Evaluation of Methods for Prediction and Computation of Delay Violation Probability in Industrial URLLC Applications

Hao-Yun Hsu, Neda Petreska

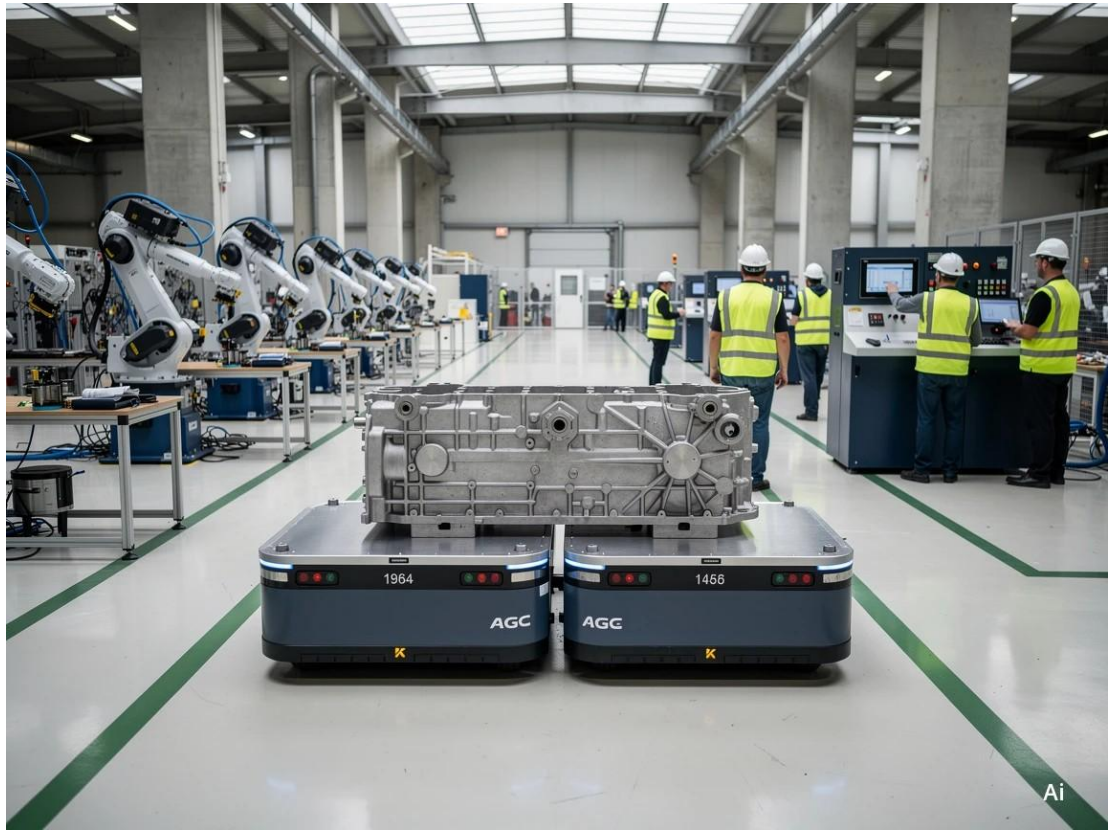
June 23, 2026, Siemens AG

Presented at OAI Workshop 2026

SIEMENS

Motivation

- **Reliable low-latency communication** is a key requirement in industrial automation applications, e.g., in the domain of cyber-physical systems, cooperative robotics and smart manufacturing.
- Average delay vs. delay violation probability
- Packet delay is influenced by **multiple interacting factors**, such as queuing, encoding/decoding, and retransmissions.
- As a result, **delay violation events** are difficult to anticipate using end-to-end delay measurements alone.
- Accurate delay violation probability analysis requires **delay decomposition**, **theoretical analysis**, and **prediction methods.**



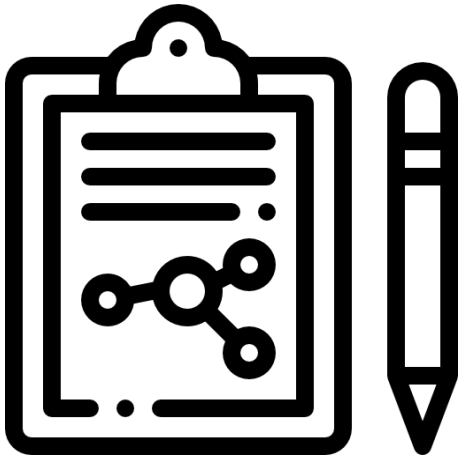
Delay Violation Probability: Definition and Analysis Approaches

Probability that total delay exceeds target threshold

$$\mathrm{DVP} = \mathrm{P}\left(\tau_{\sum} > d\right),$$

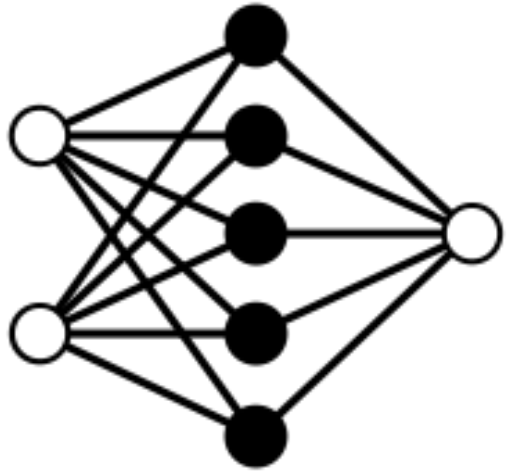
d: $target\ delay\ threshold$

$\tau_{\sum}$: $total\ packet\ delay$



Theoretical Analysis

Experimental Measurement
End-to-end Delay Analytics Framework
(USRP B210+OAI)



Learning-Based Prediction
(LSTM/Transformer)

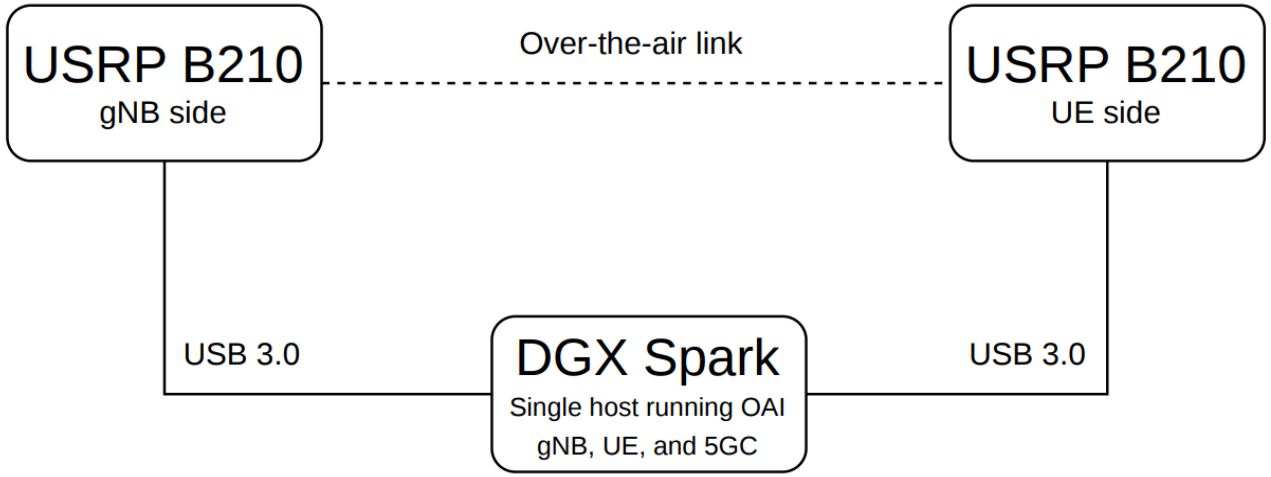
SIEMENS

Experimental Setup & EDAF Decomposition: OAI on DGX Spark with USRP B210

- OpenAirInterface (OAI) provides a flexible and open 5G research platform.
- The testbed uses OAI on DGX Spark with two USRP B210 devices.
- The setup is analyzed using EDAF delay decomposition.

DGX Spark

Our setup



Based on Cammerer et al., 2025.
DGX Spark image source: NVIDIA.

SIEMENS

Experimental Measurements

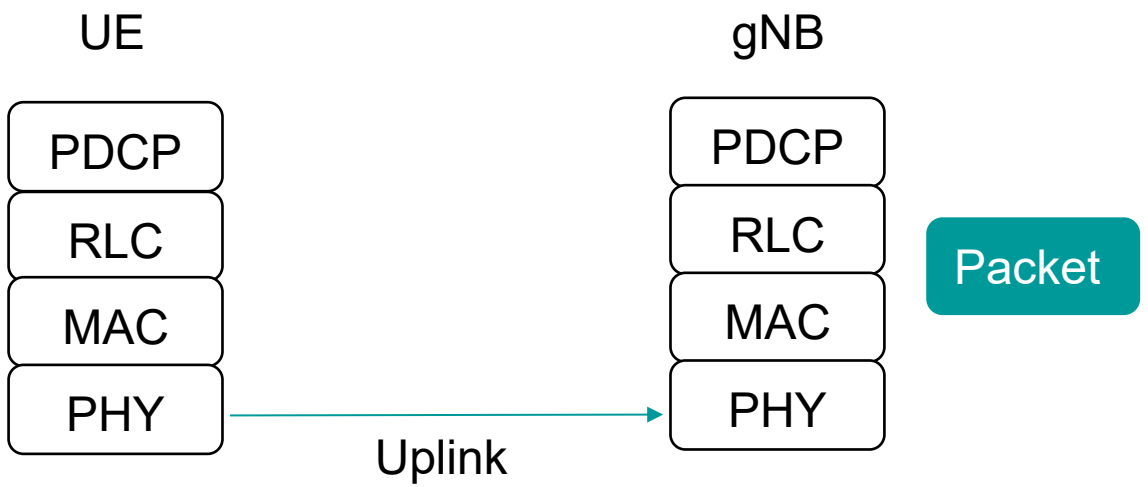
- Measure UDP packets transmission under various:
  - Physical Distance (0.5m/1m/1.5m)
  - Physical Resource Blocks (51/106)
  - Packet Inter-arrival Time (2ms/5ms/10ms/20ms)
  - Packet Size (50B/100B)

- Introduce End-to-end Delay Analysis Framework (EDAF) into our setup:
  - Decompose a packet's journey into different components
  - Inserting timestamps into the OAI code for log output
  - Use the format: t dir src--dest prop:globalId:localId

SIEMENS

EDAF – A graphical example

- We want to know the time of a packet stays in RLC layer.
  1. Packet MAC --> RLC



Based on Mostafavi et al., 2025; Moothedath et al., 2025

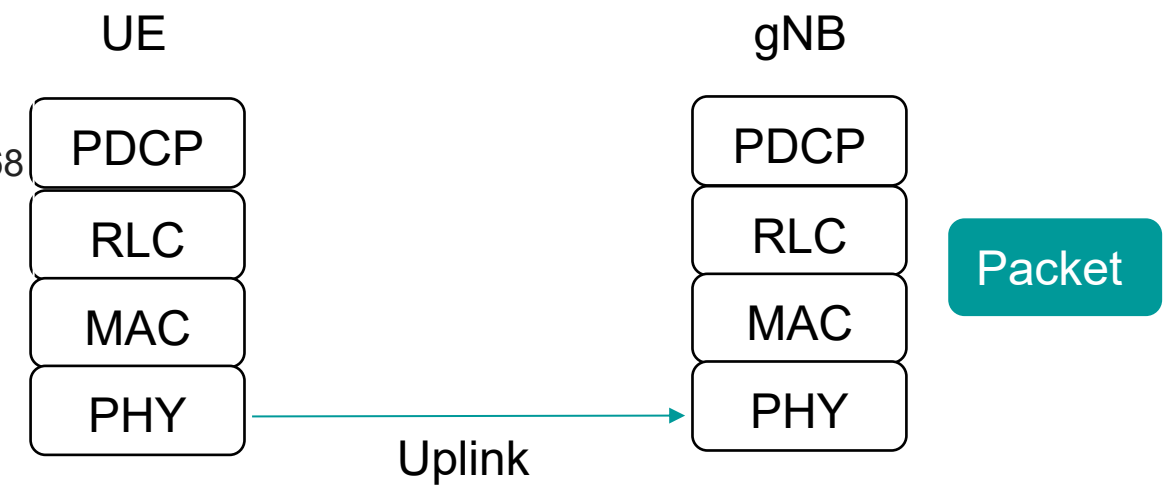
SIEMENS

EDAF – A graphical example

- We want to know the time of a packet stays in RLC layer.
  1. Packet MAC --> RLC
  2. Starts a thread to write the string into the ring buffer

Thread 1

14183704354389 U mac.demuxed--rlc.decoded len109::fm467.sl8.lcid4.hqpid1.Hbuf221047680.MRbuf221047683.rnti7f22



Based on Mostafavi et al., 2025; Moothedath et al., 2025

SIEMENS

EDAF – A graphical example

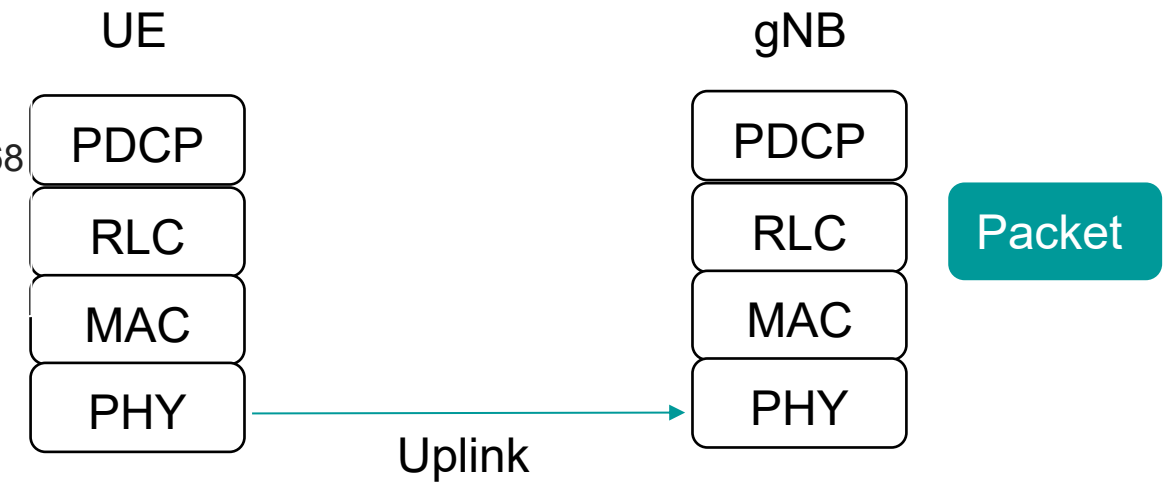
- We want to know the time of a packet stays in RLC layer.
  1. Packet MAC --> RLC
  2. Starts a thread to write the string into the ring buffer
  3. Starts a thread in RLC layer

Thread 1

14183704354389 U mac.demuxed--rlc.decoded len109::fm467.sl8.lcid4.hqpid1.Hbuf221047680.MRbuf221047683.rnti7f22

Thread 2

14183704358463 U rlc.decoded--rlc.reassembled len104::MRbuf221047683.p0.si2.sn6.so427



Based on Mostafavi et al., 2025; Moothedath et al., 2025

SIEMENS

EDAF – A graphical example

- We want to know the time of a packet stays in RLC layer.
  1. Packet MAC --> RLC
  2. Starts a thread to write the string into the ring buffer
  3. Starts a thread in RLC layer
  4. Packet RLC --> PDCP

Thread 1

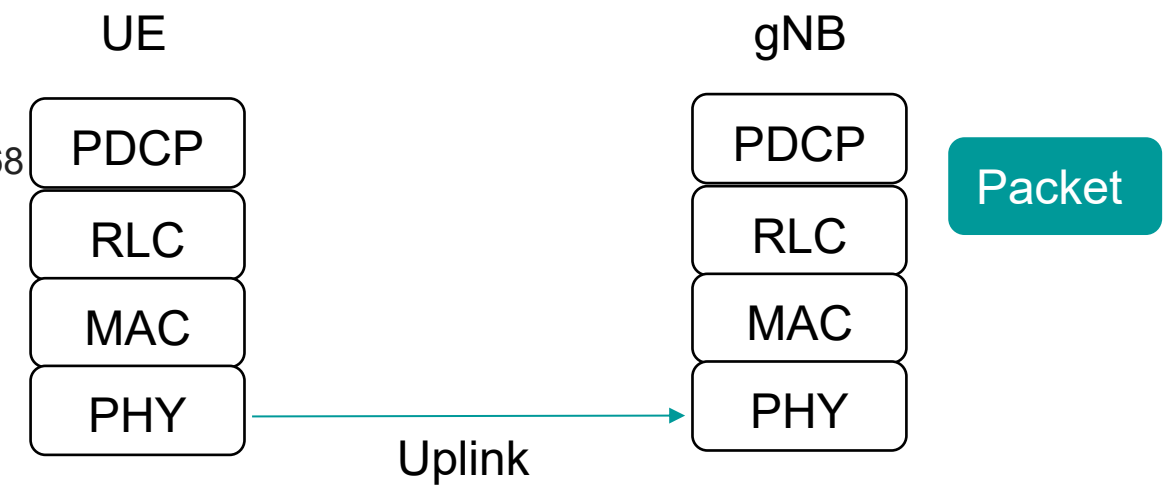
14183704354389 U mac.demuxed--rlc.decoded len109::fm467.sl8.lcid4.hqpid1.Hbuf221047680.MRbuf221047683.rnti7f22

Thread 2

14183704358463 U rlc.decoded--rlc.reassembled len104::MRbuf221047683.p0.si2.sn6.so427

Thread 3

14183704362612 U rlc.reassembled--pdcp.ind len531::sn6.PIBuf3749066208



Based on Mostafavi et al., 2025; Moothedath et al., 2025

SIEMENS

EDAF – A graphical example

- We want to know the time of a packet stays in RLC layer.
  1. Packet MAC --> RLC
  2. Starts a thread to write the string into the ring buffer
  3. Starts a thread in RLC layer
  4. Packet RLC --> PDCP

Thread 1

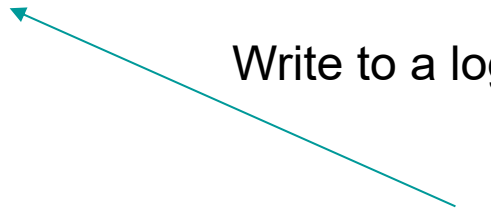
14183704354389 U mac.demuxed--rlc.decoded
len109::fm467.sl8.lcid4.hqpid1.Hbuf221047680.MRbuf221047683.rnti7f22

Thread 2

14183704358463 U rlc.decoded--rlc.reassembled
len104::MRbuf221047683.p0.si2.sn6.so427

Thread 3

14183704362612 U rlc.reassembled--pdcp.ind
len531::sn6.PIBuf3749066208



Write to a log file

Data Collector Thread
(Round Robin)

Based on Mostafavi et al., 2025; Moothedath et al., 2025

SIEMENS

EDAF – A graphical example

- We want to know the time of a packet stays in RLC layer.
  1. Packet MAC --> RLC
  2. Starts a thread to write the string into the ring buffer
  3. Starts a thread in RLC layer
  4. Packet RLC --> PDCP

Thread 1

14183704354389 U mac.demuxed--rlc.decoded
len109::fm467.sl8.lcid4.hqpid1.Hbuf221047680.MRbuf22104768
3.rnti7f22

Thread 2

14183704358463 U rlc.decoded--rlc.reassembled
len104::MRbuf221047683.p0.si2.sn6.so427

Thread 3

14183704362612 U rlc.reassembled--pdcp.ind
len531::sn6.PIBuf3749066208

Data Collector Thread
(Round Robin)

Based on Mostafavi et al., 2025; Moothedath et al., 2025

SIEMENS

EDAF – A graphical example

- We want to know the time of a packet stays in RLC layer.
  1. Packet MAC --> RLC
  2. Starts a thread to write the string into the ring buffer
  3. Starts a thread in RLC layer
  4. Packet RLC --> PDCP

Thread 1

14183704354389 U mac.demuxed--rlc.decoded
len109::fm467.sl8.lcid4.hqpid1.Hbuf221047680.MRbuf221047683.rnti7f22

Thread 2

14183704358463 U rlc.decoded--rlc.reassembled
len104::MRbuf221047683.p0.si2.sn6.so427

Thread 3

14183704362612 U rlc.reassembled--pdcp.ind
len531::sn6.PIBuf3749066208

Data Collector Thread
(Round Robin)

Based on Mostafavi et al., 2025; Moothedath et al., 2025

SIEMENS

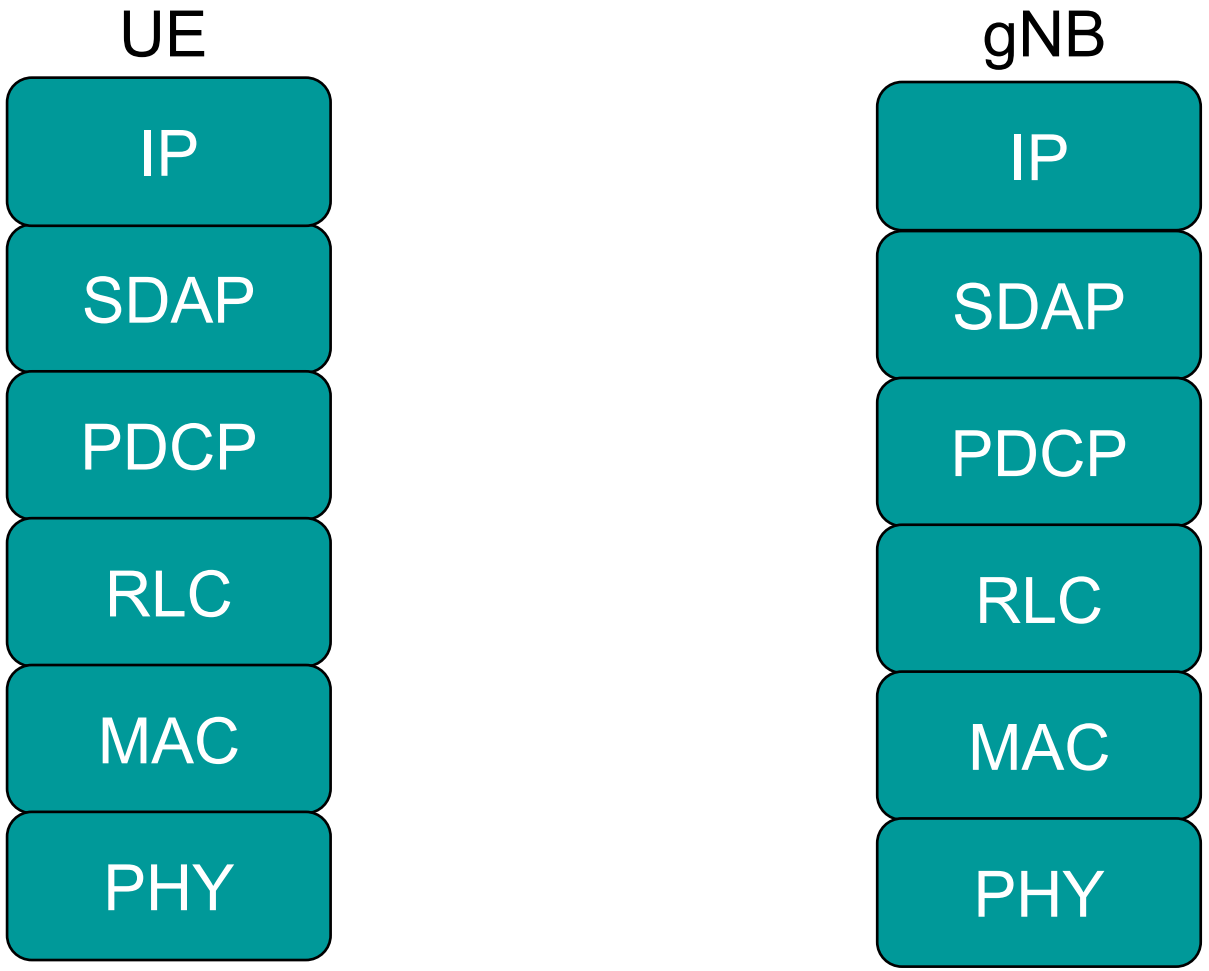
Time Conversion

- Runtime: Low-overhead delay tagging
  1. Record the virtual timer count from system register
  2. Once per second, record the Unix time and virtual timer count as follows (“S logs”)
     - 3530958760381 S rdtsc—gettimeofday 1778358211.798450864

- Timeline reconstruction for post-processing
  Calculate linear interpolations between every two "S logs"

SIEMENS

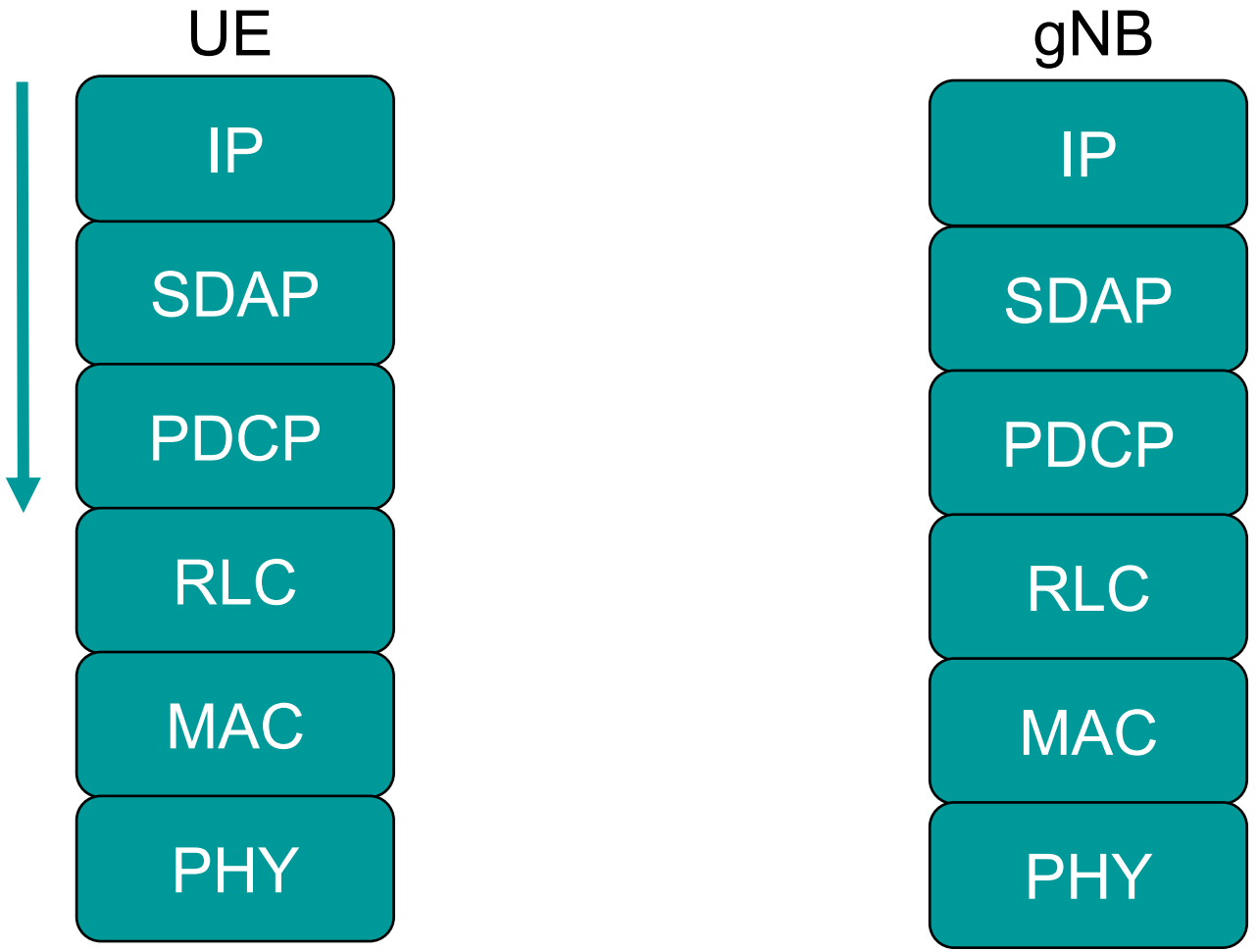
A complete End-to-end Delay Decomposition - Uplink



SIEMENS

A complete End-to-end Delay Decomposition - Uplink

ip2rlc delay
(From IP to RLC queue)

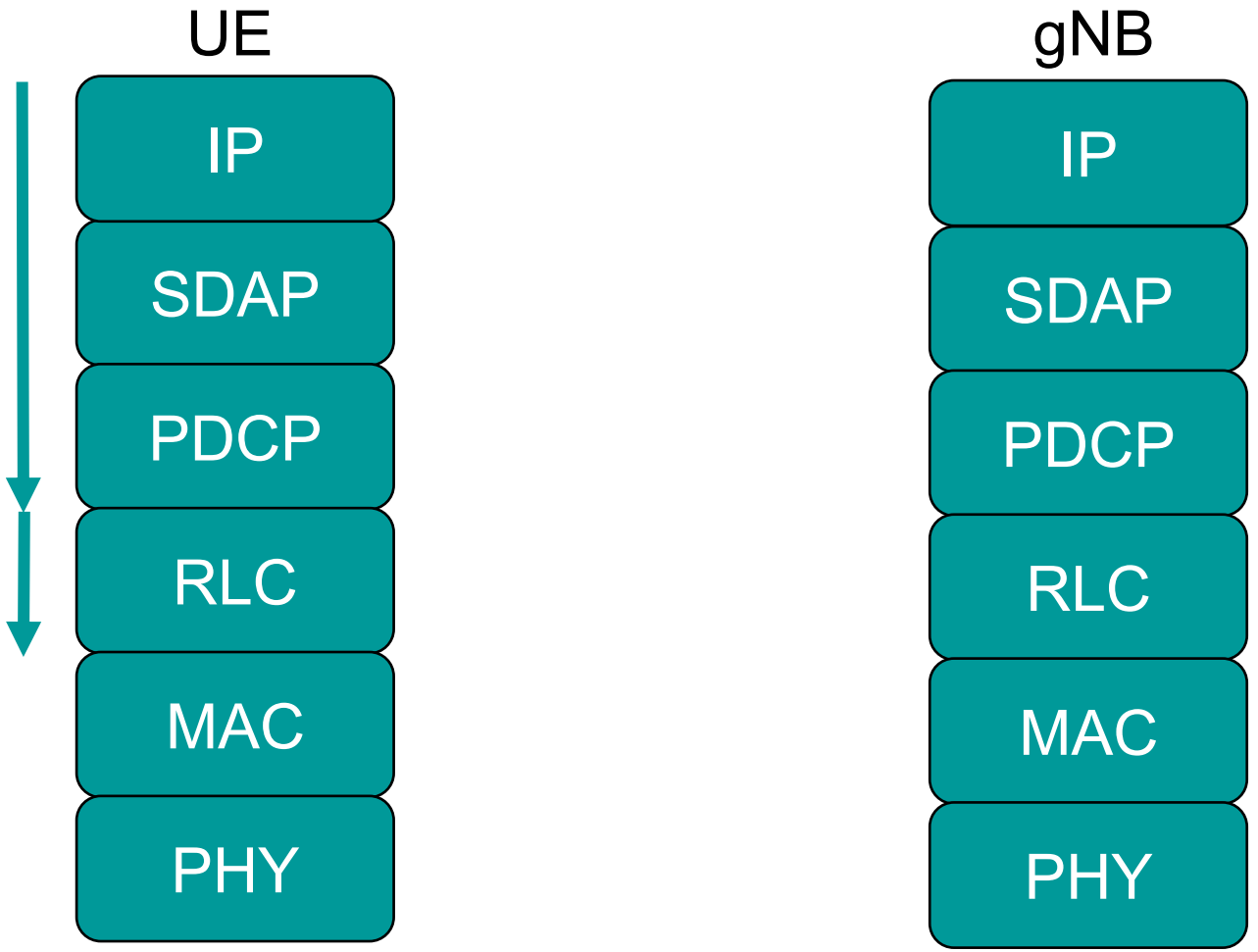


SIEMENS

A complete End-to-end Delay Decomposition - Uplink

ip2rlc delay
(From IP to RLC queue)

rlc_queuing_delay
(From RLC queue to RLC pdu)

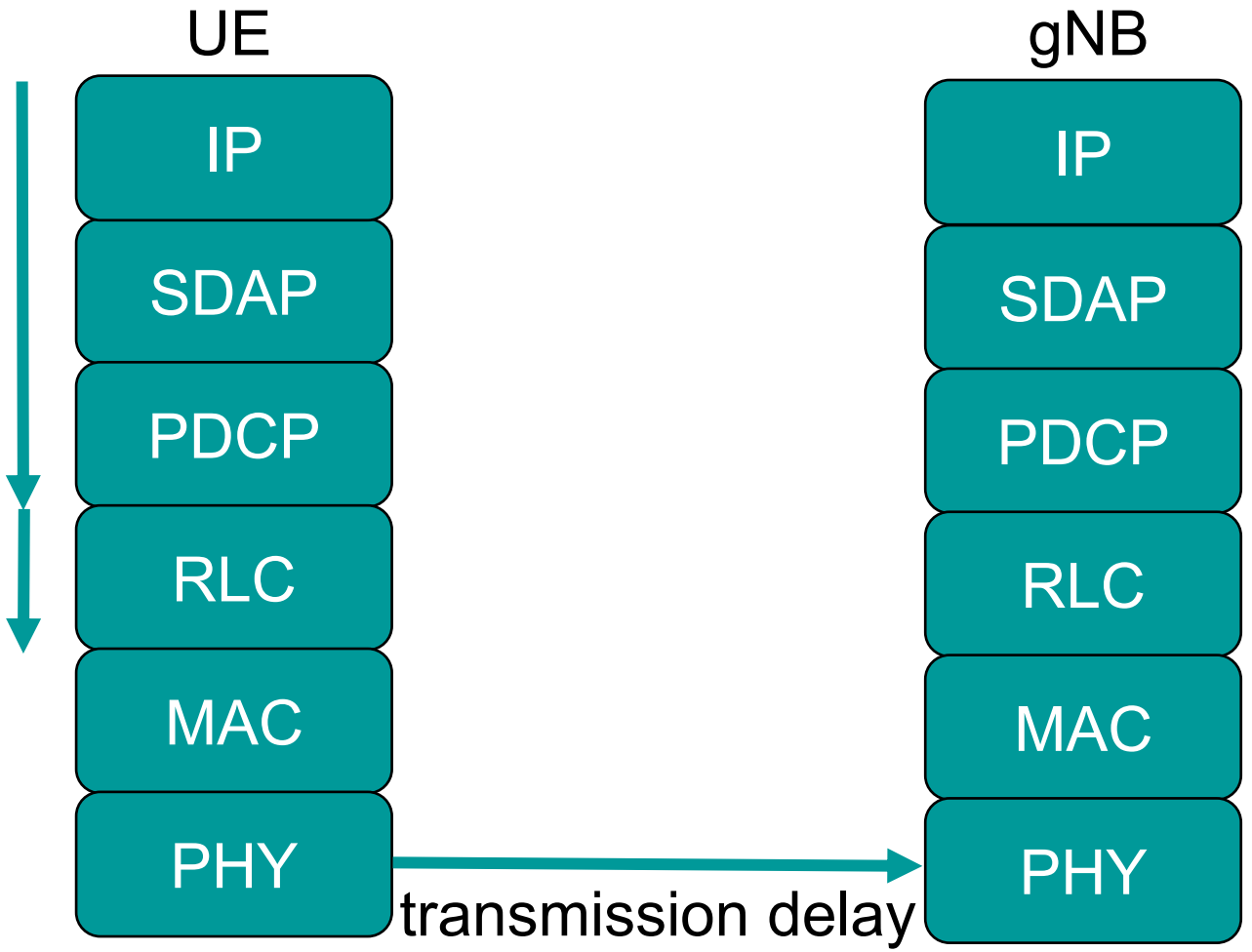


SIEMENS

A complete End-to-end Delay Decomposition - Uplink

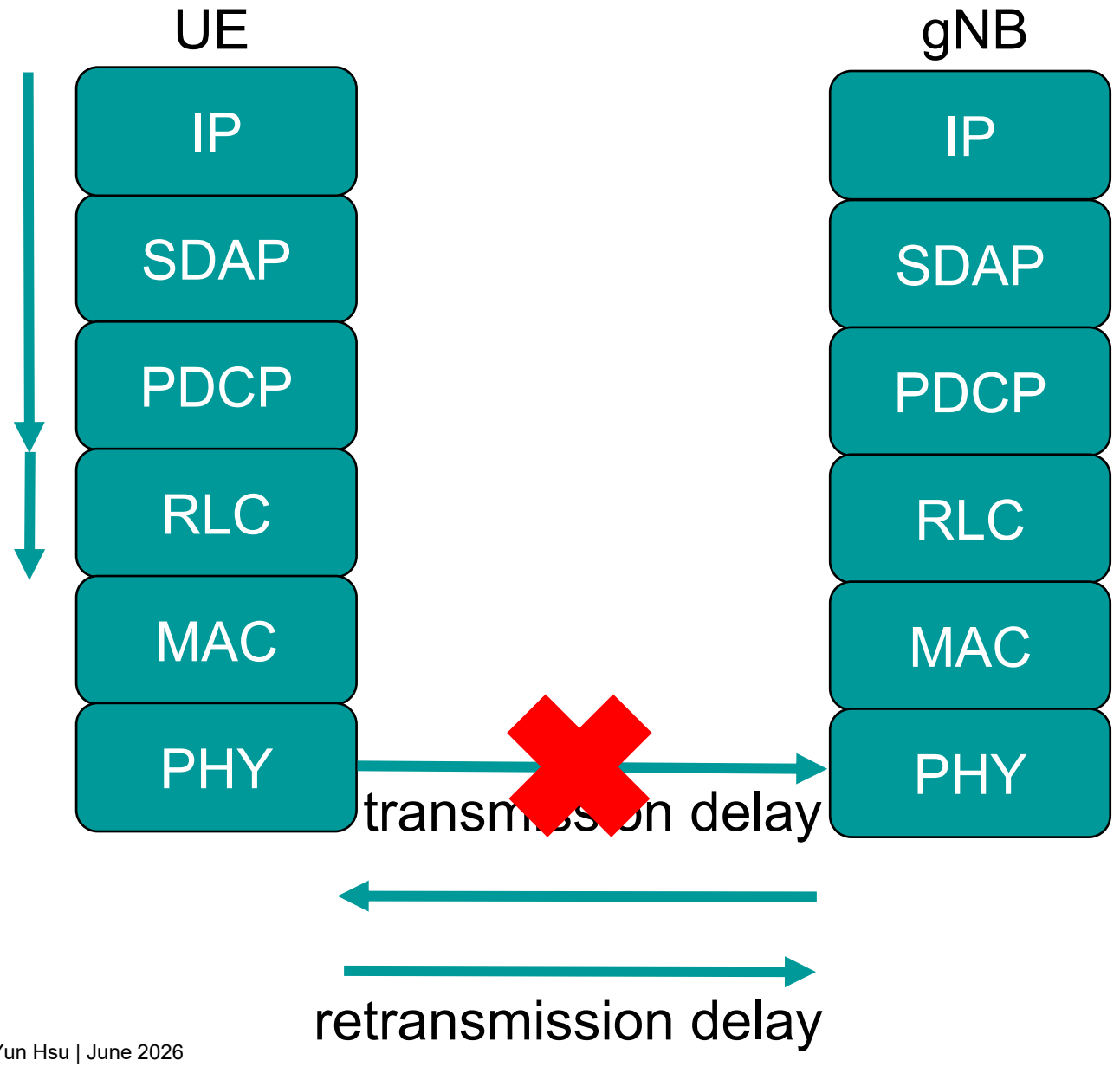
ip2rlc delay
(From IP to RLC queue)

rlc_queuing_delay
(From RLC queue to RLC pdu)



SIEMENS

A complete End-to-end Delay Decomposition - Uplink

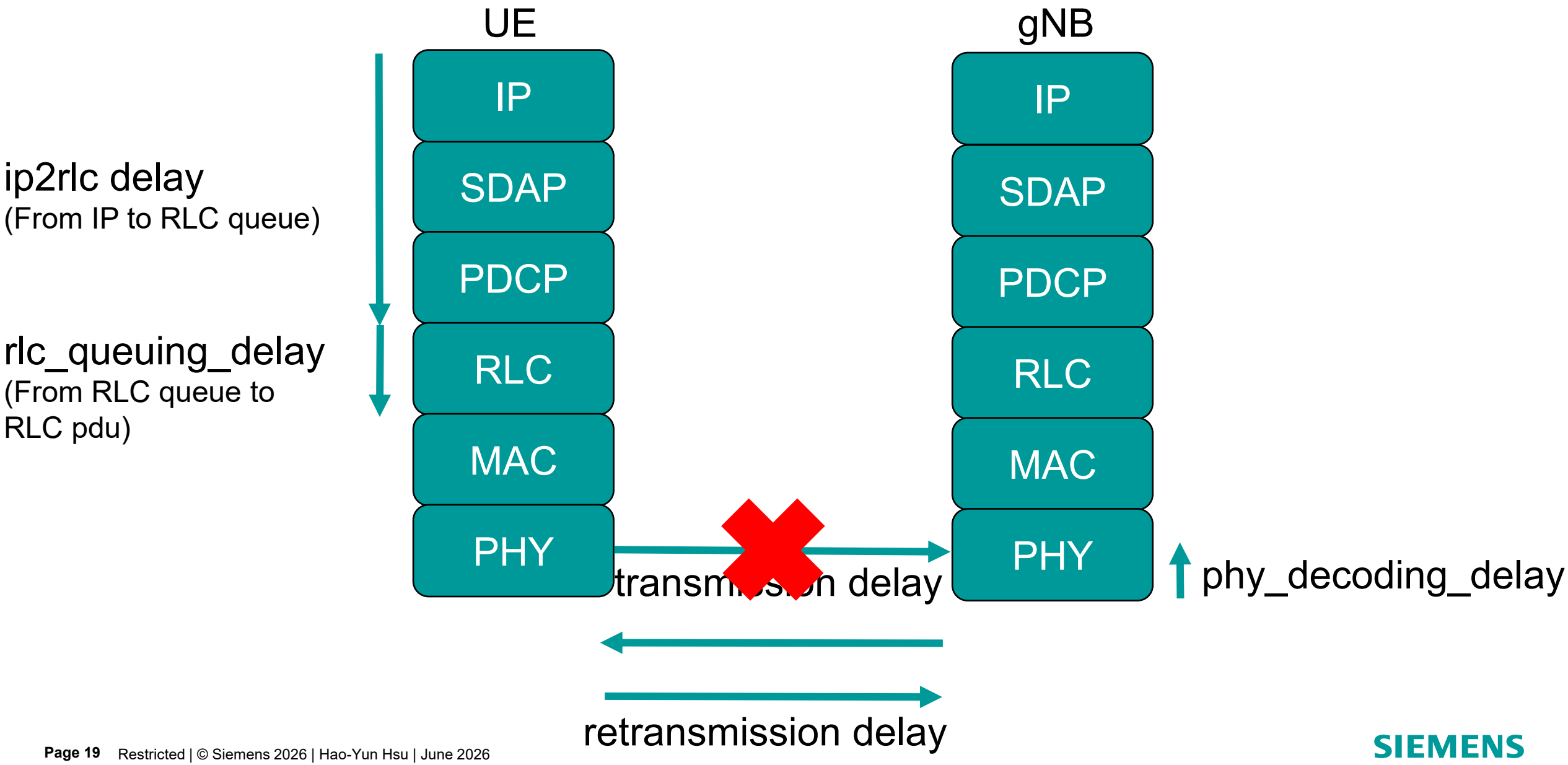


ip2rlc delay
(From IP to RLC queue)

rlc_queuing_delay
(From RLC queue to RLC pdu)

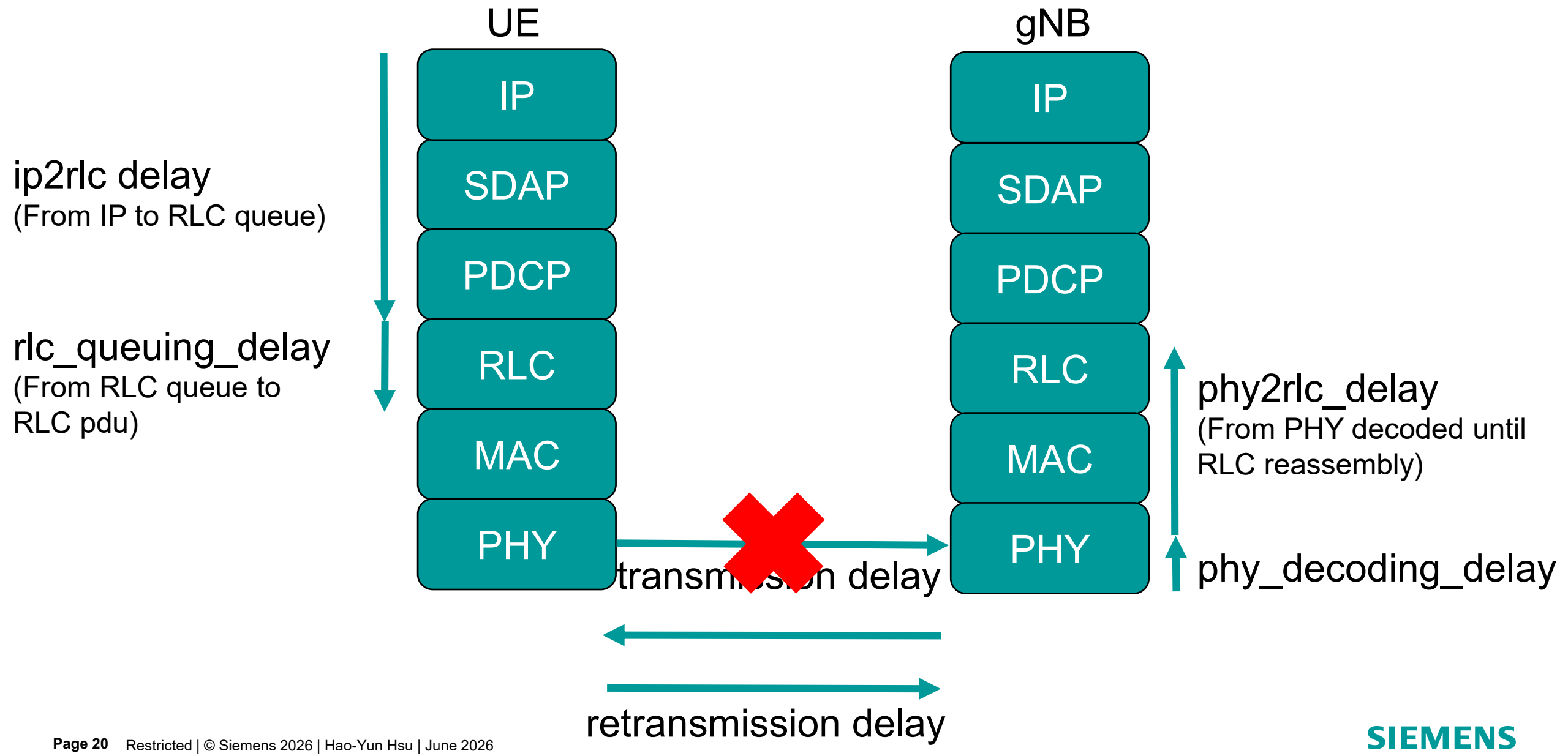
SIEMENS

A complete End-to-end Delay Decomposition - Uplink



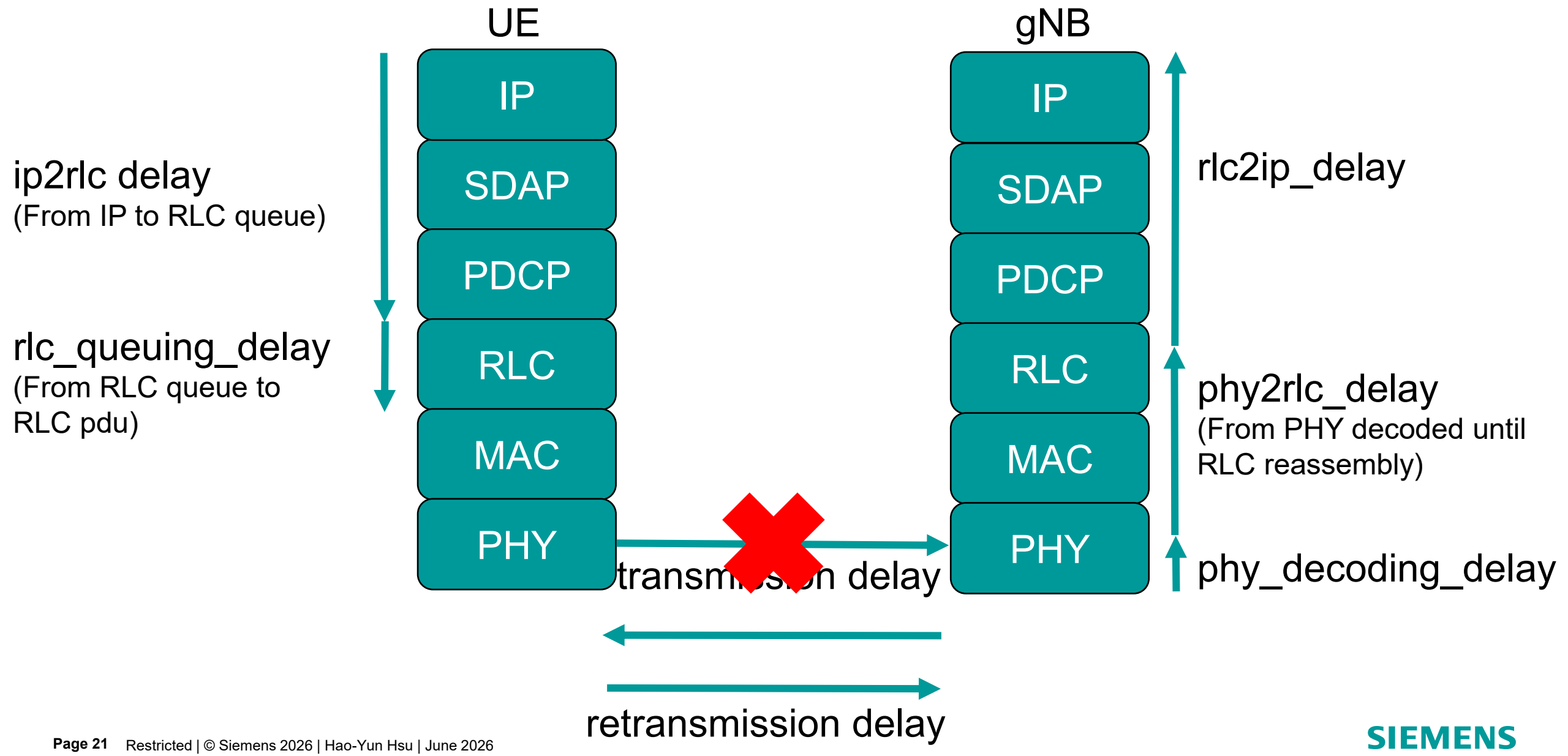
SIEMENS

A complete End-to-end Delay Decomposition - Uplink



SIEMENS

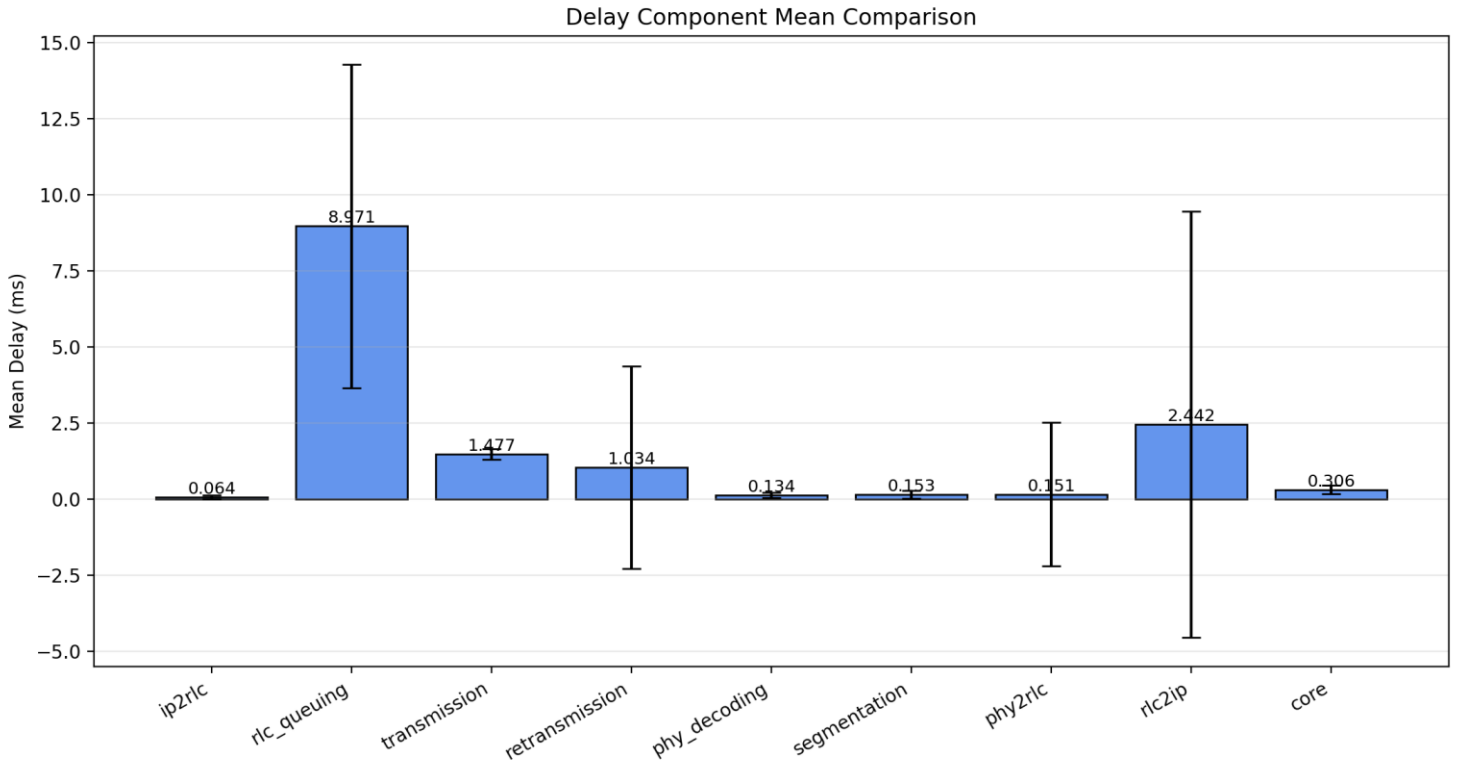
A complete End-to-end Delay Decomposition - Uplink



SIEMENS

Delay decomposition reveals the dominant component of uplink delay

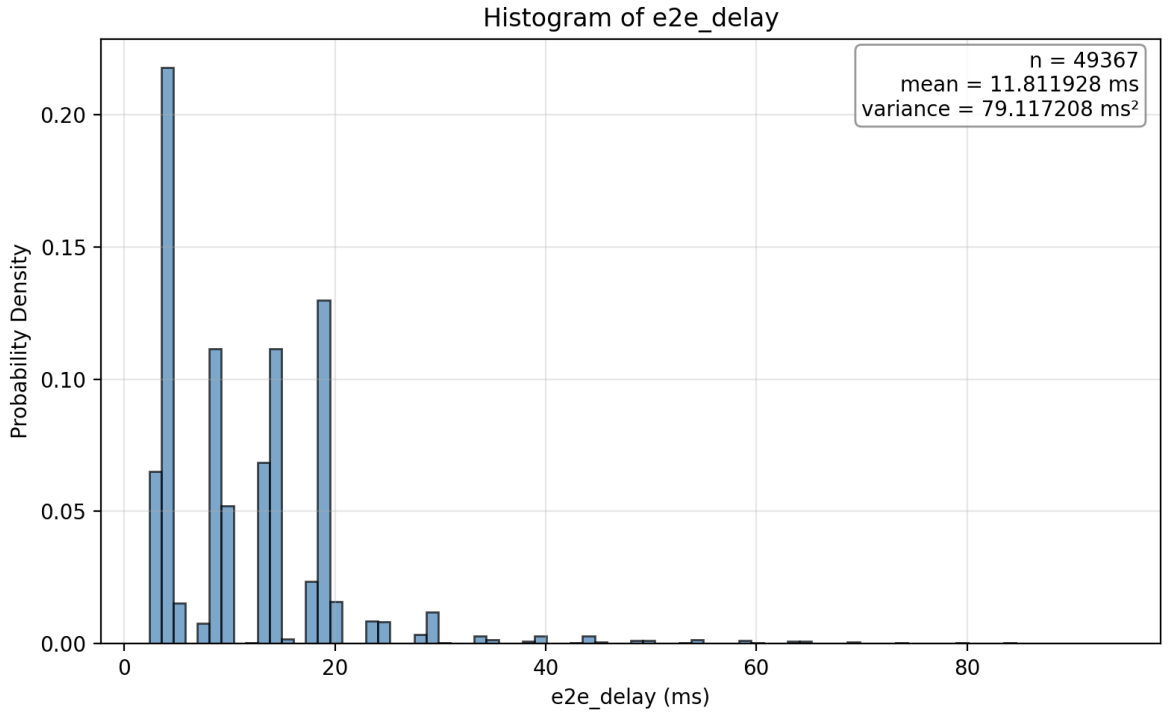
- End-to-end delay is not evenly distributed across protocol layers.
- In this setup, RLC queuing dominates the mean uplink delay.
- Transmission contributes less than queuing, while the remaining components are relatively small.

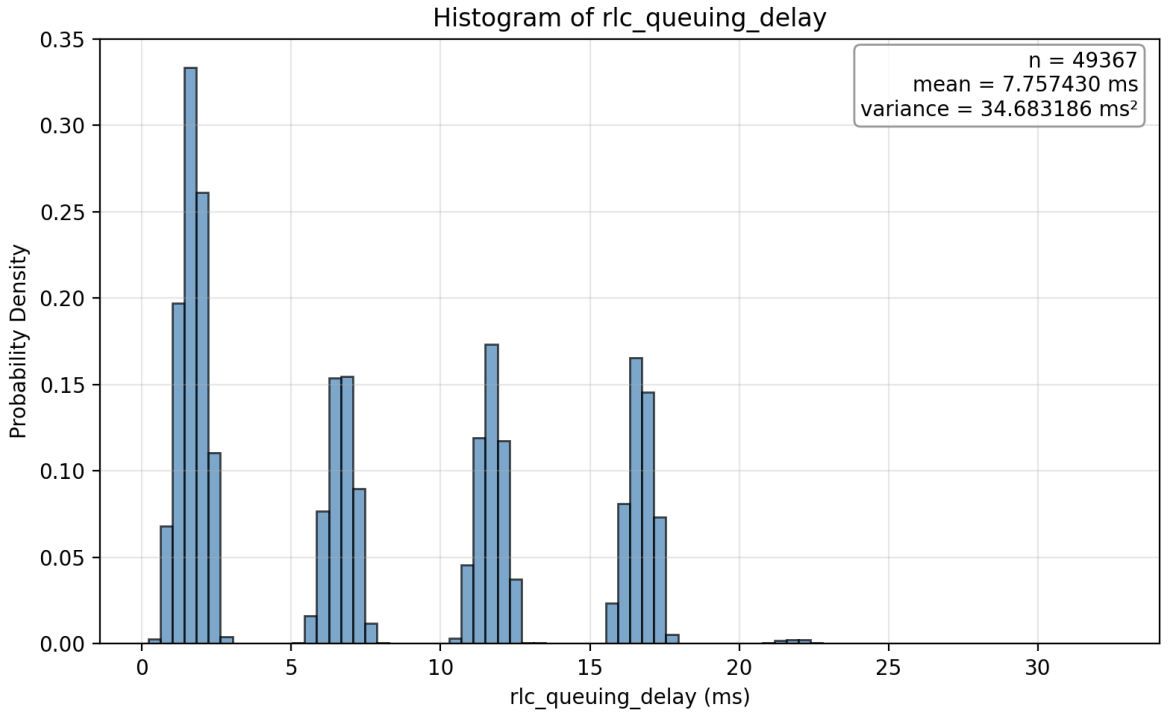


SIEMENS

End-to-end delay closely follows RLC queuing

- The histogram of end-to-end delay closely resembles that of RLC queuing delay.
- This similarity is visible in the overall distribution shape.
- Hence, the variability of end-to-end delay is largely inherited from the queuing process.
- This makes RLC queuing a key component for explaining DVP.

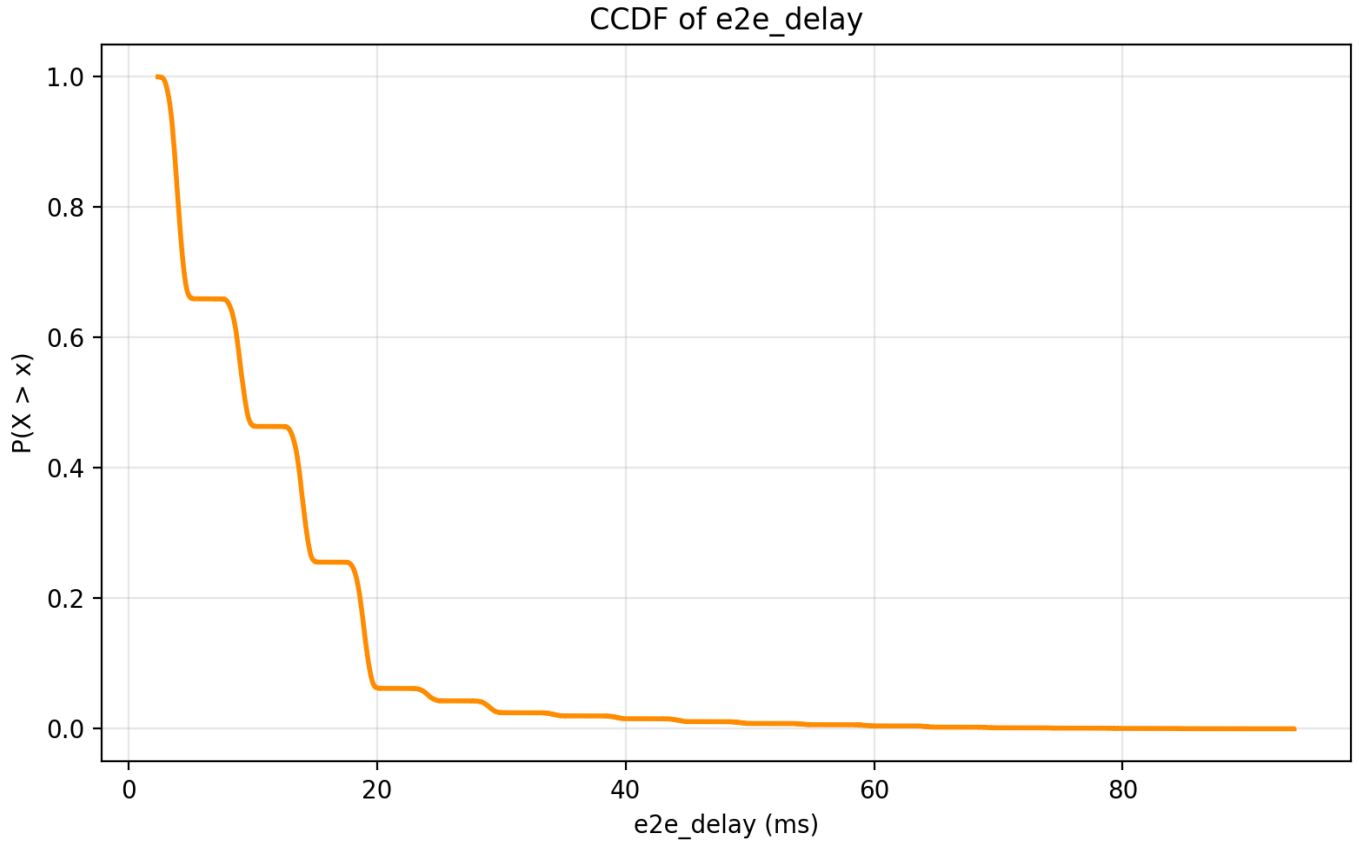




SIEMENS

Deriving DVP from end-to-end delay

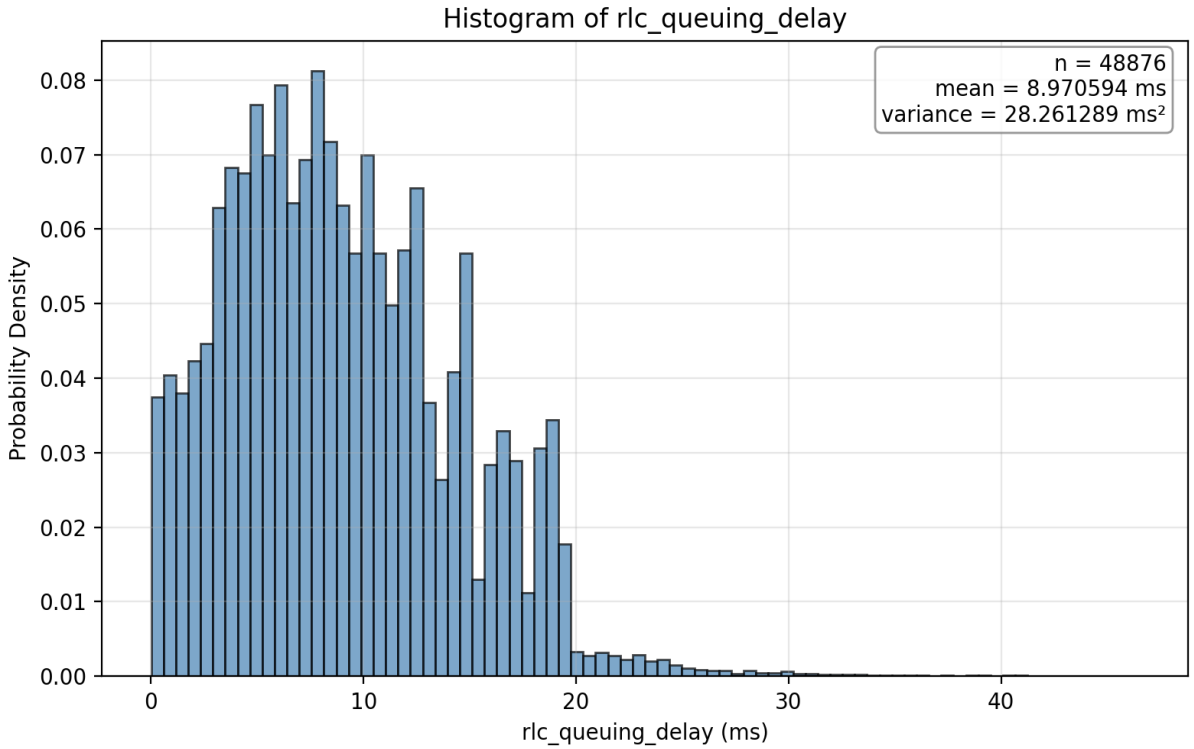
- Delay Violation Probability (DVP) is defined as the probability that packet delay exceeds a target threshold.
- Therefore, the CCDF of end-to-end delay directly provides the empirical DVP curve.
- Since the delay distribution is strongly influenced by RLC queuing, the DVP is also expected to depend heavily on queue dynamics.



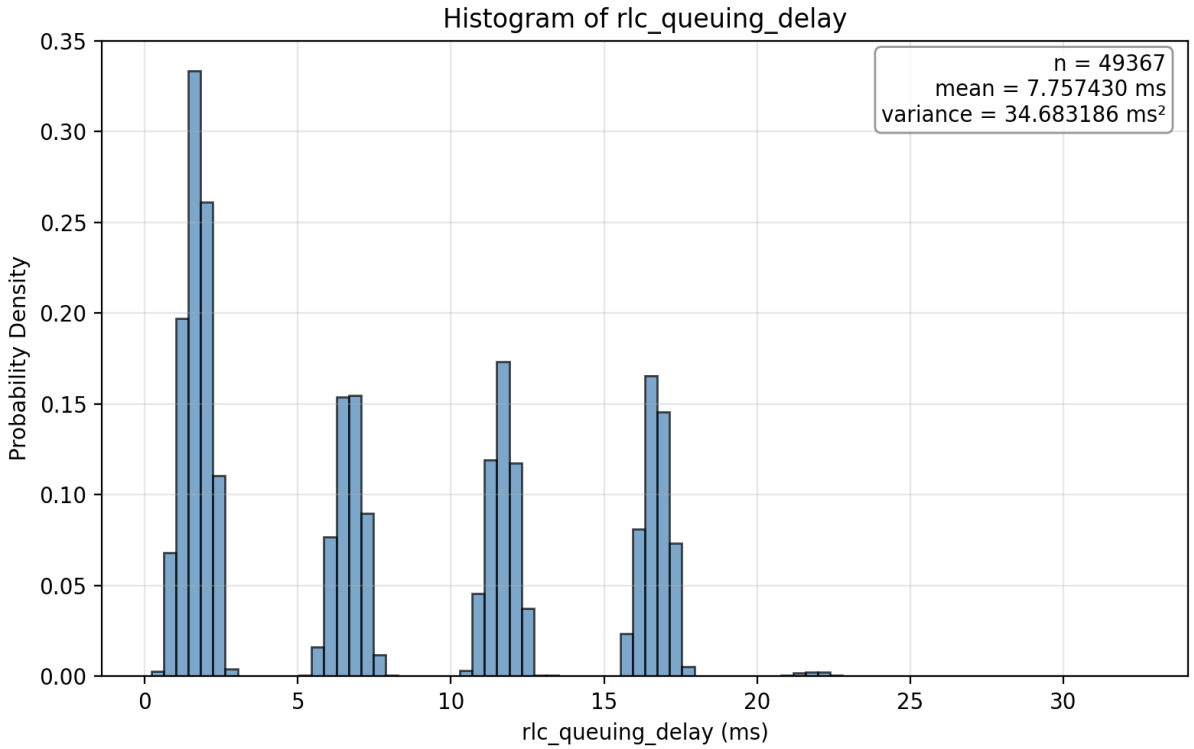
SIEMENS

Different RLC queuing delay patterns under different packet inter-arrival times

Queuing Delay at inter-arrival time 2ms



Queuing Delay at inter-arrival time 5ms



- The RLC queuing delay distribution changes significantly with packet inter-arrival time.
- At 5 ms, the histogram shows structured / multi-modal peaks, indicating phase-dependent waiting times.
- At 2 ms, the distribution becomes smoother, suggesting a more persistently backlogged queue.
- Traffic periodicity affects not only the average delay, but also the underlying queueing mechanism.

SIEMENS

Conclusion

- Delay decomposition matters — aggregate e2e delay alone hides the dominant mechanism.
- RLC queuing dominates the measured uplink delay in our setup.
- DVP is therefore tightly linked to queue dynamics rather than only transmission time.
- Traffic periodicity shapes delay behavior:
  - Sparse arrivals can create discrete delay modes.
  - Persistent arrivals can sustain backlog and long tails.

SIEMENS

Next Step: Probabilistic DVP Prediction

- Goal
  - Use LSTM and Transformer to predict the violation probability of the future delay.
  - Move from delay analysis to proactive delay prediction.
  - Evaluate how well the prediction works on our testbed.

- Key Questions
  - How much training data is needed?
  - Which model performs better?
  - Which features (e.g., MCS, retransmission) contribute the most?
  - Can the model generalize to unseen scenarios?

Inspired by prior work: Mostafavi et al., Probabilistic Delay Forecasting in 5G Using Recurrent and Attention-Based Architectures.

SIEMENS

Contact

Hao-Yun Hsu
hao-yun.hsu.ext@siemens.com

Neda Petreska
neda.petreska@siemens.com

SIEMENS