SegaNet: An Advanced IoT Cloud Gateway for Performant and Priority-Oriented Message Delivery

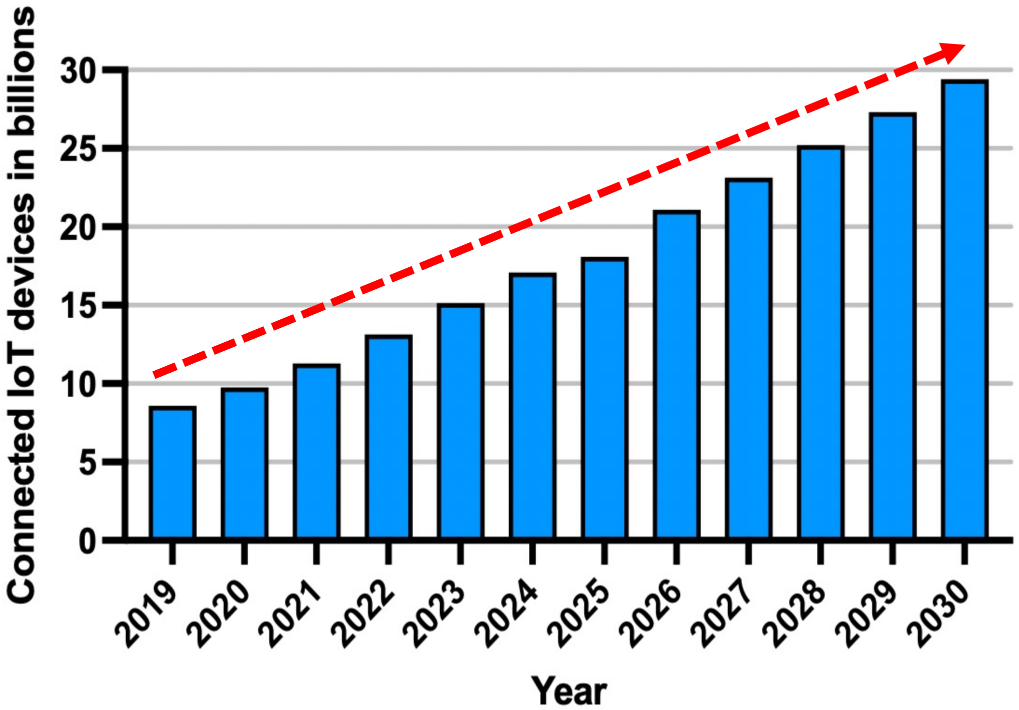
Yeonho Yoo[1,2], Zhixiong Niu[2], Chuck Yoo[1], Peng Cheng[2], Yongqiang Xiong[2]

[1]Korea University (Seoul, Republic of Korea)
[2]Microsoft Research (Beijing, China)

7th Asia-Pacific Workshop on Networking (APNET 2023)

Fast Increase of IoT Devices and Messages

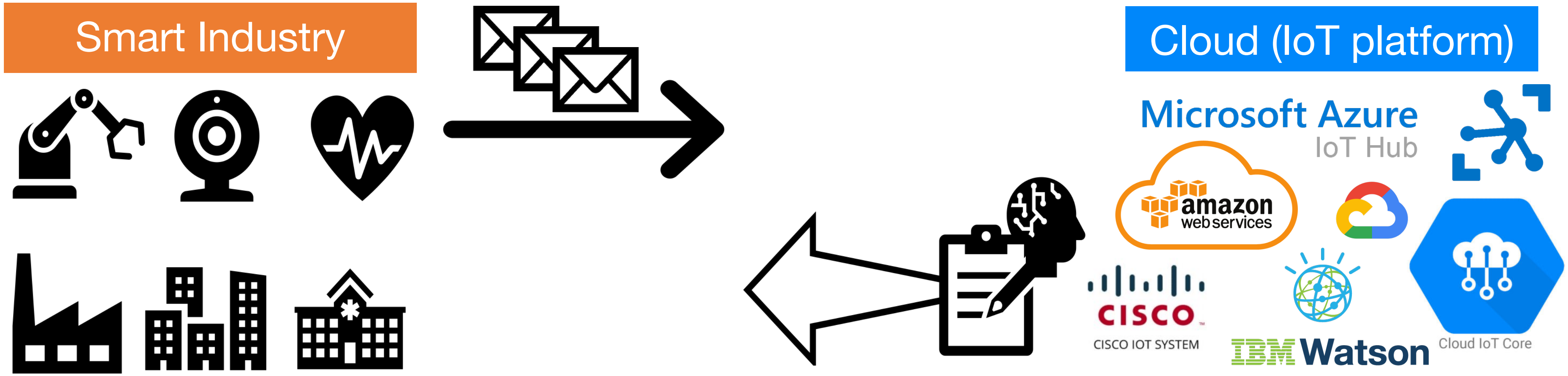
- IoT—a vital device in smart industries
- Significant expansion (post COVID-19 pandemic)[1]
  - ~29.4 billion IoT devices (2030) generate ~73.1 ZB messages/data (2025)



[1] https://www.ridge.co/blog/iot-and-the-cloud/

IoT-Cloud Connectivity

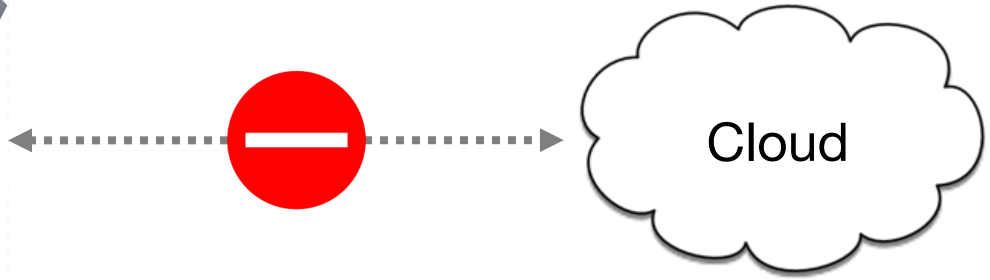
- Massive IoT messages transferred to cloud
- Cloud takes over all tasks on messages[2]
  - Scalable & reliable data management
  - Advanced analytics (w/ big data + machine learning techniques)
  - Real-time IoT device management



[2] https://learn.microsoft.com/en-us/azure/iot-hub/iot-concepts-and-iot-hub

Challenge for IoT Devices

- IoT devices typically do not support TCP/IP networking stack[3]
  - Instead, LLNs
  - e.g., 802.15.4 (Zigbee), BLE (Bluetooth), LoRa

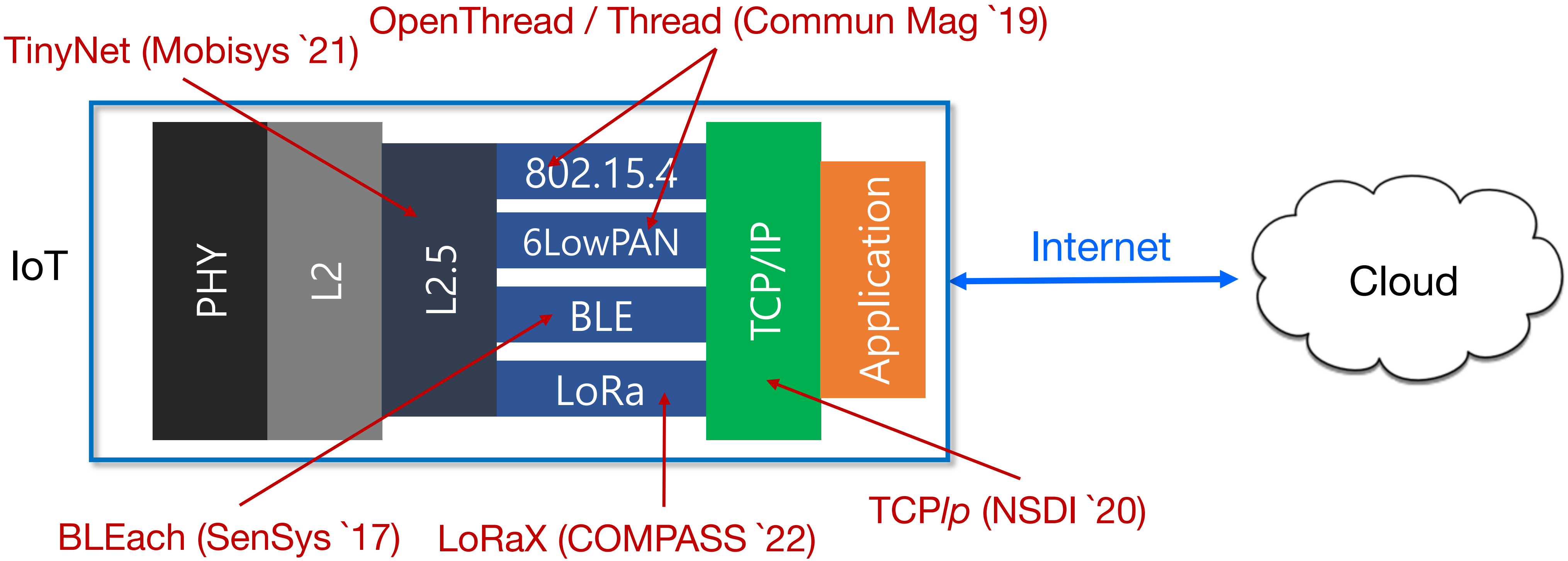


Device itself: lack of Internet connectivity

[3] Dong, Wei, et al. "TinyNet: A lightweight, modular, and unified network architecture for the internet of things." *Mobisys.* 2022.

Efforts for IoT-Cloud Connectivity

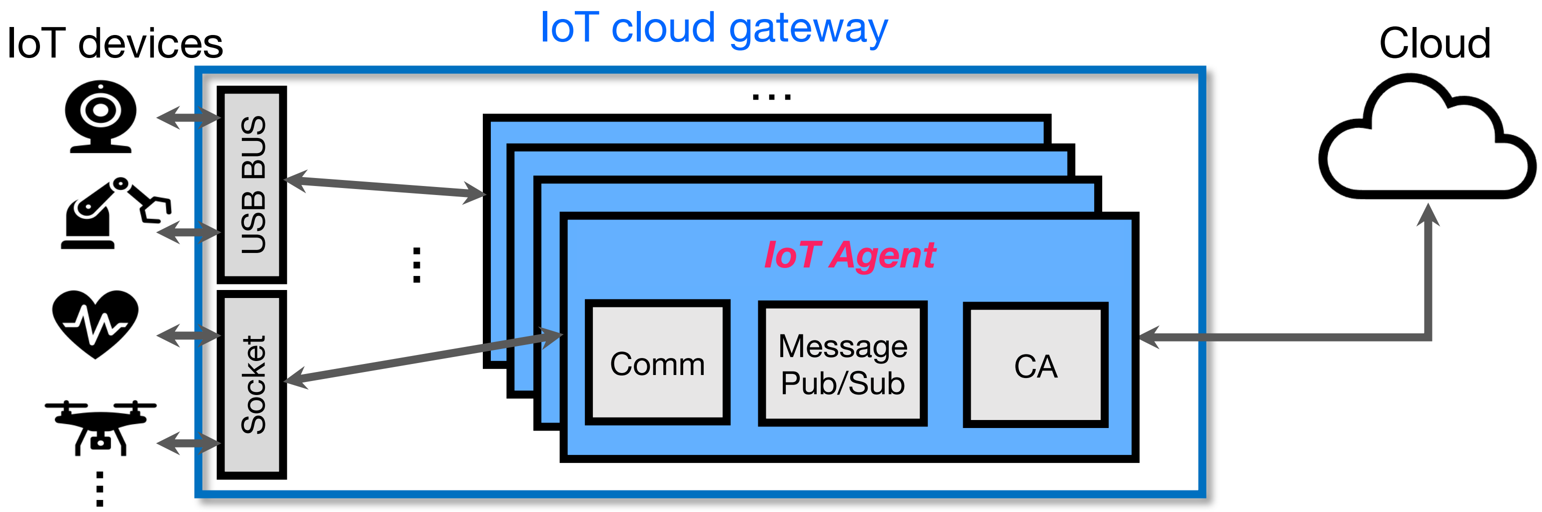
- Recent studies on enabling IoT-cloud connectivity of IoT devices
- Performance: problematic due to energy and resource constraints of devices[4]



[4] Kumar, Sam, et al. "Performant TCP for Low-Power Wireless Networks." *NSDI*. 2020.

Missing Spot: IoT Cloud Gateway

- **IoT gateway**: essential device for Internet connectivity
  - Standard solution in industry (e.g., Azure IoT hub, AWS IoT)

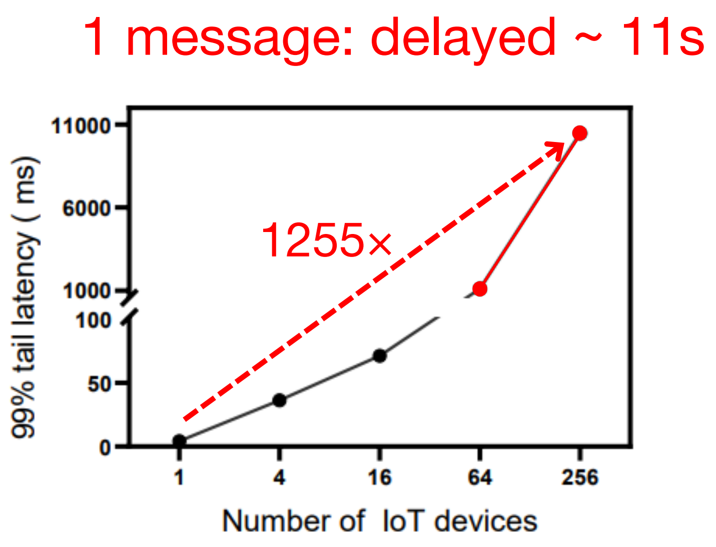


- Internally run **IoT agent** per device for
  - Communication across various protocols
  - Message delivery (e.g., MQTT, HTTP, AMQP protocols)
    - Generate messages / Sending messages
  - Certificate authority (CA) and data encryption (e.g., TLS)

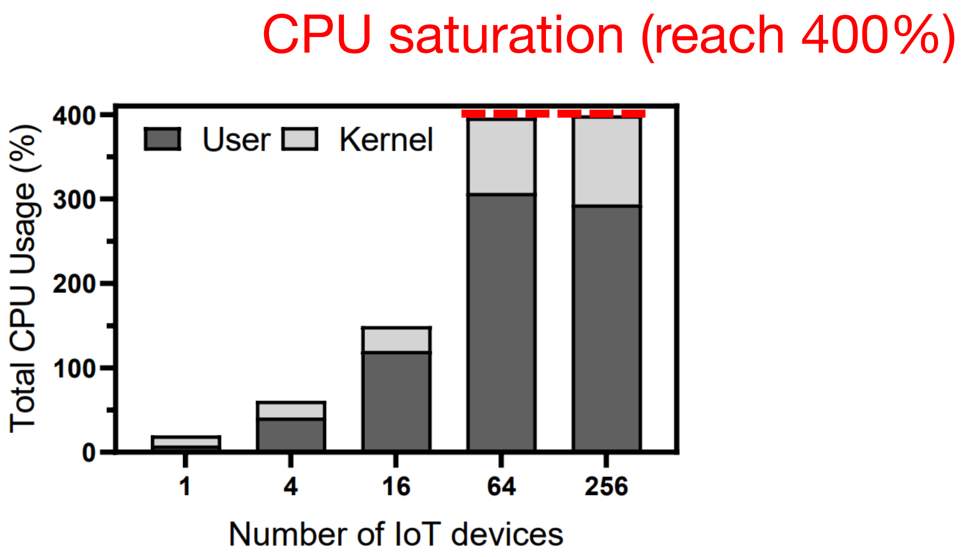
Observation: Scalability Problem

- IoT gateway must be scalable enough to manage multiple IoT devices
- Conduct experiments on closely mimicked real-world IoT scenarios
  - Emulate IoT gateway solutions of MS Azure and AWS
  - Emulate usecases such as smart farm sensors, heartbeat sensors, and drone cameras
- **Our findings on scalability**
  1) Poor latency
  2) Poor CPU usage
  3) Inefficient CPU mismatch
  4) TLS encryption overheads
  5) Message priority problem

Poor Latency and CPU Usage



High message latency

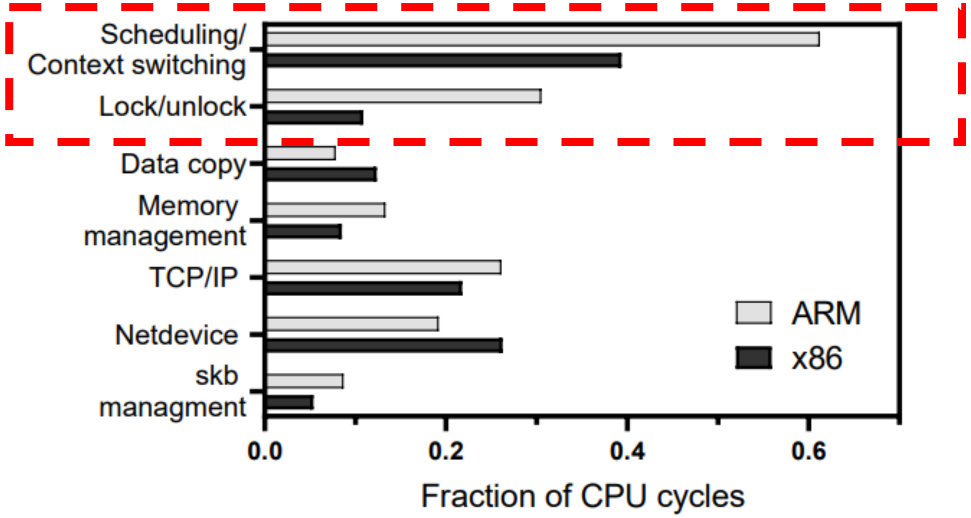


CPU bottleneck

- Exceed 1 second latency for only 64 devices (up to 11 s,1255×)
  - Serious problem (e.g., Health messages like ECG require < 1 s delivery)[5]
- Four cores saturated for only 64 devices

[5] Shukla, Saurabh, et al. "An analytical model to minimize the latency in healthcare internet-of-things in fog computing environment." *PloS one*. 2019.

CPU Profiling: Inefficient CPU Architecture

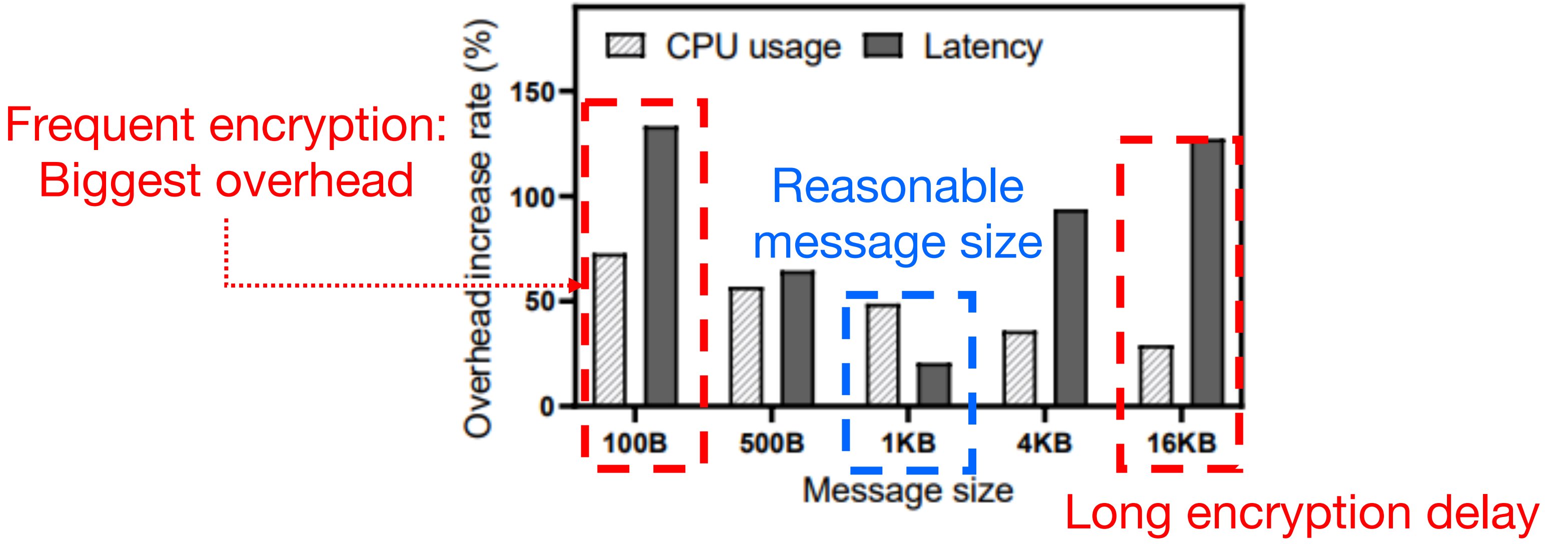


Inefficient CPU usage of ARM

- Profiling has been conducted on two architectures, ARM and x86 (by Perf)
- **ARM**—frequent and widely used for IoT gateways but **show inefficient usage**
  - Scheduling/context switching: 55.7% more than x86
  - Lock/unlock: 3.4x more than x86
    - Especially, spin-lock becomes too expensive for IoT devices

TLS Encryption Overhead

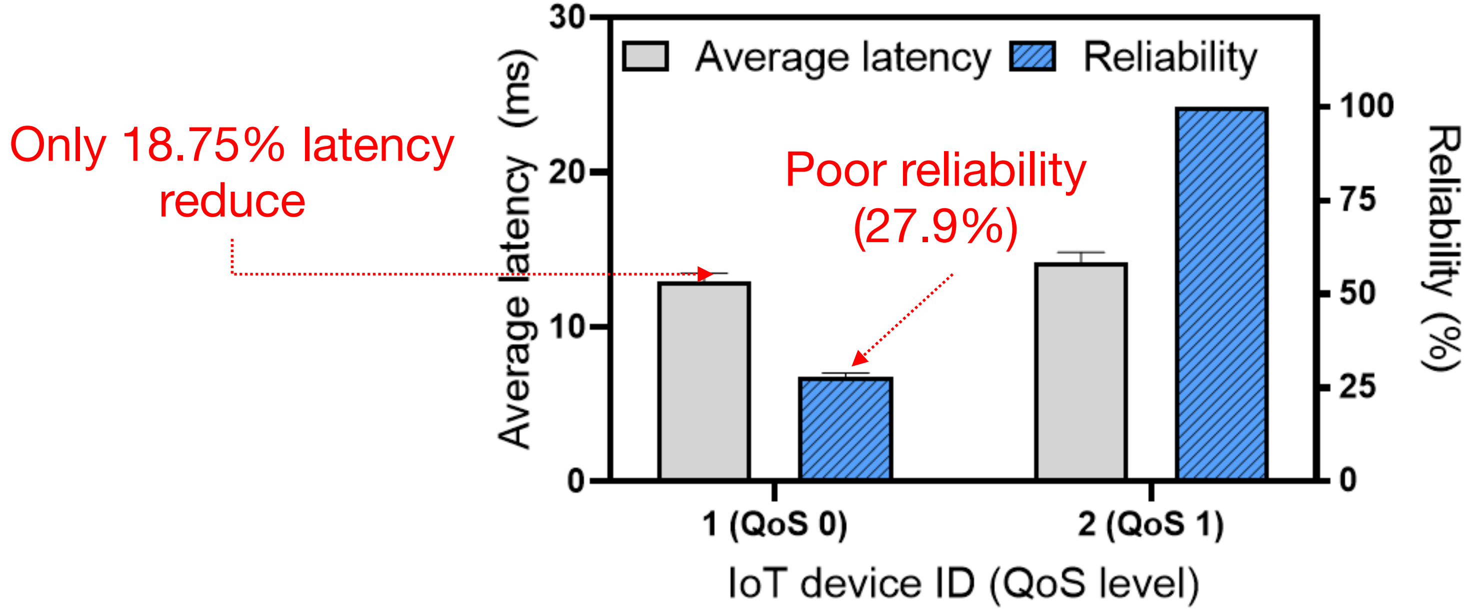
- Increase in CPU usage and latency when msgs are encrypted by TLS



- Small msg (100 B): 73% ↑ CPU cycles + 134% ↑ latency
- Large msg (16 KB): 128% ↑ latency
- Reasonable point—1KB: smallest overhead

Message Priority Problem

- Msg priority in IoT (baseline): MQTT`s QoS levels
  - QoS 0: lower latency & reliability / QoS 1: higher latency & reliability



- QoS 0: **18% better latency** but **62% higher packet drops** than QoS 1
  - Severe packet drops (reliability) with very small improvement in latency

Challenges

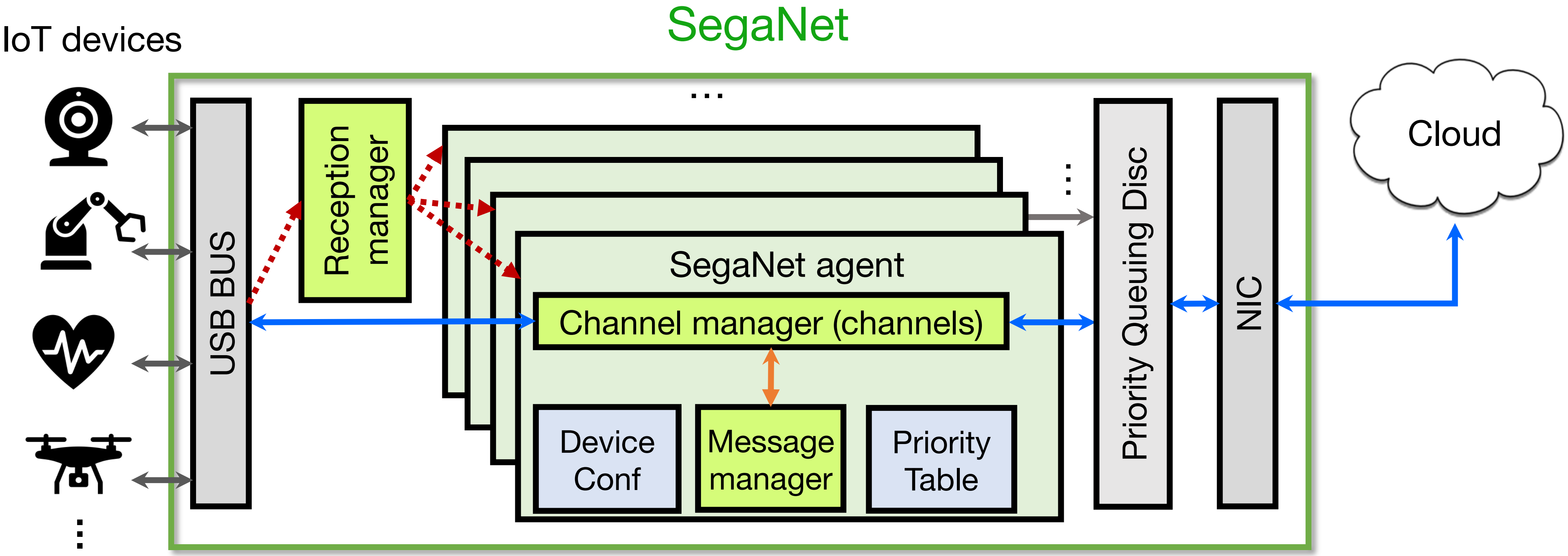
1. Inefficient CPU architecture
2. TLS encryption overhead
3. Message priority problem

Our approach

1. Architecture-aware IoT agent management
2. Efficient message batching
3. Priority guaranteed packet processing

SegaNet

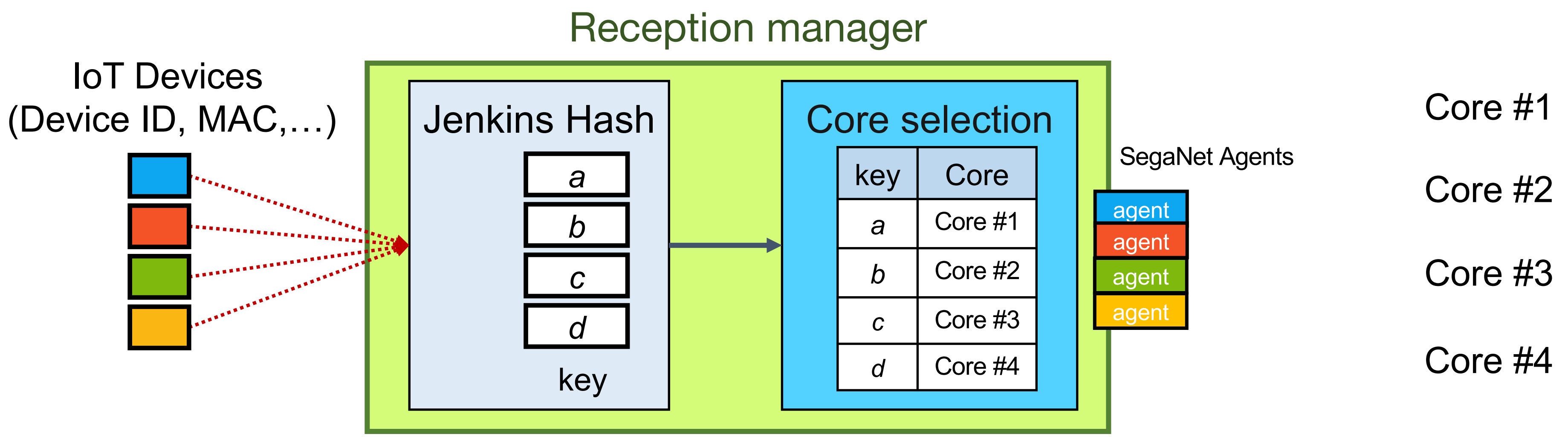
- Advanced IoT gateway architecture
  for performant & priority-oriented message delivery



Architecture-aware Agent Management

Challenge: CPU architecture mismatch

- ARM: high overheads in scheduling and lock
- Core affinity (pinning) to avoid costly operations
  - Reception manager: determine the specific core on which each agent operates
- How to decide core affinity?
  - Leverage *Jenkins* hash function to equally distribute IoT devices across cores
  - Future work: more intelligent schemes on core allocation



Efficient Message Batching

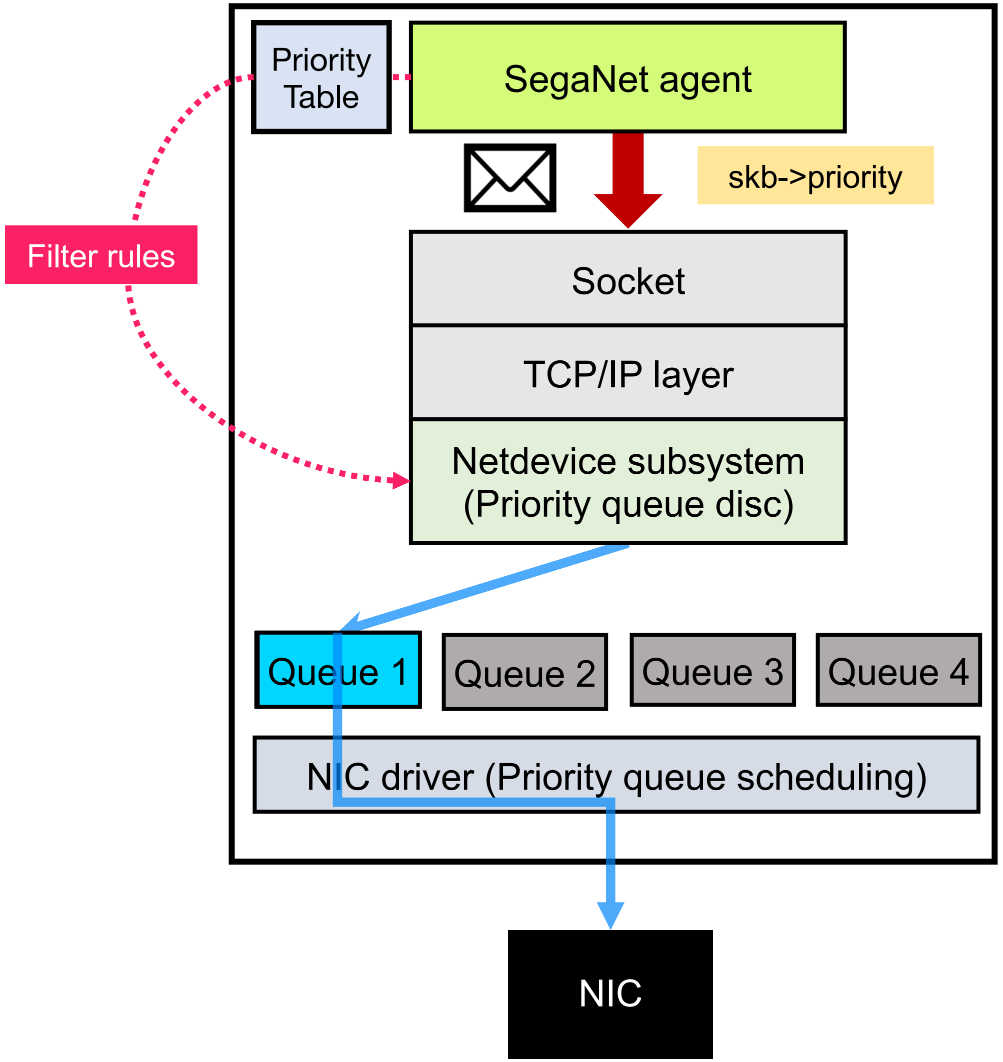
Challenge: TLS encryption overhead

- Avoid frequent TLS encryption by message batching
- Efficient message batching with three thresholds
  1) Number of connected IoT devices (n)
     - Small IoT devices ($n < 50$) → no batching
     - Numerous IoT devices ($n \geq 50$) → batching
  2) Message size (m)
     - Aggregating record data until $m \fallingdotseq$ 1KB
  3) Waiting time (t)
     - Aggregating record data until $t \fallingdotseq$ 1 second

Priority Guaranteed Packet Processing

Challenge: Message priority problem

- Prototype with two priority levels
  - 1: high-priority, 2: normal
  - Future work: Can be extended for various levels
- High priority msgs **w/o** message batching
- Msg classification per priority
  - Implemented by priority qdisc filters
- Dequeue packets based on priorities
  - Priority queue scheduling at the driver level



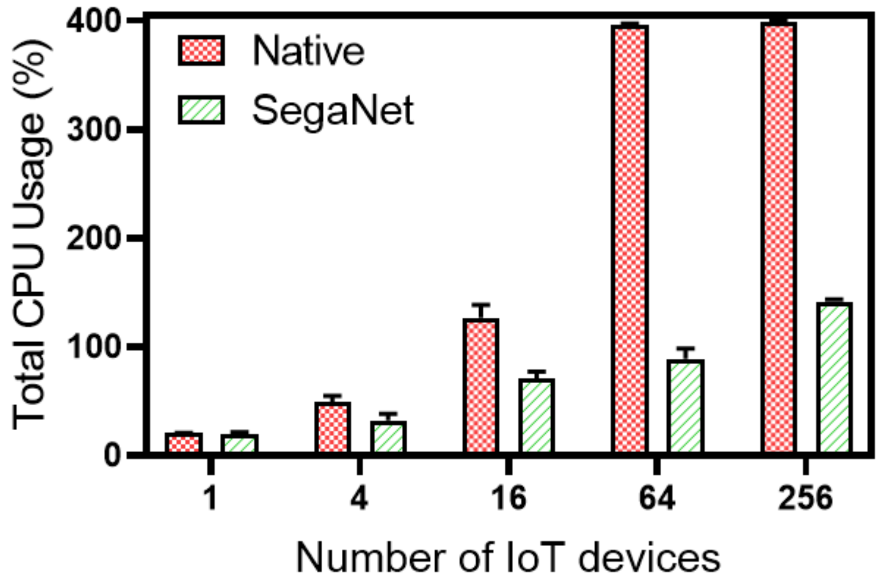
Evaluation

- Evaluation metrics
  1) CPU usage
  2) Message latency
  3) Message priority (latency and reliability)

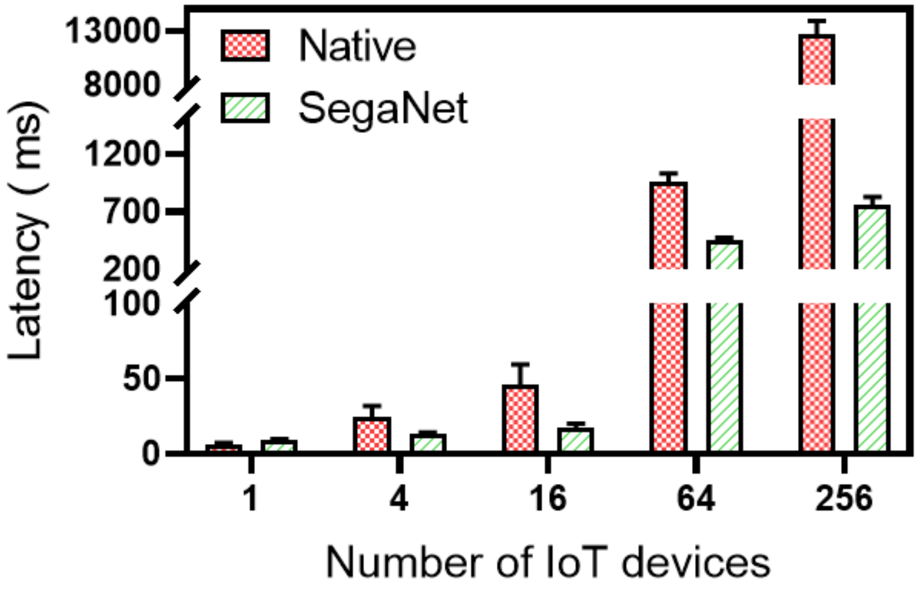
- Baseline (native IoT gateway): Raspberry pi 4 + MQTT (QoS 1) + TLS v1.3

- Future work
  - Real-world & large-scale experiment
  - Comparison with others (e.g., Interoperability IoT devices, Azure IoT gateway, …)

CPU and Latency Improvement



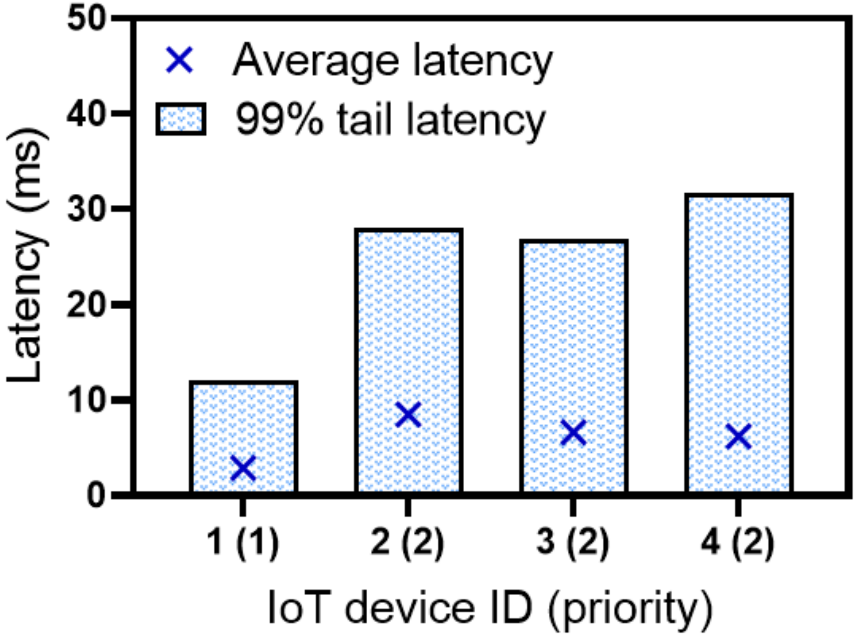
Total CPU usage



Message latency

- Total CPU usage: **~77.6%** reduced (***not saturated!***)
- Message latency: **~16.7×** improved (under than **0.8 s**)

Message Priority



- Setting different priorities (1 or 2) on four IoT devices
- Higher priority (1): **~43.2% better latency** than normal priority (2)
- **No packet drops** for all priorities

Summary

- Scalability issues of IoT gateway from real-world experiments
  - Poor message latency and CPU usage
  - CPU architecture mismatch results in ~56% CPU cycles waste
  - TLS encryption leads to ~73% more CPU usage & ~134% longer latency
  - Existing message protocol: only 18% latency reduce with ~62% packet drop

- SegaNet: An advanced IoT gateway architecture addressing key challenges:
  - 77% lower CPU usage and 16.7× better latency
  - Prioritized processing of messages with 43% faster delivery and zero drop

Future Works & Vision

Container
at IoT Gateway

Cooperation with
IoT and Cloud

Large-Scale
Environments
(e.g., OpenNetLab)

SegaNet: potential to serve **advanced IoT gateway** and **IoT-cloud framework**

Thank you

APNet
2023

QnA

Appendix – Experiment settings

Environment

- IoT gateway: Raspberry pi 4
  - ARM Cortex-A72 64-bit quad core@1.5GHz CPU, 8GB RAM, 1 Gbps Ethernet
- Each IoT device generates a 100B–1KB message per 100 ms
- IoT gateway publishes MQTT message with QoS 1
- Messages are encrypted with TLS v1.3
- Message broker is placed at cloud (emulate with separate server machine)

Latency measurement

- Elapsed time of individual messages processed by IoT gateway
- Measure 99% tail latency of all messages

CPU measurement

- Average CPU usage while processing messages using *mpstat & Perf (Linux*)