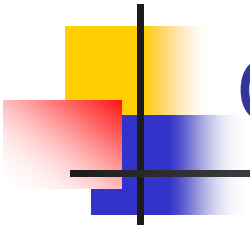
AirTight: A Resilient Wireless Communication Protocol for Mixed-Criticality Systems

Alan Burns, James Harbin, Leandro Indrusiak, Iain Bate, Robert Davis and David Griffin

Real-Time Systems Research Group
University of York



Outline

- Background
  - Context and motivation for this work and aims of AirTight
  - Wireless communications: assumptions, challenges, and mixed-criticality
- AirTight Protocol
  - Basic approach and support for mixed-criticality packet flows
  - Schedulability analysis for AirTight
- Case study
  - Experimental verification on small system
  - Simulation results for a larger system
- Conclusions
  - Summary and future work

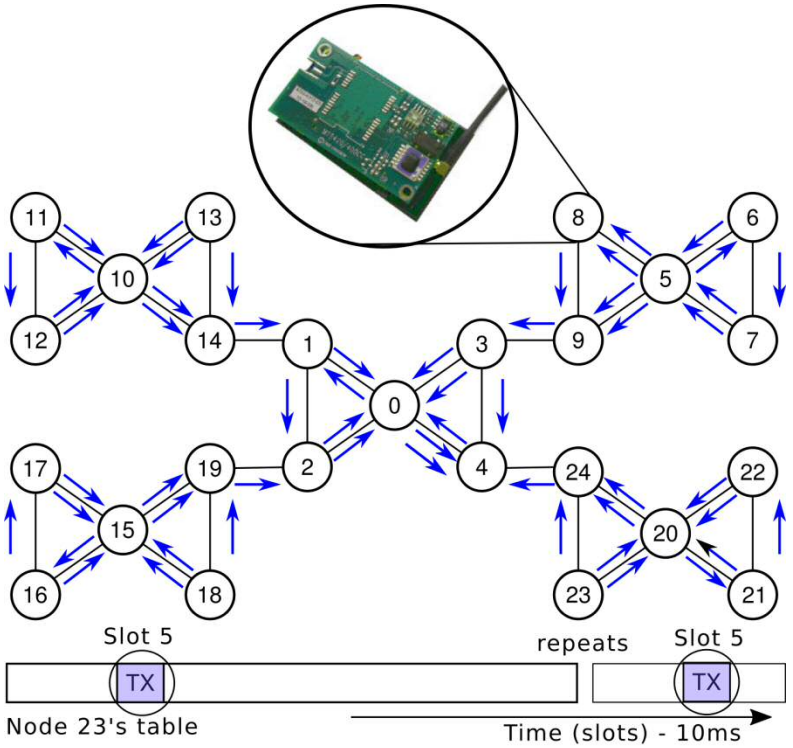
Context and Motivation

- Cyber-Physical System (CPS)
  - A distributed system of embedded devices or nodes (including sensors and actuators)
  - Connected via wireless communications (subject to interference)
  - System supports functionality of different criticality levels (mixed criticality)
  - Examples: Aircraft engine monitoring system
    Factory industrial control system

- Aims of AirTight: real-time wireless protocol
  - Enable analysis of worst-case traversal times across the network (given a general fault model)
  - Provide resilience in the presence of transmission failures due to interference
  - Provide higher resilience for HI-criticality packet flows by degrading service for LO-criticality packet flows

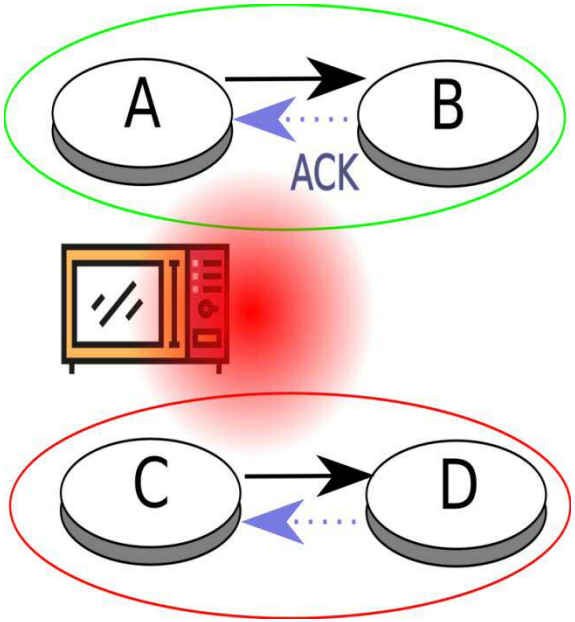
Wireless Communications in CPS

- Requirements and assumptions
  - Peer-to-peer packet-switched communication with different priorities for the packet flows
  - Multi-hop routing is used due to the limited transmission range
  - Packets are sent as a series of one or more frames
  - Buffers on each node are needed to store frames in transit
  - Successful frame transmission is acknowledged by the receiver
  - Nodes have synchronised clocks, so for analysis purposes time can be structured in terms of transmission (TX) slots



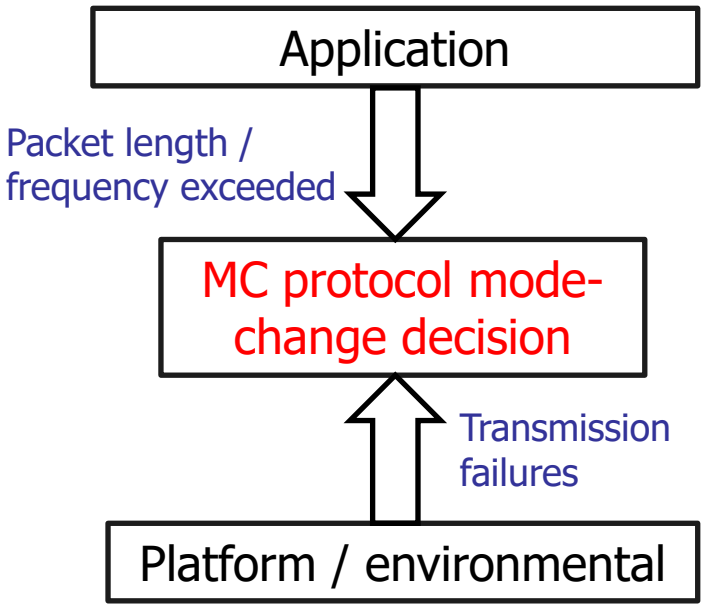
Challenges of CPS Environments

- Challenges
  - Limited range: Communications Graph indicates which nodes can communicate directly with each other
  - Interference Graph: indicates which nodes can cause interference on each others transmissions
  - Wireless channels are inherently unreliable: sources of Electromagnetic Inference (EMI) can cause transmission faults
  - Multiple channels can be used but nodes can only be active on one channel at a time
  - We assume multi-hop, single-channel, single-interference domain in this first work on AirTight



Mixed Criticality in Wireless CPS

- Mixed-criticality CPS
  - Support sharing resources between different safety assurance (criticality) levels
- Normal operation (LO-criticality mode)
  - System ensures that timing guarantees are met for LO- and HI-criticality applications
- Criticality mode changes
  - Driven by application behaviour e.g. longer than budgeted execution, packet length etc.
  - Driven by environment behaviour e.g. transmission failures (this paper)
- HI-criticality mode
  - System ensures that timing guarantees are met for HI-criticality applications (LO-criticality applications can be degraded)



Fault Model and Requirements on the AirTight Protocol

- Fault model
  - General fault load function $F_k(L,t)$ giving the maximum number of faulty transmission slots (caused by interference) that must be tolerated by node k in a time interval t for packets of criticality level L
  - Fault load that must be tolerated is greater for HI-criticality packets, so

$$F_k(HI,t) \geq F_k(LO,t)$$

- Requirements on the AirTight protocol
  - If faults experienced are no worse than implied by LO-criticality fault model – all packets will meet their deadlines
  - If faults experienced are no worse than implied by HI-criticality fault model – all HI-criticality packets will meet their deadlines (LO-criticality packets may be discarded)
  - Otherwise no guarantees, but HI-criticality packets can utilise all available bandwidth

AirTight Protocol: Basics

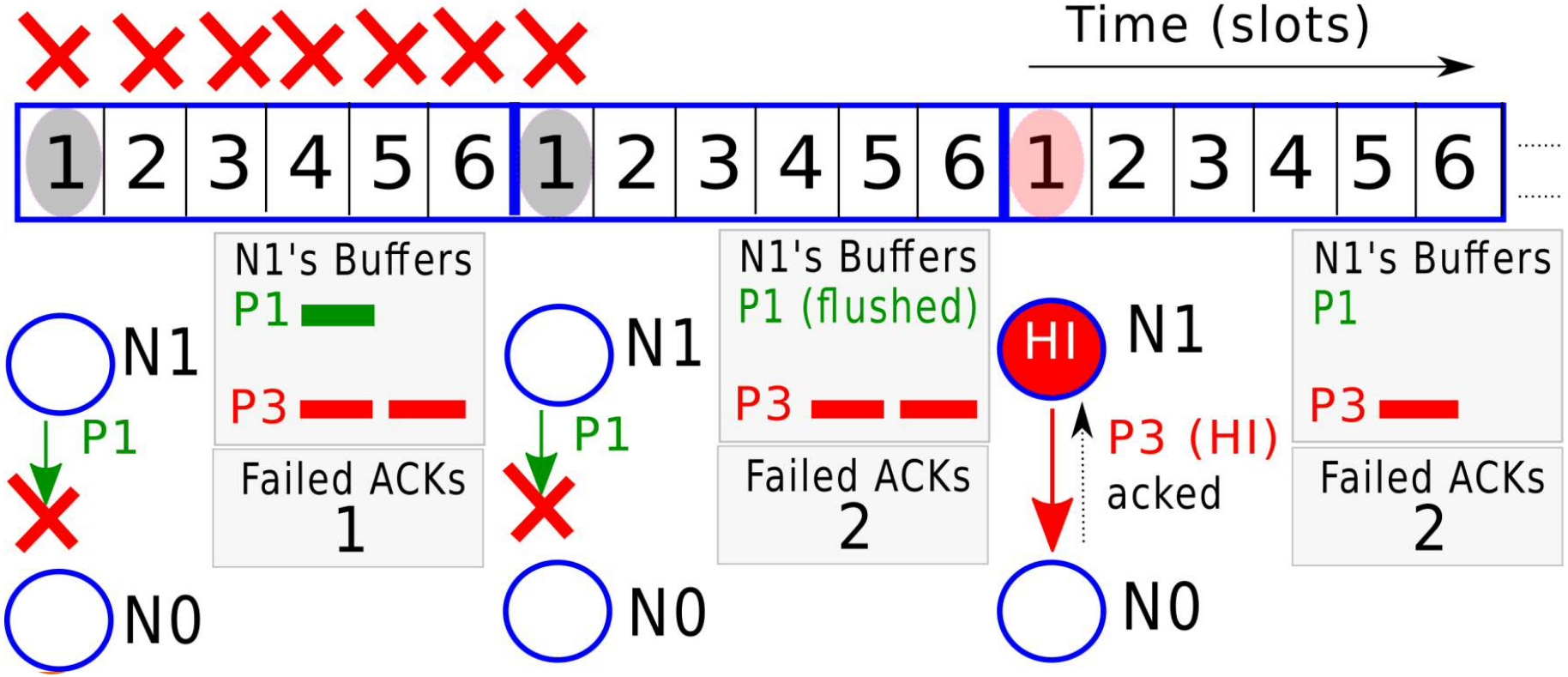
- Static slot table scheduling
  - Each node has a **slot table** specifying a cyclic schedule of slots indicating when and on which channel the node can transmit / should listen to receive
  - In this first work a single channel is assumed and the slot table is the same for every node – it just specifies which node may transmit in which slot
  - *The slot table does not indicate which packets will be transmitted*
- Dynamic fixed priority scheduling:
  - Each packet and hence its frames have a fixed priority (at each node)
  - Each node has a buffer for each priority level that can hold the frames of the corresponding packet
  - At runtime the nodes cycle through the slot table
  - When a node has an available transmission slot it transmits a frame from the highest priority non-empty buffer
  - When a frame is successfully transmitted (ACK received) it is removed from the buffer

AirTight Protocol: Mixed-Criticality behaviour

- Normal (LO-criticality) mode
  - When a node has an available transmission slot it transmits a frame from the highest priority non-empty buffer
  - If transmission is faulty (no ACK) then the **Failed-ACK** counter for the node is incremented and the frame remains in the buffer (for latter re-transmission)
- Mode switch
  - If the **Failed-ACK** counter exceeds the limit for LO-criticality mode operation then all frames of LO-criticality packets (in this node) are discarded and the node switches to HI-criticality mode
- HI-criticality mode
  - Only HI-criticality packets are transmitted – any failures cause the **Failed-ACK** counter to increment, but the frame remains in the buffer for re-transmission
  - Any LO-criticality packets arriving at the node are discarded
  - If the node has a transmission slot and there are no HI-criticality frames pending then the **Failed-ACK** counter is reset and the node returns to normal mode

AirTight Protocol: Example

- Mixed-criticality
  - Node 1 is transmitting to node 0 (Sink) and uses slot 1 in the 6-slot table
  - Node 1 has a LO-criticality packet flow P1 of (priority 1 high) with 1 frame in the buffers and also a HI-criticality packet flow P3 (priority 3 low) with 2 frames in the buffer



AirTight: Schedulability Analysis

- Aim to verify the end-to-end timing constraints of a set of packet flows on the network
- Assumptions
  - Parameters of each packet flow are given (Minimum inter-arrival time T_i, #frames C_i, source and destination)
  - Slot table supply function $S_k(X)$ gives the maximum time that the slot table can take to supply node k with X transmission slots
  - Fault function $F_k(L,t)$ gives the worst-case number of faulty transmissions by node k in time t that must be tolerated in mode L
  - Packet flow priorities are assumed to be given (return to this later)
- Analysis
  - Builds on the AMC approach for mixed-criticality fixed priority task scheduling
  - Considers each node (and packet flow) in turn

AirTight: Schedulability Analysis

- Normal (LO-criticality) mode:
  - Analysis uses fault function $F_k(LO, t)$ that gives worst-case number of faulty transmissions by node k in time t that must be tolerated in LO-criticality mode
  - Compute the number of transmission slots X needed to accommodate the frames of the packet of interest and all higher priority packet flows on the node accounting for all of the transmission slots that can be disrupted by faults

$$X = C_i + F_k(LO, S_k(X)) + \sum_{j \in \mathbf{hp}(i)} \left\lceil \frac{S_k(X)}{T_j} \right\rceil$$

  - The response time is then the time taken by the node to supply X transmission slots

$$R_i(LO) = S_k(X)$$

  - Iteration starts with $X = C_i$ and ends on convergence or when $R_i(LO)$ exceeds the deadline

AirTight: Schedulability Analysis

- HI-criticality mode:
  - Similar structure using the fault function $F_k(HI, t)$ that gives the worst-case number of faulty transmissions by node k in time t that must be tolerated in HI-criticality mode
  - Separates impact of higher priority LO-criticality and HI-criticality packets, since the former are discarded after at most time $R_i(LO)$
  - Number of slots required:

$$X = C_i + F_k(HI, S_k(X)) + \sum_{\tau_j \in \mathbf{hpH}(i)} \left\lceil \frac{S_k(X)}{T_j} \right\rceil C_j + \sum_{\tau_k \in \mathbf{hpL}(i)} \left\lceil \frac{R_i(LO)}{T_k} \right\rceil C_k$$

  - Response time:

$$R_i(HI) = S_k(X)$$

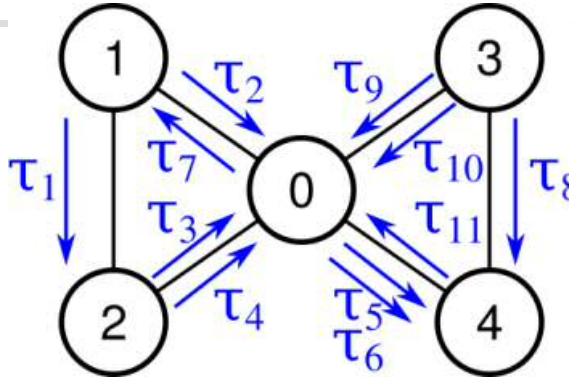
LO-criticality packets can interfere only as long as $R_i(LO)$ then discarded

Multi-hop analysis and other aspects

- Multi-hop
  - Analysed by splitting the problem into multiple single-hop packet sub-flows
  - Each sub-flow has same parameters but the deadline is some fraction of end-to-end deadline
  - Sum the worst-case response times of single hop sub-flows to obtain multi-hop response time
- Release jitter elimination
  - Method adapted from work on CAN which is used to ensure that in the worst-case each sub-flow has a periodic behaviour (no jitter accumulation on route)
- Priority assignment
  - Global priorities can be assigned according to deadlines or Audsley's algorithm used to provide optimal local (per node) priorities to each packet flow
- Slot table
  - Simple table construction method outlined in the paper

Case Study: 5 Node system

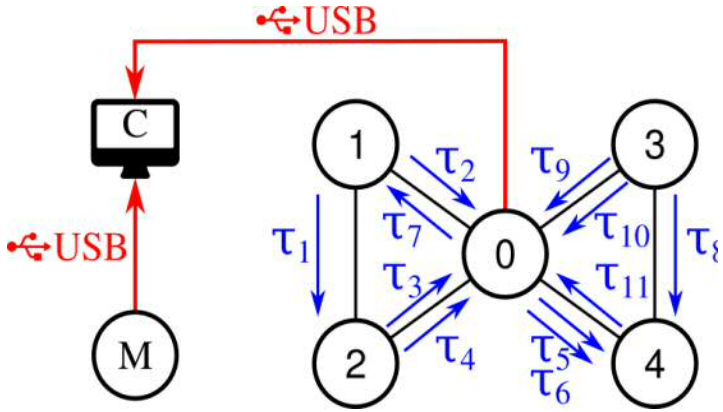
- Properties
  - 9 packet flows (two of them via the central node are split into 2 sub-flows)
  - Fault model assumes that a black-out period of 5 slots every 100 must be tolerated by LO-criticality flows, and 15 in 100 by HI-criticality flows
  - Slot table constructed has 6 slots (node-0 has one extra slot used for time synchronisation)
  - System is schedulable with response times given in the table (for the criticality level of each flow)



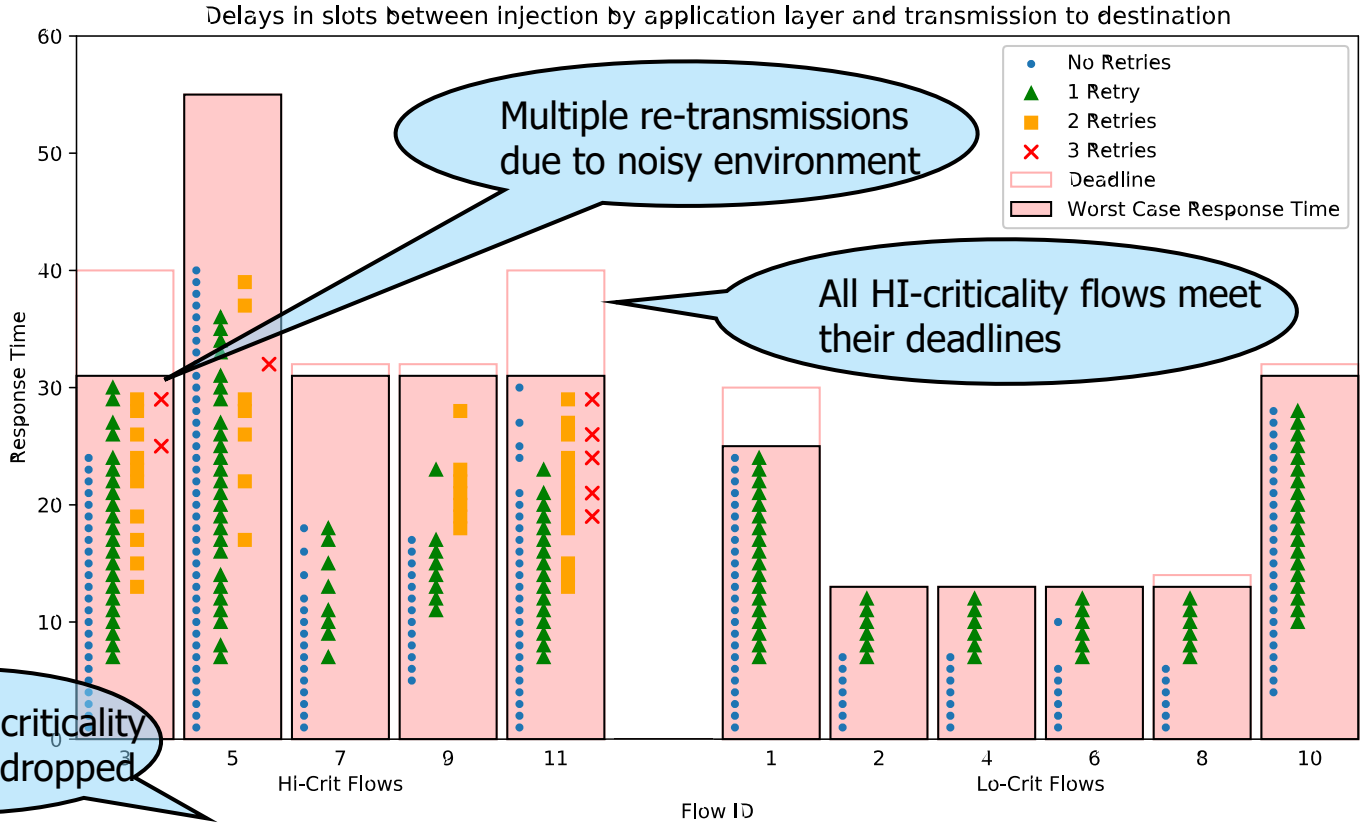
Name	From	To	Criticality	T	D	C	P	R
τ_1	n_1	n_2	LO	30	30	2	2	25
τ_2	n_1	n_0	LO	26	13	1	1	13
τ_3	n_2	n_0	HI	40	40	1	2	31
τ_4	n_2	n_0	LO	13	13	1	1	13
τ_5	n_0	n_4	HI	55	55	3	3	55
τ_6	n_0	n_4	LO	26	13	1	1	13
τ_7	n_0	n_1	HI	64	32	1	2	31
τ_8	n_3	n_4	LO	32	14	1	1	13
τ_9	n_3	n_0	HI	64	32	1	2	31
τ_{10}	n_3	n_0	LO	32	32	2	3	31
τ_{11}	n_4	n_0	HI	40	40	2	1	31

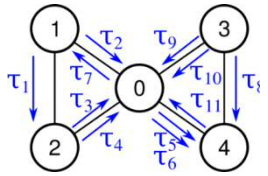
Experiments: 5 Node system

- Prototype hardware
  - IEEE 802.15.4 compliant Iris XM-2100 nodes running TinyOS version 2
  - 5 nodes (0-4) running the AirTight protocol
  - Monitoring node M listens to all packets and collects the results
  - Experiment runs for hyper-period (100ms slots ≈ 5 hours)
  - Results collected include LO-criticality packet dropping rates and response time distributions
  - No intentional fault injection but the office environment has interference from wireless access points

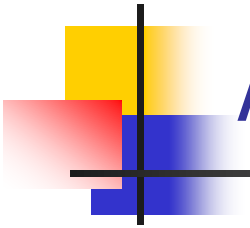


Experiment Results: 5 node system





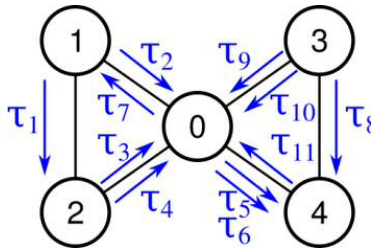
LO drop	Flow 1	Flow 2	Flow 4	Flow 6	Flow 8	Flow 10
10ms	1.13%	0.79%	1.59%	0.40%	0.32%	1.56%
100ms	0.19%	0.34%	0.90%	3.88%	0.21%	0.83%

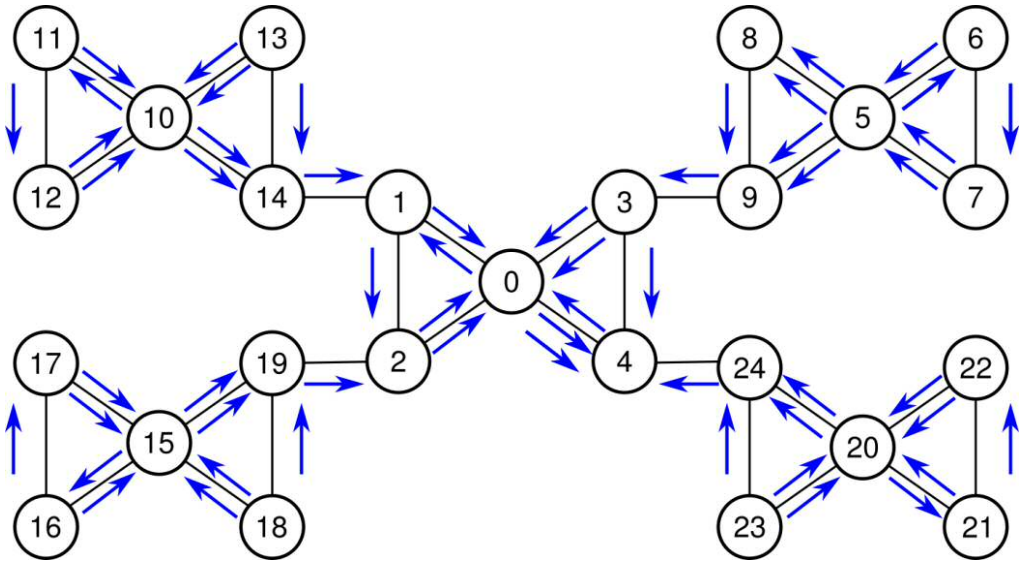


AirTight: Protocol Simulator

- Aims to provide a consistent environment for protocol testing allowing fault injection free from environmental interference
- High level simulation rather than emulation
  - Slot based with time incremented by transmission slots
  - Does not model internal component structure but rather abstractions of the node state (e.g. node data structures containing buffers, criticality mode, Failed-ACK counter and so on)
  - Allows for fault simulation of varying lengths and different faults on different links
  - Facilitates simulation of much larger systems

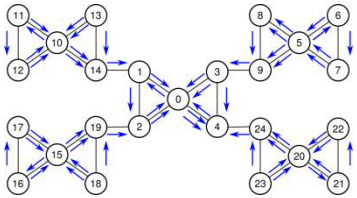
Simulation: 25-node system

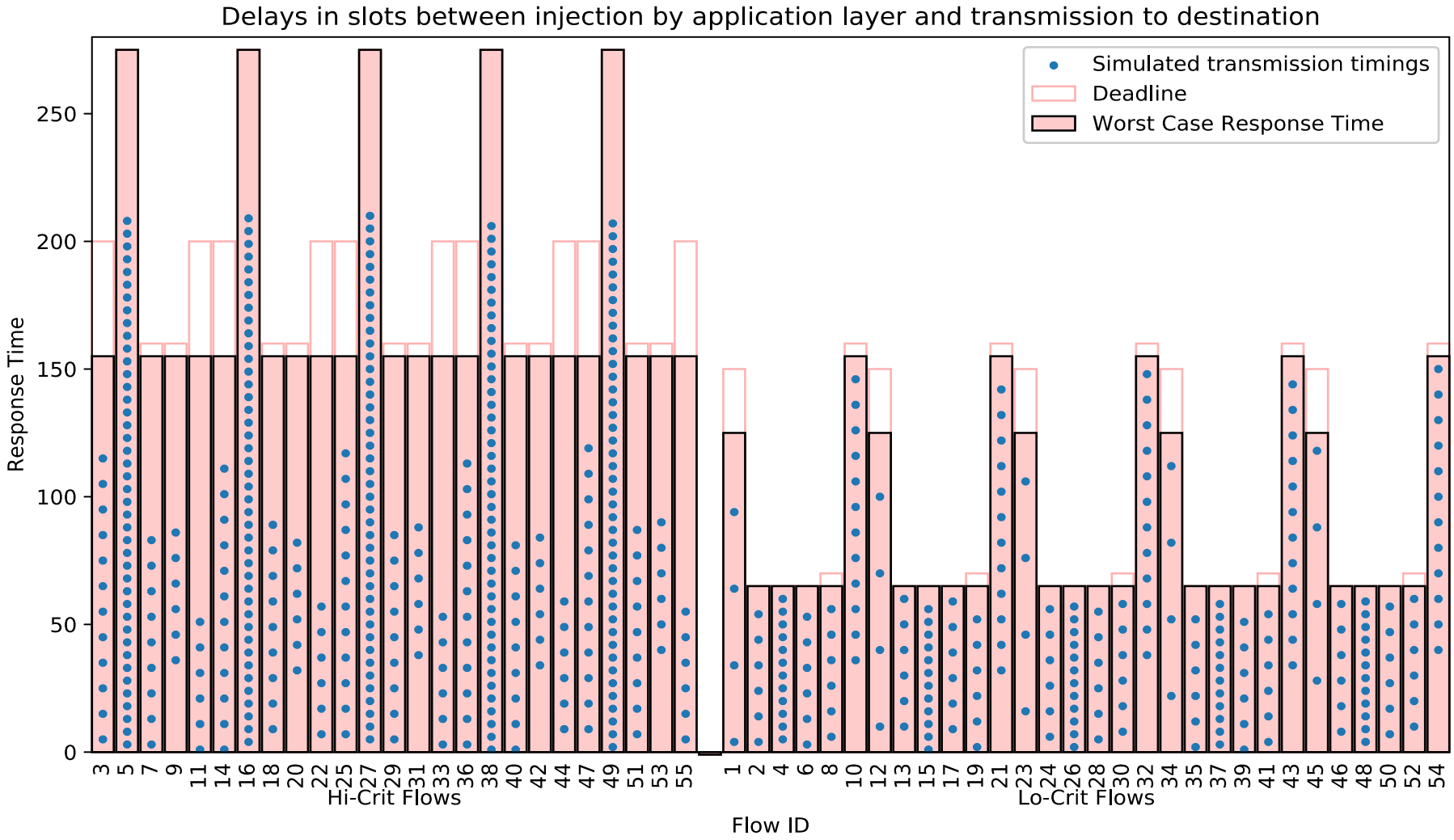




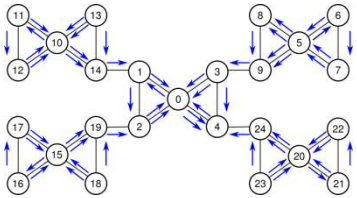
25 node example with 55 packet flows generated from topology of small example

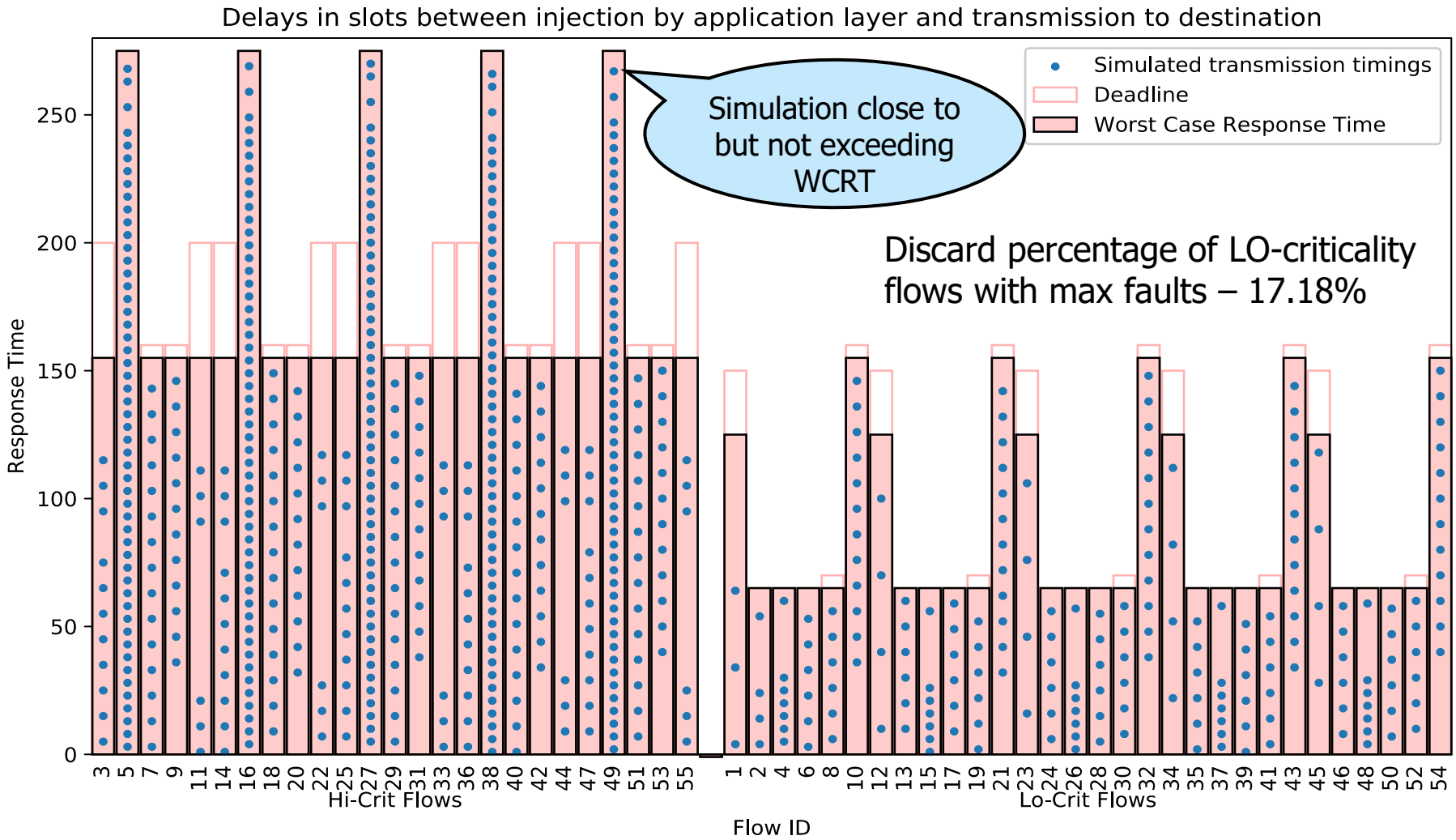
Simulation Results: Faults within LO-Criticality Fault Model





Simulation Results: Faults within HI-Criticality Fault Model



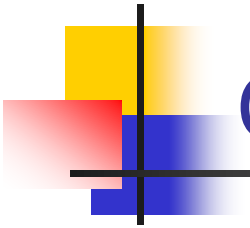


Conclusions

- AirTight Protocol
  - Combines (global) static slot tables with (local) dynamic fixed priority scheduling of packet flows
  - Address transmission errors through temporal redundancy
  - Mixed-criticality approach provides resilience according to a specified fault model for both LO- and HI-criticality packet flows
  - Achieves increased resilience for HI-criticality packet flows by dropping LO-criticality packets
  - Criticality mode changes are actioned locally by each node
- Initial experiments and simulation
  - Verify timing behaviour with respect to analytically computed worst-case response times
  - Demonstrate that the protocol meets its goals in the presence of transmission faults

Ongoing and future work

- Improvements to the analysis for multiple hops and high levels of faults
- Extensions to the multi-channel case
  - Including packets with affinities (i.e. that can use multiple channels) – related analysis techniques and methods for generating slot tables
- Extensions to the multi-domain case
  - Where more than one node is allowed to transmit simultaneously on a single channel - related methods for generating slot tables
- Dynamic slot table modification and changes
  - How to update the slot table on the fly



Questions?