WANDA — Active Message and IPv6 support

Table of contents

1 Introduction

As effort to offer a common networking protocol for low-power networks the Task group 4 of the IEEE 802.15 Working Group has defined a standard which specifies the physical layer and media access control for low-rate wireless networks. This standard, called IEEE 802.15.4, has been explicitly designed for short range communication of low-cost devices without relying on infrastructure. There are several alternative protocols for WSN, but IEEE 802.15.4 is available on many commercially widespread sensor nodes. IEEE 802.15.4 was specifically developed for low-cost and low-power sensors and actuators and is not usually available on commodity computers. Therefore to access a WSN a user needs help from a third party that can support communication both with him and the WSN itself. The node – or nodes – that connect a user to a WSN can basically behave in two different ways: it can work as a proxy and totally decouple the user's side of the network from the WSN's side or it can work as edge router and route to the WSN the same network layer packets used locally by the user.

As part of our WANDA middleware we developed a software component supporting TinyOS operating system on sensor nodes using IEEE 802.15.4 that allows an application to access a WSN reachable either by a proxy or an edge router, thus simplifying the communication logic necessary to establish a connection with a sensor node.

2 Translucent communication support

IEEE 802.15.4 is not used on common computer networks, therefore to allow access to a WSN from commodity hardware it is necessary to introduce a device which can act as gateway. A gateway is network device that has at least two different network interfaces: one to communicate with the WSN and one or more to communicate with normal computers. The gateway is a required hardware, but its role can differ based on the kind of WSN integration chosen. The integration philosophy of a WSN in a commodity network can achieve vertical or horizontal access.

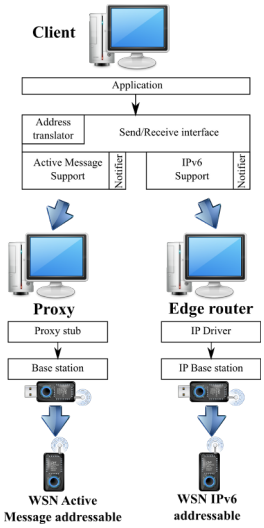
Vertical access sees WSN as autonomous networks fundamentally decoupled from computer networks: the gateway acts as proxy and the IEEE 802.15.4 protocol is used as a peripheral interconnection method; clients can not directly connect to sensor nodes and all interactions are managed by the proxy, making the WSN totally opaque to other networks. On TinyOS the message format of the IEEE 802.15.4 frames is called Active Message.

Horizontal access is the kind of connectivity we have on the Internet, as network that connects hosts, devices and other networks together. The devices that hold the networks together are routers. The routers that allow hosts using different physical layer protocols to connect using the same network layer protocol are called *edge routers*. To horizontally access a WSN therefore means to configure the gateway as edge router allowing the clients to use normal network protocols like IP or ICMP to connect to WSNs' sensor nodes. The recent Berkeley IP implementation for low-power networks project (blip) is the most complete implementation of a full fledged IPv6 stack for TinyOS.

TinyOS supports coexistence between Active Message-based and IPv6-based WSNs (ref.: TEP125). The client software running on commodity networks has to access those different kind of WSN using a different API, respectively the Active Message interface library provided with TinyOS and the Berkeley sockets API. This heterogeneity is hence visible to higher level applications. However it may be desirable to have a single API to access both kind of WSN. A client software may need to access different types of WSN in several scenarios: in a technological shift from Active Message to IPv6 based WSN there may be nodes capable of using both communication techniques, or Active Message and IPv6 can be deployed on sensor nodes with different duties respectively to achieve higher performances or ease of deployment.

To allow an easy deployment and access to WSN using Active Message and IPv6 we developed a component that provides the same interfaces to both communication techniques (see following figure). At the higher level on clients runs a *send/receive interface* able to use abstract addresses, independent from the communication protocol effectively used by the sensor nodes. The *Address Translator* translates the abstract addresses into technology dependent addresses, using for its choice a database which stores the configuration of each node in the WSN. The effective addresses are 16 bit identifier for nodes using Active Message and link local IPv6 addresses. From this point onward the the messages follow different paths accordingly to the communication protocol adopted by the destination sensor node. The message, encapsulated in a packet of the correct format, is then forwarded to the communication support modules, wich in turn forward them alternately to either the Active Message proxy or edge router. The proxy writes every packet it receives to a serial port connected to a sensor node that runs a specific TinyOS application called *Base Station*; this sensor node forwards every message that it reads from the serial port to its IEEE 802.15.4 transceiver. If the destination node is IPv6 enabled, the packet is routed based on the client's routing tables; when the packet reaches the edge router it is forwarded to the WSN using a sensor node using the analogous of the *Base Station* software, called *IP Base Station*, which supports IPv6 and cand handle IPv6 addresses.

Details about the WANDA abstract address communication support can be found in the documentation section.



3 Active Message and IPv6 support: technical details

The IEEE 802.15.4 standard, endorsed by the IETF, has gained a remarkable market share and is available on many commecially available sensor nodes. Its complete description can be found in IEEE 802.15.4-2006.pdf. We will only describe the most important aspects of IEEE 802.15.4 and how it can be used to support IPv6.

3.1 IEEE 802.15.4

The physical layer protocol of IEEE 802.15.4 states that a frame must consist of three parts:

- a preamble to allow receiving devices to synchroniza and lock onto the bit stream;
- a 7 bt header which contains the frame length, thus making the payload at most 127 octets long;
- a variable length payload, which carries the MAC sublayer frame.

The MAC layer header has a variable length depending on the chosen options. This is its complete structure:

Octets: 2	1	0/2	0/2/8	0/2	0/2/8	0/5/6/10/14	variable	2
Frame Control	Sequence Number	Destination PAN Identifier	Destination Address	Source PAN Identifier	Source Address	Auxiliary Security Header	Frame Payload	FCS
		Addressing						

Table 1: IEEE 802.15.4 MAC frame format.

The header fields are:

- *frame control*: it specifies the frame type, the addressing fields available and other control flags;

- *sequence number*: it specifies the sequence identifier for the frame;
- *destination PAN identifier*: it specifies the unique identifier of the network to which the destination node belongs to;
- *destination address*: it contains the address of the destination node;
- *source PAN identifier*: it specifies the unique identifier of the network to which the sender node belongs to;
- *source address*: it contains the address of the sender node;
- *auxiliary security header*: it reports information required for security processing if the application wants to use the encryption features supported by IEEE 802.15.4;
- *frame payload*: it is the application specifica data carried by the frame;
- *FCS*: the Frame Check Sequence is used to detect frame corruption.

3.2 Active Message

The IEEE 802.15.4 protocol is highly configurable. The TinyOS developers chose to use a fixed format on every node model that support it (ref.: TEP111). The resulting IEEE 802.15.4 header is composed by: frame control, sequence number, destination PAN identifier, destination address, source address. To these fields TinyOS adds an 8 bit field called *type* that, similarly to a UDP port, permits to identify different commpunication endpoints (applications) running on the same sensor node. Hence, the IEEE 802.15.4 header configured by TinyOS is 10 octets long.

3.3 IPv6 over IEEE 802.15.4

Allowing IPv6 packets over IEEE 802.15.4 networks poses many different challenges, stated in the RFC 4919; among them the most apparent are the different packet sizes and the fragmentation management:

- given that the maximum physical layer packet is 127 bytes, the resulting maximum frame size at the media access control layer is 102 octets, that can be further reduced down to 81 octets if the link layer security features are used. Since the IPv6 header, without optional headers, is 40 octets long, carrying it in its complete representation can take between about 40% and 50% of the MAC layer payload depending of the IEEE 802.15.4 network configuration, hence causing a huge overhead.
- The IPv6 standard mandates that an IPv6 compliant network must be able to handle a maximum transmission unit of at least 1280 bit, far larger than the payload that a IEEE 802.15.4 frame can carry.

To address these problems the IETF 6lowpan working group defined a header compression and encapsulation mechanisms called **6LoWPAN** (acronym of *IPv6 over Low power Wireless Personal Area Networks*). The header compression mechanism tries to compact the IPv6 header removing all the information that can be inferred by the IEEE 802.15.4 MAC frame, like the source or destination addresses and the packet length. In the ideal case the header compression can reduce the 40 octets long IPv6 header down to 2 octets. The standard also describes a specialized compression format for the

UDP header. The problem of handling packets up to a minimum of 1280 bits is solved by the fragmentation mechanisms, that defines how IPv6 packets that don't fit in a single IEEE 802.15.4 MAC frame should be fragmented and how the information necessary to reassemble the packet on the destination node should be encoded.

The base specification developed by the 6lowpan IETF group is RFC 4944.

3.4 WANDA translucent support

We developed a component, whose architecture has been described in the previous sections, that supports both Active Message and IPv6 as communication protocols. The full technical details che be found in the documentation section and in the source code, here we will give a brief summary of the most important aspects of the communication support of WANDA.

Following the style of TinyOS split-phase operations, the functions provided to applications by the communication component are non-blocking: every request to send data immediately returns without guarantee of success. The communication protocol (either Active Message or IPv6) is selected by the *Address translator*, which relies on a database that stores the communication protocol supported by every accessible node.

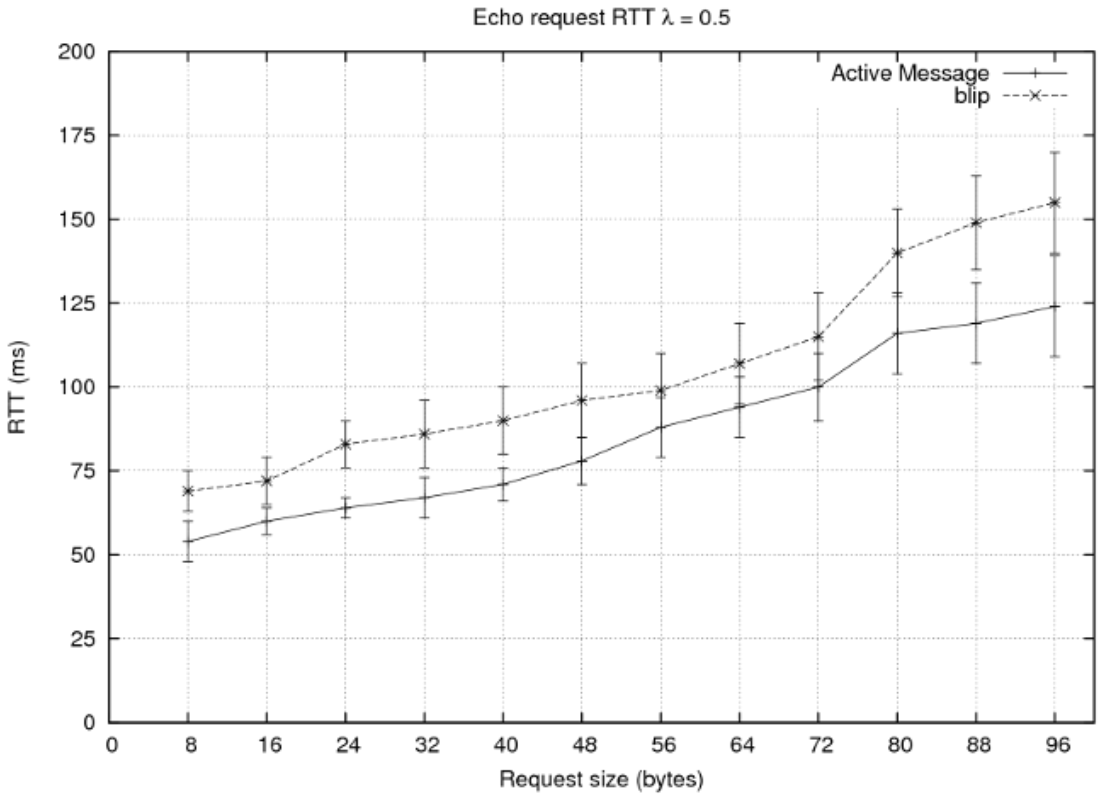
Since WANDA supports both Active Message and IPv6, the abstract addresses that it uses are the lowest common denominator between Active Message addresses and IPv6 addresses, i.e. 16 bit identifiers.

Whenever the WANDA component running on a client receives a message it notifies every registered listener of the event, providing them the sender's address and the data payload.

4 Performances

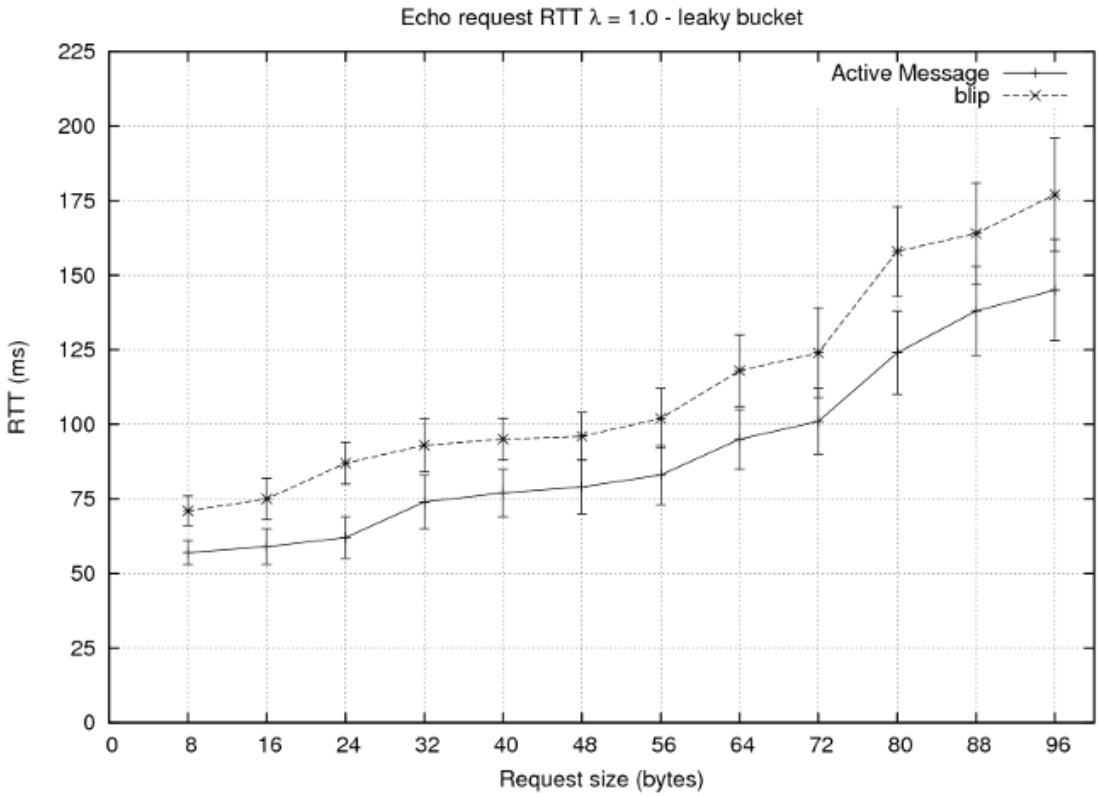
To evaluate the performances of the WANDA communication support, we developed a echo server, that simply sends back to the sender any message it receives, for TinyOS using both Active Message and blip. A gateway acted respectively as proxy and edge router. A client host on the commodity network generated echo requests separated by a random interval of time generated with an exponential distribution of rate # and mean 1/#, thus the number of requests followed a Poisson distribution. We measured the round trip time (RTT) for varying packet length and values of #.

In the following plot we can see the RTT time for #=0.5 at increasing packet size.



We can see that, given the limited IEEE 802.15.4 bandiwdth, there is a strong relation between packet size and RTT. The Active Message implementation is always faster than the IPv6 one, the difference is due to the handling of IPv6 addresses.

In the following plot we can see the RTT time for #=1 at increasing packet size.



To achieve these performance we modified the proxy deployed with TinyOS to introduce a "leaky bucket" behavior adding a small minimum delay between the dispatch of two messages. These performance can't be obtained using the default implementation of the TinyOS proxy (*Serial Forwarder*) since it floods the sensor node running the Base Station application, causing a high packet drop rate and a much worse RTT.

The good performance obtained allows us to say that the translucent communication component developed for the WANDA middleware provides an homogeneous message send/receive interface that grants both higher performance of Active Message and ease of integration of IPv6.