# **ENS** <u>Center for Embedded Networked Sensing</u>

# Non-Intrusive Analysis of Sensor Network MAC Protocols





Tyler McHenry, John Heidemann

## ISI Laboratory for Embedded Networked Sensing Experimentation - http://www.isi.edu/ilense

Introduction: Some experiments require MAC-layer analysis of live sensor networks

### **Example: High-Density Sensor Networks**

- High density networks cannot be simulated reliably
  - As potential traffic increases, the actual collision rate must be measured rather than modeled statistically to discover realistic MAC performance In real networks, packets do not collide or corrupt at a known rate.
  - Low-level MAC layer *timings* need testing in light of real collision rates.

#### **Needs/Requirements:**

- Software to collect and analyze the behavior of active sensor networks
  - Must not interfere with the operation of the network
  - Must be able to deliver state information which is not explicitly communicated between nodes

# **Problem Description: MAC-Level analysis is hindered by the addition of reporting code**

• Extra traffic generated

**Important MAC timings affected** 

**T7 T1** 

- Sending reporting data over the network requires either
  piggybacking it on normal packets or creating entirely new
  packets for debugging purposes, increasing traffic on the network.
- Plus, the delivery of this data depends on the *reliability* of the network, which may be a variable!
- **Tethered nodes** 
  - Sending reporting data over backchannels restricts the placement of nodes and cannot be used in an existing deployment.
- In S-MAC, the extra processing and transmission time necessary to add extra debugging information to SYNC packets causes the S-MAC period to elongate by 1.04 ms, which may affect performance under dense or heavy-use conditions.
- Inability to detect some collisions
  - Relying on the source to report its transmissions ignores collisions entirely.
  - Relying on the sink to report its receptions will still fail to capture *"total loss" collisions* (in which preambles are corrupted), without modifying the MAC.

#### Key Idea:

Completely avoid modifying all mote-side software

#### **Proposed Solution:** A snooper mote can collect data to be processed later with Ethereal addons

#### The "Radio Traffic Analysis" (RTA) Suite

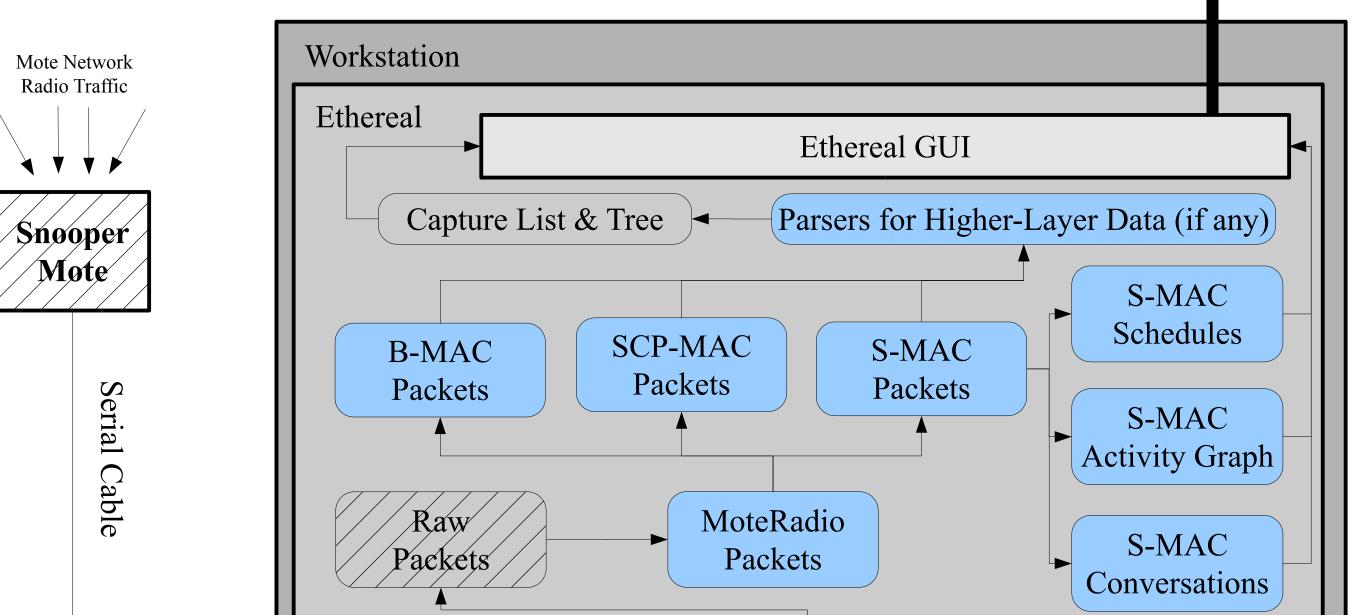
#### Data collection based on snooper motes

- Constantly listens using the same physical layer as the MAC being analyzed, but need not implement the MAC protocol itself.
- Echoes every byte it overhears to a Linux host over some backchannel (most simply, a serial cable). Other motes do not even know it exists.
- Can also monitor for "total loss" packet collisions by detecting changes in RSSI.
- Packet-level processing emulates a Linux network interface
  - The *moteradio driver* accepts bytes from a snooper mote and presents them as whole packets on a network interface with a microsecond timestamp added.
  - Any program can read this interface as it would read any other network card.
- Reporting and analysis software are addons to Ethereal
  - A familiar and widely-used network traffic analysis package
  - Provides a portable framework and familiar GUI that can capture packets from any network interface and feed them to custom processing code depending on the type of packet captured.

### The Ethereal GUI with RTA Modifications

smac-experiment-run1-38nodes - Ethereal	111111111			1-1-1-1	11111111	1-	1111111111	1110700	00	
<u>File E</u> dit <u>View Go</u> <u>Capture Analyze Statistics</u> <u>H</u> elp										
	■ 🔿 轮 🥇			€ (	२, 🔍 🖭		5 💥 🔯			
Eilter:	▼ <b>4</b> <u>E</u> xp	ression 💡	<u>s C</u> lear	<u>A</u> pply						
No Time Source Destination	Protoco	l Info								
5790 404.100022 Node 19 Broadcast					ation Data					
5791 404.149026 Node 35 Broadcast					ation Data					
5792 404.199012 Node 3 Broadcast					ation Data					
5793 404.255001 Node 15 Node 12	S-MAC Request-to-Send									
5794 404.274003 Node 12 Node 15 5795 404.329997 Node 15 Node 12	S-MAC Clear-to-Send S-MAC Linkstate Data									
5796 404.329997 Node 13 Node 12 5796 404.347993 Node 12 Node 15	S-MAC Acknowledgement									
5797 404.395982 Node 11 Node 13	S-MAC Request-to-Send									
5798 404.413984 Node 13 Node 11	Conversation				odes	+++++++++++++++++++++++++++++++++++++++	+++++++++++++++++++++++++++++++++++++++	1111	· O ×	
5799 404.470978 Node 11 Node 13										
▷ Frame 5792 (19 bytes on wire, 19 bytes captu	Ethernet Fib	re Channel F	DDI IPv4	IPX S	CTP S-MAC: 17	1 TCP Toke	n Ring UDP W	LAN		
▼ Mica-2 Mote Radio										
Time Received by Kernel: 476867787	S-MAC Conversations									
Packet Length: 10	Address A	Address B	Packets *	Bytes	Packets A->B	Bytes A->B	Packets A<-B	Bytes A<-B		
CRC-16 Checksum: 0×1e9f	Node 25	Node 32	90	3240	34	646	56	2594	-	
▼ Sensor MAC	Node 4	Node 37	89	3401	38	722	51	2679		
0100 = Message Type: Sync Packet (4)	Node 13	Node 15	88	3292	49	2551	39	741		
0000 = Message Subtype: 0	Node 16	Node 17	88	3562	42	798	46	2764		
Source Address: 3	Node 27	Node 30	88	3382	39	741	49	2641		
State: 2	Node 4	Node 7	86	3434	43	817	43	2617		
Sequence Number: 5	Node 24	Node 30	86	3524	45	2745	41	779	-	
Next Sleep Time: 503										
0000 7e cb 6c 6c 1c 00 00 00 00 0a 40 03 00 0		Copy								
0010 01 9f 1e	🛛 Name reso	lution								

- New *dissectors* (packet format definitions) are easily created for new MAC protocols or higher-level protocols
  - Packets progress through a tree of analysis code from physical layer to MAC layer to higher layers (if applicable).
  - Adding a new dissector for a new MAC or higher layer protocol is straightforward.
  - Dissectors currently exist for B-MAC, S-MAC, SCP-MAC, the SMACTest application, and Linkstate data.
- Designed from the bottom up to be as portable as possible
  - Ethereal is already portable to Linux, Windows and more.
  - Porting to a different version of Linux or a new platform entirely necessitates modifying or replacing the moteradio driver only.



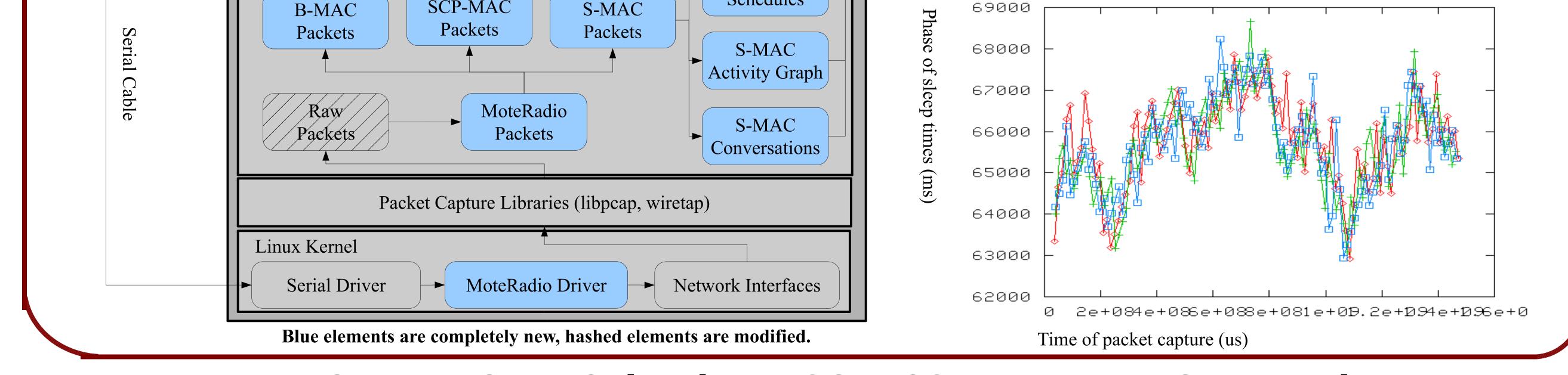
#### Type is the major type of the S-MAC packet (smac.type), 1 byte

•

Screenshot of Ethereal displaying an S-MAC SYNC packet and a list of point-topoint conversations determined from this flow of packets

#### **Goal: Automatic Adaptability in MAC Analysis**

- Each protocol with a dissector may provide more complex data analysis
  - For S-MAC, RTA provides: point-to-point conversation tracking, enumeration of schedules and schedule tracking, and node activity graphs.
  - Each of these requires intensive post-processing of captured packets.
- Analysis provided by the dissector should not make assumptions
  - A key use of Radio Traffic Analysis is testing the effectiveness of modifications to protocol parameters.
  - Dissectors should *detect* parameters whenever possible.
- Example: S-MAC sleep/listen cycle length (period) and schedules
  - The S-MAC dissector *detects* the period in use within some margin of error and is able to determine that 3 nodes are on the same schedule
  - It can then constrain the timing of sleeps in that schedule within a +/- 5ms window for prediction of future sleep times.



X Close

#### UCLA – UCR – Caltech – USC – CSU – JPL – UC Merced