LTB 2021
Load Testing & Benchmarking

PIERES: A Playground for Network Interrupt Experiments on Real-Time Embedded Systems in the IoT

19th April 2021


Franz Bender


Jonas Brune


Nick Keutel

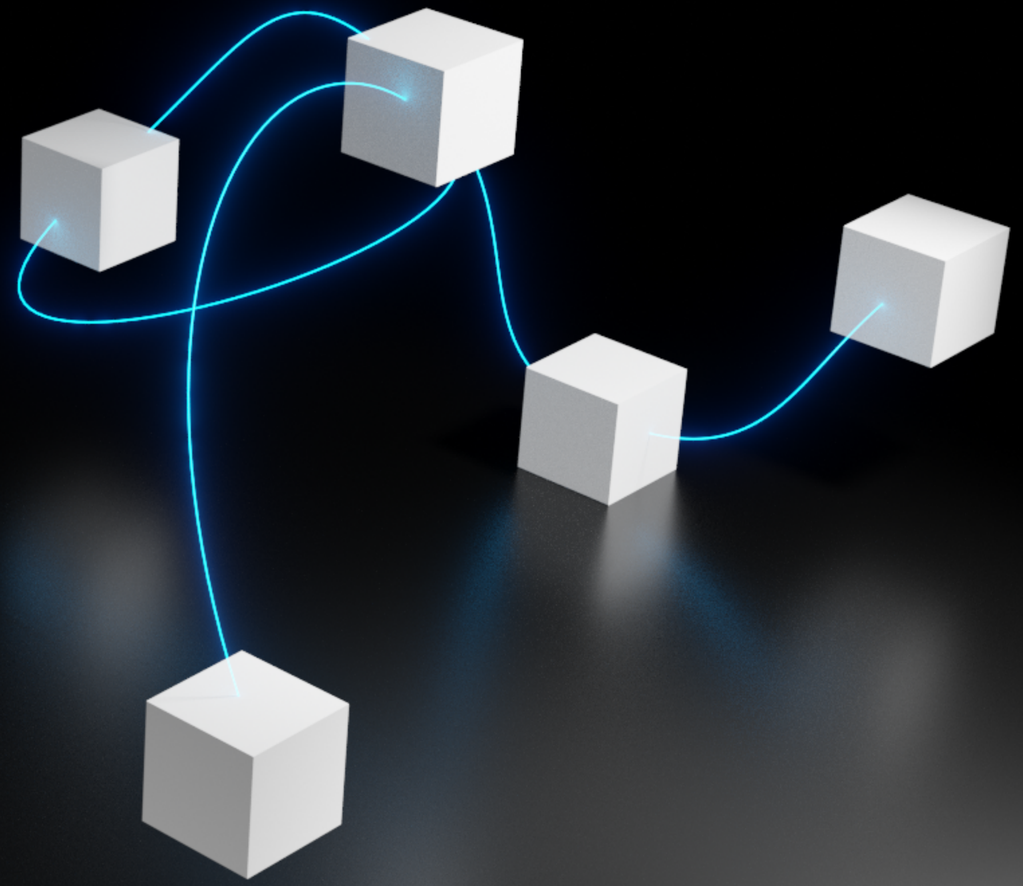

Ilja Behnke


Lauritz Thamsen

TU berlin

Outline

- Motivation
- Approach
- Experiments
- Conclusion



Motivation

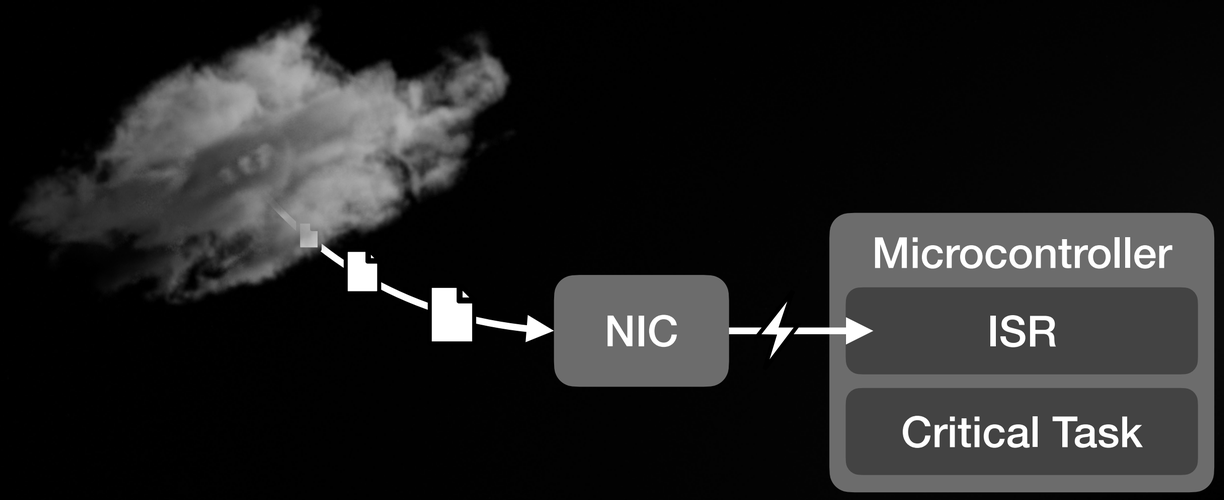
Problem Setting
Real-Time Systems in the IOT

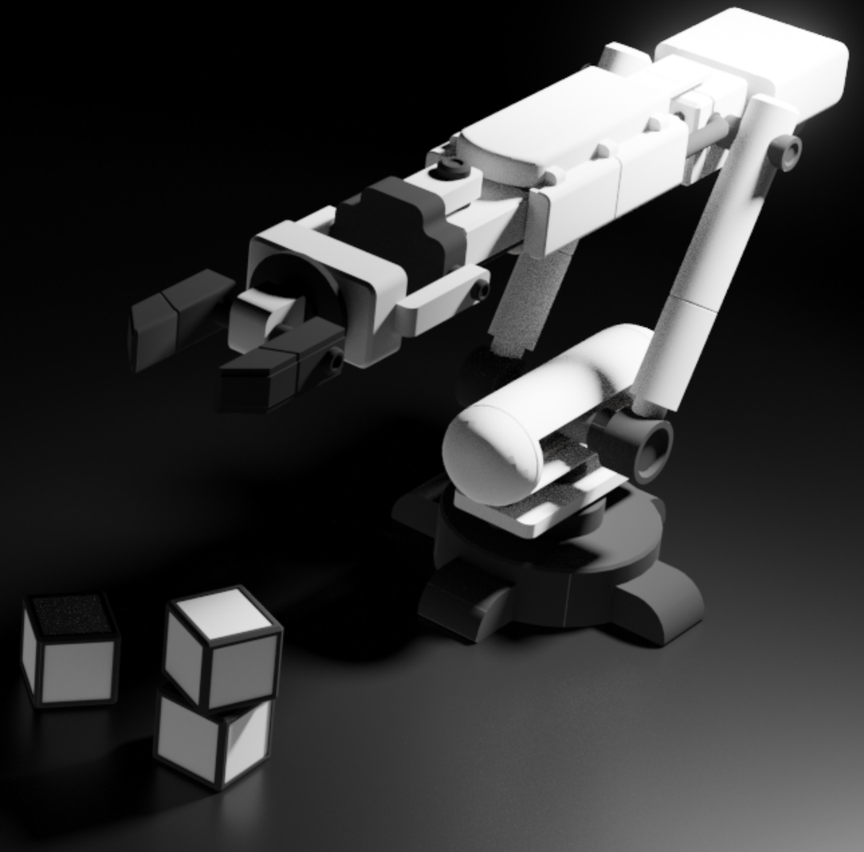
Current Situation

More and more real-time system get connected to the IOT

What happens when a lot of network packets arrive?

Handling Network Packets





Model: Basic Robot Arm by Blender3D

Problem Setting
Real-Time Systems in the IOT

Current Situation

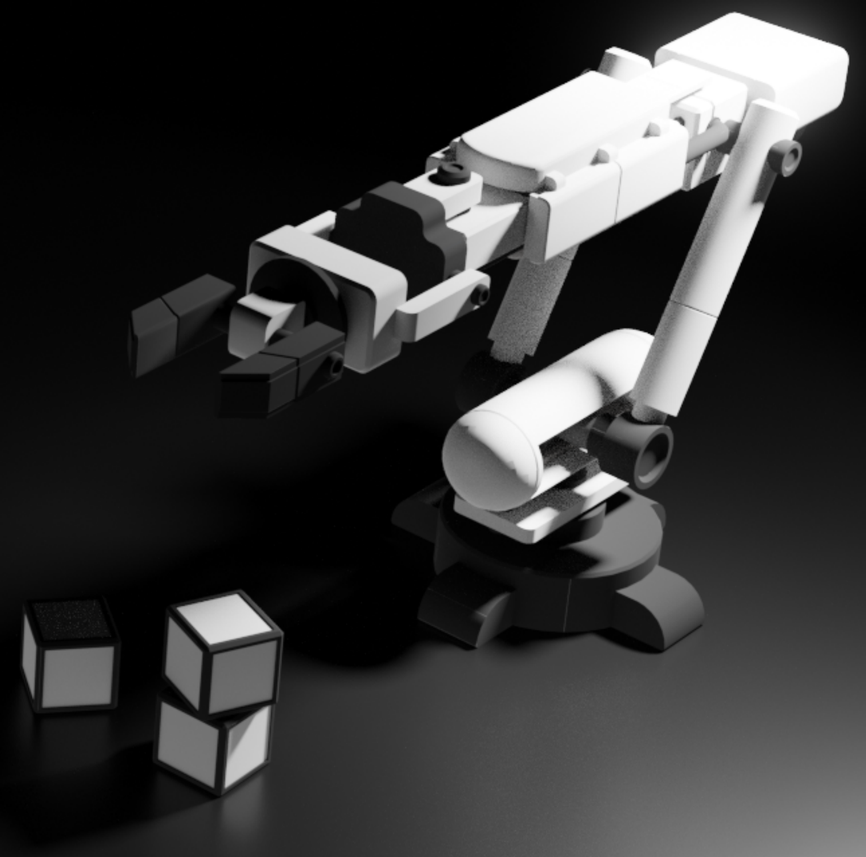
More and more real-time system get connected to the IOT

What happens when a lot of network packets arrive?

Resulting Questions

➜ Do network interrupts pose a threat to the real-timeness?

➜ How can we conduct research on this?



Model: Basic Robot Arm by Blender3D

Problem Setting
Real-Time Systems in the IOT

Current Situation

More and more real-time system get connected to the IOT

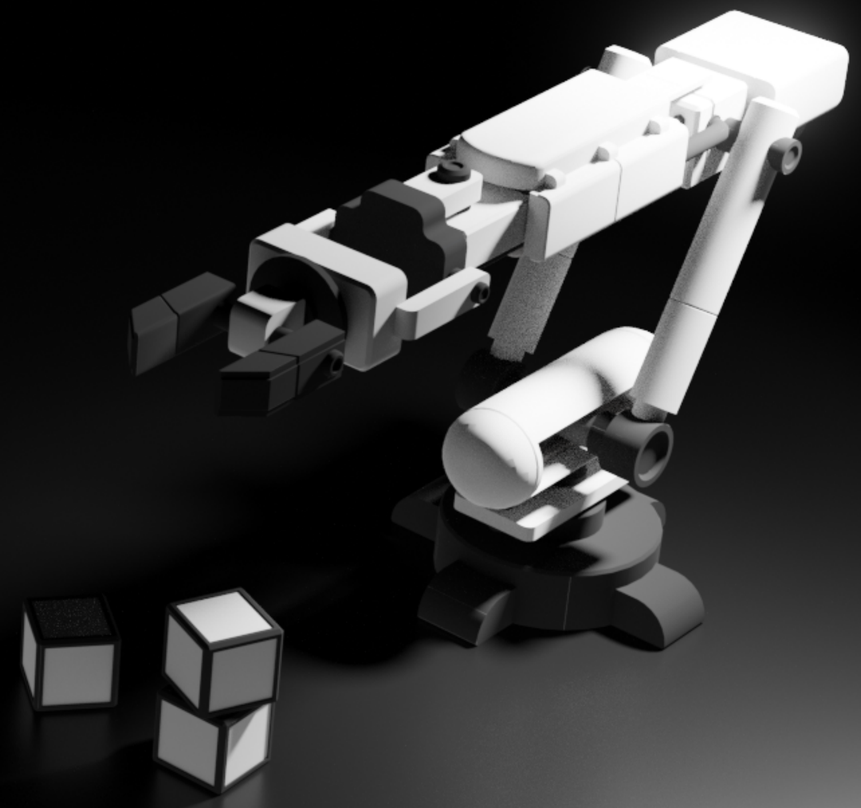
What happens when a lot of network packets arrive?

Resulting Questions

➜ Do network interrupts pose a threat to the real-timeness?

➜ How can we conduct research on this?

First Question: It depends, see [1]

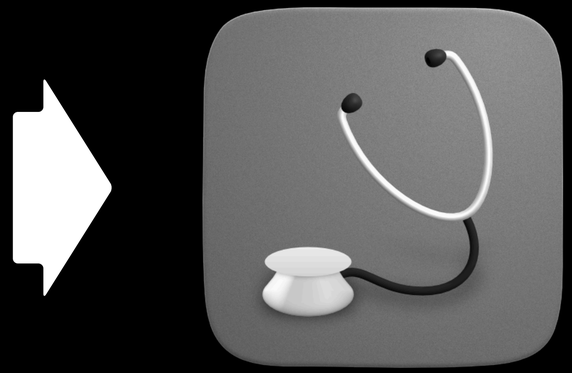


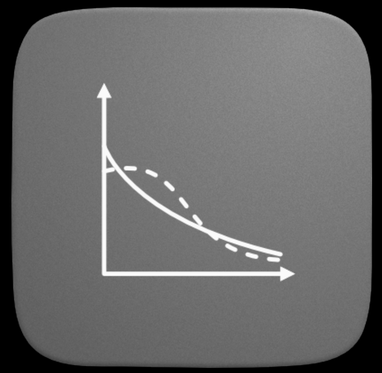
[1] Behnke, Ilja & Pirl, Lukas & Thamsen, Lauritz & Danicki, Robert & Polze, Andreas & Kao, Odej. (2021). Interrupting Real-Time IoT Tasks: How Bad Can It Be to Connect Your Critical Embedded System to the Internet?

Model: Basic Robot Arm by Blender3D

Our Goal

A playground for testing the effects of network interrupts on some given code





Source code (to be tested)

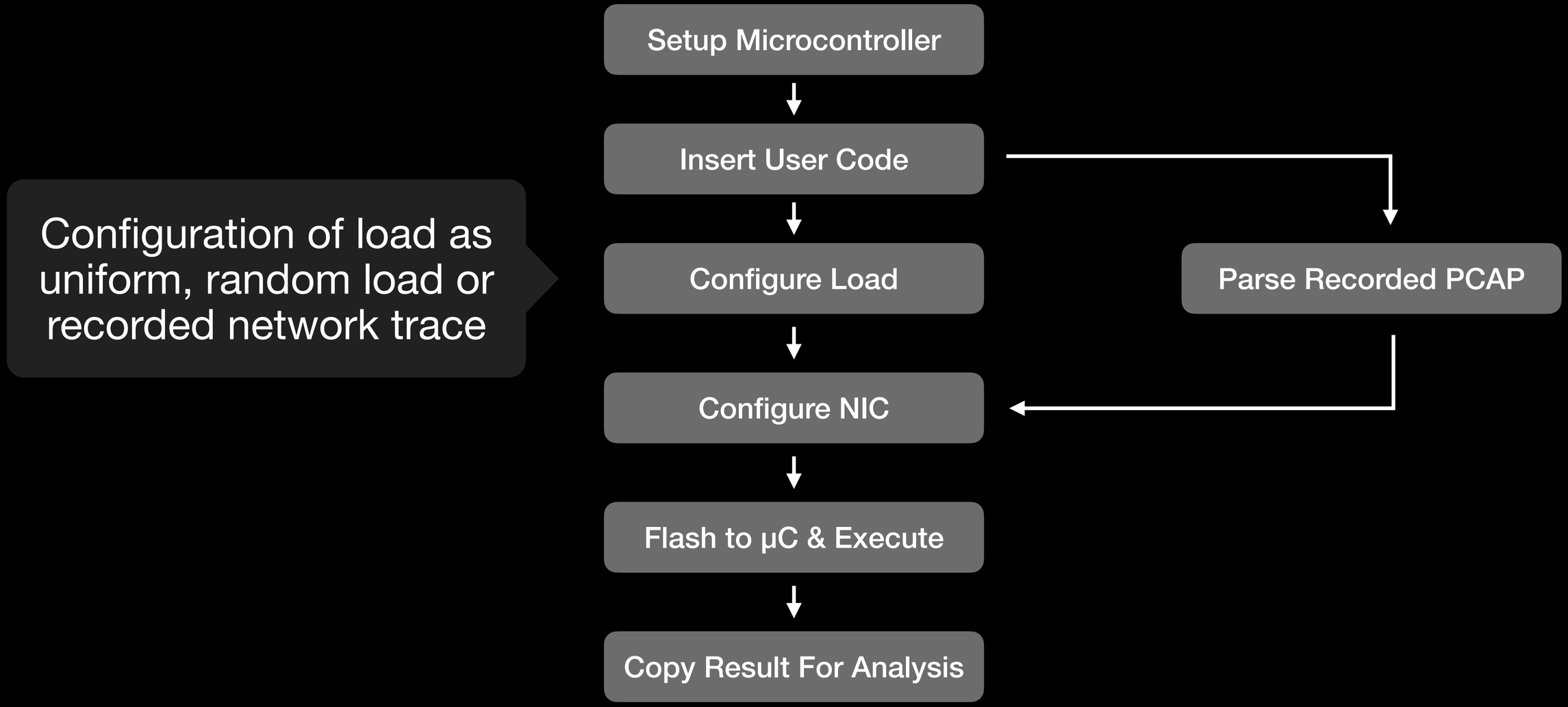
Analysis

What's the impact of interrupts on runtime?

Approach

Operation of the Playground

General procedure of a user operating the playground



Network Interface Controller (NIC) Implementations

Simple NIC model

$$d(l) = d_l \cdot l + d_c$$

$d(l)$: total duration per packet

l : packet length

d_l : length dependent delay

d_c : length independent delay

NIC Implementations

NIC implementations with interrupt moderation

Interrupt moderation allows for the CPU to decide whether it wants to be informed about new packets immediately or later

The playground supports NICs to be defined with

- a counter mode
- a timer mode
- or a combination out of the box (mixed mode)

NIC Implementations

NIC implementations with counter mode

A NIC with the counter mode does not trigger an interrupt for every received packet, but triggers one after a specified number of packets

The packets are stored in a buffer and after a specified count a single interrupt is invoked for them all

Counter and buffer are reset after an interrupt has been triggered

NIC Implementations

NIC implementations with timer mode

a packet arrives

It gets written to memory
No interrupt is invoked
A packet delay timer is set

a packet arrives

It gets written to memory
Still no interrupt is invoked
The timer gets reset

interrupt gets fired

The timer has expired because no other packet was received since the last timer reset so the interrupt gets fired

Combining timer and counter mode ensures that an interrupt gets triggered eventually

Network Traffic Scenarios as Load

Multiple implemented load generators offer flexibility

Definition of load scenario: the arrival of packets with corresponding time stamps over some observed time

The playground offers:

uniform loads with constant receive frequency

random loads

user defined or recorded loads

Network Traffic Scenarios as Load

Random loads

Poisson distribution is used to model random loads

Assuming the number of incoming packets per interval p_i is Poisson distributed then the inter-arrival time d_i is exponentially distributed

$$p_i \sim P(\lambda)$$
$$d_i = t_{i+1} - t_i$$
$$d_i \sim E(\lambda)$$

Inverse transform sampling enables the sampling of packet delays from uniform distribution [2]

$$\hat{d}_i = F^{-1}(u_i) = -\frac{1}{\lambda}\ln(1-u_i) \hat{=} -\frac{1}{\lambda}\ln(u_i) \qquad u_i \sim U(0,1)$$

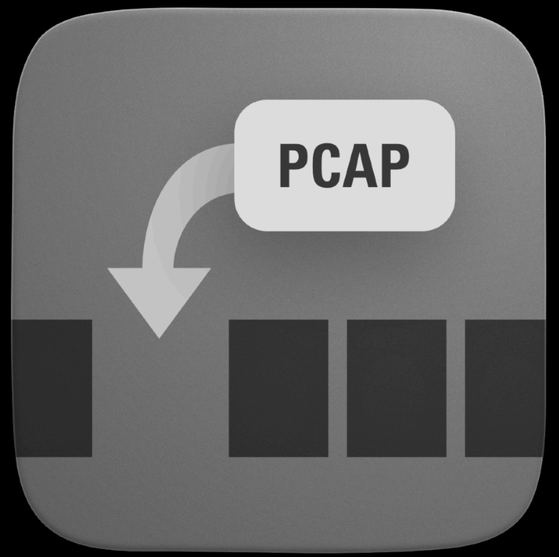
[2] L. Devroye, Non-Uniform Random Variate Generation. New York, NY: Springer New York, 1986.

Network Traffic Scenarios as Load

User defined loads

The playground allows for recorded network scenarios to be replayed

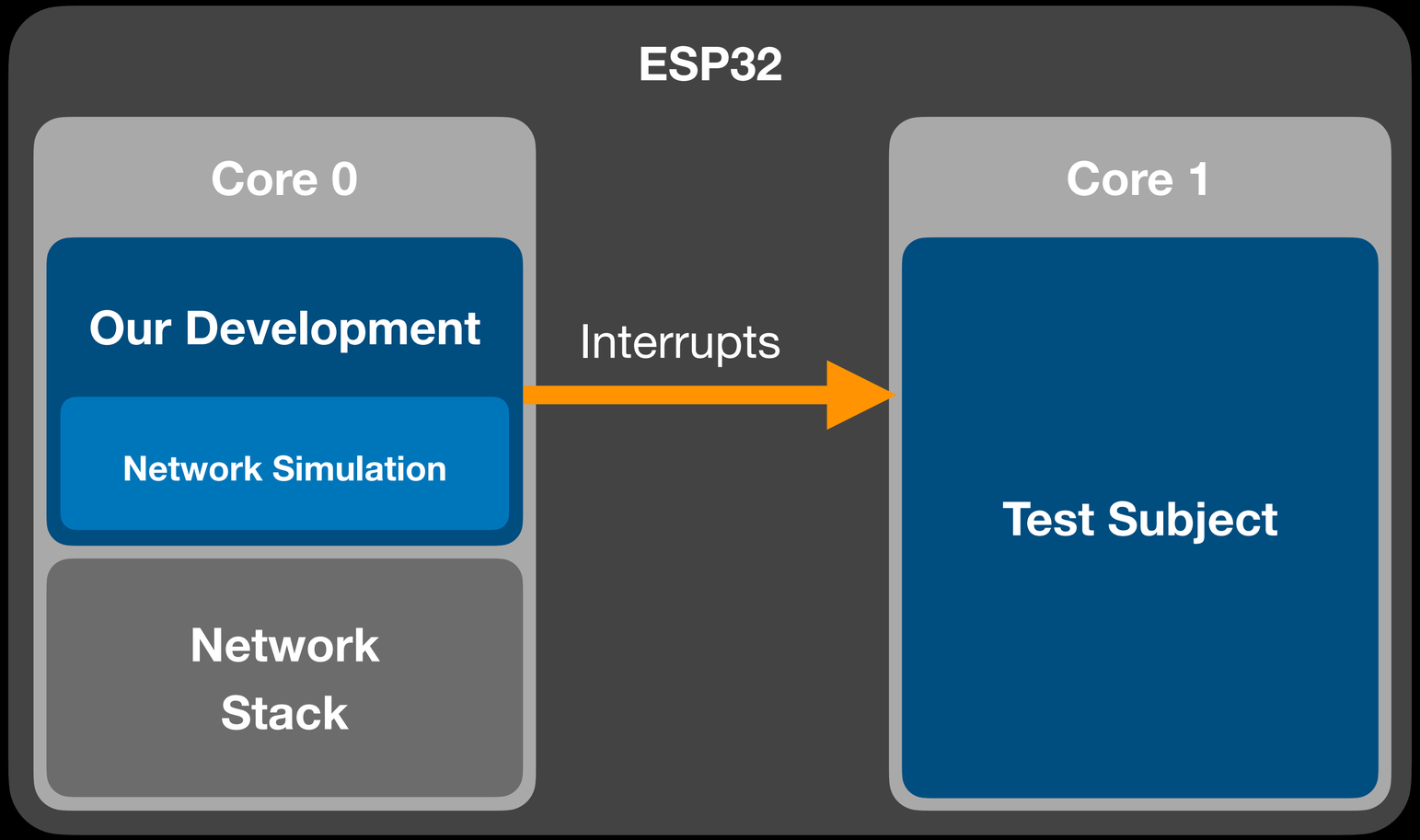
The network scenarios have to be provided as Packet CAPtures

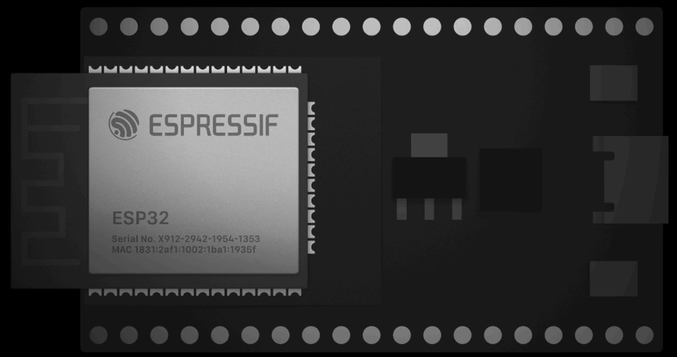


➜ Custom Reproducible Loads

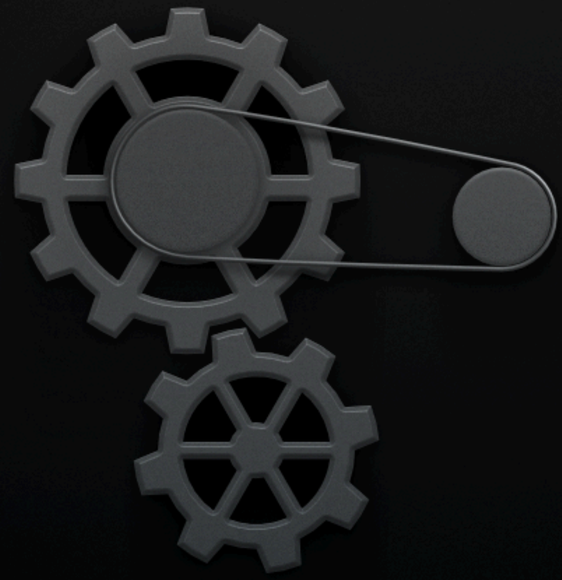
Prototype Implementation

Implementation of the playground on the ESP32



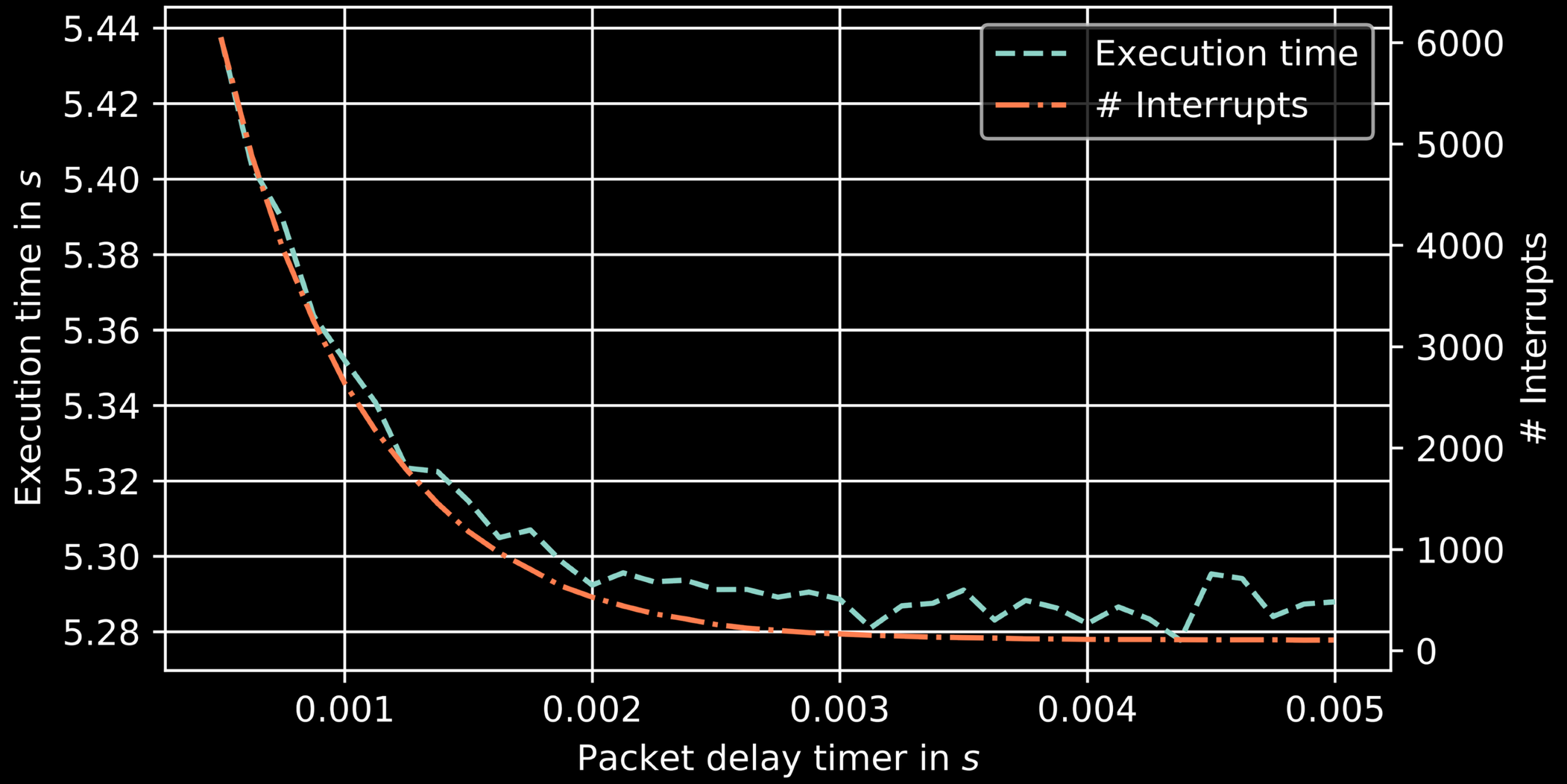


Experiments



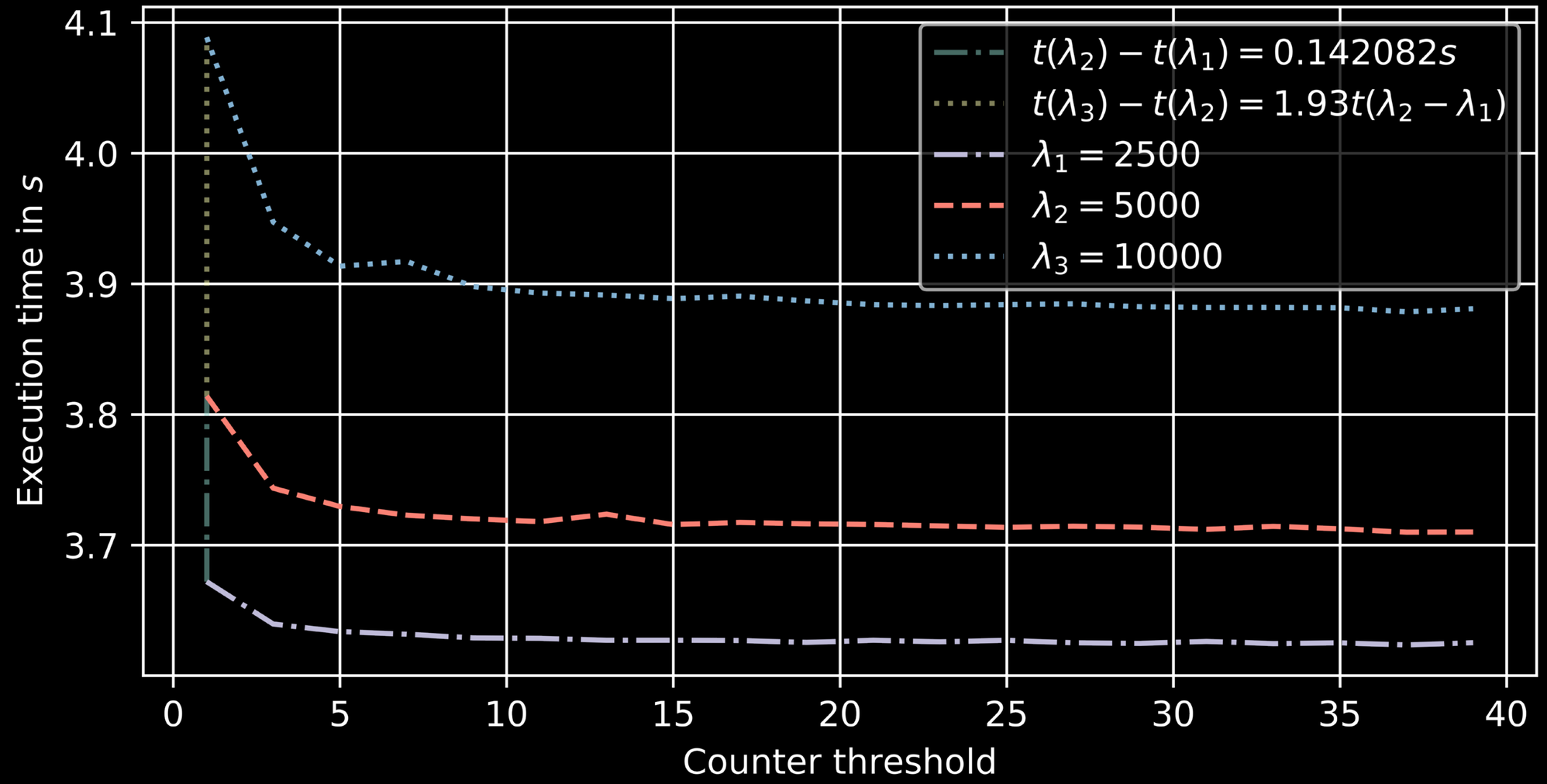
Validation Experiment I

Expectation: Longer packet delay timer results in shorter execution time and less interrupts



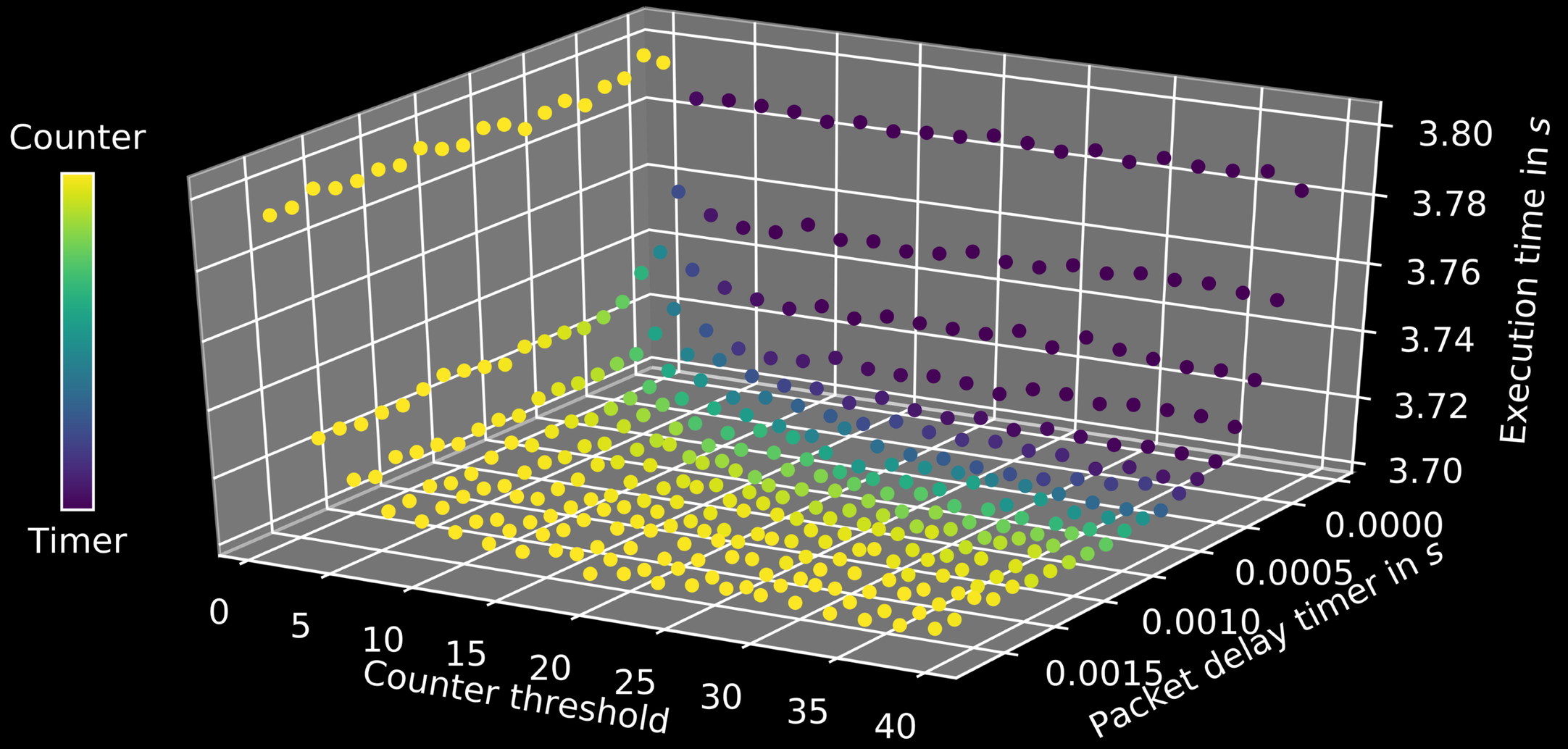
Validation Experiment II

Expectation: Counter mode scales linearly with load



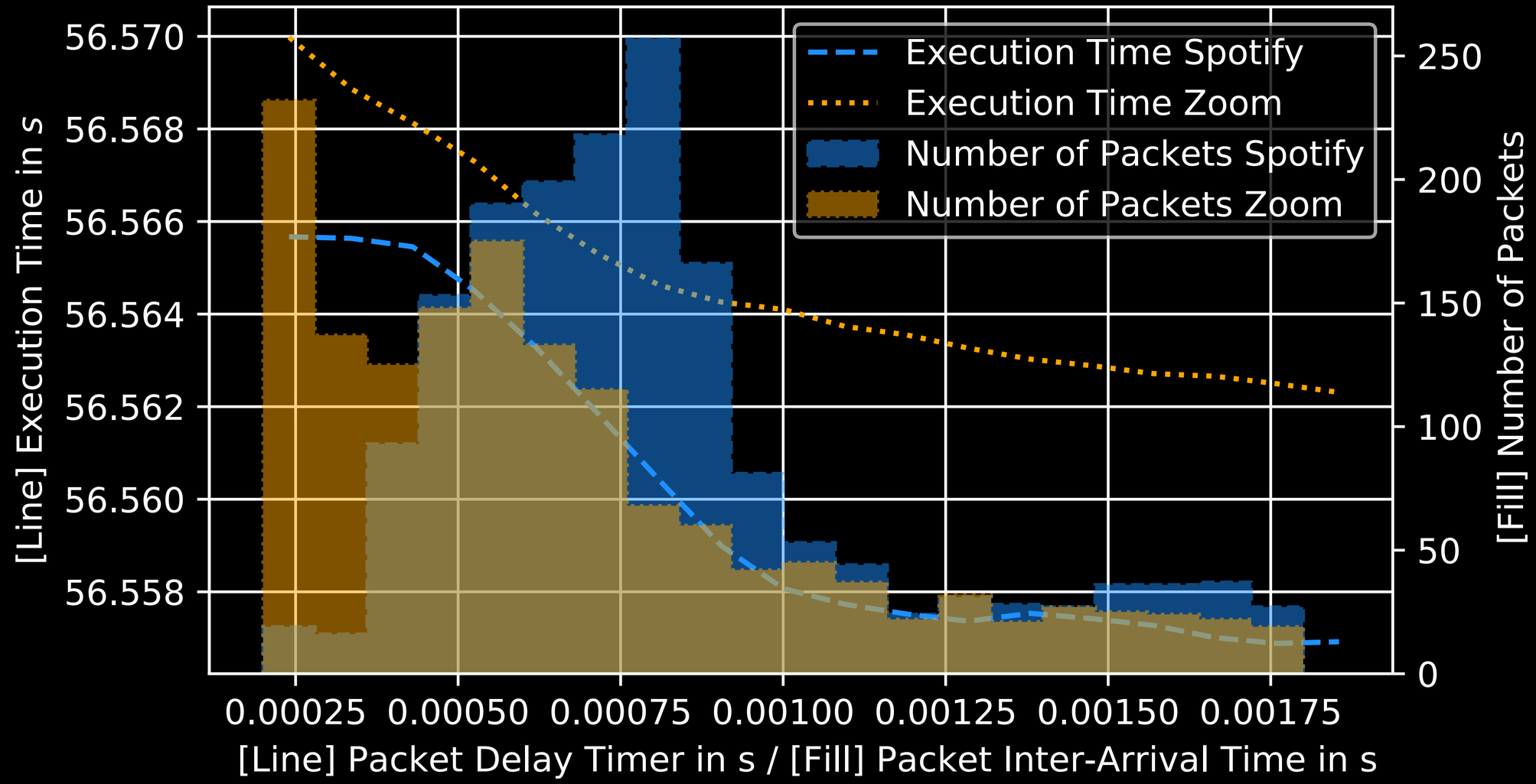
Practical Example I

What causes the interrupts?



Practical Example II

Using prerecorded PCAP: Spotify vs. Zoom



Conclusion

Our playground

- enables researchers to conduct experiments in the context of network interrupt simulation in real-time scenarios
- offers multiple load generators including random and custom prerecorded settings as well as logging
- was validated through a series of tests

Future Work

- Add more (complex) NIC models and random load sources
- Create a repository of PCAP files