TinyRPL

From TinyOS Wiki

Contents

Overview

TinyRPL, is the TinyOS implementation of the IETF's IPv6 Routing Protocol for Low-power and Lossy Networks (RPL). The current implementation of TinyRPL, is baed on the most up-to-date revision of the RPL Internet Draft (-17). Other drafts that support RPL (e.g., trickle, of0, metrics) are based on their respective updates up to the same point.

Goal/Prerequisites

This tutorial will guide you through the process of installing a RPL network of nodes which includes an edge router, routing nodes, and leaf nodes. Note that the TinyRPL implementation is highly interconnected with the BLIP/6LoWPAN implementation in TinyOS.

RPL and TinyRPL introduction

Given the new challenges that low power and lossy networks pose, the IETF has recently proposed the IPv6 Routing Protocol for LLNs (RPL), which provides a mechanism to deal with multipoint-to-point traffic (i.e., collection), as well as point-to-multipoint and point-to-point traffic. RPL finds neighbors and establishes routes using ICMPv6 message exchanges and manages routes based on the a 'rank' value that represents nodes' relative position to the root of the routing tree.

TinyOS' implementation of RPL, TinyRPL, implements routing support for the three different traffic that RPL supports. Once a mote boots up with TinyRPL, TinyRPL will operate in the 'background' of an application to exchange route related messages with other RPL-using nodes. The RPL Routing Engine begins its operations once a global address is allocated using the DHCPv6 process. Once the RPL routing engine starts exchanging DIO and DAO messages, it can receive packets from the application layers. The packet-sending IP interface can be connected identically to the ways that they are wired in blip. Once the packet reaches the point to discover the next hop address (on the blip stack), RPL's routing table will be called to retrieve the next hop node's IPv6 address for the specified destination.

Platform support

Currently TinyRPL is supported on all platforms supported by BLIP 2.0.

TinyRPL interfaces

Note that the application does not need direct connections to any of these interfaces. These interfaces are internally connected to the blip/6lowpan layer (see Section on "RPL and TinyRPL introduction"). These interfaces are positioned within **tos/lib/rpl/**.

- interface RPLRank

- interface RPLRoutingEngine

- interface RPLDAORoutingEngine

- interface RPLOF

Related interfaces

- interface RootControl

TinyRPL uses the generic RootControl interface to set the root of the RPL DODAG.

Makefile configuration

- PFLAGS += -DRPL_ROUTING

Indicates that the RPL routing protocol will be used.

- PFLAGS += -DRPL_STORING_MODE

Indicates the storing mode of the RPL routing protocol will be used. While the non-storing mode has not yet been implemented, it will have a PFLAG of **-DRPL_NON_STORING_MODE**.

- PFLAGS += -I$(TOSROOT)/tos/lib/net/rpl

Points the Makefile to the TinyRPL implementation directory.

Configuration file modifications

- components RPLRoutingC;

The configuration file for the application code should include the RPLRoutingC component.

Implementation file modifications

For the node that will act as the root of the routing tree (e.g., the edge router), the implementation file should use the **RootControl** interface. The RootControl interface should be connected to the **RPLRoutingC** component or the **RPLRoutingEngineC** component.

Packets can then be sent through a transport layer component such as the UDP component supported by BLIP 2.0.

Usage Tips

1. To get the IPv6 address of the edge router (RPL DODAG Root), add **components RPLRoutingEngineC** to use the **RPLRoutingEngine interface**. The command getDodagId() will return the IPv6 address of the edge router

2. It is desirable to increase BLIP_L2_RETRIES to a higher value (3 is the BLIP 2.0 default) so that the ETX calculation gains finer granularity.

Sample application

A sample application of TinyRPL can be found in **$(TOSROOT)/apps/tests/TestRPL/udp/**. You can use **make <platform> blip** to test out the application.

Limitations

The current implementations of TinyRPL supports only the storing mode of RPL with the default upwards routes. The non storing mode implementation for downwards routes are soon to be supported. Also TinyRPL supports only a single RPLInstanceID while supporting multiple DODAGIDs and does not support the security options.

Currently TinyRPL is only supported on the telosb and epic platforms. Therefore, TinyRPL is not supported on the TOSSIM simulator which requires a micaz binary.