# Framing and Schedule Dissemination for Multi-hop TDMA-based Wireless Networks

Gaurav Chhawchharia

Department of Computer Science
Indian Institute of Technology Kanpur

Supervisors:
Prof. Bhaskaran Raman & Prof. Dheeraj Sanghi

Thesis Defense
August 2, 2008

# Outline

# Outline

# Wireless Mesh Networks (WMNs)

In WMNs, each node acts **both** as a host and a router, forwarding packets on behalf of other nodes which are within radio range of one another.

- The above property enables to extend connectivity in hard-to-reach locations.

## Other advantages of WMNs

- Low setup cost.
- Easy network maintenance.
- Robustness and reliable service coverage.
- Technology independence.

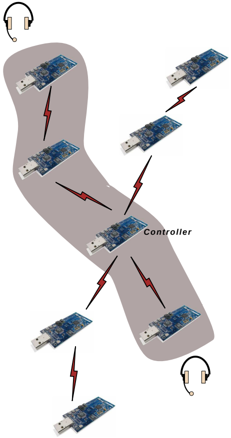
# QoS

QoS guarantees are essential in a network to support real-time audio/video flows.

- No provision for QoS guarantees in the existing CSMA/CA MAC of IEEE-802.11 and IEEE-802.15.4 based networks.
- TDMA-based MAC protocols can provide QoS guarantees:
  - Due to changing traffic requirements, dynamic scheduling needs to be supported.
  - Scheduling can be centralized or distributed.
- To provide QoS guarantees in a WMN, a multi-hop TDMA- based protocol can be used.

# $Lo^3$: Voice Over Motes [1]

## Use-case of Multi-hop TDMA-based Network using IEEE-802.15.4-based hardware platform

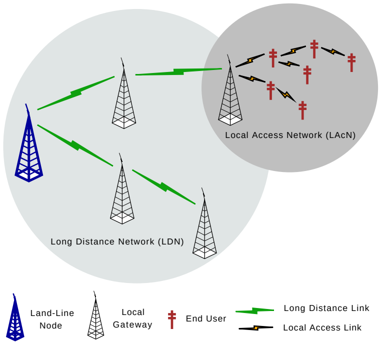



### Intended Features of $Lo^3$

- Provide communication within a village.
- Provision for multiple voice connections simultaneously.
- Motes with attached audio sensors as the hardware platform.
- Since the data rate is low, QoS guarantees are essential.

# FRACTEL: Rural WiFi Connectivity [2]

## Use-case of Multi-hop TDMA-based Network using IEEE-802.11-based hardware platform




### Intended Features of FRACTEL

- Provide connectivity to several villages.
- Low cost by using 802.11 based hardware.
- Support for real-time audio/ video and web surfing.
- Support QoS
- Tree structure with combination of short and long distance links.

# Problem Statement

*To design, implement and evaluate a centrally-controlled TDMA-based MAC protocol for multi-hop wireless networks, which incorporates a connection setup mechanism, multi-hop framing, and support for dynamic scheduling, in order to provide QoS guarantees for real-time applications.*

# Challenges

- Support for platforms with limited capabilities also. Processing, memory and power constraints.
- Quick reconstruction of routing tree in the event of node failure. End-to-end connection setup mechanism.
- Synchronization of all the clocks in the network to a central clock.
- Provision for making use of any spatial reuse available in a mesh network.
- Provision for dynamic scheduling to take into consideration the varying traffic requirements in the network.

# SRAWAN and Wi-Fi-Re

## (Sectorized Rural Area Wireless Access Network) and (WiFi Rural Extension)

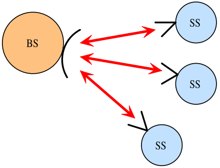



Figure: SRAWAN (Source:[3])

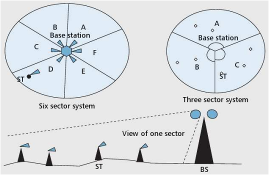
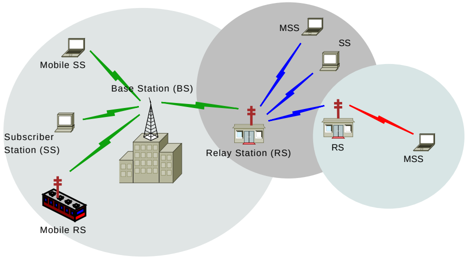



Figure: Wi-Fi-Re (Source:[4])

### Properties of SRAWAN and Wi-Fi-Re

- Applicable to single-hop based setting only.
- IEEE-802.16 style MAC on IEEE-802.11 based hardware.
- Use of sectorized antenna at BS and directional at SS.
- UL:DL duration ratio fixed in a frame in case of Wi-Fi-Re.

# 802.16j: Multi-hop WiMax




## Properties of 802.16j

- Goal is to enhance throughput and coverage using relay stations.
- Relay stations act only as routers not hosts.
- Provision for both end-to-end and link-by-link connections.
- Still in draft stage with no concrete design in place.

# BriMon: Bridge Monitoring

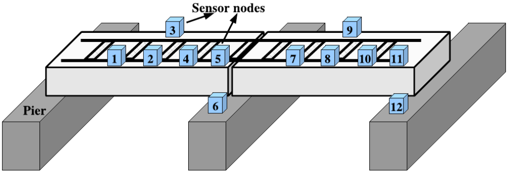



Figure: BriMon (Source: [5])

## Properties of BriMon

- Multi-hop network of sensor nodes for data collection.
- The collected data across nodes are time-aligned. Therefore, a multi-hop time-synchronization mechanism in place. We have studied and used this mechanism in our implementation, as explained later.

# Outline

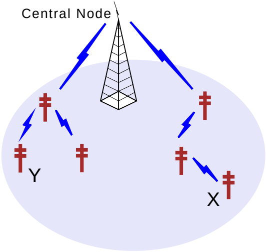
# Design: Multi-hop Framing & Schedule Dissemination

- Design not limited to any particular wireless technology, could be applied to any TDMA-based architecture.
- Design includes various protocol messages and fields in the packets.
- Many details such as number of bytes intentionally omitted. These depend on specific wireless technology.

# Requirements

- Provision for dynamic scheduling with schedule computation at a central node (schedule computation is out scope of this work).
- Multi-hop setting requires a routing tree. Routing algorithm runs using a flooding mechanism.
- Clocks need to be synchronized to a particular global clock (explained later).
- Mechanism to disseminate schedule to the nodes in the network (explained later).
- Flow and bandwidth request mechanisms required.
- Mechanism to relay data from source to destination over (possibly) multiple hops.

# Time Synchronization to a Global Clock




## Synchronize to the Fastest Clock

If X has the fastest clock, then each time upto 5 cycles of exchange of clock values are required for synchronization.

## Synchronize to a Global Clock

If global clock is that of the central node, then each time a maximum of 3 cycles of exchange of clock values are required for synchronization.

Synchronization to a global clock using the central node's clock as the global value results in faster time-synchronization.

# Schedule Dissemination

## Definition of Scheduling Element

A scheduling element specifies parameters such as transmitter address, type of packet, flowID (if required) and possibly, the channel, for one transmission in a frame. The owner of a scheduling element is the node whose address is specified in the transmitter address field. A frame may involve many transmissions.

A node on receiving the schedule, extracts the scheduling elements owned by it and also its descendants.

- It transmits a schedule consisting of scheduling elements owned by its descendants.
- It could also include its own scheduling elements in the schedule to be transmitted. Knowing when the parent and children transmit, a node could remain in low-power state at other times.

# Packet Structure

| | | |
|---|---|---|
| Generic Header | Specific Header | Payload |

- Three parts to a packet: generic header, specific header and an optional payload.
- Generic header is common to all kinds of packets whereas the specific header and the payload depends on the type of packet.

## Fields in the Generic Header

- Transmitter address
- Type of packet (this determines the specific header)
- Direction (upstream or downstream)
- CRC (for error detection)

# Type of Packets I

## Schedule and Schedule-Fragment Packets

- Carries time-sync information and start-time of the schedule.
- Size may span more than MTU, therefore, need for fragmenting.
- Payload are the scheduling elements.
- A control field is present in schedule to facilitate fragmenting.

## Flow Request and Flow Request Aggregate Packets

- Used for connection setup.
- Request specified in terms of $n$ number of bytes every $i$ interval of time along with requested flow number.
- To avoid overhead of sending many packets, aggregation done at the parent.

# Type of Packets II

## Data Packet

- Used to transfer user data between the central node and the other nodes in the network.
- Data transferred in flow, therefore, prior flow setup required.
- Source, destination addresses, flow number and length specified.

## Bandwidth Request and Bandwidth Request Aggregate Packets

- Used to clear existing transmission queues.
- Number of bytes and FlowID make up the request (FlowID is a combination of source, destination addresses and flow number).
- May also be used for terminating a flow with a special value as the number of bytes.
- Like in flow request, aggregation done here too.

# Type of Packets III

## Control Packet

- Broadcast packets which does not assume the presence of a routing tree.
- Slots specified for them are contention-based.
- Useful for application like the routing algorithm.
- Fields not specified. Can be determined on case-by-case basis.

## Tree Broadcast Packet

- Broadcast packets which assumes the presence of a routing tree.
- Nodes relay the tree broadcast packets to their descendants.
- Useful for protocols like ARP.
- Fields not specified. Can be determined on case-by-case basis.

# Other Design Aspects I

## Contention-based and Contention-free Requests

- Flow and bandwidth requests can be contention-based or slots can be allotted specifically for them. We leave this an open issue.
- Past studies have shown that contention-free mode performs better in terms of throughput and delay, and therefore, in our implementation we have used this mode.

## Central Node Reset

- Routing tree reset and existing connections terminated.
- Reset packet (subtype of the control packet) used for this.
- Central node on boot-up broadcasts the reset packet.
- A node on receiving the reset packet broadcasts it and resets its state.

# Other Design Aspects II

## Packet Error/Loss and Their Effects

- Packet losses and error treated similarly. No re-transmission mechanism in place.
- Control packet loss handled by the protocol that makes use of it.
- If any fragments of the schedule is lost, the entire schedule is considered lost. A node simply waits for the next schedule in case of a loss.
- Data packet loss handled at the higher layer.
- Loss of flow and bandwidth requests (and aggregation packets) results in them having to be re-sent.
- Loss of tree broadcast packet ignored.

# Steps in Transmitting packets

1. The node X (say) boots up and waits for a schedule packet.
2. On receiving the schedule, X synchronizes its clock and runs routing algorithm over control packet slots.
3. Once the central node receives X's routing information, X is considered to be "associated" to the network.
4. Subsequent schedule contains slot for flow request by X.
5. X sends a flow request, central node on receiving it assigns X data transmission slot.
6. X transmits according to the schedule. If its queue builds up it sends bandwidth request to clear the queue.
7. X terminates the connection by sending bandwidth request packet with a pre-decided value as number of bytes.

# Outline

# Platform Overview

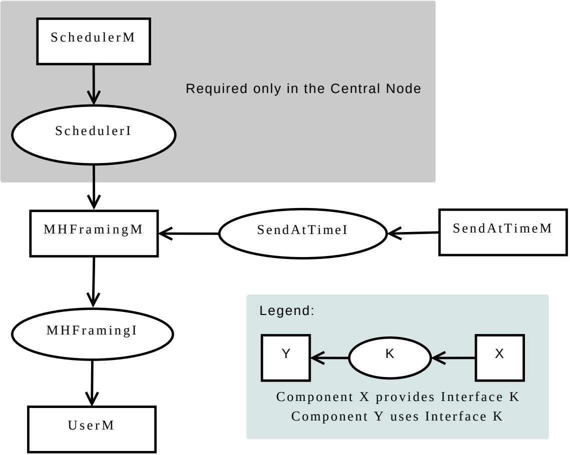
## Tmote-sky

- From Moteiv Corporation
- 8 MHz Texas Instruments MSP430 microcontroller
- 10KB RAM, 48KB flash, 1MB external flash
- 250kbps, 2.4 GHz CC2420 wireless transceiver

## TinyOS

- Operating system for motes
- Coding in nesC language
- Low memory usage
- Applications are modular consisting of components and interfaces
- Each component provides some interfaces and uses some
- Interface is a set of commands and events

# Component Diagram




*Central node not implemented

# Component Overview I

## SchedulerM

- Houses the scheduler, therefore, required in central node only.
- Needs to have access to the information on existing flows.
- Flow and bandwidth requests received by the central node are passed to this component.
- Since the central node is not implemented, this component is also not implemented.

# Component Overview II

## MHFramingM

- Principal component: co-ordinates the working of other components.
- Initializes state and buffer variables on boot-up.
- Handles received packets.
- Handles service requests from UserM.
- Stores information on existing connections.
- Checks periodically for inactive connections and terminates them.
- Schedules transmissions as per the received schedule.

# Component Overview III

## SendAtTimeM

- Receives data and management information from MHFramingM, forms packets out of it and transmits over the radio.
- If the packet size exceeds the MTU size, multiple packets are sent one after other.

## UserM

- It is the higher layer interface.
- Sends/receives data, setup/terminates flows.

# Time-Synchronization

- Time-synchronization mechanism is borrowed from BriMon.
- An offset variable maintained at each node which stores the difference between the local and global clocks. Using this offset, the global time at any instant can be calculated.
- Schedule carries sender's timestamp as well as the sender's offset value. Offset value at central node is 0.

receiverOffset=(senderTStamp+senderOffset)-receiverTStamp
globalClock=localClock+receiverOffset

Difference from BriMon: In BriMon if a node misses an update, it uses previous five updates to come up with the next. We do not have any such mechanism.

# More Details

## Disabling Random Backoff

Random backoff was disabled so as to enable transmission at a specific time.

## Routing

Routing protocol not implemented. Routing information hard-coded in each node.

## Termination of Idle Connections

Idle count maintained for each flow. On receiving a schedule, idle count of all flows incremented by 1. On transmission for a flow, idle count of that flow set to 0. If idle count reaches a threshold value, connection terminated.

# Experiment Setup

## Definition of a Frame

A frame is the number of bytes transmitted cumulatively, by all the nodes in the network (including the central node), from the start-time of a schedule till the start-time of the next schedule. A Frame consists of many packets.

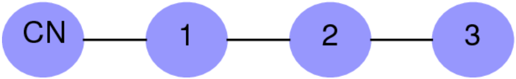


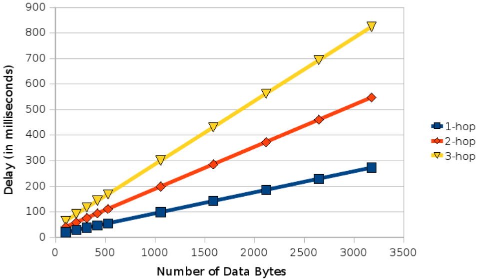

Figure: Network topology used for evaluation

1-hop, 2-hop and 3-hop experiments conducted where data sent from central node to hop-1, hop-2 and hop-3 nodes respectively.

# Procedure

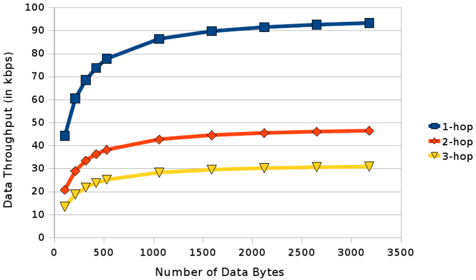
- Calculation of optimal throughput
  - Measurements carried out in the absence of the implementation of our protocol.
  - Experiments carried out over single hop only.
  - Multiple packets sent continuously.
  - Optimal system throughput (inclusive of overhead) = 125 kbps
  - Optimal data throughput (considering only data bytes) = 103 kbps
- Measurement of delay in 1-hop, 2-hop and 3-hop experiments while varying the data to be transferred from 106 to 3180 bytes.
  - Multiple frames sent one after the other and the total time averaged out to get the delay value per frame.
  - Value of delay used in calculation of data and system throughputs.
  - Hidden node scenario created while taking measurements.

# Delay



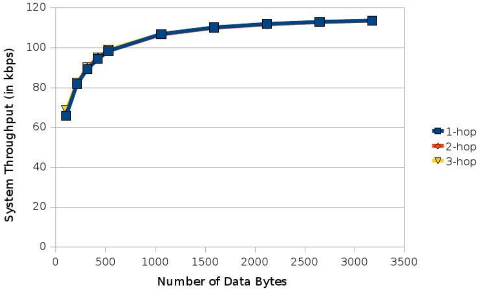

- For a particular destination, delay increases linearly with the increase in the number of data bytes per frame.
- For a particular number of data bytes in a frame, delay increases linearly with increase in the number of hops.
- Expected behaviour achieved.

# Data Throughput




- For a particular destination, data throughput increases non-linearly with increase in the number of data bytes per frame.
- For a particular number of data bytes in a frame, data throughput decreases with increasing hop count.
- Maximum data throughput achieved is 90% of optimal value.

# System Throughput




- Calculated for total number of bytes transmitted in a frame.
- Processing done on per flow basis, therefore, with increase in number of bytes in a frame the ratio of processing decreases increasing the throughput.
- System throughput constant for increasing number of hops.

# Evaluation Results

- Coarse-grained scheduling (more data bytes per frame) better than fine-grained scheduling in terms of throughput but trade-off for delay.
- Throughput of 13 - 30 kbps achieved over 3 hops. GSM quality voice call could be supported.
- Managed to achieve system and data throughputs close to the optimal value.
- Observed values specific to mote platform but pattern expected to remain the same across different wireless technologies. Shape of the graphs are expected to remain the same.

# Outline

# Feasibility of Implementation on WiFi

The answers to the following determine the feasibility of an implementation using 802.11 hardware.

- Is it possible to implement multi-hop time-synchronization to a central node using the existing 802.11 hardware? The synchronization has to be tight (sub-packet duration).
- Can we achieve throughput enough to support multiple real-time audio/video flows?

# Multi-hop Time-Synchronization to a Central Clock I

## Time-Synchronization in Adhoc Mode as per IEEE-802.11

- Multi-hop networks possible in adhoc mode.
- The standard specifies a maximum of $4\mu s$ skew across network.
- Distributed beaconing mechanism used for time-synchronization.
- At every beacon interval, one randomly chosen node transmits a beacon.
- Local clock re-synchronized if the received value is older than the local value.

# Multi-hop Time-Synchronization to a Central Clock II

## Our Approach

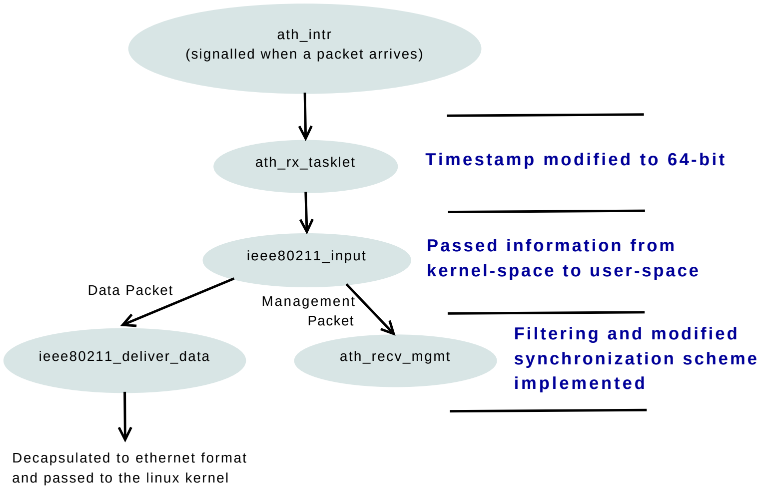
- Modifications done to the adhoc mode.
- Each node is assumed to know the MAC address of the parent.
- Beacon accepted if received from the parent only.
- If beacon accepted and the sender's and receiver's timestamps differ:
  - The receiver's clock is reset (to zero).
  - When next beacon arrives, sender's clock is older than receiver's clock and the routine adhoc mode synchronization mechanism is followed.

# Multi-hop Time-Synchronization to a Central Clock III

## Implementation Details

- Implementation carried out on open-source MADWiFi driver on Linux OS.
  - Linux driver for 802.11a/b/g chipsets manufactured by Atheros Communications.
  - Two layered MAC: proprietary HAL and open-source net80211 stack hacked from FreeBSD.
- Implementation done on MADWiFi version 0.9.3.2/HAL 0.9.18.0
- Tested on kernel 2.6.11/2.6.20/2.6.22
- MAC filtering, display of timestamp and sequence number, and modified synchronization scheme added.

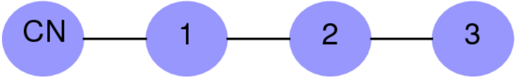
# Multi-hop Time-Synchronization to a Central Clock IV

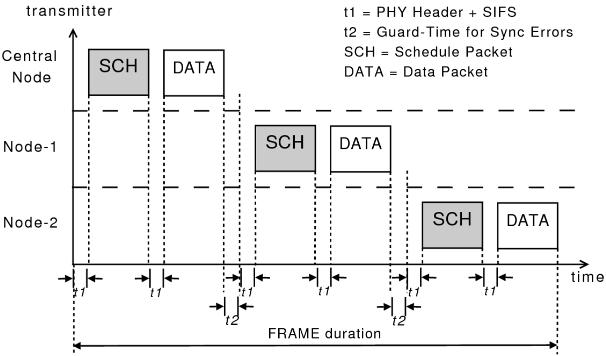


# Assumptions for Calculating Achievable Throughput

- Use of 802.11b hardware
  - Minimum/Maximum bit-rate = 1/11 Mbps
  - SIFS = 10$\mu s$
  - PHY Header = 192$\mu s$
- No random backoff
- No propagation delay
- Data sent at 11 Mbps
- Schedule sent at 1 or 11 Mbps. If Schedule is sent at 1 Mbps there is less chance of it being lost.
- Guard time = 25$\mu s$ to account for synchronization error.
- UDP payload = 1400 bytes

# Assumed Topology and Packet Sequence in a Frame



# Achievable Throughput

- Up to 8.8 Mbps over single hop (Even better than the existing CSMA/CA MAC).
- Varies from 1.64 to 2.94 Mbps over three hops. This is good enough to support multiple video flows at 384 kbps.
- Throughput doesnot drop significantly if the schedule is sent at 1 Mbps and data is sent at 11 Mbps. This option should be considered to increase reliability.
- Implementation of the protocol is feasible.

# Outline

# Conclusion

- Designed, implemented and evaluated a multi-hop framing and schedule dissemination mechanism.
- Measured throughput and delay for various frame sizes during the evaluation.
  - As frame size increases, throughput increases
  - Trade-off between throughput and delay.
  - Throughput inversely proportional to number of hops.
- Achieved throughput up to 90% of the optimal value. Good enough to support multiple GSM quality voice calls.
- Conducted studies and verified that an implementation using WiFi hardware is feasible.
  - Implemented a multi-hop time-sync mechanism.
  - Theoretically computed achievable throughputs.

# Future Work

- Optimization of our implementation for throughput gains by reducing size of various fields in the headers.
- Evaluation with upstream traffic and variable bit-rate traffic.
- Evaluation with multiple flows in a complex non-linear network topology.
- Design and implementation of push-to-talk application for motes using our protocol as the underlying MAC.
- Implementation using WiFi hardware.

Thank You!

# References I


Bhaskaran Raman *et al.*
$\text{Lo}^3$: **Lo**w-power, **Lo**w-cost, **Lo**cal Voice and Messaging for Rural Regions, In Preparation.
http://www.cse.iitb.ac.in/silmaril/br/doku.php?id=pro

Kameswari Chebrolu and Bhaskaran Raman.
FRACTEL: A Fresh Perspective on (Rural) Mesh Networks.
In *ACM SIGCOMM Workshop on Networked Systems for Developing Regions (NSDR'07)*, Aug 2007.

Pavan Kumar.
Design, Implementation, and Evaluation of new MAC Protocols for Long Distance 802.11 Networks.
Master's thesis, IIT Kanpur, May 2006.

# References II


Krishna Paul, Anitha Varghese, Sridhar Iyer, Bhaskar Ramamurthi, and Anurag Kumar.
WiFiRe: Rural Area Broadband Access Using the WiFi PHY and a Multisector TDD MAC.
*IEEE Communications vol.45(1)*, Jan 2007.

Kameswari Chebrolu, Bhaskaran Raman, Nilesh Mishra, Phani Kumar Valiveti, and Raj Kumar.
BriMon: A Sensor Network System for Railway Bridge Monitoring.
In *ACM MobiSys*, Jun 2008.