CoreKube: An Efficient, Autoscaling and Resilient Mobile Core System

Andrew E. Ferguson*, Jon Larrea*, Mahesh K. Marina

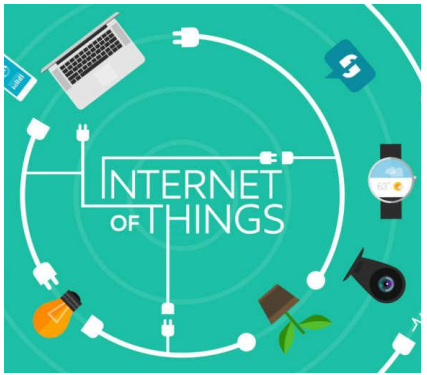
*Co-primary authors

The University of Edinburgh

Motivation

- Mobile Core: heart of any mobile network
  - Central controller and coordinator
- Control plane signalling
  - Authentication, Management, Mobility

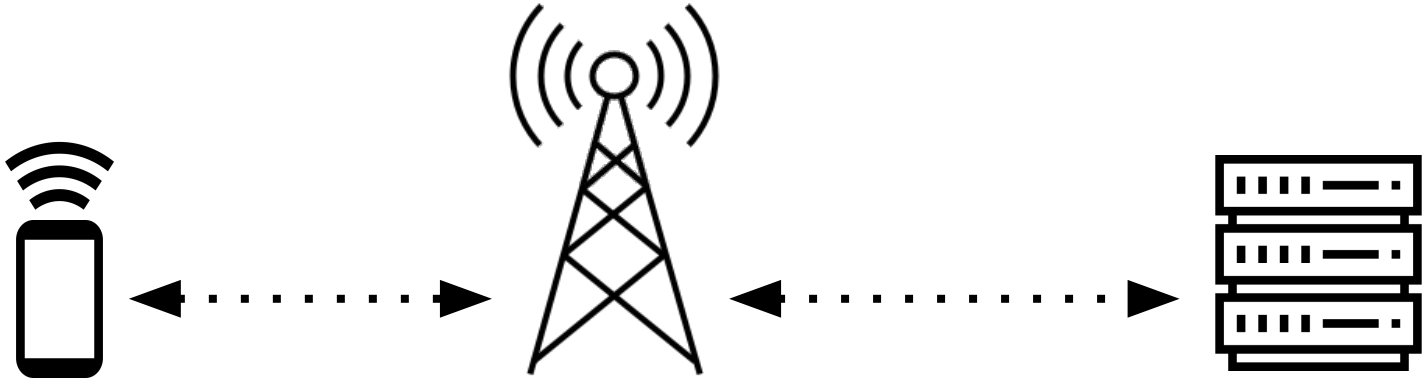
1. Growth in traffic volume
2. Bursty and unpredictable traffic



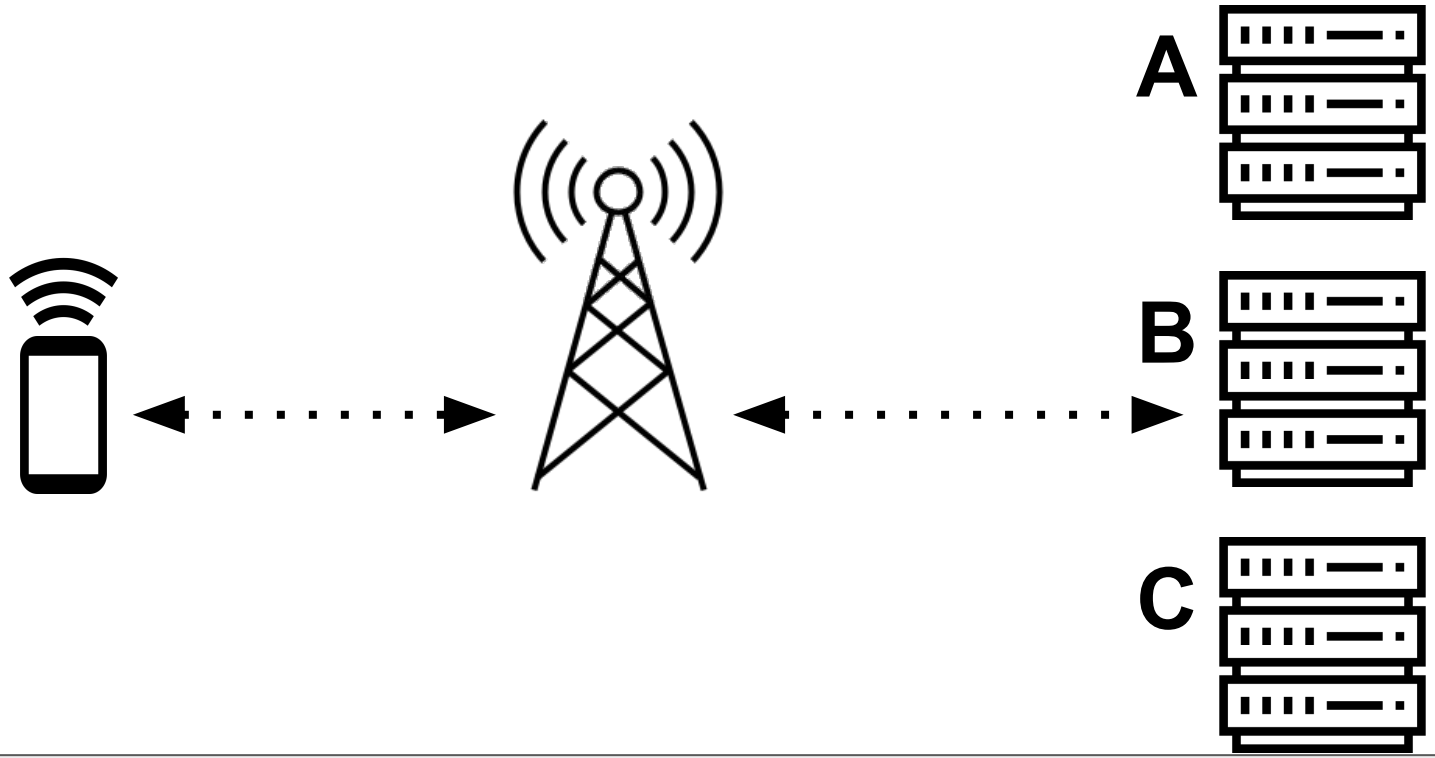
Goal: Dynamically scale the core according to demand

Whilst maintaining **efficiency** and **resiliency**

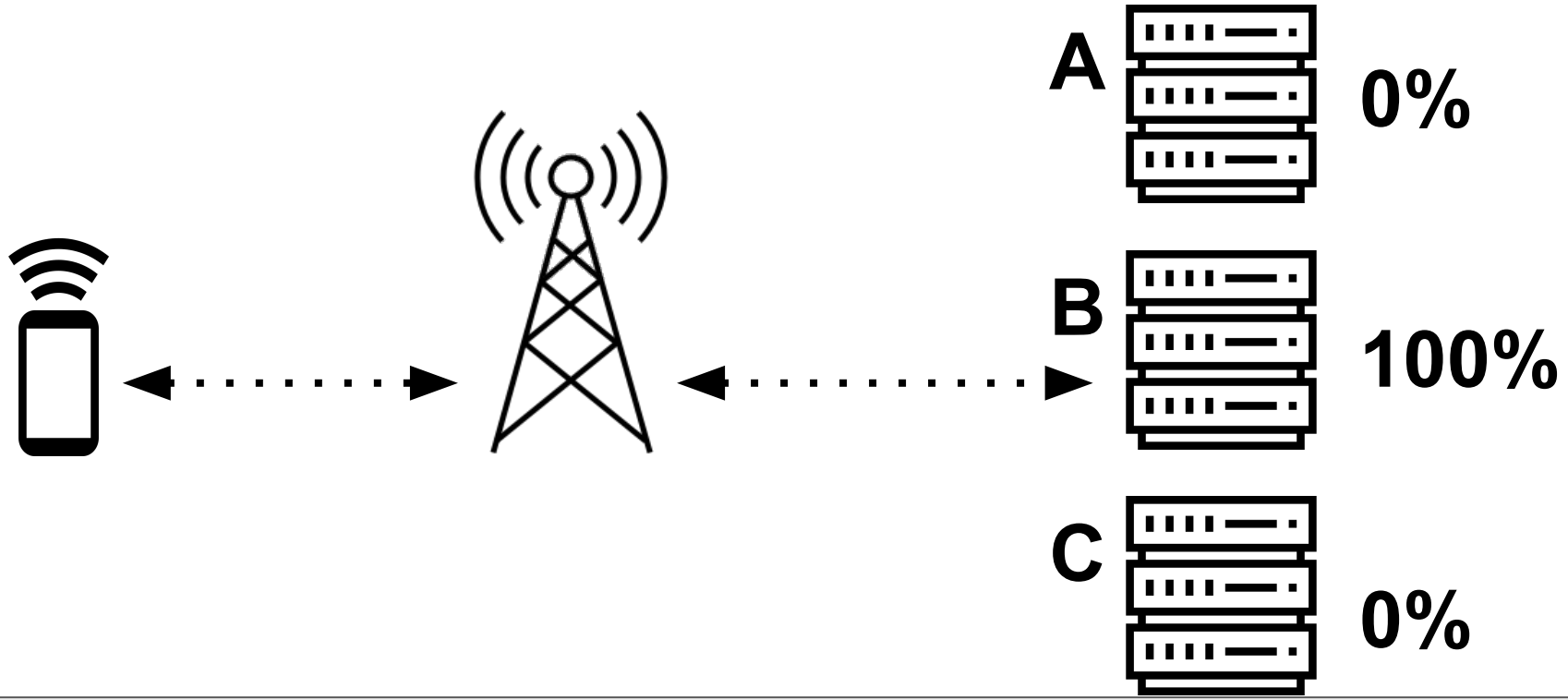
Challenge 1: Coupling between RAN and core



Challenge 1: Coupling between RAN and core

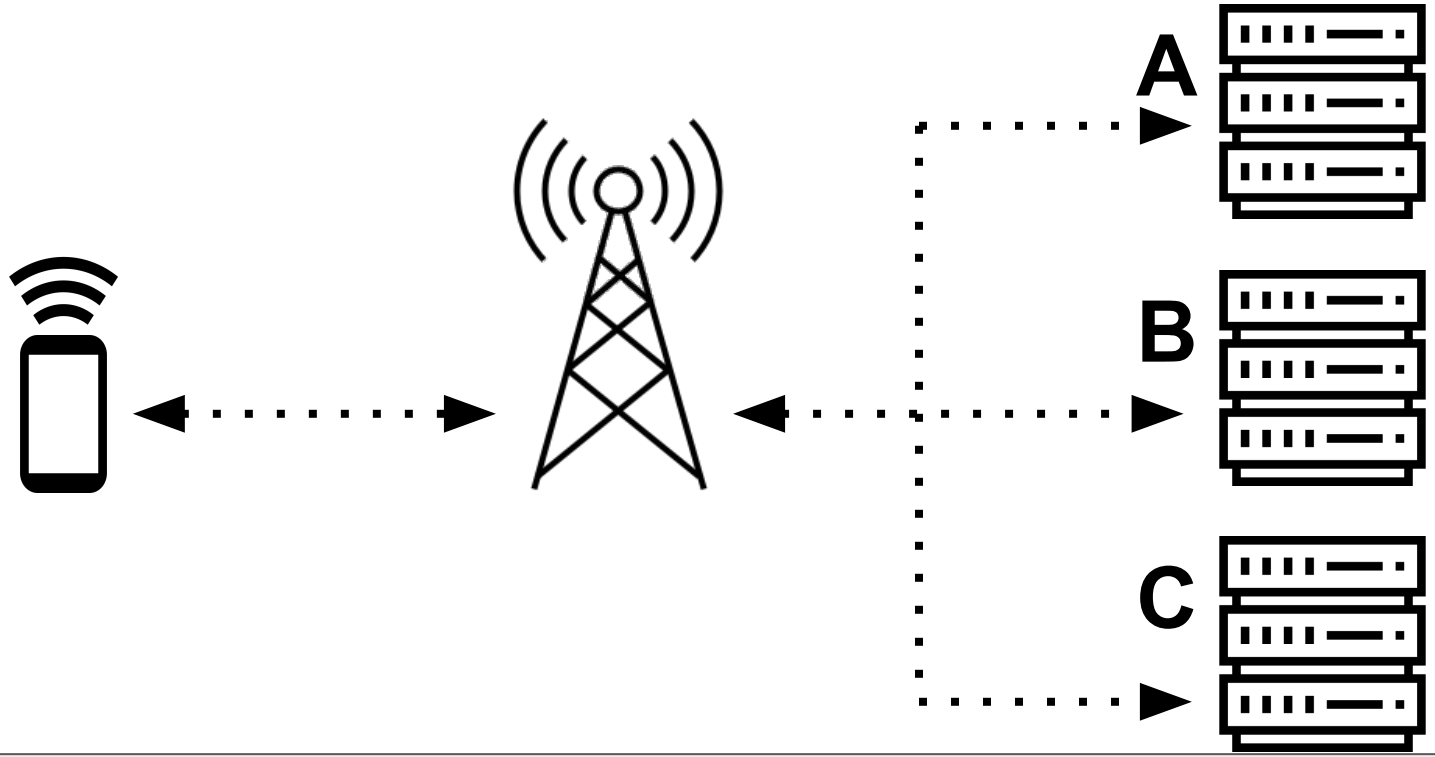


Challenge 1: Coupling between RAN and core



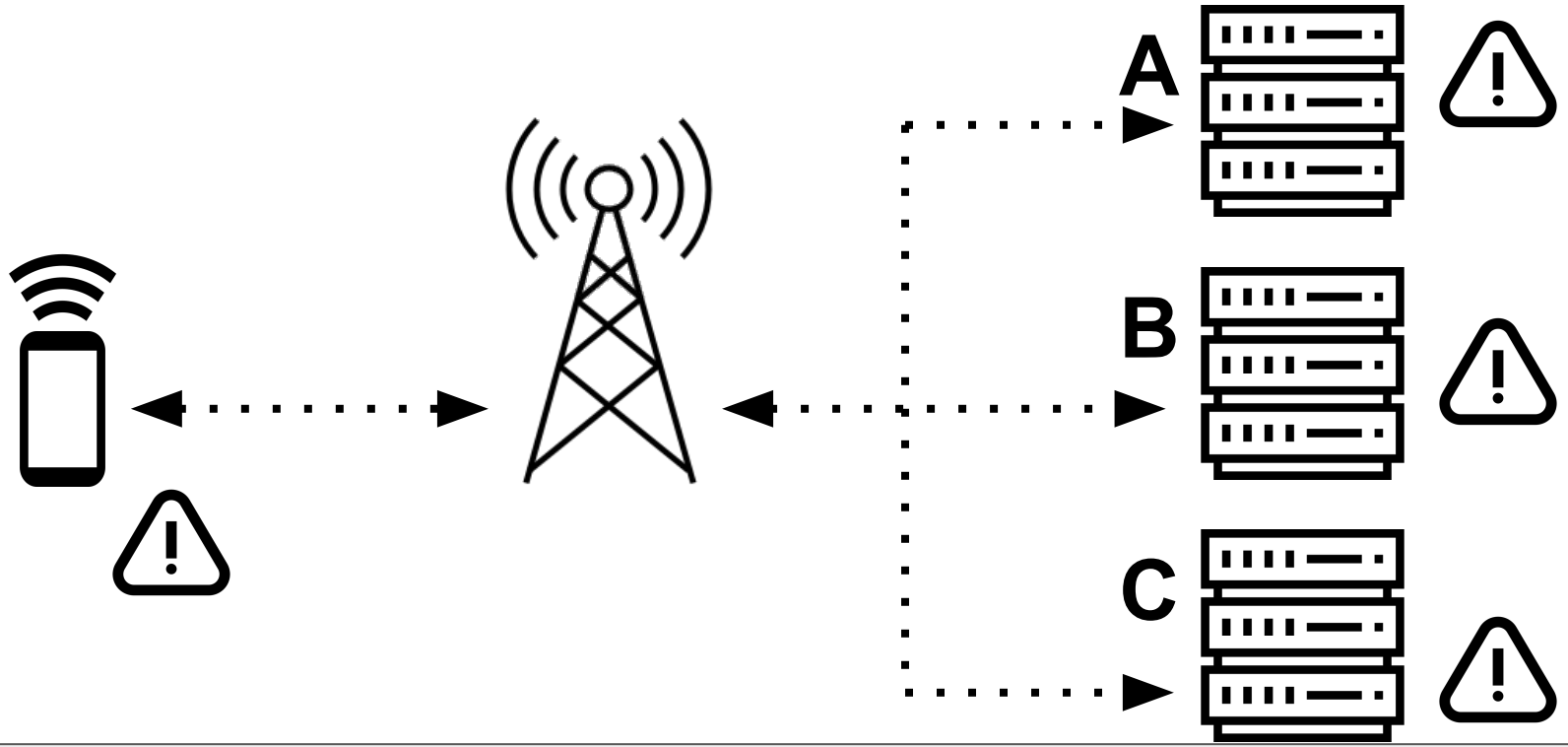
Challenge 2: Entanglement between processing and state

Many procedures contain multiple request / response pairs



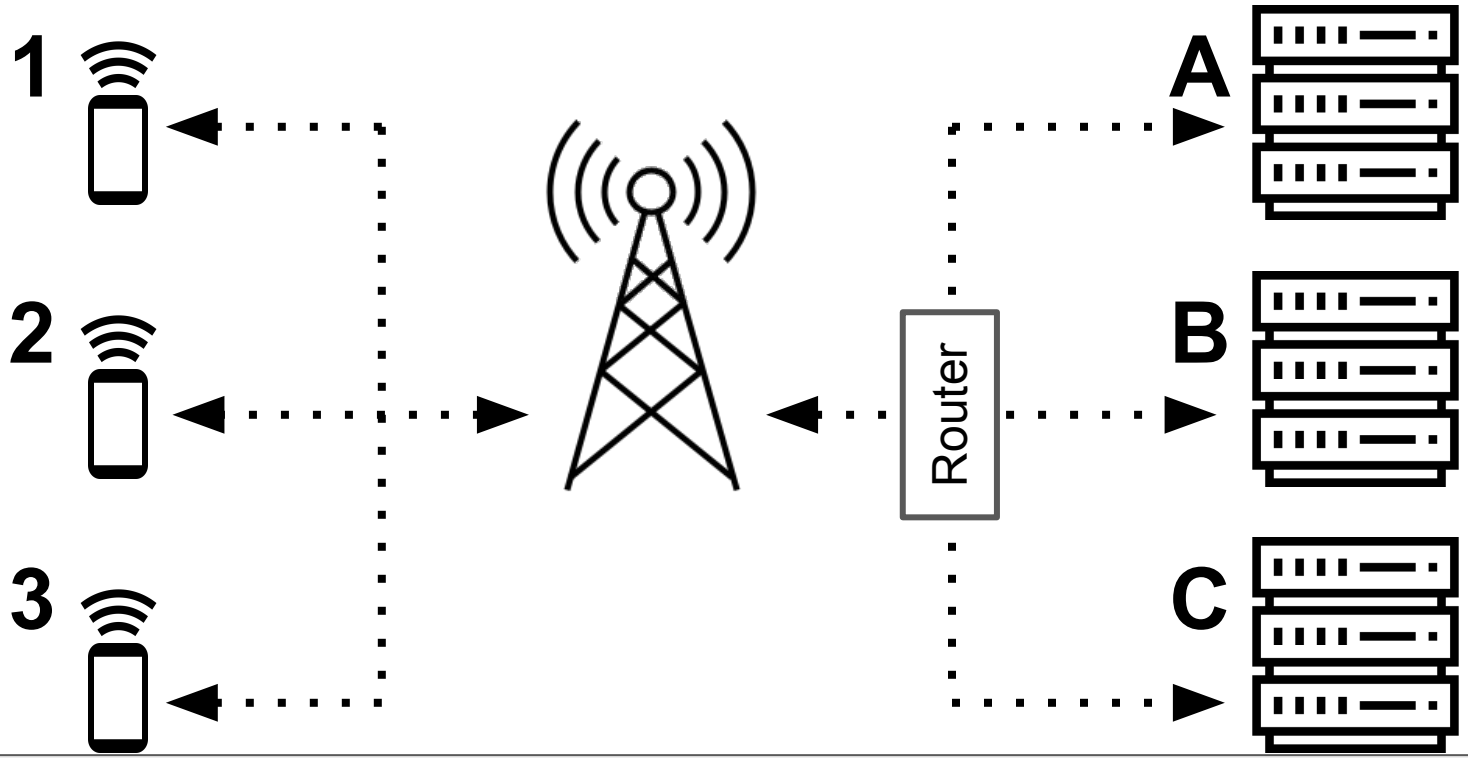
Challenge 2: Entanglement between processing and state

Many procedures contain multiple request / response pairs



Alternative Approach

Assign each device to a scaled instance

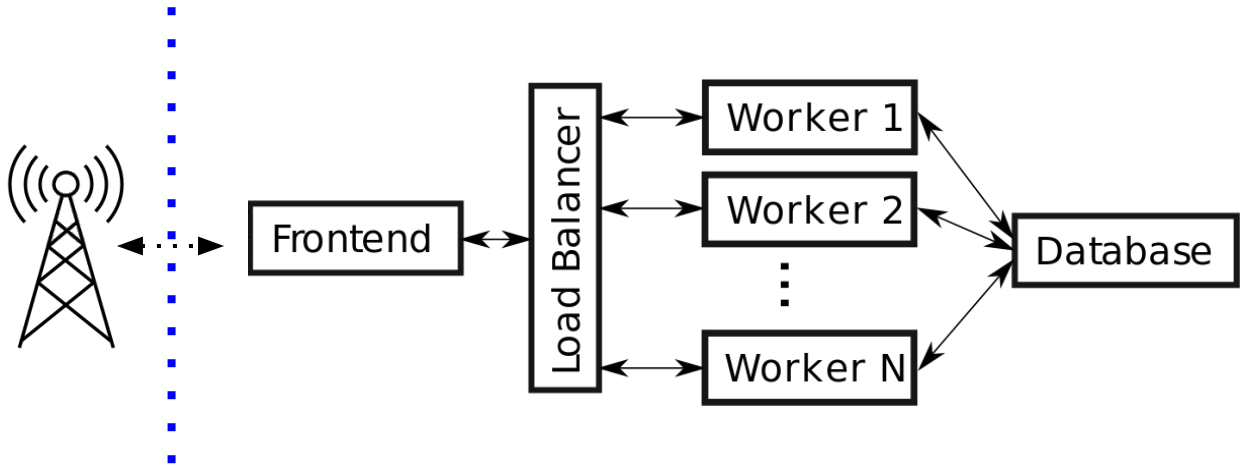


Assign each device to a scaled instance

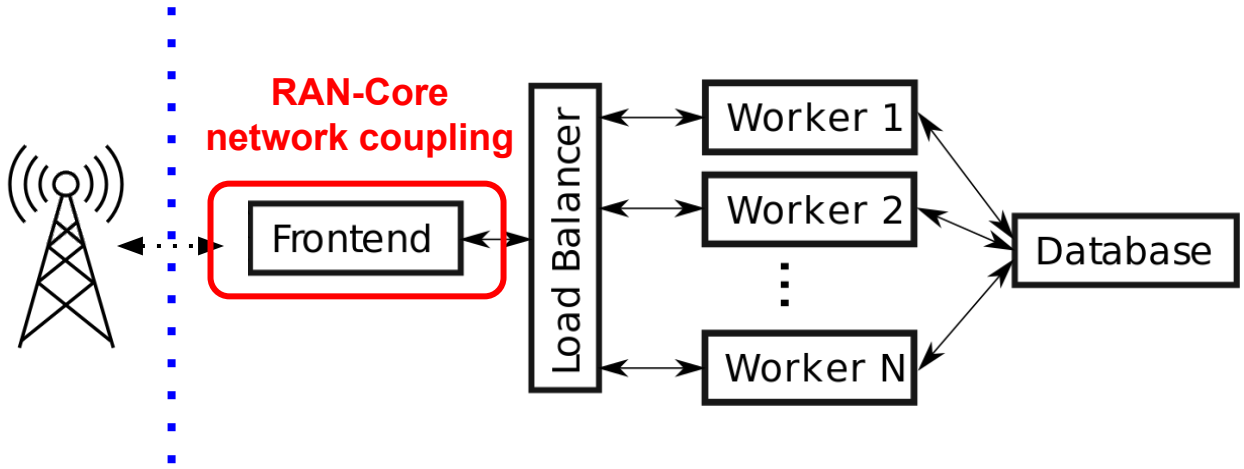
- ECHO (MobiCom'18), MobileStream (CoNEXT'18)
- Easy to scale up
- Difficult to scale down
  - Need to wait for all assigned devices to disconnect
  - Need to migrate devices between instances
- Inefficient
  - Routing messages to instances causes delays
  - Need to extract device ID from messages

CoreKube

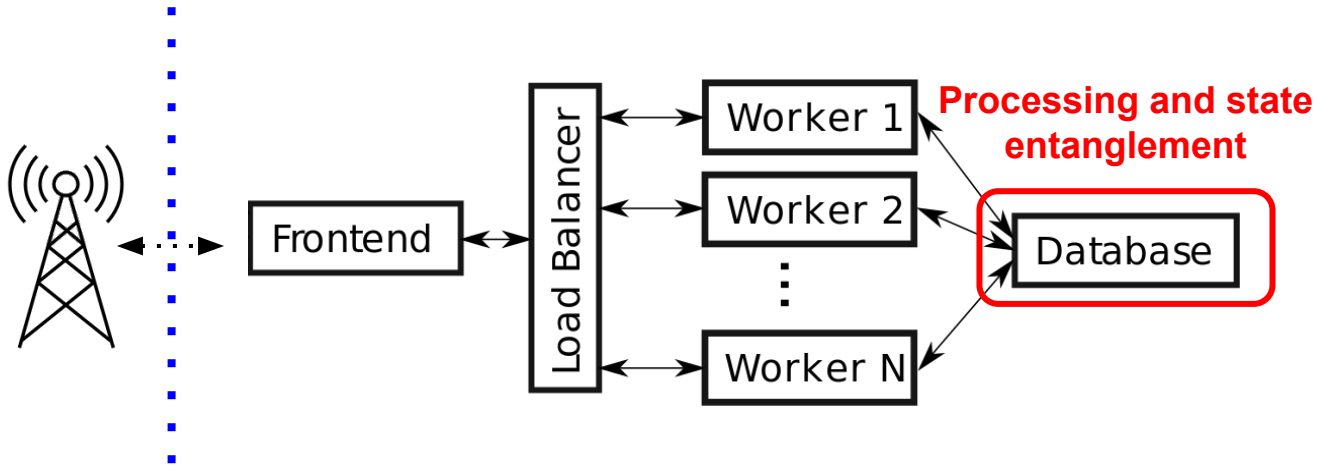
CoreKube design: Stateless architecture



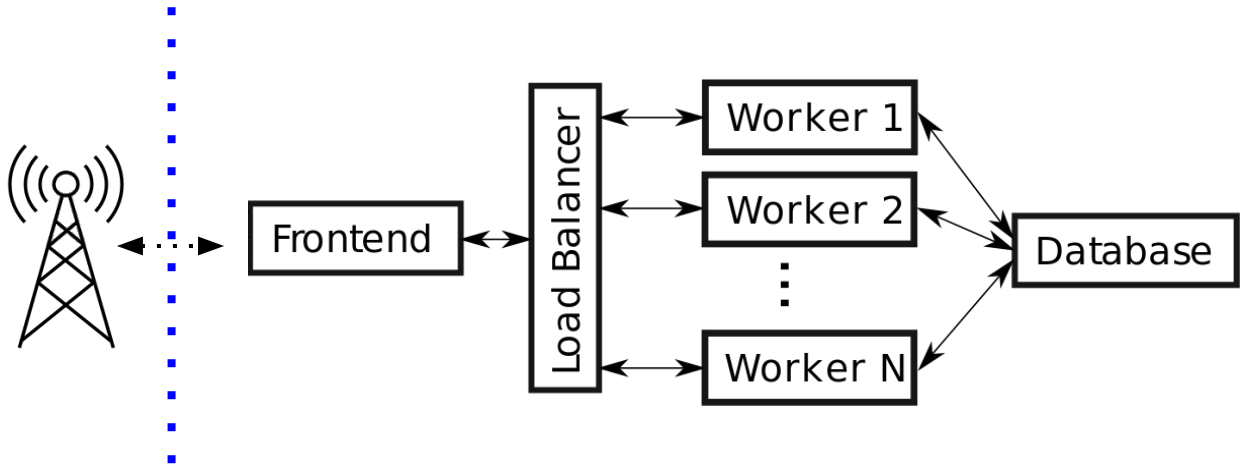
CoreKube design: Stateless architecture



CoreKube design: Stateless architecture

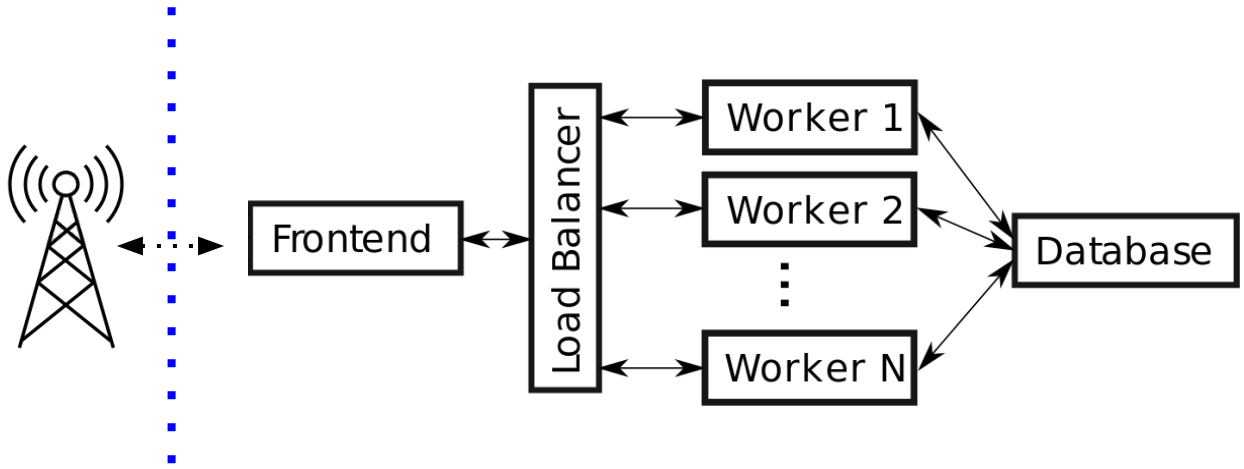


CoreKube design: Stateless architecture



1: UEs are unique

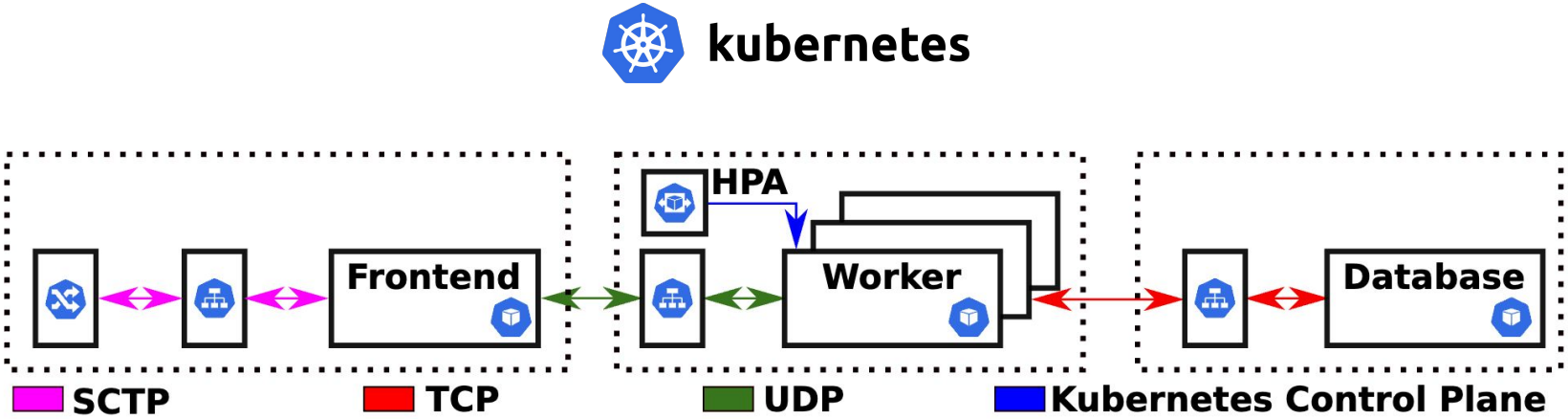
CoreKube design: Stateless architecture



1: UEs are unique

2: UE messages are sequential

Kubernetes for auto-scaling and self-healing



Implementation

- 4G and 5G standard compliant (14K LOC)
- Data plane
- Integrated with Nervion (MobiCom 2021) in Powder
- Evaluated with COTS UEs

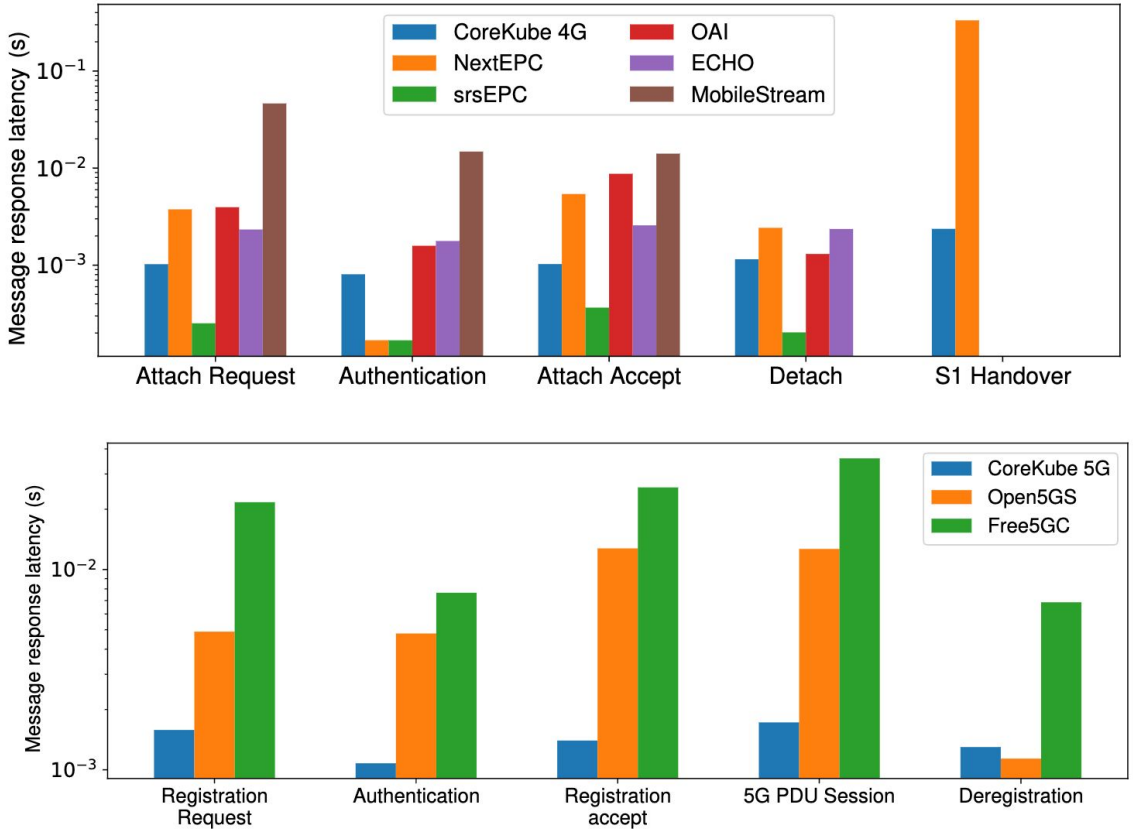
GitHub

github.com/netsys-edinburgh/CoreKube

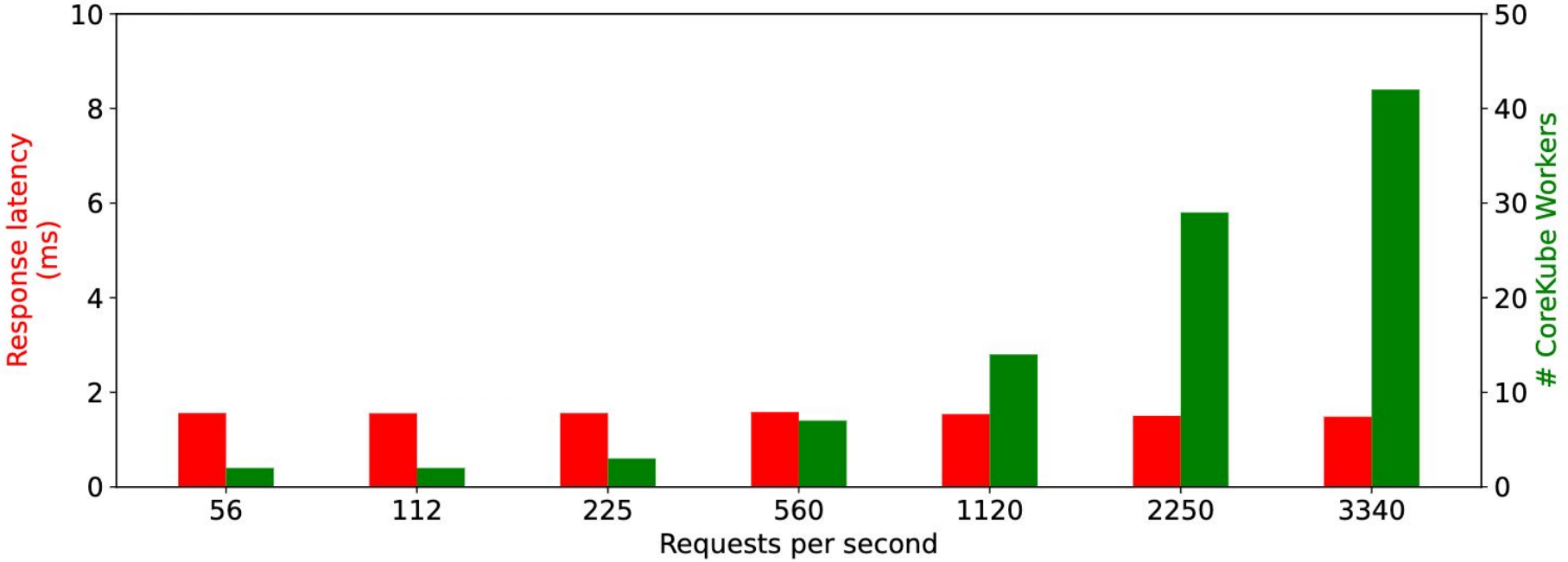
POWDER

powderwireless.net/p/Test5G/Nervion

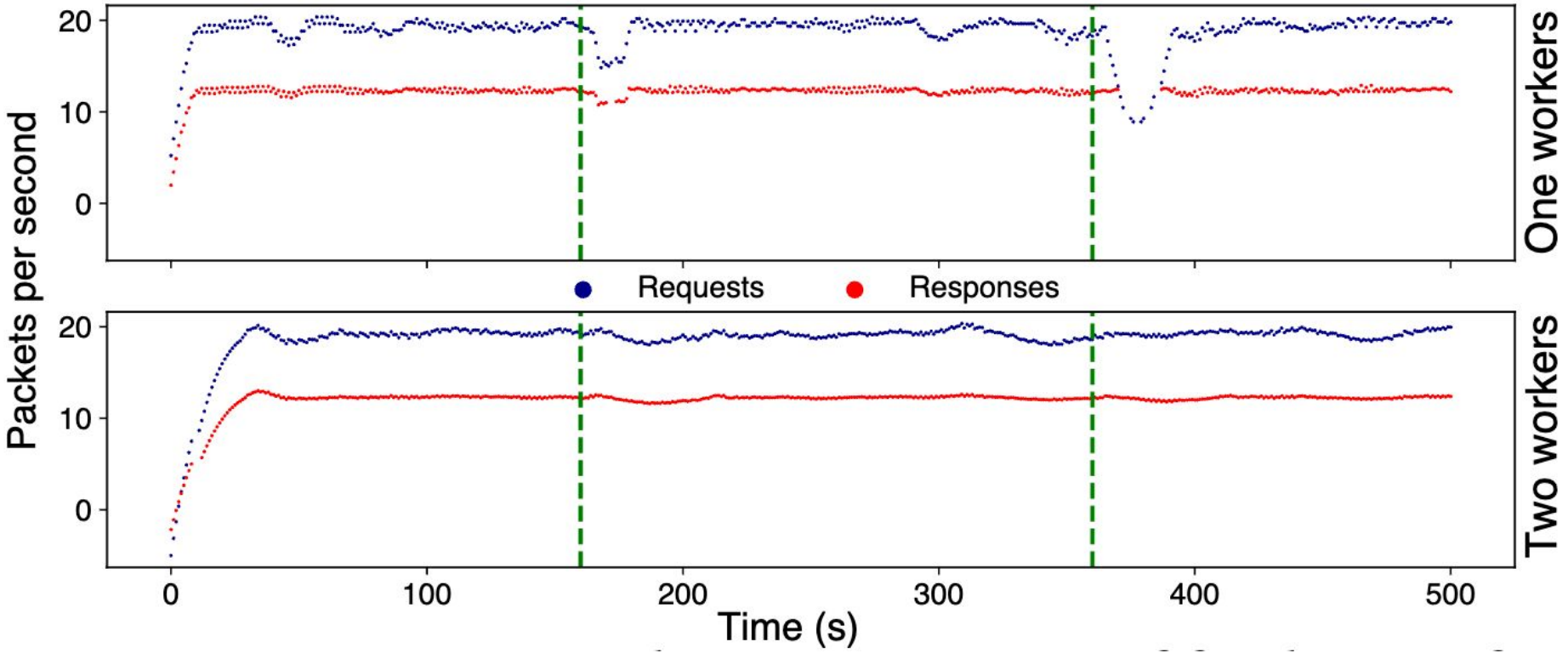
Evaluation: Efficiency



Evaluation: Scalability



Evaluation: Resilience




DEMO

Thank you!

Full paper

Documentation