OMNeT++ Community Summit, 2020

Infrastructure for
Time Sensitive Networking (TSN)
in the INET Framework

Virtual Summit – October 05-06, 2020

Levente Mészáros

Background

- Application Areas
  - Originated from large multi-media networks
  - Nowadays used in industrial and in-vehicle networks
- IEEE 802.1
- Goals
  - Bounded latency
  - Low packet delay variation
  - Low packet loss
  - Fault tolerance

Missing Pieces from INET

- Clock models
- Time synchronization protocols (not covered, e.g. gPTP)
- Aborting transmissions
- Cut-through transmissions
- Versatile queueing model
- Composable Ethernet model
- Synchronous intra-node packet streaming

Covered Topics

Clock Model

Queueing Model

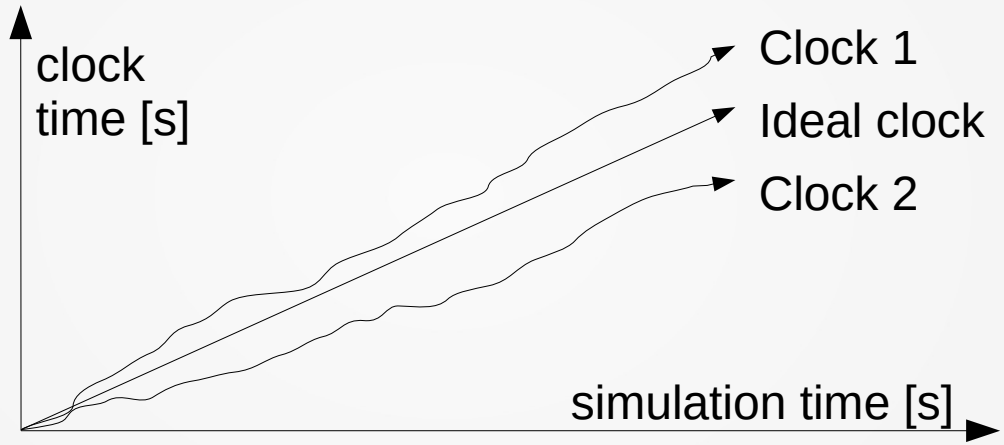
Packet Streaming

Enhanced Transmission Support

Composable Ethernet Model

Clock Model

- Clock time vs. simulation time
- Clock drift (e.g. temperature dependence, thermal noise)



- Hardware precision vs clock accuracy
- Separate oscillator models
- Avoid periodic events

Available Models

- Clock models
  - `IdealClock`
  - `OscillatorBasedClock{oscillator, initialTime}`
  - `SettableClock extends OscillatorBasedClock`
- Oscillator models
  - `IdealOscillator{tickLength}`
  - `ConstantDriftOscillator{nominalTickLength, tickOffset, driftRate}`
  - `RandomDriftOscillator{changeInterval, driftRateChange}`

Using Clocks

- Optional submodules in network interfaces and network nodes
- Subclass from `ClockUserModuleBase` or `ClockUserModuleMixin<T>`
- Use inherited methods or `IClock` and `IOscillator` C++ interfaces
- Usage is similar to standard event scheduling mechanism
  - `simtime_t` vs `clocktime_t`
  - `getClockTime()`
  - `scheduleClockEventAt(time, event)`
  - `scheduleClockEventAfter(delay, event)`
  - `cancelClockEvent(event)`

Covered Topics

Clock Model

Queueing Model

Packet Streaming

Enhanced Transmission Support

Composable Ethernet Model

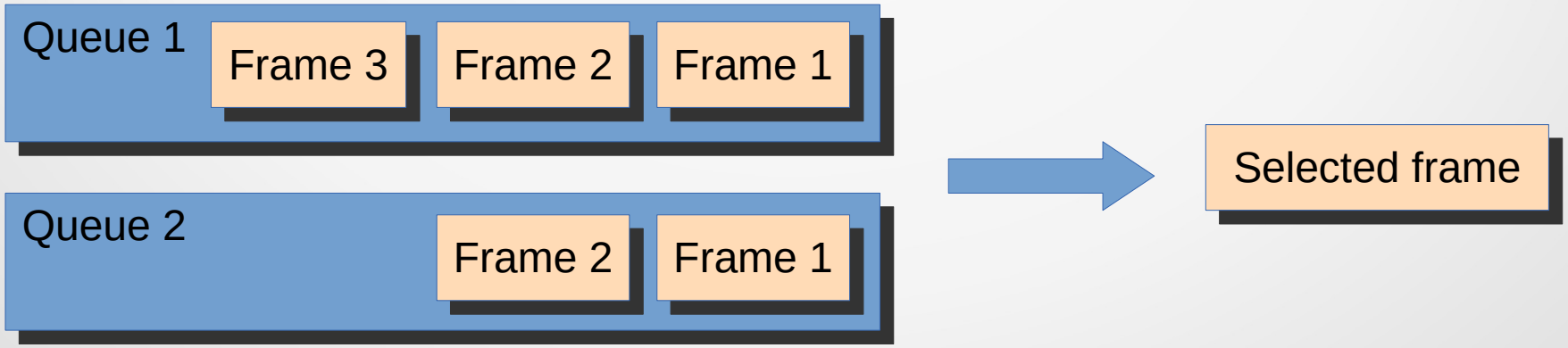
Queueing and Traffic Conditioning

- Traffic shaping
  - Delaying
  - Reordering
  - Metering
  - Classification
- Traffic policing
  - Marking
  - Dropping

- Queueing
  - Prioritizing
  - Shared buffering
  - Gating
  - etc.

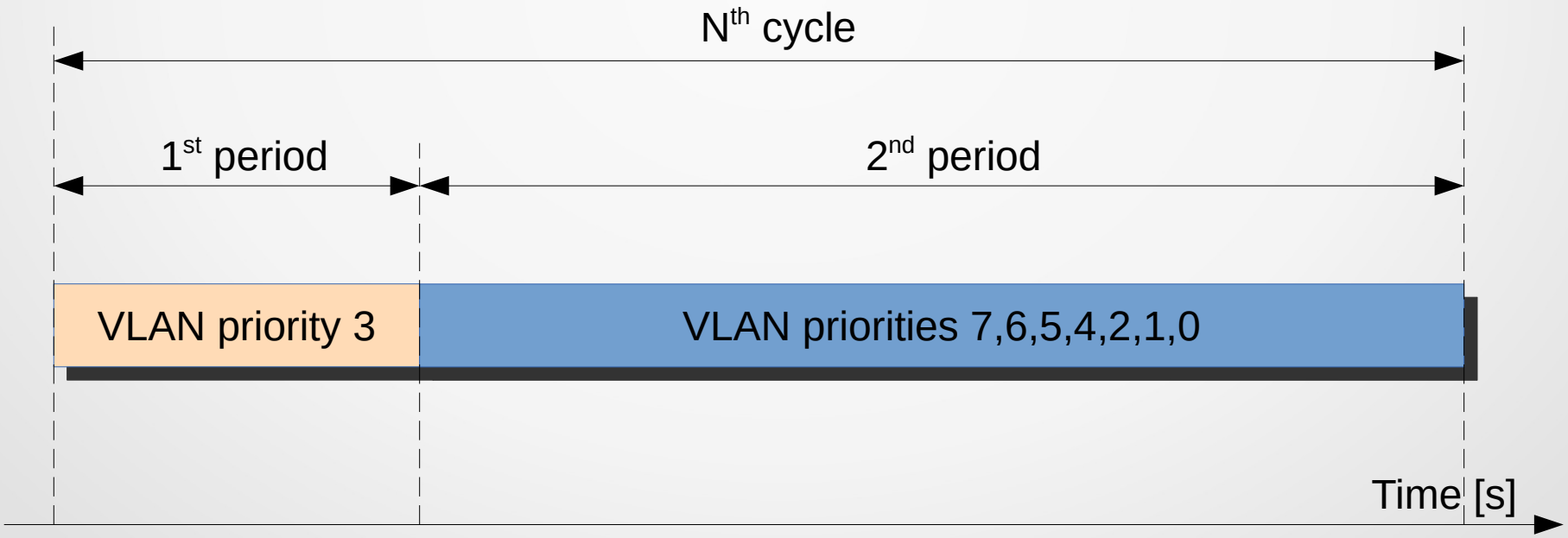
Transmission Selection

- Select next transmitted frame based on
  - Priority, frame length, packet data, meta data
  - Timing constraints and guard intervals
  - Credits for channel use
  - etc.



Gating Mechanism

- Gates can be open or closed based on
  - Predefined periodic scheme
  - Available credits
  - etc.

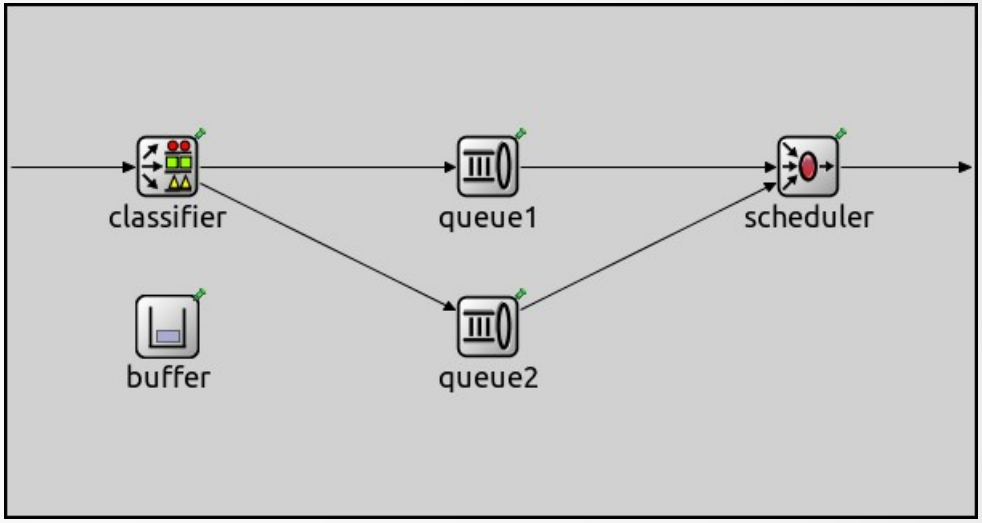


Simple Queueing Model Elements

- Sources  and sinks 
- Queues  buffers  and servers 
- Classifiers  and schedulers 
- Filters  and gates 
- Meters  and markers 
- Multiplexer  demultiplexer  and delayer 
- etc.

Compound Queueing Model Elements

- Priority queues
- Shared buffer queues



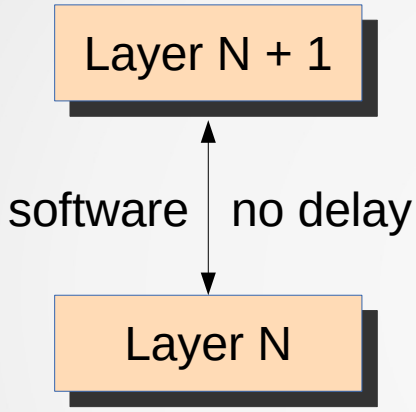
- Queues with gating mechanism
- Traffic shapers and traffic conditioners
- Traffic sources for applications
- Request-response traffic generator applications

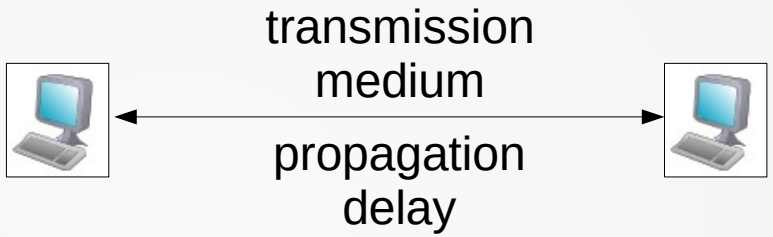
Protocol Support Model Elements

- Transmitters  and receivers 
- Inter-packet gap inserter 
- Fragmenters  and defragmenters 
- Aggregators  and deaggregators 
- Padding and CRC inserters  and checkers 

Communication between Modules

- Intra-node or inter-node





- Asynchronous (message sending)
- Synchronous (C++ function call)

Packet Processing and Queueing API

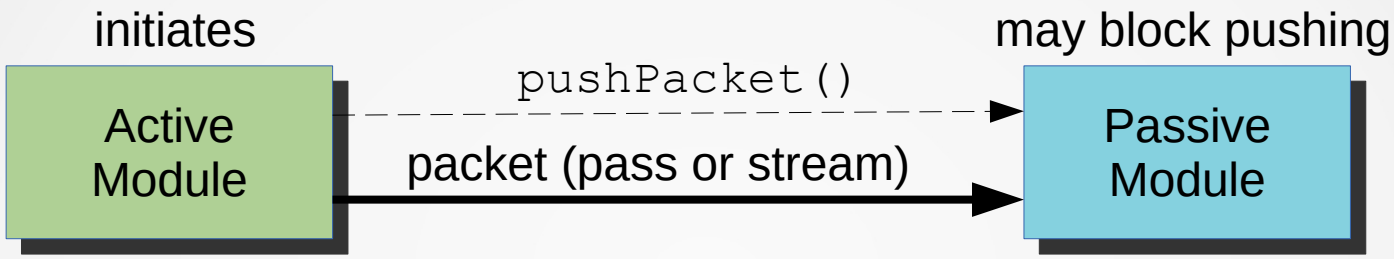
- Sources and sinks
  - `IActivePacketSource, IPassivePacketSink`
  - `IActivePacketSink, IPassivePacketSource`
- Queues and buffers
  - `IPacketCollection, IPacketQueue, IPacketBuffer`
- Classifiers, schedulers, filters, gates, etc.
  - `IPacketClassifier, IPacketScheduler, IPacketFilter, IPacketGatee, etc.`

Active Source and Passive Sink Interface

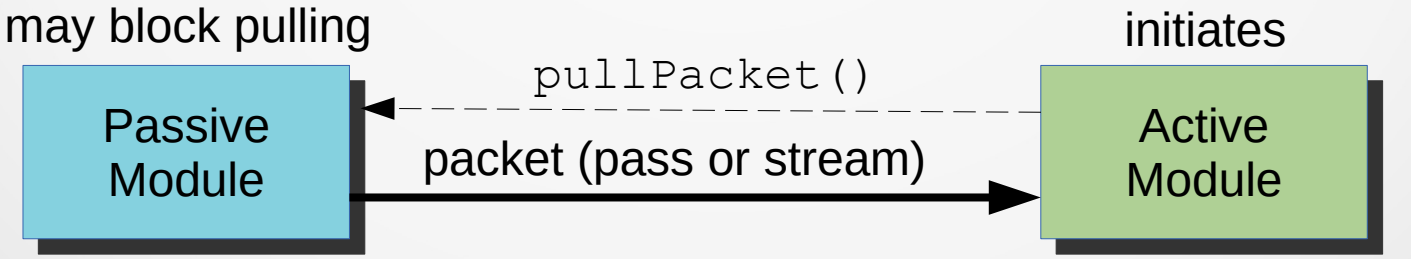
- The `IPassivePacketSink` C++ interface
  - `pushPacket(packet, gate)`
  - `canPushPacket(packet, gate)`
  - `pushPacketStart(packet, gate, datarate)`
  - `pushPacketEnd(packet, gate)`
  - `pushPacketProgress(packet, gate, datarate, position)`
- The `IActivePacketSource` C++ interface
  - `handleCanPushPacketChanged(gate)`
  - `handlePushPacketProcessed(packet, gate, bool)`

Synchronous Packet Flow

- Pushing a packet



- Pulling a packet



- Supports backpressure

Covered Topics

Clock Model

Queueing Model

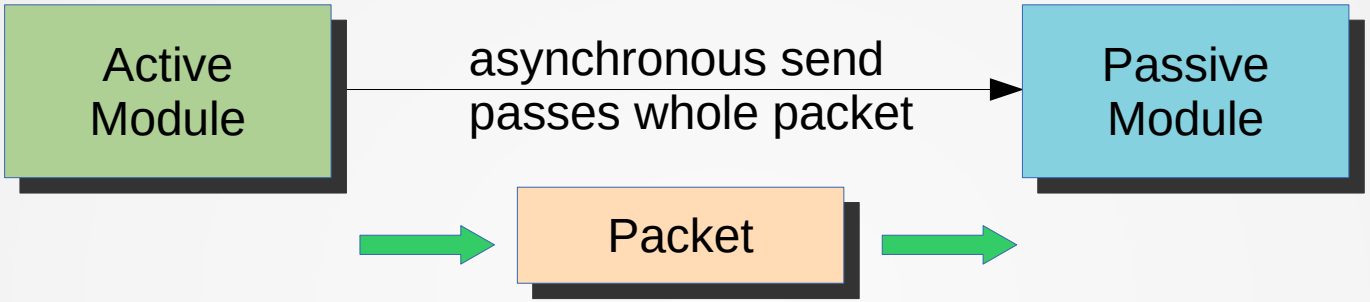
Packet Streaming

Enhanced Transmission Support

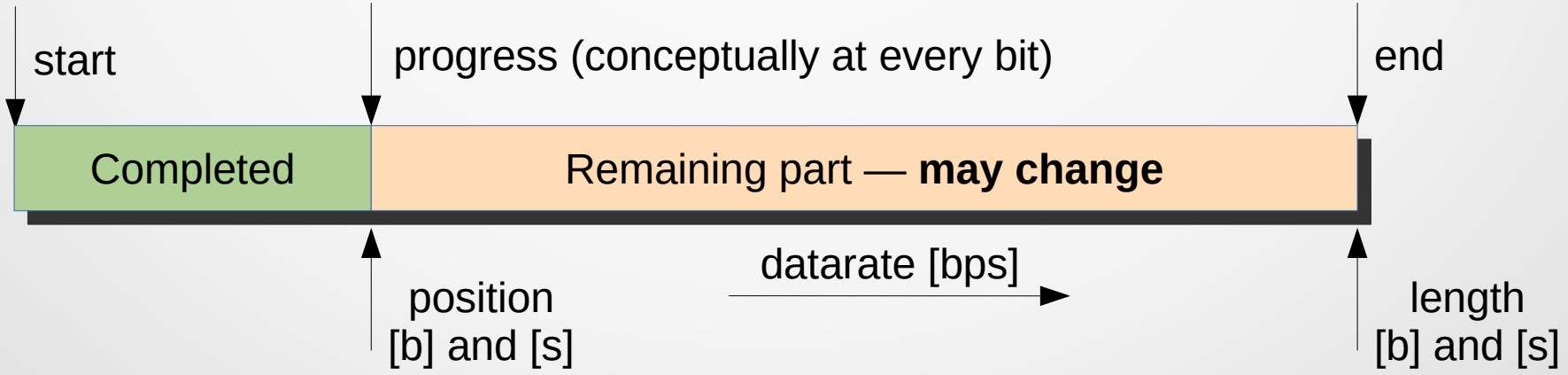
Composable Ethernet Model

Passing vs Streaming a Packet

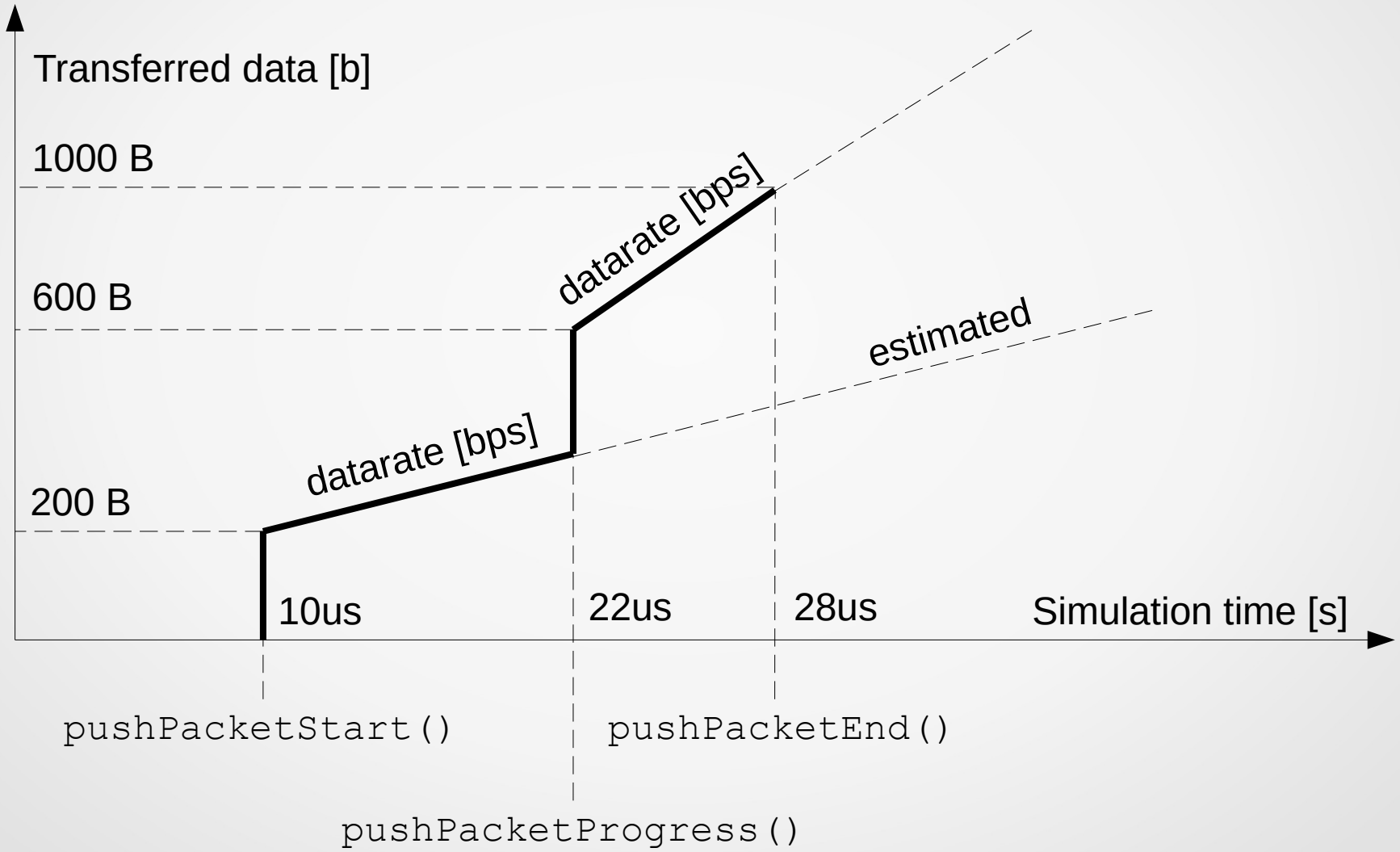
- Passing a packet



- Streaming a packet



Streaming a Packet over Time



Covered Topics

Clock Model

Queueing Model

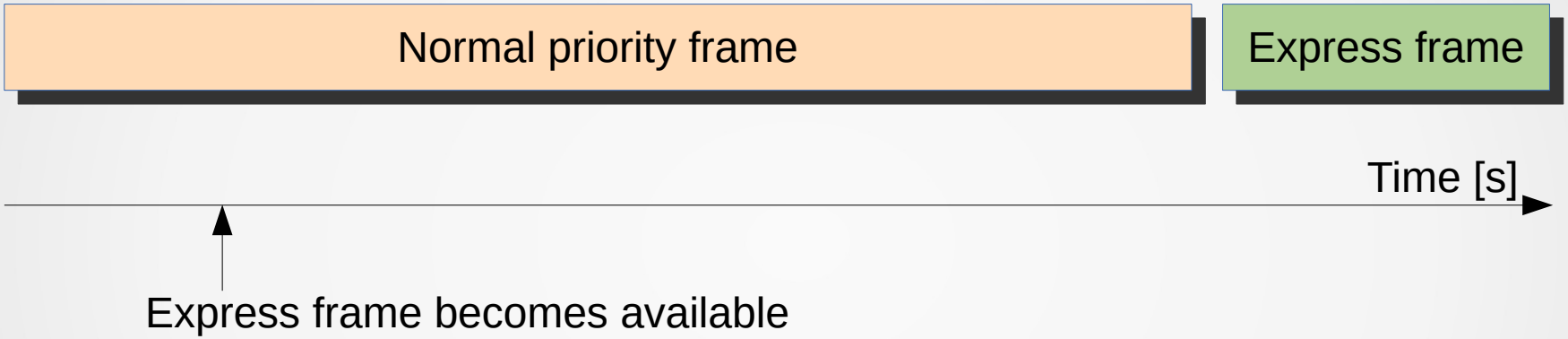
Packet Streaming

Enhanced Transmission Support

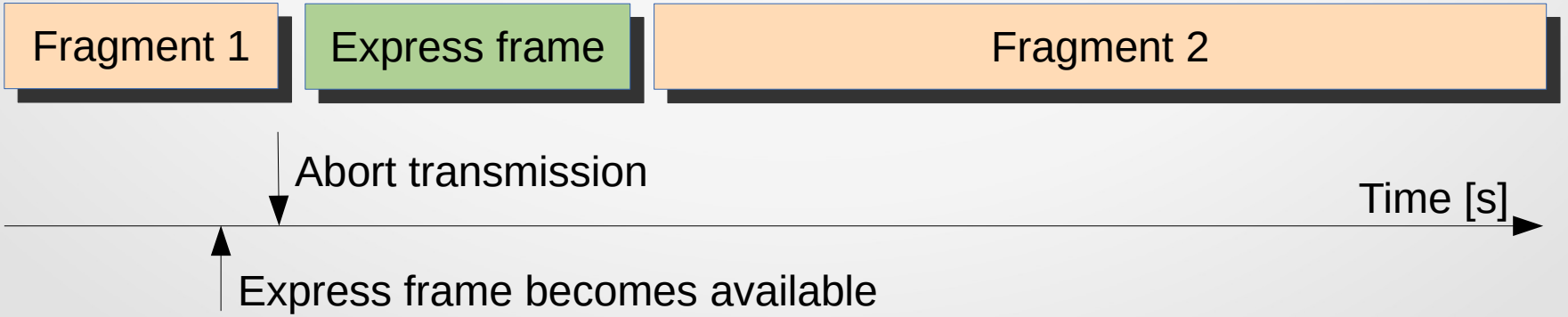
Composable Ethernet Model

Ethernet Frame Preemption

- Long frame delays high priority frame

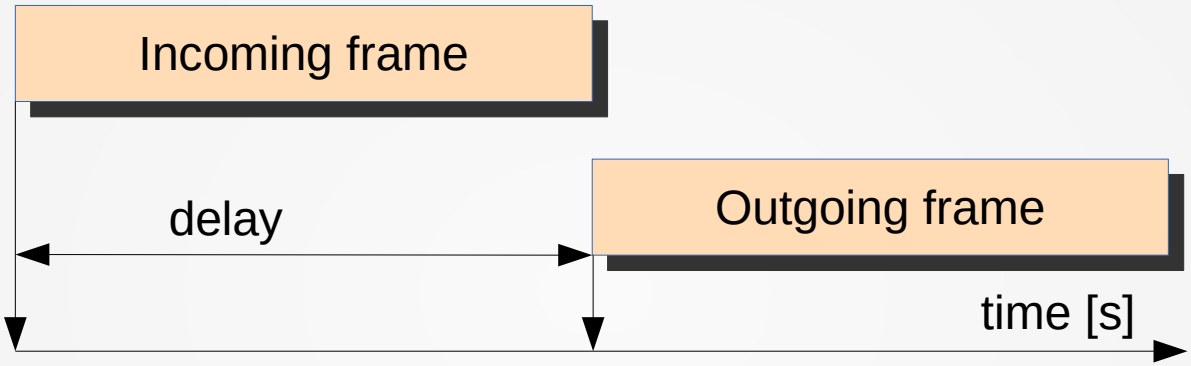


- Reduce latency by aborting transmission

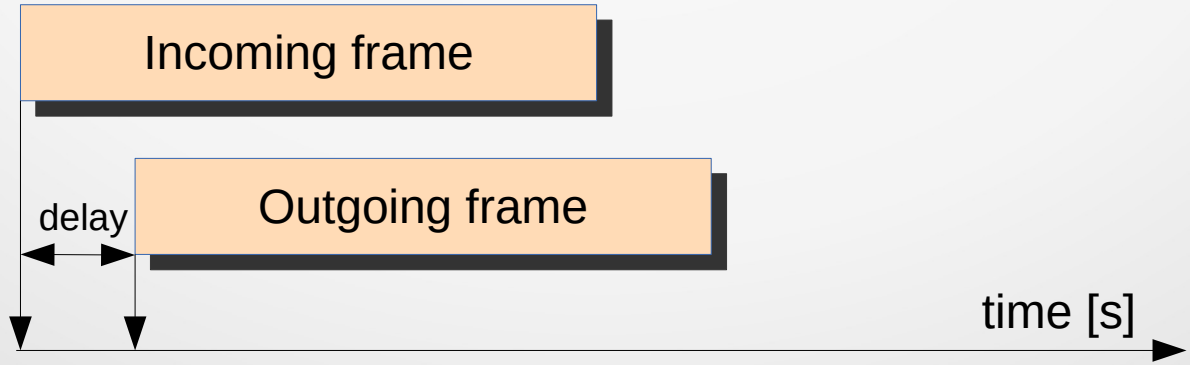


Ethernet Cut-through Switching

- Store and forward



- Start forwarding as soon as MAC header is received



Covered Topics

Clock Model

Queueing Model

Packet Streaming

Enhanced Transmission Support

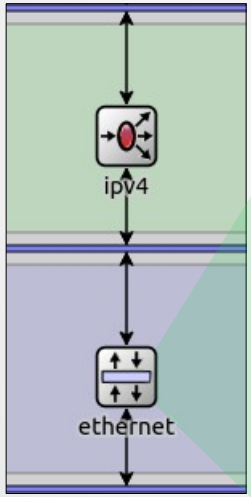
Composable Ethernet Model

Ethernet Standards

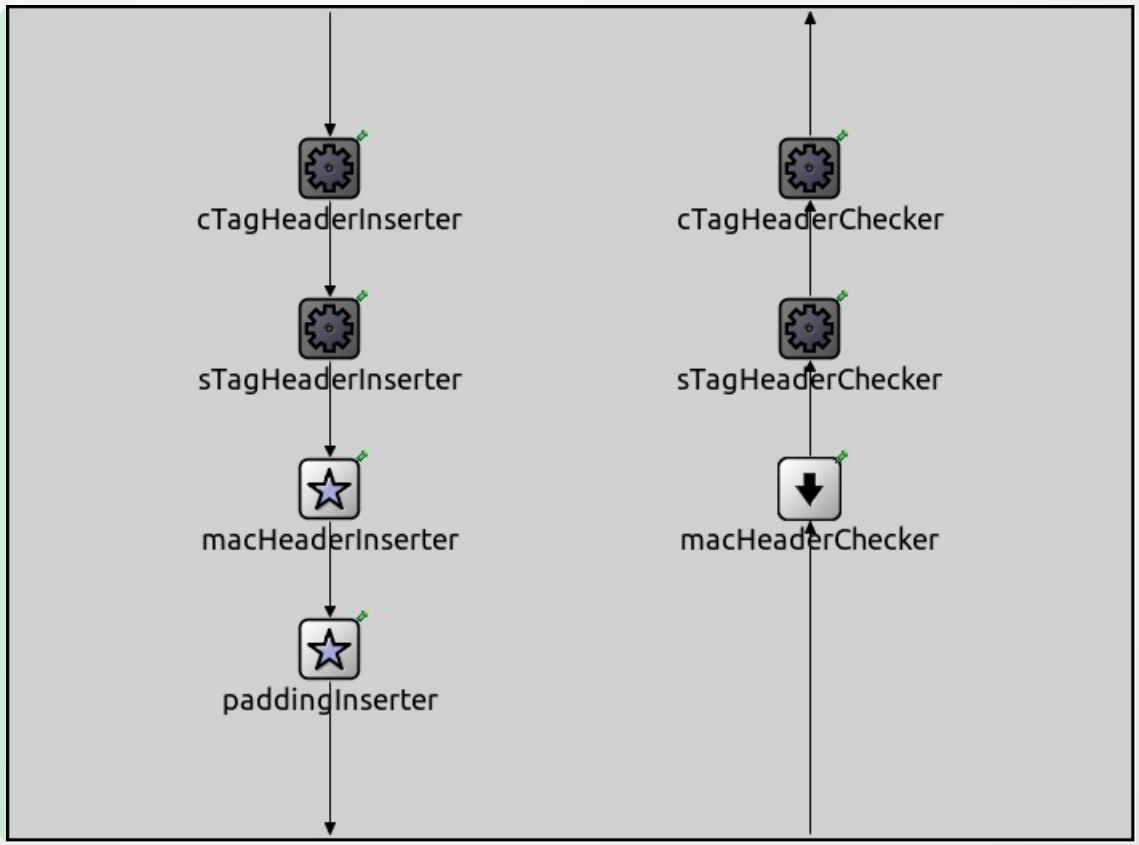
- Several related IEEE 802.1 and 802.3 standards
  - br, AS, Qav, Qat, Qcc, Qch, Qci, Qbv, Qbu, Qcr, Qca, CB, CS, Qdd, ABdh
  - Many combinations
  - Different interpretations
  - Complex behavior
- Need composable Ethernet model

Ethernet Protocol Layer

Network node fragment

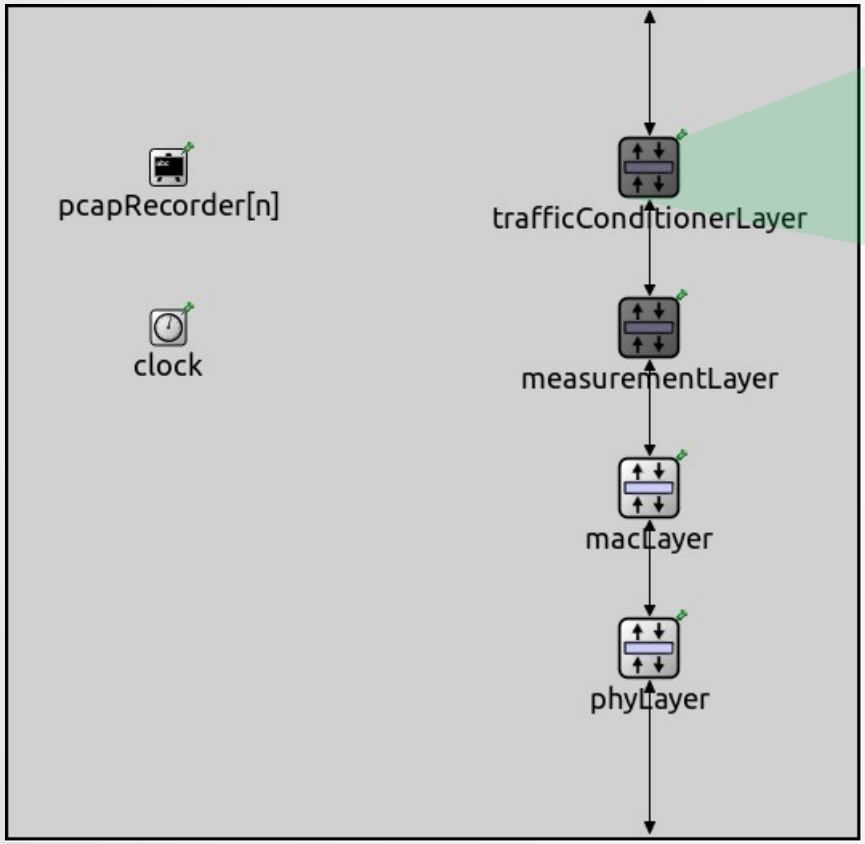


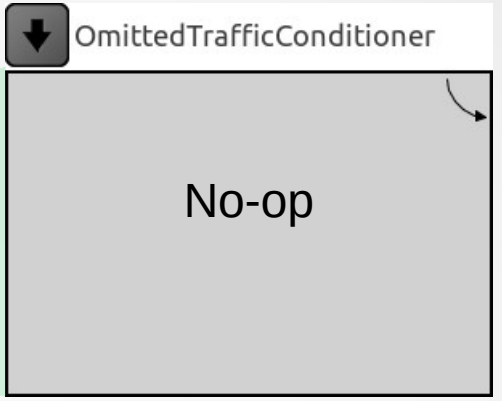
Separate protocol layer in network nodes



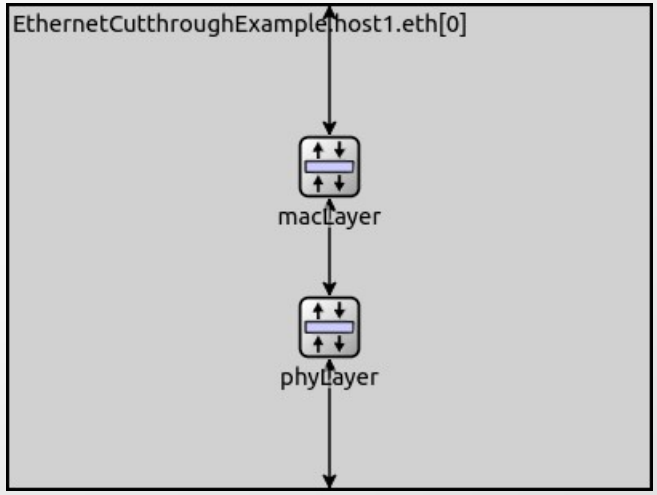
Ethernet Interface

Some submodules are optional, not used by default



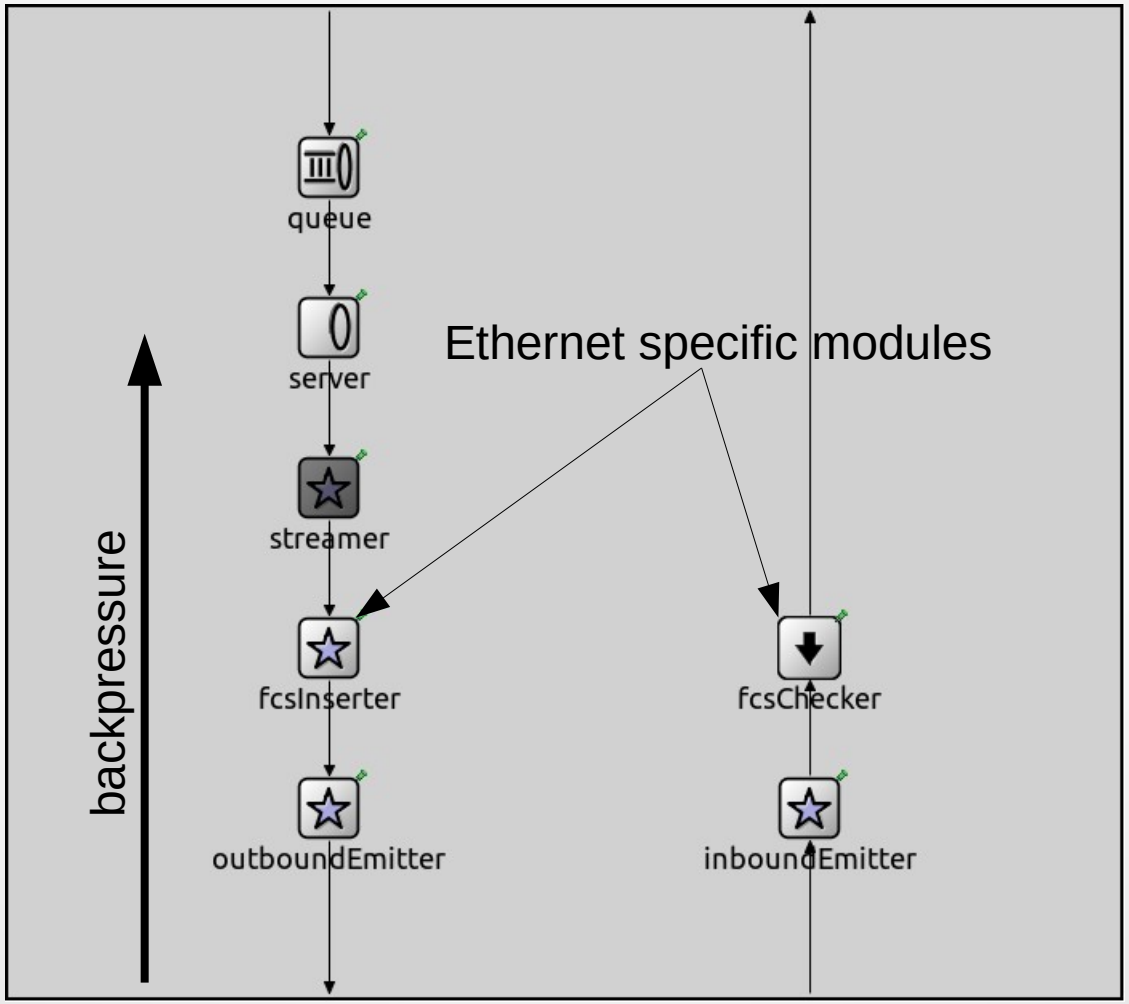


No-op submodules disappear at runtime



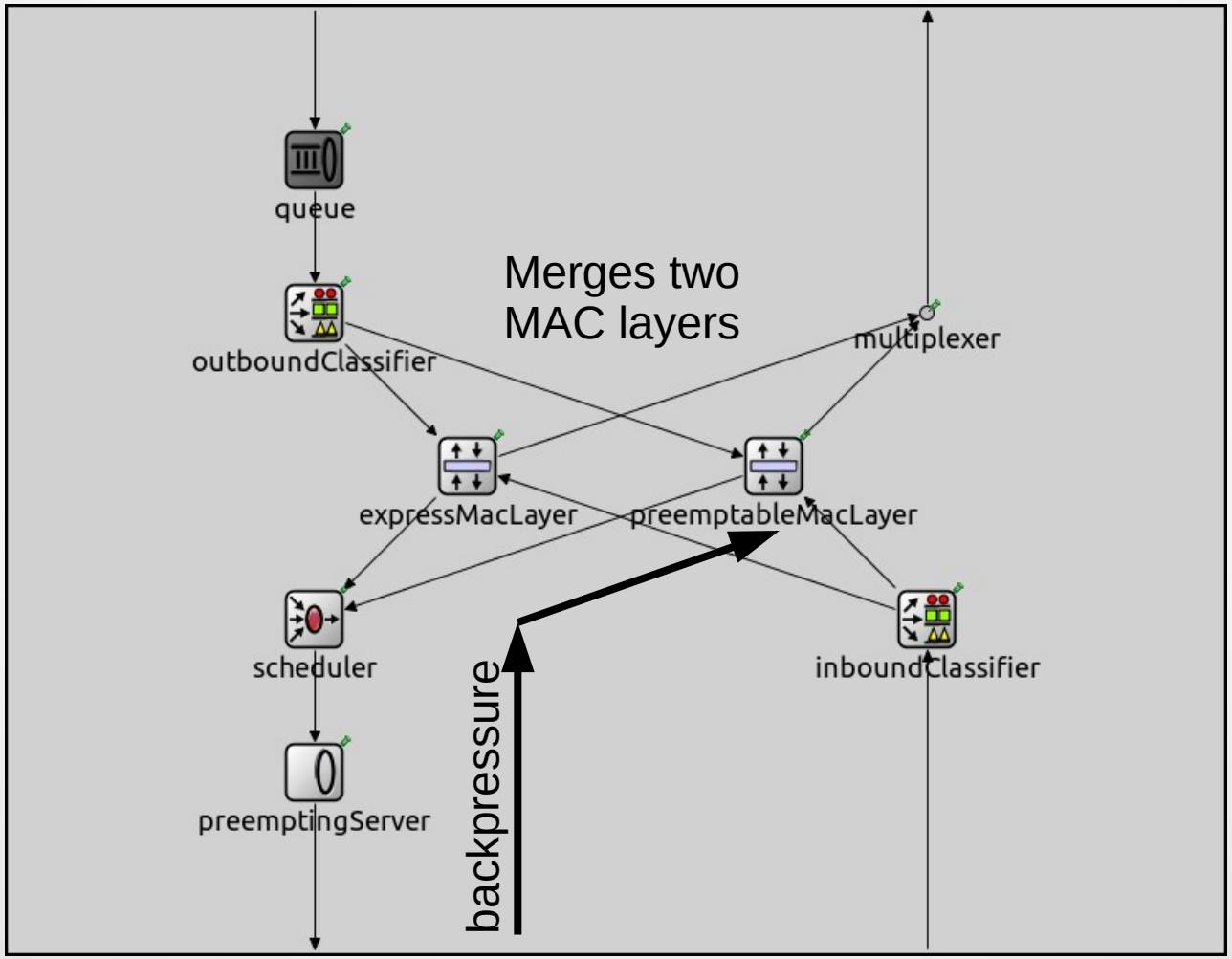
Ethernet MAC Layer

Uses several generic modules



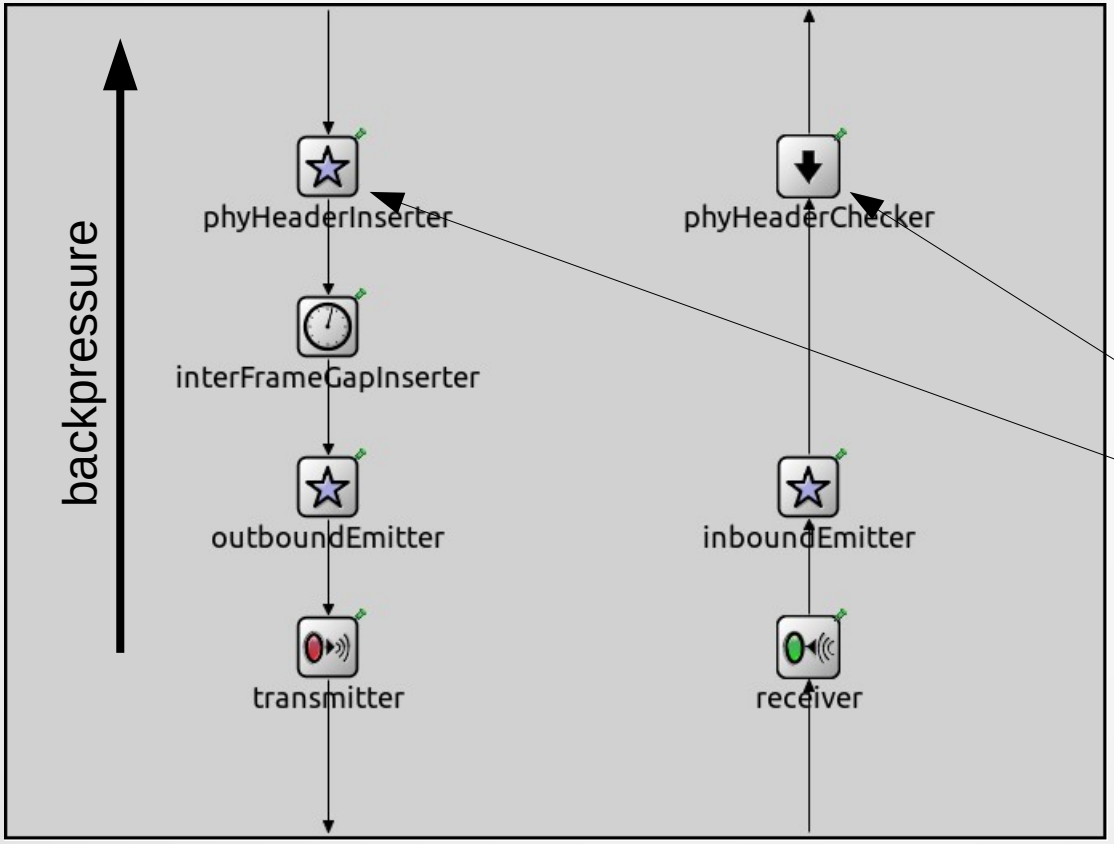
Ethernet Preempting MAC Layer

Combines two Ethernet MAC Layers with several generic modules



Ethernet PHY Layer

Backpressure from transmitter goes up to the server in the MAC layer



Ethernet specific modules

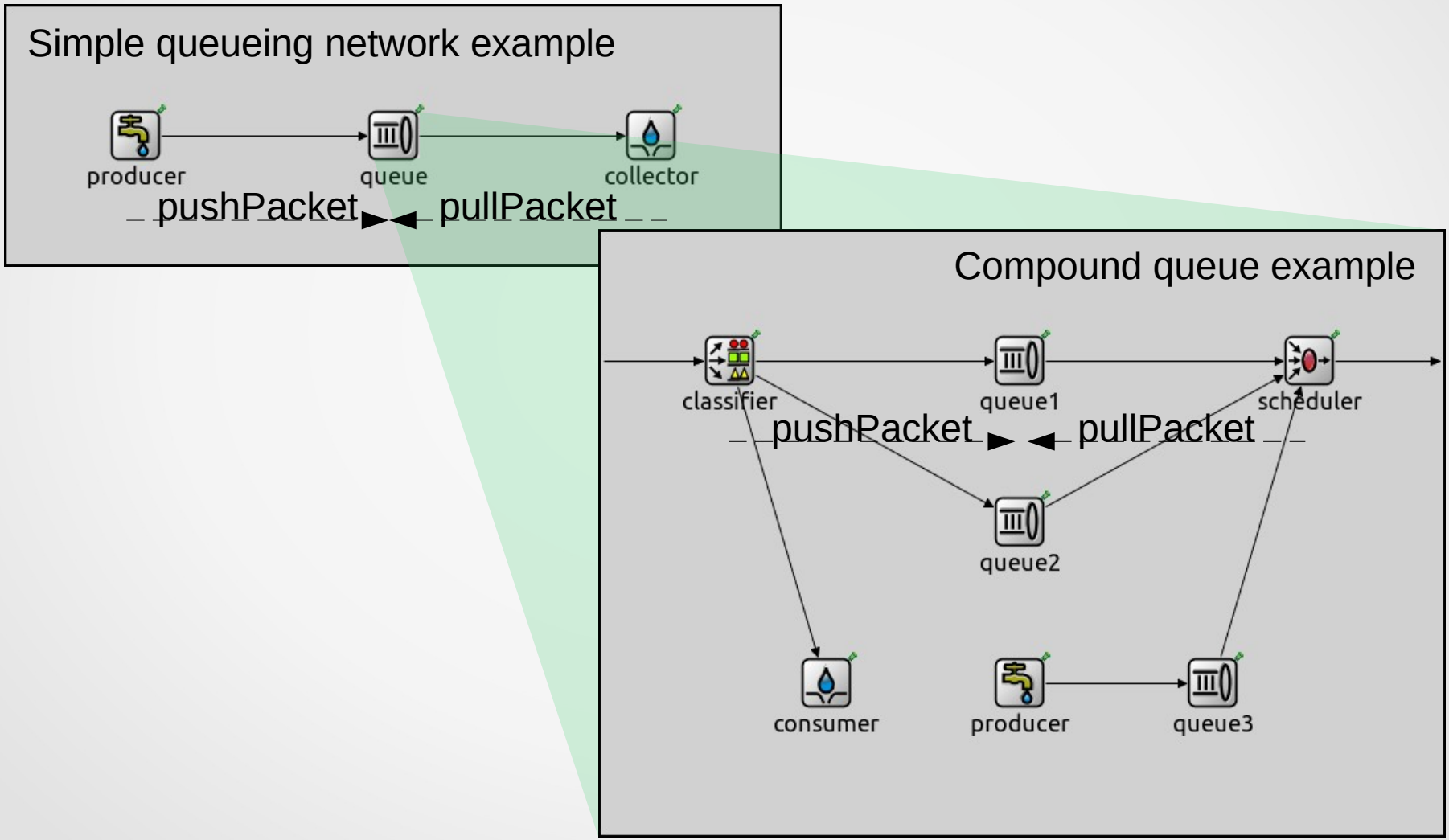
Examples

Queueing Model

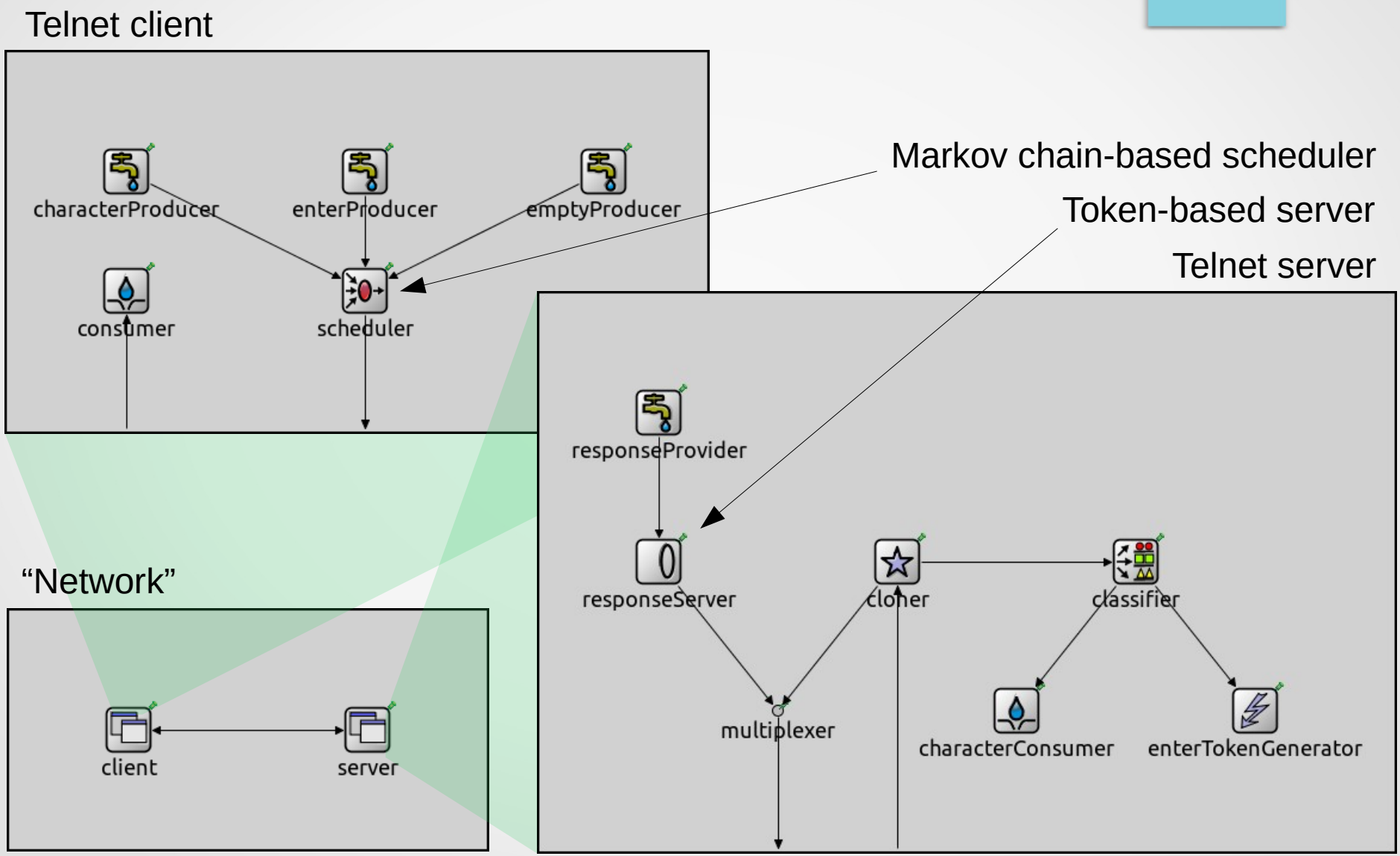
Ethernet Frame Preemption

Ethernet Cut-through Switching

Packet Queueing Example



Telnet Application Example



Examples

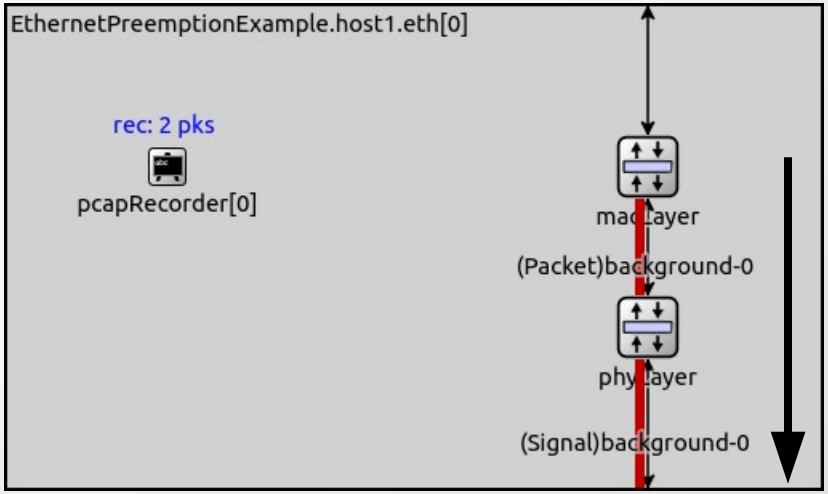
Queueing Model

Ethernet Frame Preemption

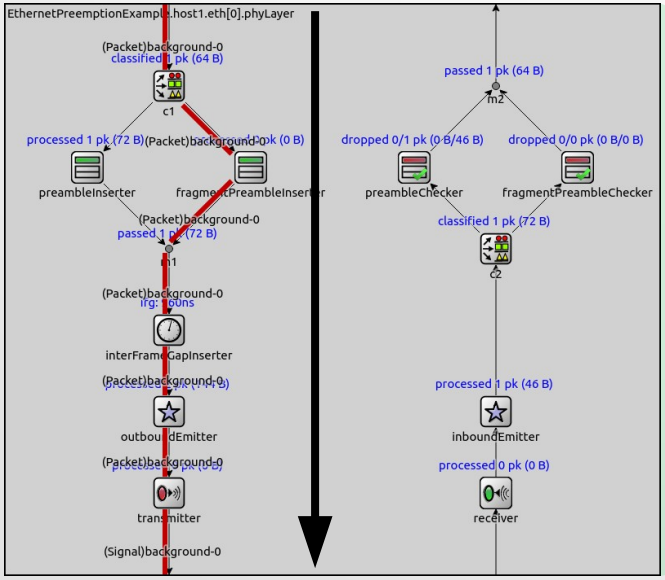
Ethernet Cut-through Switching

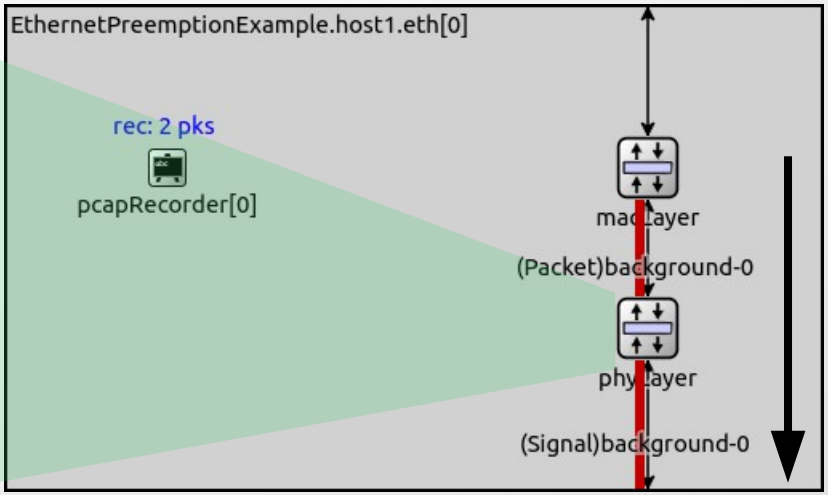
Ethernet Frame Preemption Example

- Legend
  - Ongoing packet streaming
  - Direction of information flow 

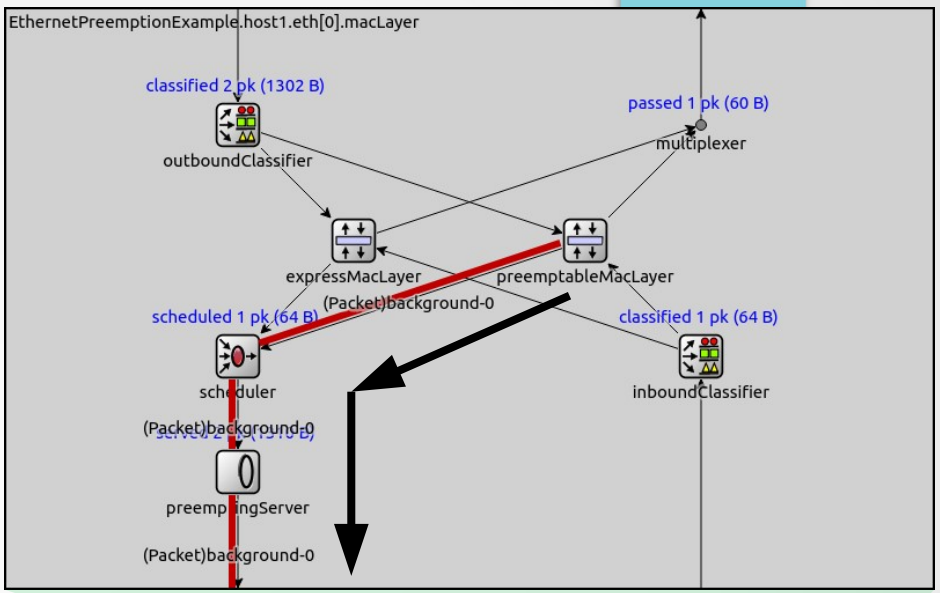


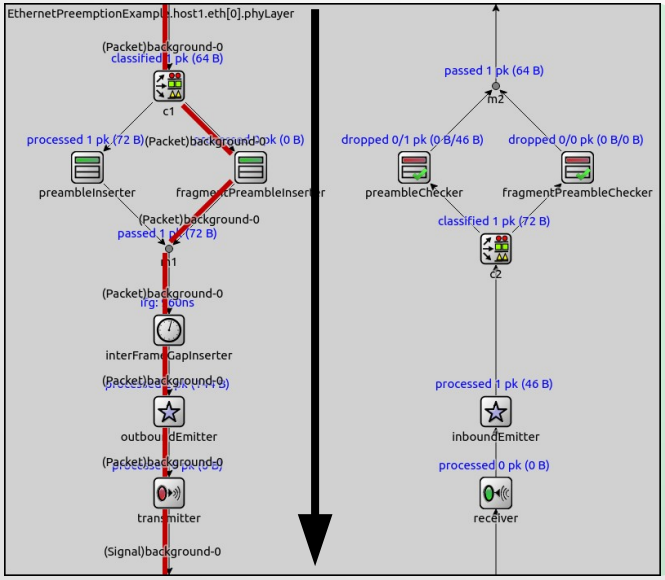
Ethernet Frame Preemption Example

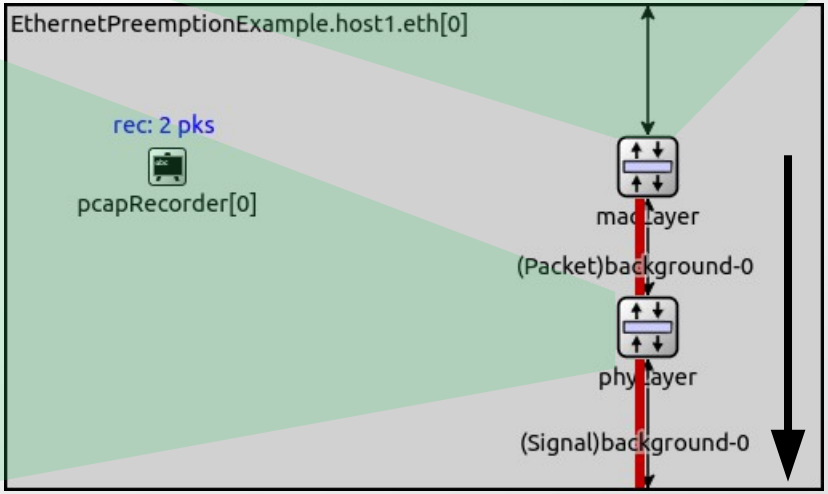




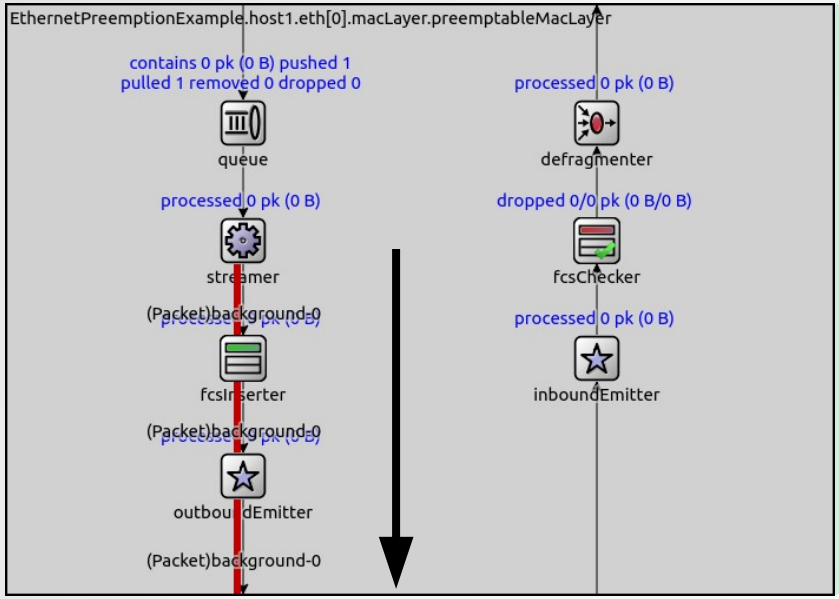
Ethernet Frame Preemption Example

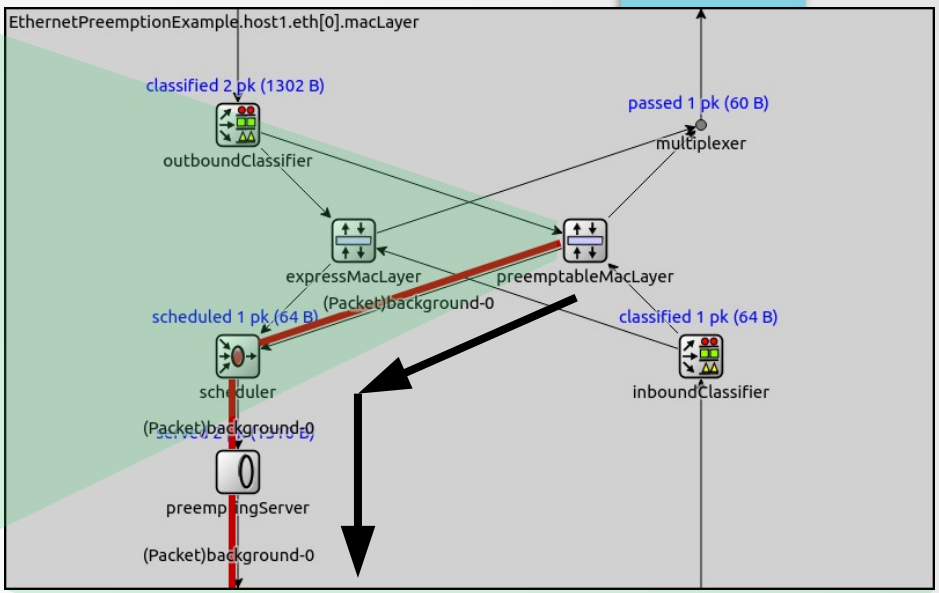


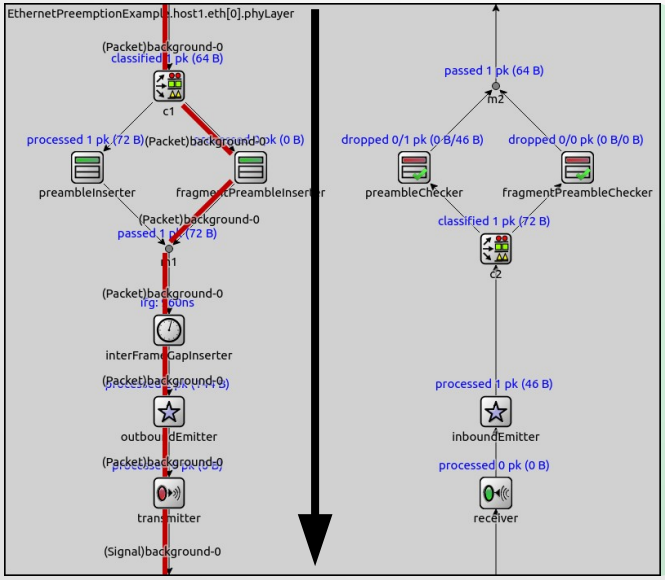


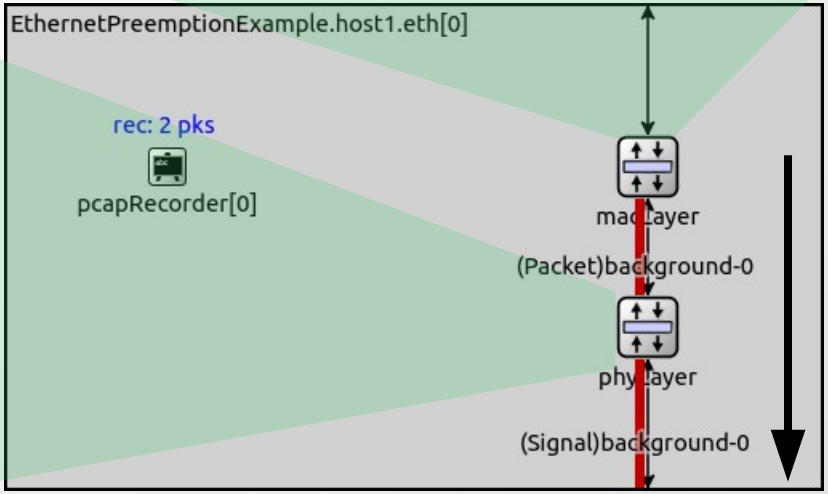


Ethernet Frame Preemption Example







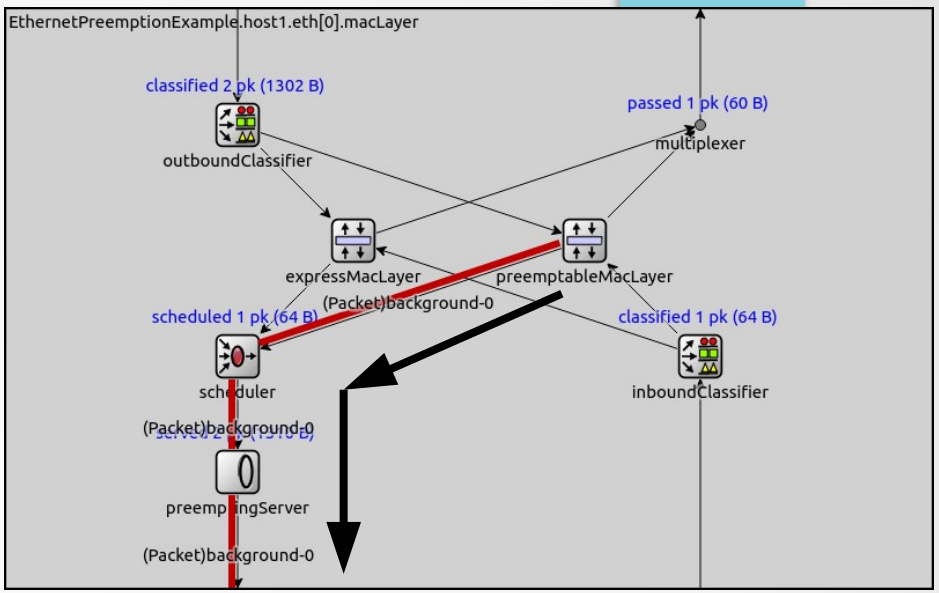


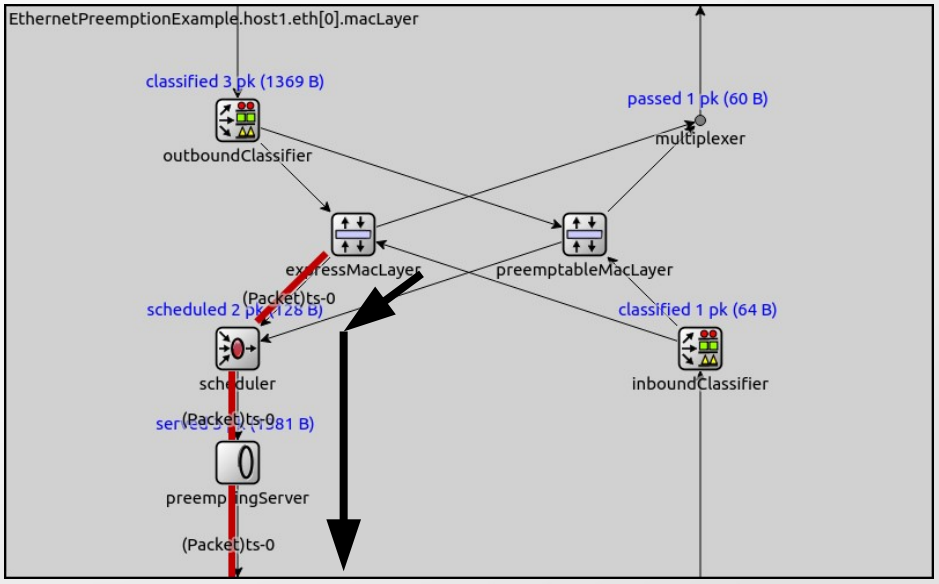
Ethernet Frame Preemption Moment

packet log

#30	0.00001162	host1 --> host2	background-0
#31	0.00001162	host1 --> host2	background-0-frag0:start
#35	0.00001738	host1 --> host2	background-0-frag0:end
#37	0.00001834	host1 --> host2	ts-0
#40	0.00002466	host1 --> host2	ts-0:end
#45	0.00002562	host1 --> host2	background-0-frag1
#53	0.00012114	host1 --> host2	background-0-frag1:end

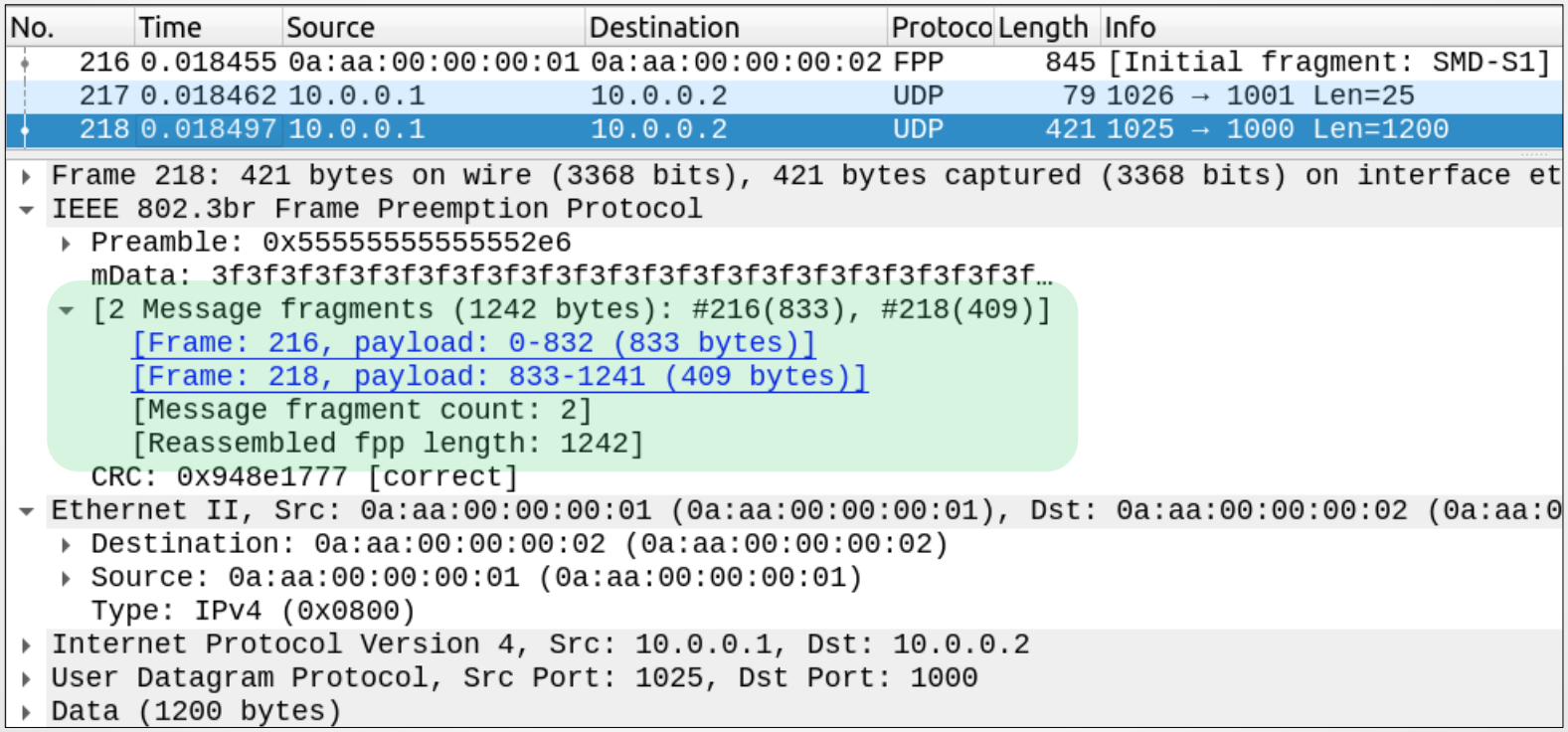
preemption





Ethernet Frame Preemption in Wireshark

Analyzing a recorded PCAPng trace



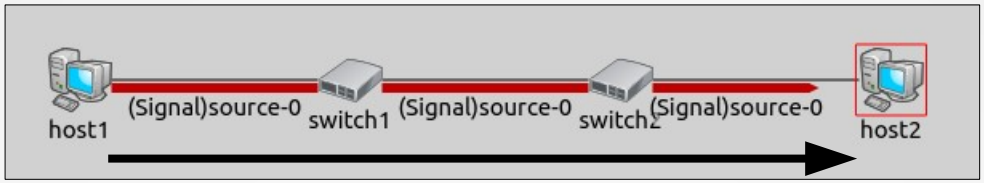
Examples

Queueing Model

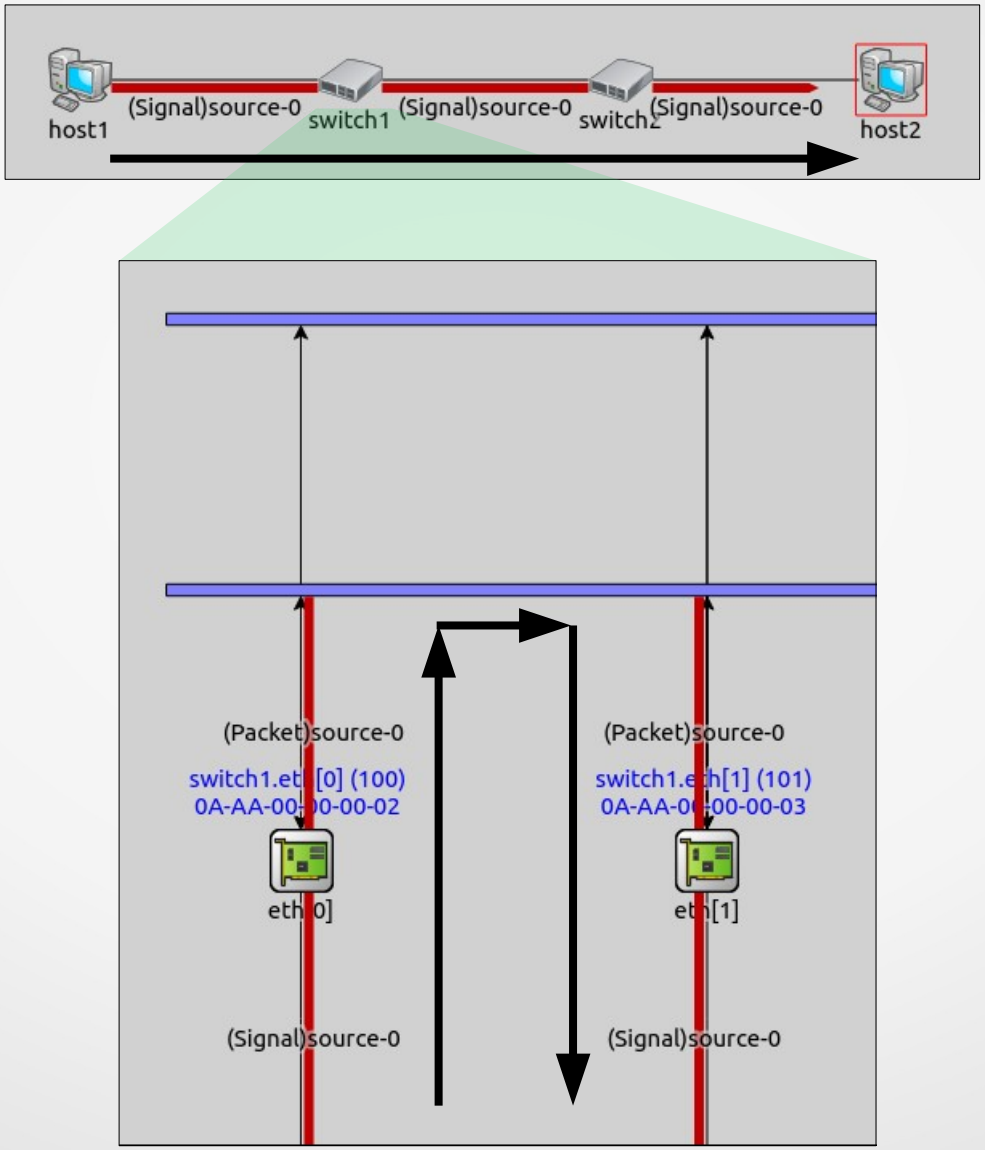
Ethernet Frame Preemption

Ethernet Cut-through Switching

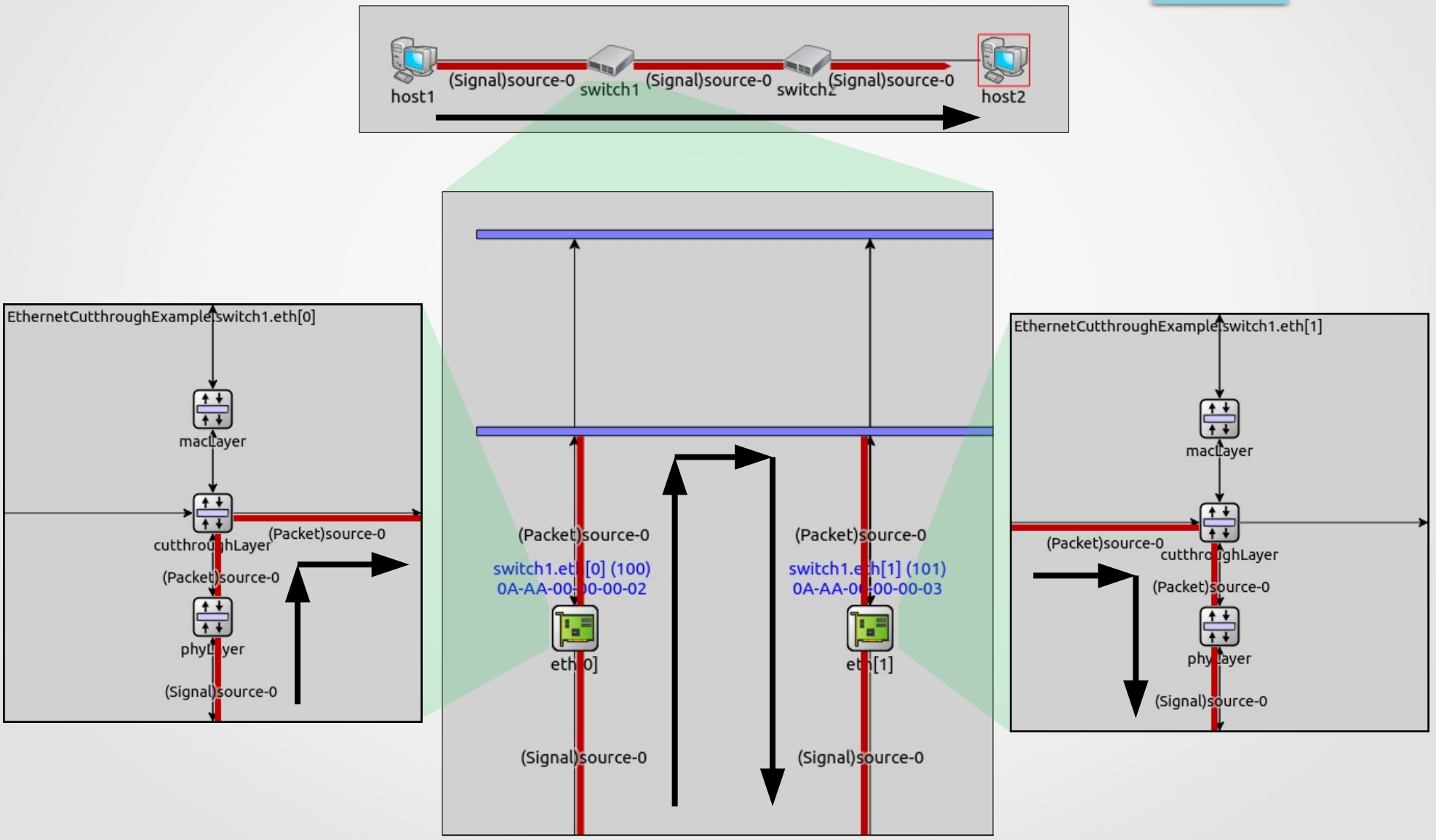
Ethernet Cut-through Example



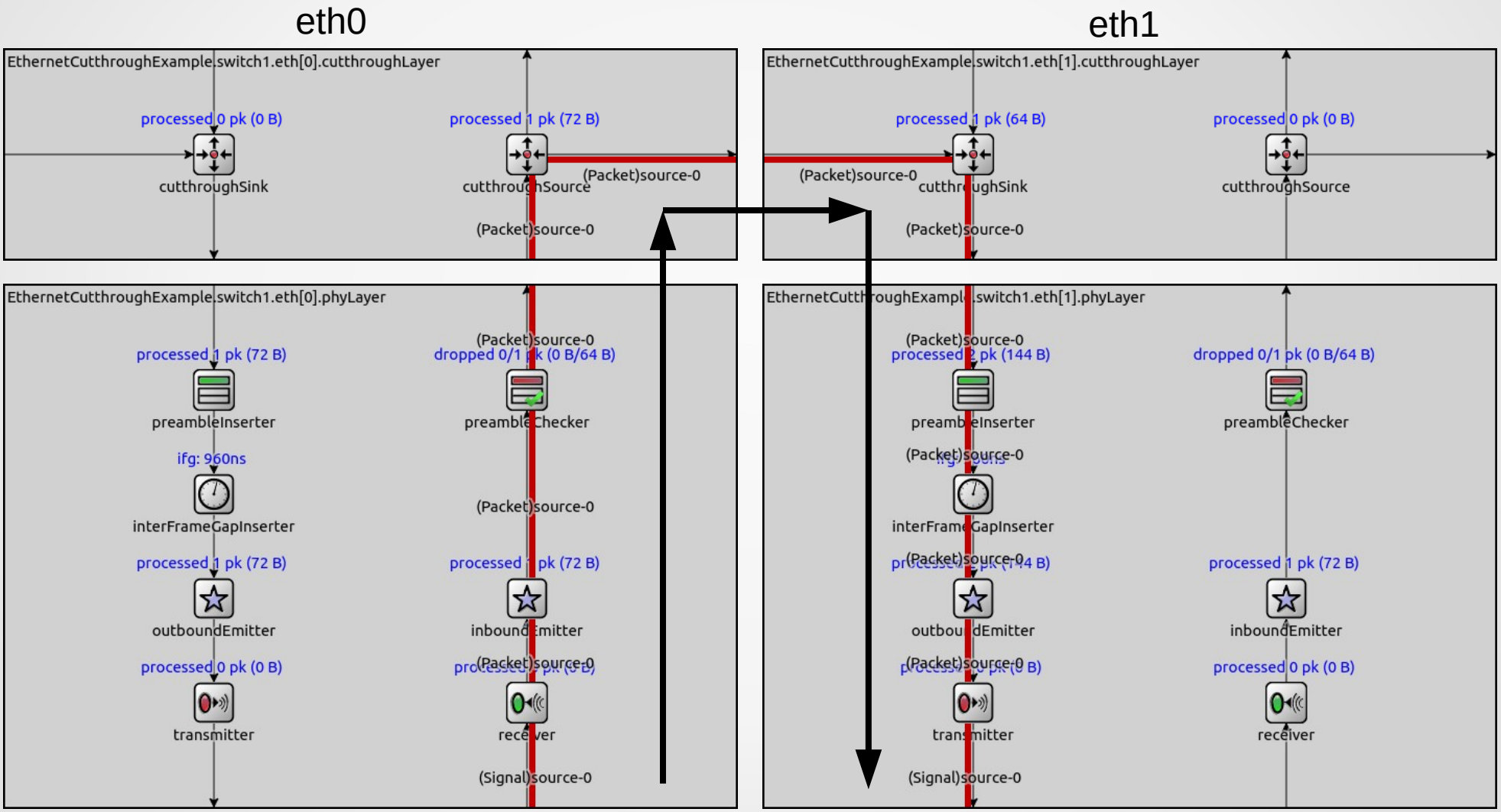
Ethernet Cut-through Example



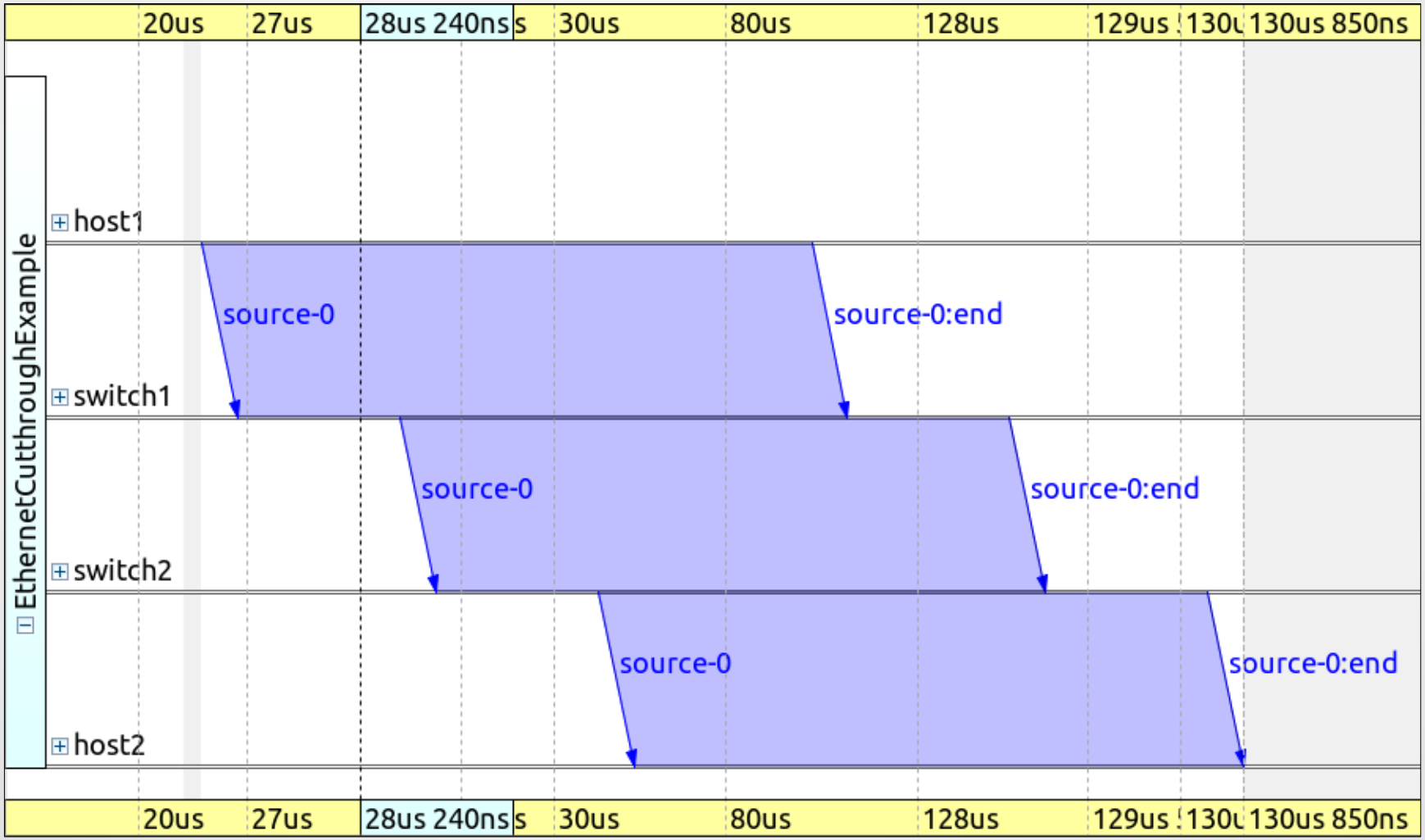
Ethernet Cut-through Example



Ethernet Cut-through Layer



Ethernet Cut-through in the Sequence Chart



Closing

- Applicability points beyond TSN
  - Queueing model
  - Packet streaming
- Standalone teaching material
  - Queueing model
  - Protocol support

Questions and Answers

Thank you for your attention!