Ad Hoc Networks 34 (2015) 252–264

Contents lists available at ScienceDirect

Ad Hoc Networks

journal homepage: www.elsevier.com/locate/adhoc

Software-defined underwater acoustic networking platform and its applications

Dustin Torres [a], Jonathan Friedman [a], Thomas Schmid [b], Mani B. Srivastava [a], Youngtae Noh [c,*], Mario Gerla [d]

[a] Electrical Engineering Department, University of California at Los Angeles (UCLA), United States
[b] Electrical and Computer Engineering Department at the University of Utah, United States
[c] Department of Computer Science and Information Engineering, Inha University, South Korea
[d] Computer Science, University of California at Los Angeles (UCLA), United States

ARTICLE INFO

Article history:
Received 2 March 2014
Received in revised form 19 December 2014
Accepted 21 January 2015
Available online 31 January 2015

Keywords:
Underwater acoustic networking
Software defined radio
Time synchronization
Channel allocation protocols

ABSTRACT

As underwater communications adopt acoustics as the primary modality, we are confronting several unique challenges such as highly limited bandwidth, severe fading, and long propagation delay. To cope with these, many MAC protocols and PHY layer techniques have been proposed. In this paper, we present a research platform that allows developers to easily implement and compare their protocols in an underwater network and configure them at runtime. We have built our platform using widely supported software that has been successfully used in terrestrial radio and network development. The flexibility of development tools such as software defined radio, TinyOS, and Linux have provided the ability for rapid growth in the community. Our platform adapts some of these tools to work well with the underwater environment while maintaining flexibility, ultimately providing an end-to-end networking approach for underwater acoustic development. To show its applicability, we further implement and evaluate channel allocation and time synchronization protocols on our platform.

1. Introduction

The underwater medium presents many challenges for digital communication. There is limited available bandwidth and high bit error rates caused by multipath, fading, and long propagation delay. The speed of sound is five orders of magnitude less than that of terrestrial radio, which can make it hard for underwater networks to synchronize, exchange data, update routes, and communicate efficiently.

Due to the severity of multipath underwater, a receiving node might be at a point where there is little energy in the received signal making it hard to receive without errors [1]. These spots of destructive multipath interference vary with time due to the movement of water, and make it quite hard to even have a static network topology. Along with multipath, other unfavorable characteristics such as ambient noise, bubbles, surface scattering, and slow propagation speed make developing underwater networks a difficult task [1].

Furthermore, the ocean varies significantly both temporally and spatially, which makes it challenging to create a model of a "typical" underwater acoustic channel [1]. Since there is no "typical" model, there is no single network architecture that works best in all situations. Therefore, depending on the current characteristics of the channel, there is a variable amount of bandwidth, various noise

* Corresponding author. Tel.: +82-32-860-7445.
E-mail address: ytnoh@inha.ac.kr (Y. Noh).

http://dx.doi.org/10.1016/j.adhoc.2015.01.010
1570-8705/© 2015 Elsevier B.V. All rights reserved.

sources in different frequency bands, varying inter-symbol interference (ISI) depending on the water depth, and many other characteristics under consideration. Even if the channel noise is known, attenuation still depends on both distance and frequency, along with space and time varying multipath [2].

We present a software defined Underwater Acoustic Networking plaTform (UANT) to aid development of underwater acoustic networks. A software defined system can help to address the constantly changing underwater acoustic channel by way of reconfigurability. A flexible approach allows for system parameters at all layers to be easily modified without the need for specialized hardware. UANT uses GNU Radio, a software defined radio framework, to achieve configurability at the physical layer. TinyOS has been widely adopted for the use on sensor network platforms and provides a full network stack. We adapted these widely supported tools that have proven effective prototyping, development, and implementation for terrestrial networks to be used in UANT.

Since the characteristics of the underwater acoustic channel cannot be properly modeled with a static configuration, it is important to be able to change the properties of an acoustic modem at run time. UANT has the flexibility of software defined radios and the advantages of the network layers of TinyOS and Linux, ultimately providing and end-to-end network for easy underwater development from the physical to application layer. This paper significantly enhances our earlier work [3] as follows:

- We add Applications and protocol implementation on UANT section to show UANT's applicability and discuss channel allocation protocol's detailed description (Section 4).
- We implement and evaluate channel allocation protocol and time synchronization protocol (THSL) on our proposed underwater acoustic networking platform (Section 6.2 and 6.3 respectively).

2. Background

Software defined techniques have been of interest in recent years not only for terrestrial radios but also acoustics and ultrasonics [4]. Jones et al. described some of the benefits to the use of software define radio for underwater use [5]. They talked about the possibility to improve underwater acoustic performance by using methods that have worked well with terrestrial radios such as Cognitive radio via software defined techniques.

Sözer and Stojanovic developed rModem [6] to achieve a similar goal of having a configurable acoustic modem. However, the rModem is a standalone board that uses an FPGA and DSP. Furthermore, rModem focuses on lower layers and does not provide an end-to-end networking environment, making it hard to study the impact on actual applications. We hope to supplement platforms such as the rModem by aiding research in MAC and PHY layer protocols that can ultimately be implemented on standalone nodes to create an underwater network.

2.1. GNU Radio and USRP

Software defined radio has received a lot of attention most notably in the research community. The ability to use software to modulate and manipulate the received and transmitted signals allows for rapid development without the need or cost of specialized hardware. GNU Radio [7], one of the most popular SDR frameworks, is comprised of a flow graph and signal processing blocks. The signal processing blocks are written in C++ and act as the “heavy lifters” whereas the flow graph is setup in Python in order to move data from one block to the next. In this way many modulation schemes can be created using standard C++ blocks (already included in GNU Radio) and connecting them together in a flow graph. There is a large community of users who have contributed to this open source project, both signal processing blocks as well as various applications. The many contributions have led to a large library of modulation schemes including GMSK, PSK, QAM, CPM, OFDM, and more.

We use GNU Radio along with the Universal Software Radio Peripheral version 1 (USRP1) to achieve underwater acoustic communication. The USRP created by Ettus Research [8], is a radio front-end that is commonly used with GNU Radio. Although the option of using a sound card provides a low cost solution, the USRP offers a wider frequency range as well as more dedicated hardware. The USRP has a total of 4 ADCs and 4 DACs allowing for up to 16 MHz of bandwidth each way, which is proficient for the underwater acoustic channel. We have modified an example application used for digital communications bundled with GNU Radio to use the USRP as a NIC via the TUN/TAP interface. This allows for wireless communications between two computers using software defined radio that can run networking applications over TCP.

2.2. TinyOS and TOSSIM

TinyOS is a widely used sensor network operating system created at University of California, Berkeley and meant for sensor nodes requiring concurrency and flexibility while being limited to resource constraints [9]. TinyOS is implemented in the NesC language, which supports the concurrency model needed for sensor networks. TinyOS is widely used both in commercial applications as well as in academics for research purposes. There is a large user base who constantly contribute to the open source project, which allows UANT to always benefit from the newest protocols. For instance, Washington University contributed a MAC Layer Architecture [10] to provide the MAC layer with many commonly needed functions for MAC protocols.

Generally TinyOS is compiled for a sensor network platform, with a network stack in accordance with the specific radio being used. However to gain the benefits of using TinyOS in UANT we have chosen to use TOSSIM [11], which simulates sensor network nodes on a PC. TOSSIM was created in order to support compiling TinyOS component graphs into a simulation for a PC. It utilizes a discrete event queue, reimplements some of the sensor nodes hardware, models the radio and ADC of a mote, and most importantly for UANT, uses communication services for external