TinyNet: A Lightweight, Modular, and Unified Network Architecture for The Internet of Things

Gonglong Chen, Yihui Wang, Huikang Li and Wei Dong
College of Computer Science, Zhejiang University
Alibaba-Zhejiang University Joint Institute of Frontier Technologies
{desword,wang1hui,lihk,dongw}@zju.edu.cn

CCS CONCEPTS

• **Networks** → **Network architectures**.

KEYWORDS

network architectures, modular, Internet of Things

ACM Reference Format:
Gonglong Chen, Yihui Wang, Huikang Li and Wei Dong. 2019. TinyNet: A Lightweight, Modular, and Unified Network Architecture for The Internet of Things. In *SIGCOMM '19: ACM SIGCOMM 2019 Conference (SIGCOMM Posters and Demos '19), August 19–23, 2019, Beijing, China.* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3342280.3342290

1 INTRODUCTION

The recent years have witnessed the rapid growth of IoT (Internet of Things) technologies and applications. Low-power wireless radio technologies form the basis of many IoT applications. These low-power wireless radio technologies have drastically *different* PHY layer designs and *different* protocol stacks. The communication among IoT nodes *does not fully* interoperate to date [8]. For example, IoT nodes without IP support (due to strict resource constraints) cannot use application layer IoT protocols, e.g., MQTT, to communicate with each other. The underlying reason is the lack of a lightweight and holistic network architecture for heterogeneous IoT nodes having different radio technologies [3, 4, 7].

In this poster, we present TinyNet, a *lightweight*, *modular*, and *unified* network architecture for different low-power radio technologies. TinyNet has the following desirable goals:

Lightweight: The implementation should be lightweight, allowing the installation on low-end IoT nodes. **Modular**: The design should be modular so that new protocols can be easily composed for specific scenarios. **Unified**: TinyNet should be able to unify many different radio technologies at the network layer so that heterogeneous IoT nodes can seamlessly communicate with each other at or above the network layer. **Holistic**: TinyNet should have a holistic network stack from the PHY layer to the application layer so that high layer protocols, e.g., UDP, and MQTT, can be used. **Efficient**: TinyNet should have a high runtime efficiency, and achieves low communication delay.

SIGCOMM Posters and Demos '19, August 19–23, 2019, Beijing, China

ACM ISBN 978-1-4503-6886-5/19/08…$15.00
https://doi.org/10.1145/3342280.3342290

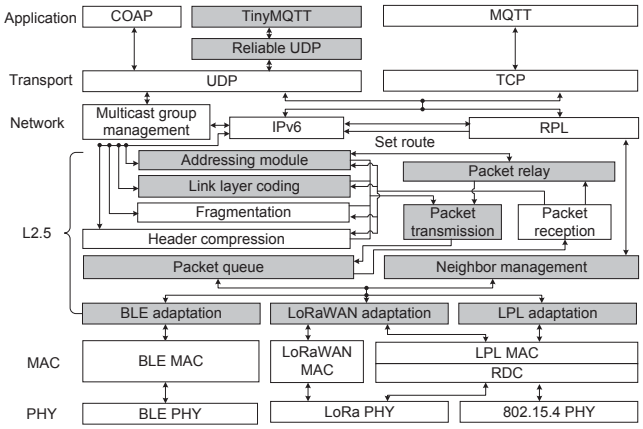

Figure 1: TinyNet architecture overview. The white boxes indicate existing modules while the gray boxes indicate newly implemented modules in TinyNet.

TinyNet consists of many *reusable* modules in different layers (Section 2.1). TinyNet needs to address several technical challenges in unifying different radio technologies. For example, how to provide neighbor discovery service for both single channel based radios and channel hopping based radios? How to enable multi-hop support for single-hop communication technologies such as BLE and LoRa? The neighbor management module and the packet relay module in TinyNet have addressed the above challenges, respectively. The modular architecture of TinyNet allows us to easily re-implement existing protocols, e.g., SP [10] as well as simplifying the creation of new ones by selecting specific modules in TinyNet (Section 2.2). We present one new protocol for example: RPL over LoRaWAN provides multi-hop wireless routing over LoRaWAN.

2 TINYNET ARCHITECTURE

Figure 1 shows the network architecture of TinyNet where the boxes indicate different modules and arrows show the interactions between modules. The current implementation of TinyNet unifies three radio technologies, including 802.15.4, BLE, and LoRa. On top of PHY layers, there are dedicated MAC protocols, e.g., BLE MAC [5], LoRaWAN [1] and LPL (Low Power Listening) MAC [2]. It is worth noting that TinyNet also allows LoRa to reuse the MAC protocols originally designed for 802.15.4 PHY, i.e., RDC (Radio Duty Cycle Control) and LPL MAC. An important part of TinyNet is located at L2.5, i.e., an abstraction layer between the link and network layer. This layer allows TinyNet to run over a broad range of radio technologies as well as provide support for IPv6 which forms the basis for interoperability at the network layer and above. At the transport layer, TinyNet supports TCP, UDP, and reliable UDP

Table 1: Decomposition of existing and composition of new network protocols.

Protocol	Packet Relay	Packet Transmission	Packet Queue	Neighbor Management	Addressing Module	Network Layer
SP[10] (*Exist*)	-	Uni./Broad.	Packet Priority	Link Estimation + Neighbor Table	Unicast Addressing	-
RPL over LoRaWAN (*New*)	Packet Relay	Uni./Broad.	FIFO	Link Estimation + Neighbor Discovery	Unicast Addressing	RPL+IPv6

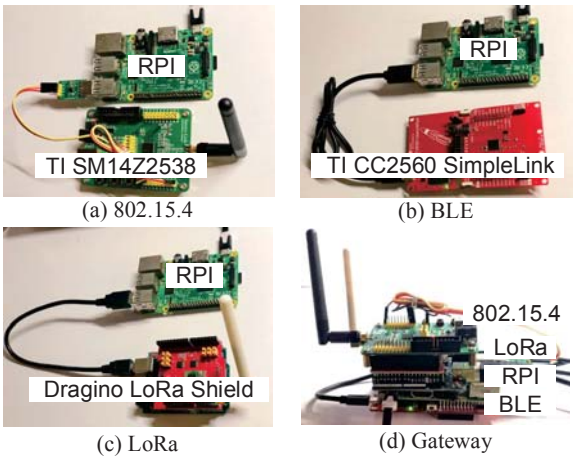

Figure 2: Four types of IoT nodes for the evaluation.

transmission. At the application layer, TinyNet supports protocols built upon TCP/UDP, including COAP and MQTT [11].

2.1 Main module description

Neighbor management. This modules provide two functionalities including neighbor discovery and link quality estimation. TinyNet provides a *unified* neighbor table: neighbor address, radio-on time, radio-off time, link quality, channels. The channel entry is used for channel hopping based protocols for recoding the synchronized transmission channels. Considering different underlying mechanisms, TinyNet has two neighbor discovery implementations: LPL based neighbor discovery (e.g., for 802.15.4 and LoRa) and channel hopping based neighbor discovery (e.g., for BLE and LoRaWAN). TinyNet provides two implementations for link quality estimation: RTT (round trip time) based and PRR (packet reception ratio) based.

Packet relay. This module is important to enable multi-hop communications for all three radios. Like neighbor discovery, TinyNet has two implementations for packet relay: LPL based packet relay (e.g., for 802.15.4 and LoRa) and channel hopping based packet relay (e.g., for BLE and LoRaWAN). The latter approach carefully allocates the reception/transmission slots for receiving/forwarding packets.

Packet queue. The packet queue module performs the buffer management and the packet scheduling. TinyNet provides three packet scheduling mechanisms: FIFO [9], priority based and multi-radio packet scheduling.

2.2 Protocol composition

The modular architecture of TinyNet allows us to easily reimplement existing and new protocols (Table 1).

SP. SP [10] is L2.5 layer protocol designed for 802.15.4 based sensor networks. SP allows link layers and network protocols to cooperate by maintaining and exposing a shared neighbor table and message pool. Priority based packet scheduling is utilized so that urgent packets, e.g., control packets, can get higher priority in SP. It is also worth mentioning that SP does not maintain synchronized channel information and thus the channel entry of neighbor table can be discarded.

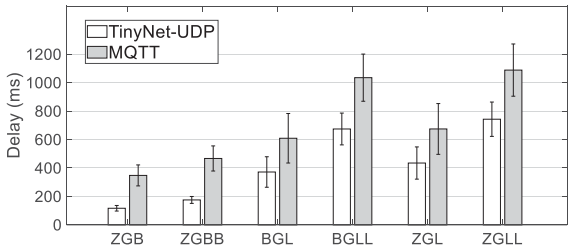

Figure 3: Multi-hop delay performance between heterogeneous IoT nodes through a gateway for different protocols.

RPL over LoRaWAN. RPL-over-LoRaWAN is a new protocol that uses RPL for multi-hop routing. RPL is a multi-hop wireless routing protocol based on 6LoWPAN [6] which compresses IPv6 headers to meet the resource constraints of IoT nodes. We can use PRR based link estimation since the underlying MAC protocols in LoRaWAN provides packet reception status. For neighbor discovery, we use channel hopping based neighbor discovery since LoRaWAN is channel hopping based. The packet relay module is required to allocate the time slots for packet transmission and reception.

3 EVALUATION AND FUTURE WORKS

TinyNet is implemented on Raspberry Pi 3 running Linux kernel version 4.14 and unifies three radio technologies, including 802.15.4, BLE, and LoRa (as shown in Figure 2). For the full implementation for three radios, TinyNet consumes 9.138~10.242 KB in RAM and 71.272~ 83.053 KB in ROM, indicating that it is lightweight for many existing IoT nodes. In Figure 3 shows the delay performance of different nodes through a gateway at different layers, i.e., the transport layer and the application layer. ZGBB means a 802.15.4 node communicates with a BLE node two hops away through a gateway. We can see that TinyNet allows interoperability among heterogeneous nodes at the network layer and above (e.g., MQTT and TinyNet-UDP), even if they are several hops away. TinyNet *automatically* handles all communication details such as multi-hop forwarding, neighbor management, etc. There are multiple future research directions. First, we would like to incorporate more radio technologies in TinyNet. Second, we would like to expose more tunable parameters for performance optimizations.

ACKNOWLEDGMENTS

This work is supported by the National Science Foundation of China under Grant No. 61772465, and Zhejiang Provincial Natural Science Foundation for Distinguished Young Scholars under Grant No. LR19F020001.

REFERENCES

[1] LoRa Alliance Technical Committee. 2017. LoRaWAN 1.1 Specification. *Standard V1* (2017).

[2] Adam Dunkels. 2011. The contikimac radio duty cycling protocol. (2011).

[3] Adam Dunkels, Fredrik Österlind, and Zhitao He. 2007. An adaptive communication architecture for wireless sensor networks. In *Proc. of ACM SenSys*.

[4] Cheng Tien Ee, Rodrigo Fonseca, Sukun Kim, Daekyeong Moon, Arsalan Tavakoli, David Culler, Scott Shenker, and Ion Stoica. 2006. A modular network layer for sensorsets. In *Proc. of USENIX OSDI*.

[5] Bluetooth Special Interest Group. 2014. *Bluetooth Specification Version 4.2*. Technical Report.

[6] Jonathan W Hui and David E Culler. 2008. IP is dead, long live IP for wireless sensor networks. In *Proc. of ACM SenSys*.

[7] Kevin Klues, Gregory Hackmann, Octav Chipara, and Chenyang Lu. 2007. A component-based architecture for power-efficient media access control in wireless sensor networks. In *Proc. of ACM SenSys*.

[8] Sam Kumar, Michael P. Andersen, Hyung-Sin Kim, and David E. Culler. 2018. Bringing Full-Scale TCP to Low-Power Networks. In *Proc. of ACM SenSys*.

[9] MS Momanyi, EO Oduol, and S Musyoki. 2014. First in first out (fifo) and priority packet scheduling based on type of service (tos). *Journal of Information Engineering and Applications* (2014).

[10] Joseph Polastre, Jonathan Hui, Philip Levis, Jerry Zhao, David Culler, Scott Shenker, and Ion Stoica. 2005. A unifying link abstraction for wireless sensor networks. In *Proc. of ACM SenSys*.

[11] Andy Stanford-Clark and Hong Linh Truong. 2013. Mqtt for sensor networks (mqtt-sn) protocol specification. (2013).