicle

AUTONOMOUS EMERGENCY BRAKING (AEB) SYSTEM

System Overview

- **Platform**: 1/10 scaled autonomous vehicle with AEB.
- **Navigation**: LIDAR-based SLAM.
- **Function**: AEB brakes in emergency situations by predicting collision risks.

Real-Time Flows

1. **LIDAR**: Used for SLAM localization.
2. **STREAM**: Video frames for user entertainment.
3. **CAM**: Depth camera frames for AEB obstacle detection.

Flow	Period (ms) LO	Period (ms) HI	Priority LO	Priority HI	Size (KB)	Src.	Dst.
LIDAR	40	40	high	high	8	s_1	d_1
STREAM	40	40	mid	drop	432	s_2	d_2
CAM	200	22	low	low	154	s_1	d_1

TABLE IV: Flow set specification in the car system

Experimental Logic and Results

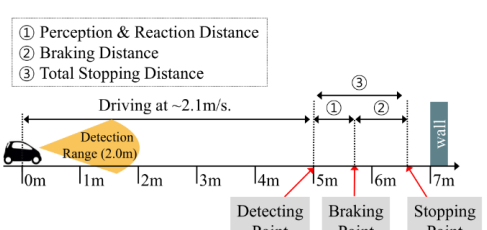
Mode Change Operation

- **Trigger**: Mode changes to HI when CAM detects an obstacle.
- **HI Mode**: Drops the STREAM flow to prioritize critical LIDAR and CAM flows.

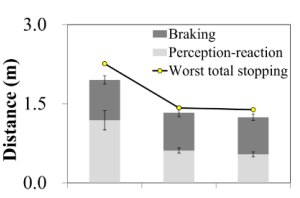
Performance Metrics

1. Perception and reaction distance.
2. Braking distance.
3. Total stopping distance.

Performance depends on the perception and reaction distance , governed by how fast depth images are delivered to the control node.



(b) Evaluation overview



(c) Stopping distances

Part IX

Discussion and Reflection

Personal Reflection: System-level Insights

Analysis over Complexity

- While the mathematical complexity is modest, the rigorous systematic analysis to identify real-world bottlenecks was highly insightful.
- True real-time guarantees come from understanding how system layers (OS, Driver, Protocol) interact.

The "Swap & Switch" Paradigm: MC-SDN vs. RT-Swap

- **Common Strategy**: Both employ Proactive Management.
  - **RT-Swap**: Pre-caches weights/data to bridge the memory capacity gap.
  - **MC-SDN**: Pre-caches HI-rules to bridge the mode-change delay gap.
- **Implementation**: Modifying the datapath/kernel code and validating via schedulability analysis to ensure hard deadlines.

HYPOTHETICAL RESEARCH JOURNEY

"If I were the author..."

1. **The Spark**: Aiming to apply mode-based MC scheduling to automotive switched Ethernet.
2. **The Search**: Discovering SDN as a potential solution for runtime flexibility.
3. **The Discovery**: Realizing standard SDN is unsuitable for real-time due to controller-driven delays.
4. **The Investigation**: Identifying specific overhead points (Arrangement, Rule Update, Queue) through empirical testing.
5. **The Solution**: Analyzing and modifying the OVS system code to eliminate bottlenecks and create MC-SDN.