Pragmatic Low-Power Interoperability: ContikiMAC vs TinyOS LPL

JeongGil Ko[1], Nicolas Tsiftes[2], Adam Dunkels[2], Andreas Terzis[1]
[1] Department of Computer Science, Johns Hopkins University
[2] Swedish Institute of Computer Science

***Abstract*—Standardization has driven interoperability at multiple layers of the stack, such as the routing and application layers, standardization of radio duty cycling mechanisms have not yet reached the same maturity. In this work, we pitch the two *de facto* standard flavors of sender-initiated radio duty cycling mechanisms against each other: ContikiMAC and TinyOS LPL. Our aim is to explore pragmatic interoperability mechanisms at the radio duty cycling layer. This will lead to better understanding of interoperability problems moving forward, as radio duty cycling mechanisms get standardized. Our results show that the two flavors can be configured to operate together but that parameter configuration may severely hurt performance.**

I. Introduction

Interoperability is essential for the success of the emerging Internet of Things (IoT), but interoperability should not come at the cost of performance. To achieve lifetimes of years, radios must be duty cycled but standardization has not yet emerged as standardization thus far has occurred at the application, network, and adaptation layers. Examples of these efforts include the IETF 6LoWPAN working group's effort to provide a standardized IPv6 header compression layer for IEEE 802.15.4 [4], the IETF RoLL working group's effort to design and standardize RPL [1], the standard routing protocol for low-power and lossy networks (LLNs), and the IETF CoRE working group's CoAP as a lightweight application-layer protocol for accessing the resources of individual devices [6]. Despite having protocol standards available, it is still difficult to attain interoperability and good performance at the same time, because there is a high degree of freedom for implementations in choosing the network parameters. We have previously demonstrated the performance implications of interoperability of 6LowPAN, IPv6, and RPL through two independent implementations: BLIP/TinyRPL and uIPv6/ContikiRPL [3].

The next step is to achieve low-power interoperability. Although the aforementioned protocols are designed for low-power radios, the radios need to be duty-cycled in order to make them truly energy efficient and help the devices operate on batteries for years. However, given the diverse set of application-level requirements of IoT systems, we have not yet standardized the radio duty cycling layer. As a result, different vendors deploying their systems in the same geographical area can introduce different low-power MAC protocols. When these systems share the sample PHY layer standards (e.g., IEEE 802.15.4) along with the IPv6 standards for LLNs, these systems can start interoperating with each other and join each others' network. However, it is still unknown how different low-power MAC protocols (i.e., with duty cycling) will perform against each other. Moving forward, even with a standardized MAC protocol, we are still unsure how different parameters affect an interoperating system's performance when each system is configured to meet the goals of their own respective applications. Such increase in uncertainty of interoperating LLN systems' performance limits the practical deployment of LLNs in real environments.

In this work, we take the first steps in identifying the performance of interoperating low-power MAC protocols by using the low-power MAC protocols implemented for the widely used Contiki and TinyOS software stacks. Specifically, we use the ContikiMAC and the TinyOS LPL implementations, both sharing the sender-initiated duty cycling paradigm, to explore the effect of various low-power MAC level parameters in the interoperating systems' performance. We find that parameters such as periodic wake-up intervals and inter-packet transmission intervals can significantly affect the resulting performance of interoperating low-power systems. Our evaluations are one of the initial efforts to better understand the interoperability problems associated with low-power MAC protocols.

II. ContikiMAC and TinyOS LPL

Both TinyOS and Contiki provide *de-facto* standard low-power MAC protocols that use the sender-initiated duty cycling paradigm [2], [5]. In other words, for both low-power MAC protocols, the sender will try to send its packet continuously until the receiver wakes up by turning its radio on and sends back an acknowledgment packet.

When nodes use ContikiMAC [2], the duty cycling low power MAC protocol provided with the Contiki operating system, if a sender wants to transmit a packet, it repeats the packet transmission for up to a full wake-up cycle. Each packet is separated by an interval of t_{IPS}. During each wake-up cycle, the receiver samples the channel for energy twice. The time t_c between each sample must be larger than t_{IPS} to ensure that one of the sender's transmissions is detected. This timing constraint creates a challenge when communicating with other duty cycling protocols.

Similar to ContikiMAC, the TinyOS Low-Power Listening (LPL) implementation allows nodes to duty cycle their radios with a user-defined interval [5]. All nodes will wake up every t_{dc} to sample the channel with n CCA checks. Like ContikiMAC, when a node wants to send a packet, it will transmit the packet continuously for a time that is at most t_{dc}, and with an interval of t_{IPS} between the consecutive packet transmissions. If the receiver node wakes up and realizes that there is activity on the channel it will turn its radio on to listen for incoming packets, and return back an acknowledgment packet once the packet is received. The sender, upon receiving the acknowledgment packet will stop its continuous packet transmissions and return to its periodic wake-up behavior.

Using the two low-power MAC protocols above, we first try to examine if these protocols, originally designed and implemented without interoperability in mind, can interoperate and achieve high packet delivery performance and low radio on times. We then try to identify different parameters at the MAC layers that can affect the performance of an interoperating low-power wireless system. Based on our findings we will try to provide suggestions to MAC standard designers of what points should be further considered to allow interoperability between low-power wireless systems with high performance.

III. Low-Power MAC Interoperability Performance

We examine the performance of a network consisting of Contiki nodes and TinyOS nodes running their respective low-power MAC protocols, ContikiMAC and TinyOS LPL. We use the IPv6 network stack implementations in the two software stacks and examine the performance of the two different types of nodes communicating while varying different MAC layer parameters. Specifically, we focus on varying the inter-packet spacing (IPS) values and the wake up intervals for the two duty cycling protocols. The IPS value determines how frequent packets are sent at the transmitter while waiting for the target receiver node to to wake up and the wake up interval determines how often the receiver turns on its radio to listen to the channel for any potential incoming packets.

A. Adjusting the Inter-packet Spacing

The inter-packet spacing (IPS) time defines how frequent packets are transmitted at the sender while waiting for the receiver to wake up when using a sender initiated duty cycling protocol. When the IPS is configured to be too high, the receiver may wake up in between the sender's packet transmissions and not be able to detect the transmitter's intent to send a packet. If the IPS is set to low, the transmitter can be using too much of the limited wireless bandwidth, and it can also result in missed acknowledgment packets. ContikiMAC, by default, has a short inter-packet spacing time (i.e., 0.4 ms), whereas the IPS for the default TinyOS LPL implementation is very large (i.e., $\sim$ 8 ms). As explained in Section II, a node with the ContikiMAC will wake up only very shortly to perform just two channel sensing operations and turn its radio off again. Therefore, the long IPS value for a TinyOS LPL transmitter may result in low packet delivery performance. To validate this, we configure our simulation environment so that a Contiki node declares itself as a RPL root and a single TinyOS node tries connect to this RPL network and periodically sends a packet to the Contiki node every 10 seconds.

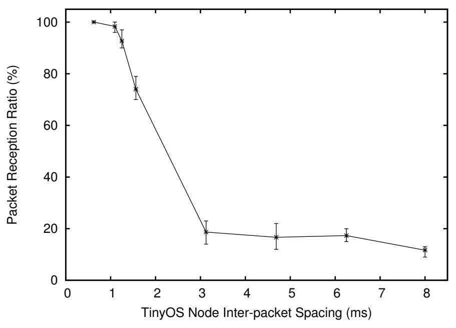


Fig. 1. Packet reception ratio of a single hop topology with a TinyOS-based transmitter and Contiki receiver. The transmitter sends 100 packets with an interval of 10 seconds. We vary the inter-packet spacing for the TinyOS transmitter while the default case is the case with 8 ms.

Figure 1 plots the packet reception ratio (PRR) at the Contiki-based RPL root for different IPS values set in the TinyOS-based transmitter. The experiment is set up so that no packets are dropped due to the link quality, and the two types of nodes are configured to have the same wake up intervals (e.g., 250 ms). The Contiki node, using ContikiMAC, makes two channel samplings at each wake up, and stays awake to receive a full packet if it senses sufficient energy (e.g., a packet transmission by the sender) in the wireless channel. We can notice in Figure 1 that when the IPS is high for the TinyOS node (e.g., default value of 8 ms), the average PRR is only 11.67%. On the other hand, as the IPS decreases, the PRR increases to a high value.

The next question we seek to answer with the same network configuration is the actual duty cycle (i.e., percentage of the radio on time) for the TinyOS transmitter node. In Figure 2, we plot the actual radio duty cycle achieved by the TinyOS-based node while sending packets to the Contiki receiver. We can notice that the duty cycle is at the lowest point when the IPS is $\sim$1.1 ms, and noticeably higher when the IPS is at different values. This behavior can be explained as follows. When the IPS is too high, the receiver node will not be able to properly receive the packets (i.e., low PRR in Figure 1); therefore, the transmitter will try to send its packets for the entire period and also for the maximum number of retransmissions requested by the application (in our case 3). On the other hand, when the IPS is too low (e.g., $\sim$0.6 ms), the transmitter will initiate its next continuous transmission before noticing the the acknowledgment frame sent by the Contiki receiver. As a result, the TinyOS node believes that its packet was not transmitted successfully (e.g., the receiver's radio is not on), and will continue sending for the entire period and also for two subsequent periods because of its application level retransmission request.

B. Effect of Different Wake Up Intervals

Along with t_{IPS}, another parameter that affects the performance of a duty cycling-based low-power MAC protocol is t_{dc}, the interval used by each node to wake up and sample energy on the wireless channel. We now try to determine how varying the configured wake-up rate affects the packet

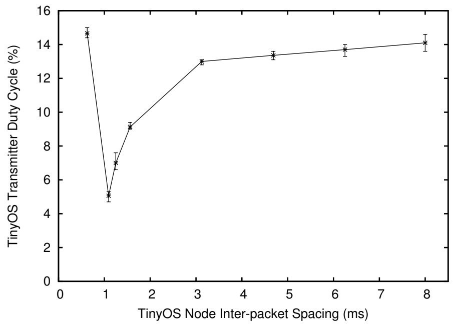


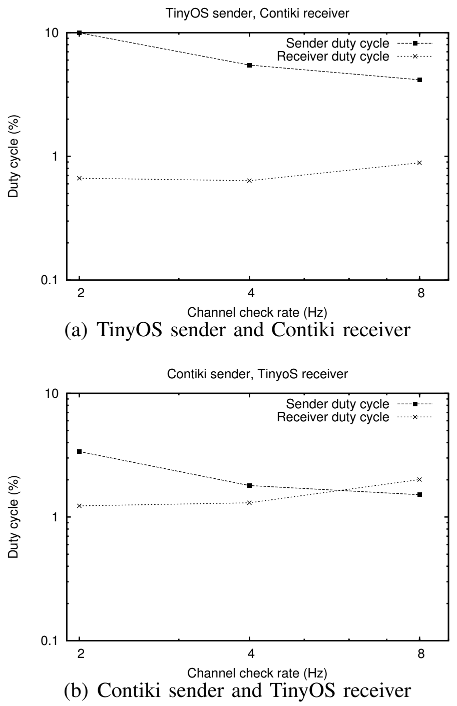
Fig. 2. Duty-cycle values of the TinyOS-based sender node for a single hop topology to the Contiki-based receiver with varying inter-packet spacing values for the TinyOS sender.

reception rate (PRR) and the observed duty cycle. This configuration only sets a lower limit on the energy consumption of the radio. Beside the effect of the data traffic in the network, the energy consumption and PRR depends on how well the implementations communicate with each other.

Our simulation environment is similar to the one in Section III-A, but in this case, we fix the TinyOS node to have an IPS of ∼1.1 ms and test for cases where we change the wake-up rate of the Contiki and TinyOS nodes. We test for two cases, the first when Contiki node is the receiver and TinyOS node is the transmitter, and in the second case, the TinyOS node receives packets from the Contiki node.

Figure 3(a) shows the observed duty cycle performance of the nodes when the TinyOS node sends packets periodically every 10 seconds to the Contiki node, while Figure 3(b) shows the performance when the roles are switched. We point out that both nodes are configured with the same wake-up interval in all cases. As expected, we see that as the wake-up interval decreases, the sender uses more energy and the receiver uses less. The PRR between the two nodes stay high (e.g., PRR $>$ 95%) for all cases, despite the reconfiguration of the wake-up intervals. Note that if the sender is configured with a shorter wake-up interval compared to the receiver, the PRR can be affected due to the fact that the sender may give up its transmissions before the receiver wakes up.

Along with the results presented in Section III-A, our evaluations suggest that regardless of the original target performance of individual implementations, the performance of an interoperating network can be heavily affected by the parameter configurations of each different implementation. Although our evaluations are done with two different low-power MAC implementations, ContikiMAC and TinyOS LPL, they share the same paradigm of sender-initiated duty cycling. Therefore, we believe that our findings can be shared with researchers that are designing a standardized sender-initiated duty cycling low-power MAC protocol. While the freedom to configure parameters should be given to the system developers, our results indicate that the standards should provide guidelines on parameter selections so that interoperating systems can achieve high performance while meeting the application requirements of individual systems.



(a) TinyOS sender and Contiki receiver

(b) Contiki sender and TinyOS receiver

Fig. 3. Observed duty cycle of a single-hop topology with different wake-up intervals.

IV. Conclusion and Future Directions

In this work we explore the interoperability performance at the radio duty cycling MAC layer using the *de-facto* standard low-power MAC protocols in Contiki and TinyOS, ContikiMAC and TinyOS LPL, respectively. We show that various parameters at the two low-power MAC protocol implementations can be configured so that they can communicate well. On the other hand, our results show that when poorly configured, the interoperating performance can severely suffer. We envision this work to be a first step in better understanding interoperability issues as we standardize a low-power MAC protocol for IoT systems.

V. Acknowledgments

This work was funded SSF through the Promos project, and by NSF Award #0855191.

References

[1] A.Brandt, J. Hui, R. Kelsey, P. Levis, K. Pister, R. Struik, JP. Vasseur, and R. Alexander. RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks. RFC 6550, March 2012.
[2] Adam Dunkels. The ContikiMAC radio duty cycling protocol. Technical Report T2011:13, Swedish Institute of Computer Science, 2011.
[3] JeongGil Ko, Joakim Eriksson, Nicolas Tsiftes, Stephen Dawson-Haggerty, Jean-Philippe Vasseur, Mathilde Durvy, Andreas Terzis, Adam Dunkels, and David Culler. Beyond interoperability: Pushing the performance of sensor network ip stacks. In *ACM SenSys*, 2011.
[4] G. Montenegro, N. Kushalnagar, J. Hui, and D. Culler. Transmission of IPv6 Packets over IEEE 802.15.4 Networks. RFC 4944, September 2007.
[5] D. Moss, J. Hui, and K. Klues. TEP 105: Low Power Listening. Available at http://www.tinyos.net/tinyos-2.x/doc/pdf/tep105.pdf, 2008.
[6] Z. Shelby, K. Hartke, C. Bormann, and B. Frank. Constrained Application Protocol (CoAP). Internet Draft, IETF (work in progress), 2012.