WisperNet

Anti-Jamming for Wireless Sensor Networks

Miroslav Pajic and Rahul Mangharam
Department of Electrical and Systems Engineering
University of Pennsylvania

Penn Engineering

mLAB Real-Time & Embedded

What this talk is about?

Jamming **and** **Anti-Jamming**

Outline

- **Jammers and trade-offs with jamming**
- **Anti-jamming:**

  Coordinated Spatio-temporal Randomization

  ➔ WisperNet-Time

  ➔ WisperNet-Space
- **Performance analysis**
- **Implementation Experiences**

Jamming

- Jamming – radiation of electromagnetic energy
  - loss of link reliability,
  - increased energy consumption,
  - extended packet delays,
  - disruption of end-to-end routes
- Malicious and non-malicious
  - The hard part: distinguishing and concealing
- Anti-jamming
  - Resilience to jamming and its avoidance
  - Must be native to the communication protocol

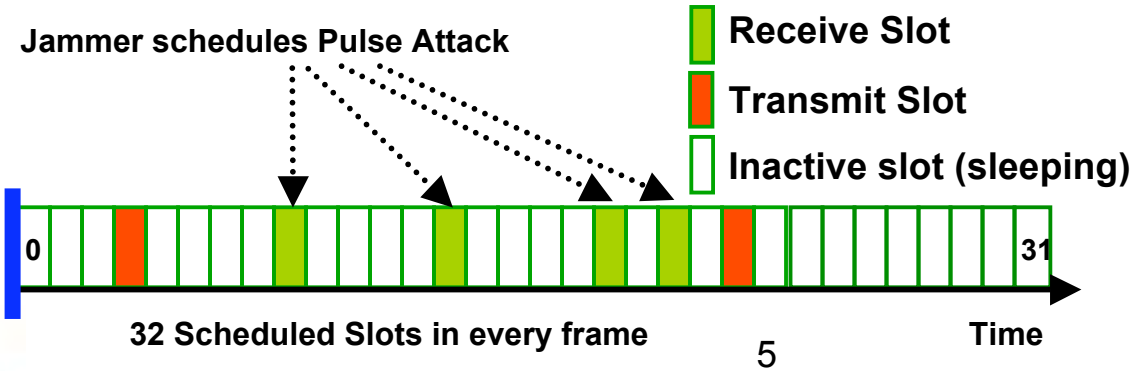
What is so special about Wireless Sensor Networks?

- Nodes scheduled to maximize the common sleep duration and coordinate communication

  → **Temporal patterns** in communication

  → **Predictable intervals** of transmission activity

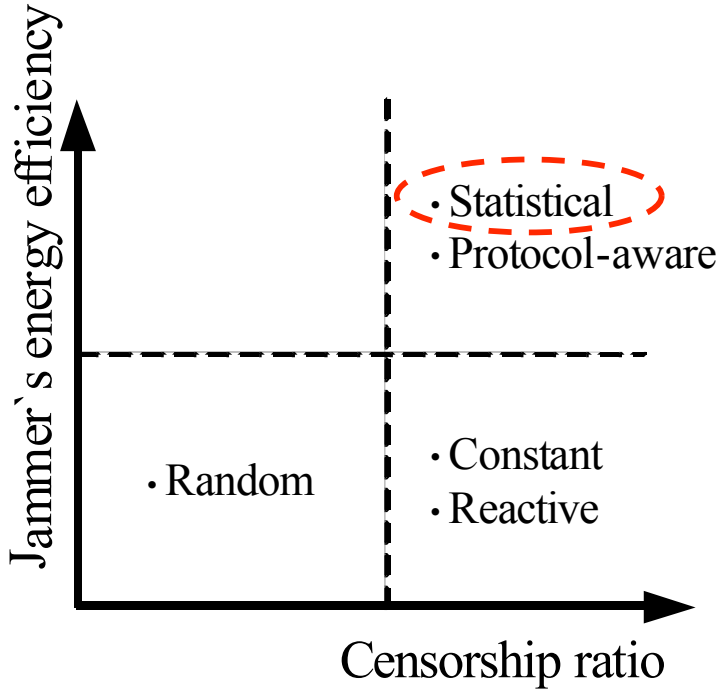
- Efficient to scan and jam only during activity
- Jamming Competitive Ratio



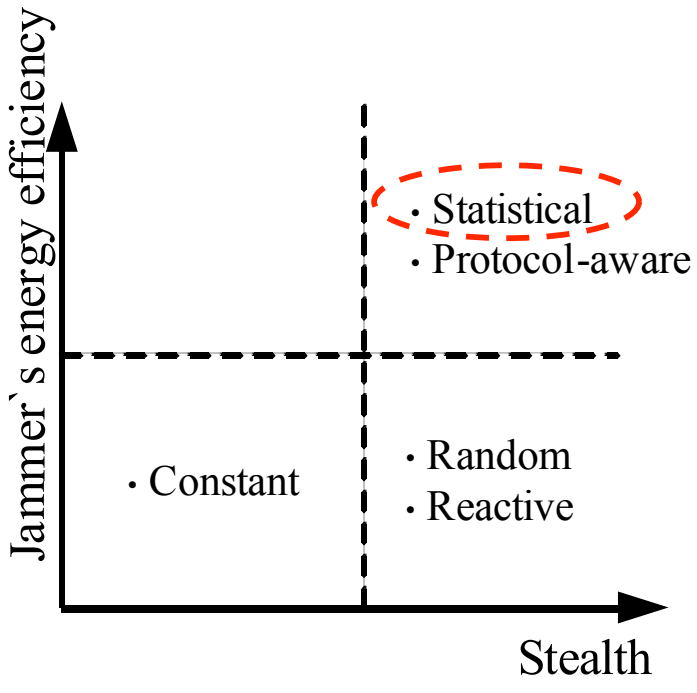
Properties of anti-jamming protocol

1. Non-predictable schedules
2. Non-predictable slot sizes
3. Coordinated and scheduled transmission
4. Coordinated changes of slot sizes
5. Collision-free transmission

Comparison of Jamming Models



Jammer's Energy Efficiency vs. Censorship Ratio

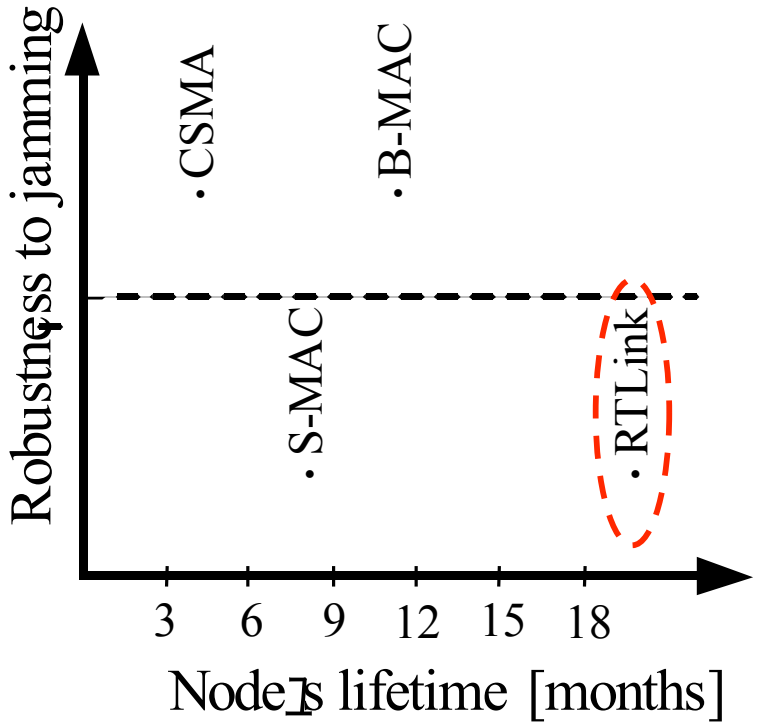


Energy Efficiency vs. Stealth

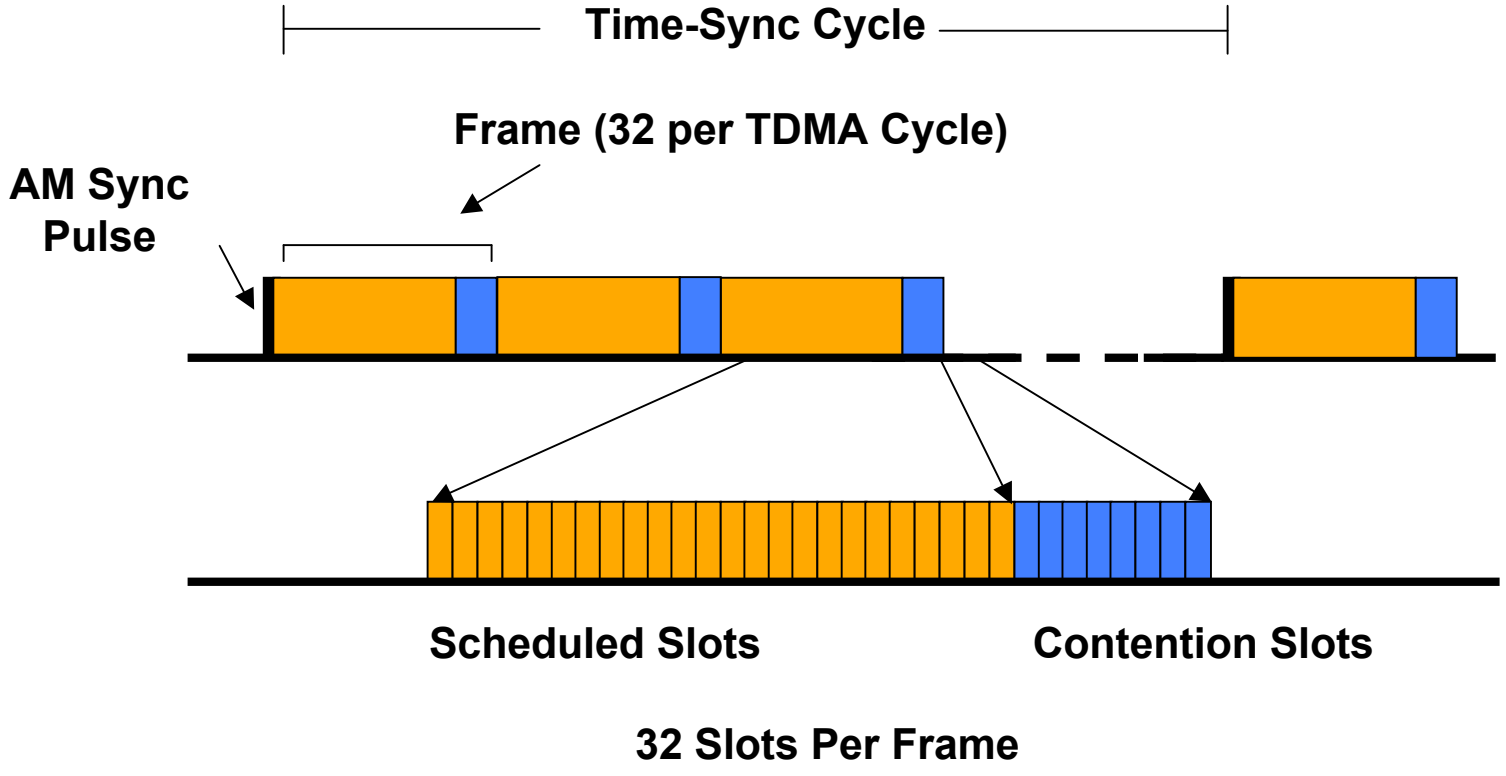
Impact of Jamming on MAC Protocols

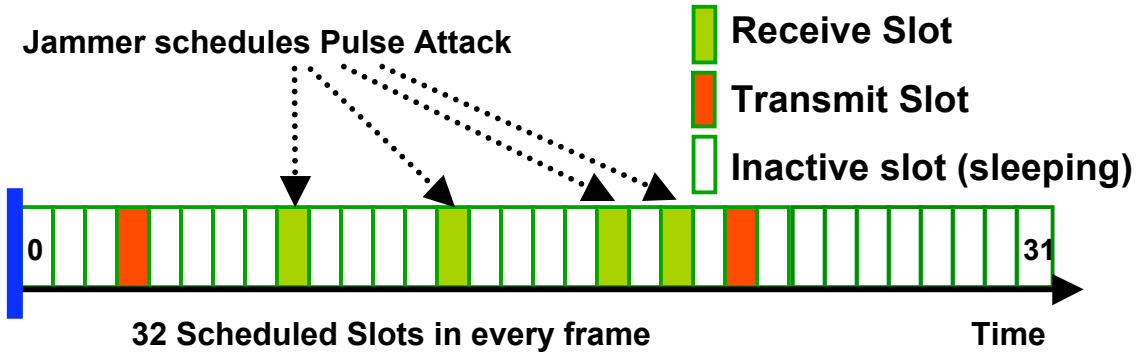
Energy-Efficient MAC protocols

- Asynchronous protocols (B-MAC)
- Loosely-synchronous (S-MAC)
- Synchronous protocols (RT-Link)



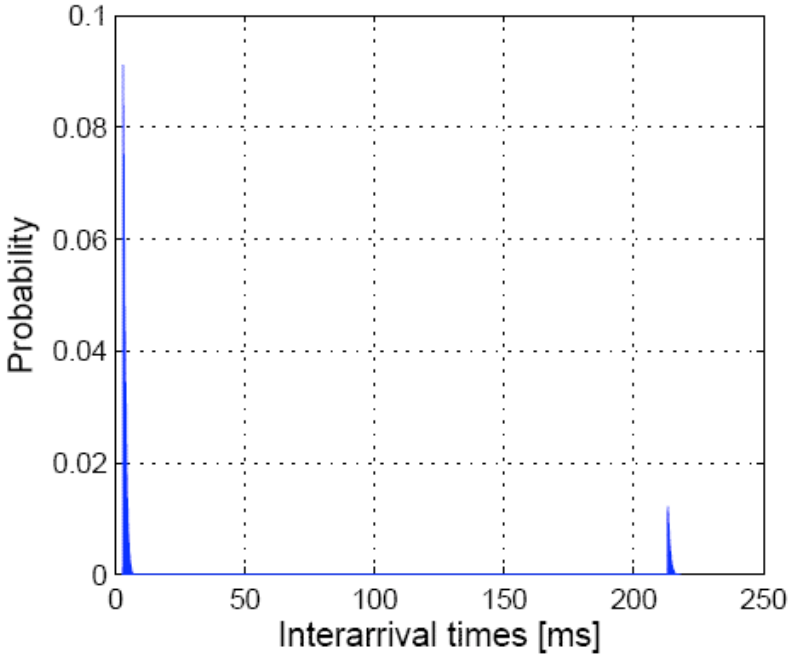
RT-Link: Real-Time Link Protocol



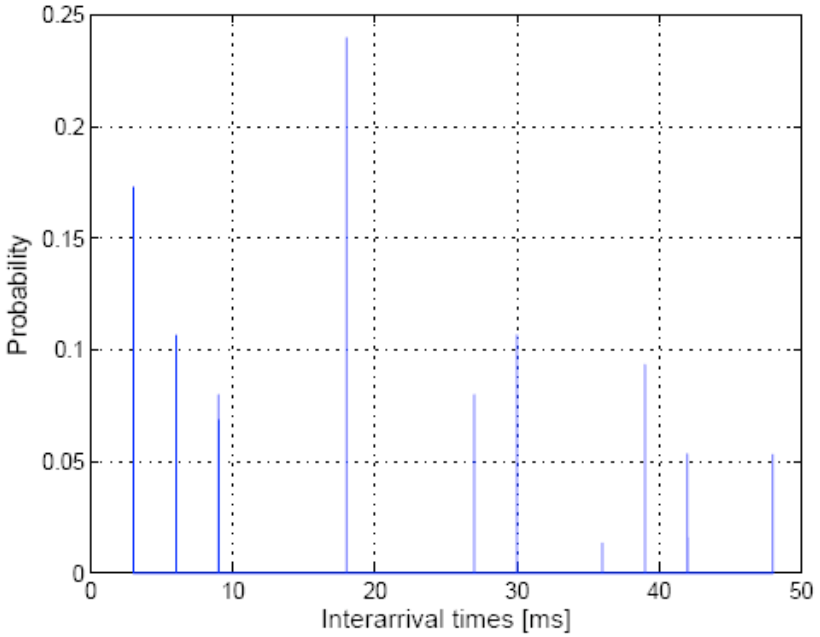


Statistical jamming

Histogram: S-MAC



Histogram: RT-Link



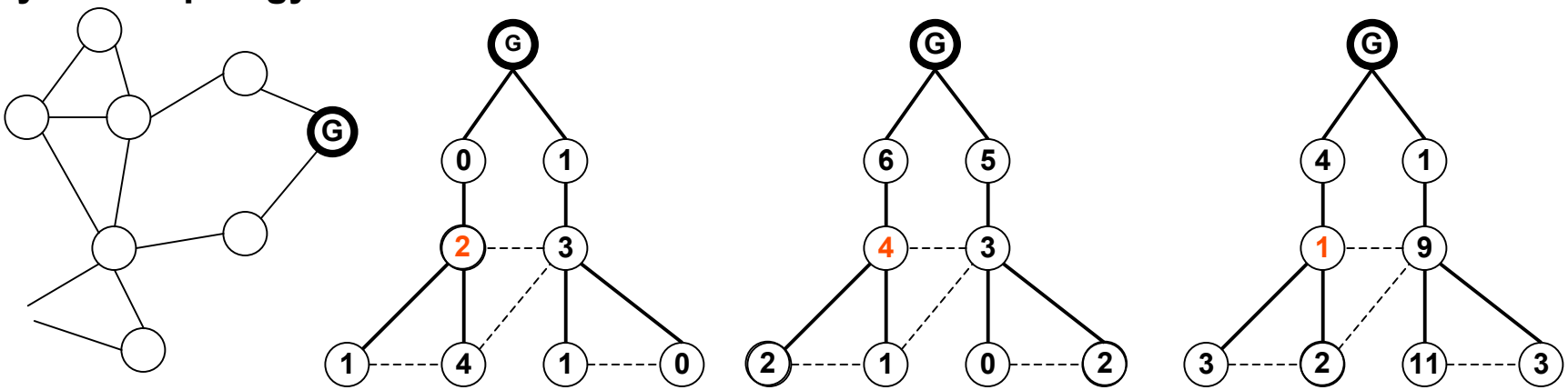
10 nodes in network, average transmission 3ms

Assumptions

- Jammer is as energy constrained as a legitimate node
- Jammer is not protocol aware
- Both malicious and non-malicious are considered
- Transmission power 0dBm
- Interference range is equal to transmission range (1 hop)

Schedule Randomization

Physical Topology

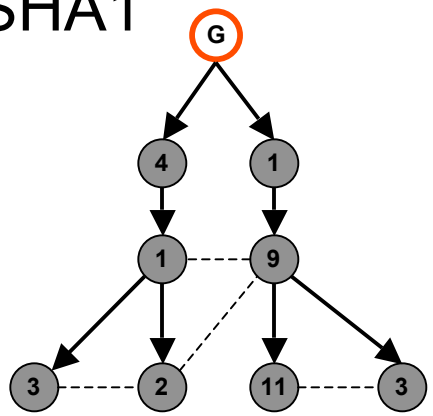


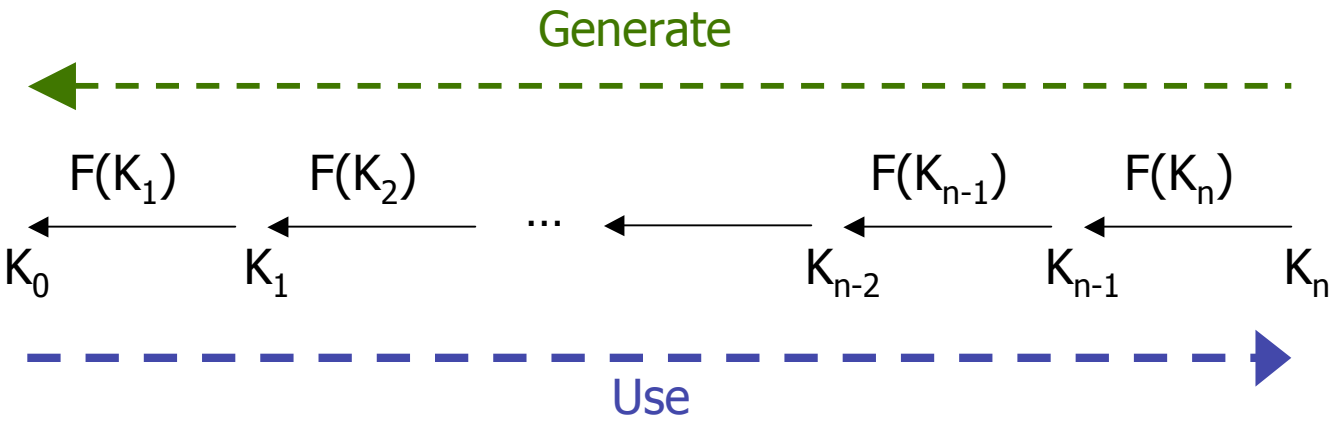
Scheduled Frame *i* **Frame *i* +1** **Frame *i* +2…**

- Gateway distributes new keys periodically to each node (~5secs)
- Slot schedules based on SHA1_HMACkey (nodeID)
- Non-repeating schedules
- Tight time synchronization between nodes
- Schedules must not cause packet collisions

HMAC Schedule Randomization

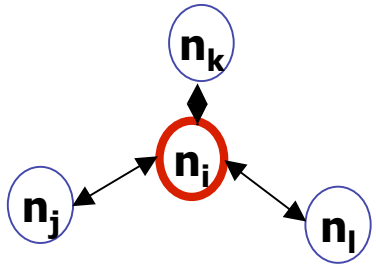
- Gateway generates One-way key chain using SHA1
  - One key per 5 sec cycle



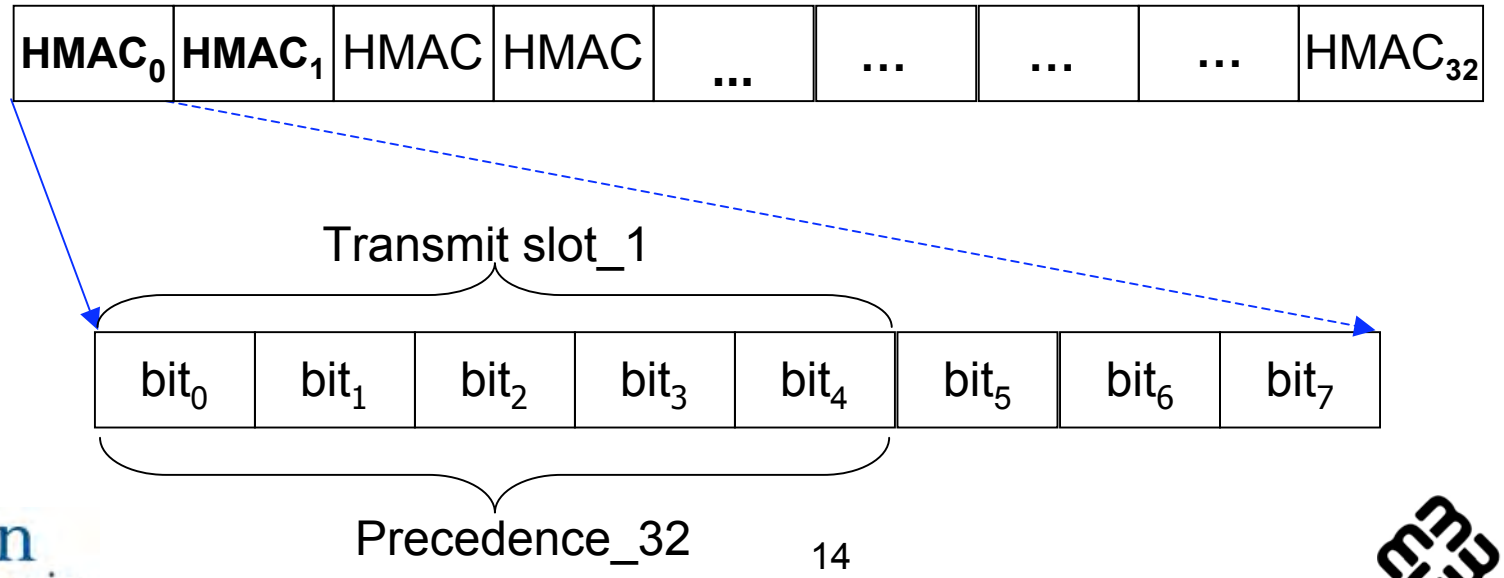


Schedule based on SHA1_HMAC

- Each node computes (using SHA1_HMAC):
  - its **own transmit slots** and their **precedence**
  - its neighbors' transmit slots and their precedence
  - Collision-free operation

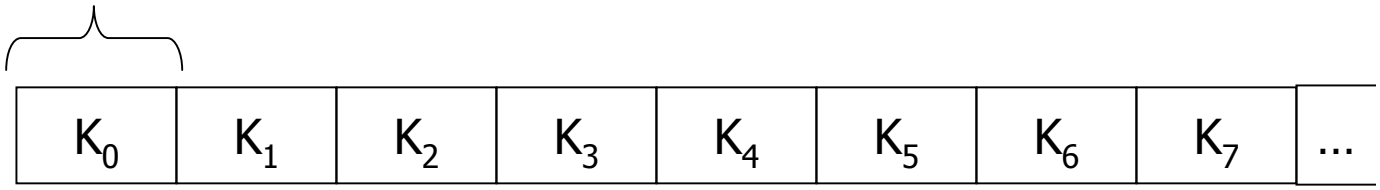


SHA1_HMAC output:



Implicit Schedule Conflict Resolution (1)

Valid for 5 secs



Gateway broadcasts

Node determine their schedules

n_i : $HMAC_{K_0}(n_i) \rightarrow (Tx_{ij}, TxP_{ij}); (Tx_{ik}, TxP_{ik}); \ldots$

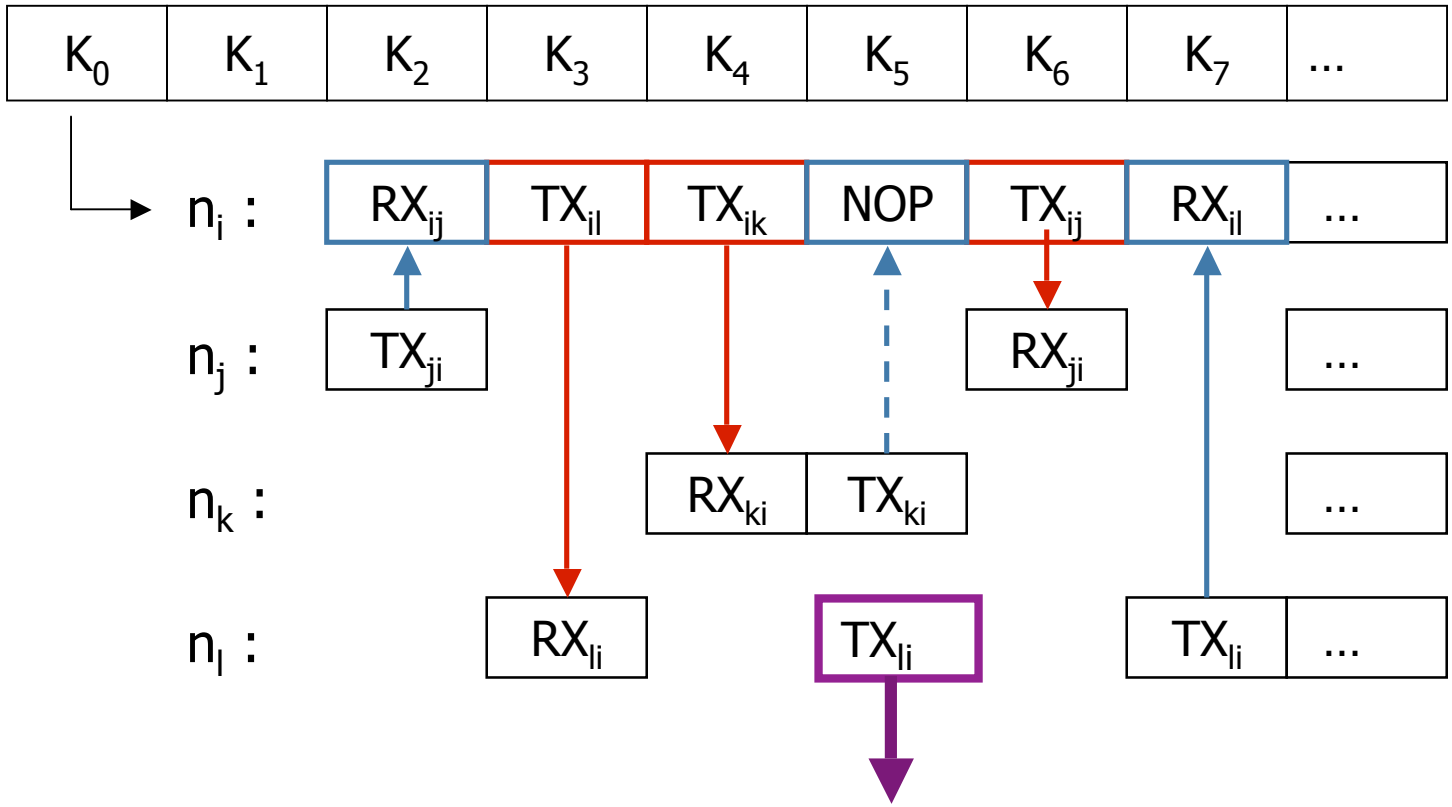
n_j : $HMAC_{K_0}(n_j) \rightarrow (Tx_{ji}, TxP_{ji}); \ldots$

n_k : $HMAC_{K_0}(n_k) \rightarrow (Tx_{ki}, TxP_{ki}); \ldots$

n_l : $HMAC_{K_0}(n_l) \rightarrow (Tx_{li}, TxP_{li}); \ldots$

Implemented SHA1 and HMAC_SHA1 in 8-bit fixed-point

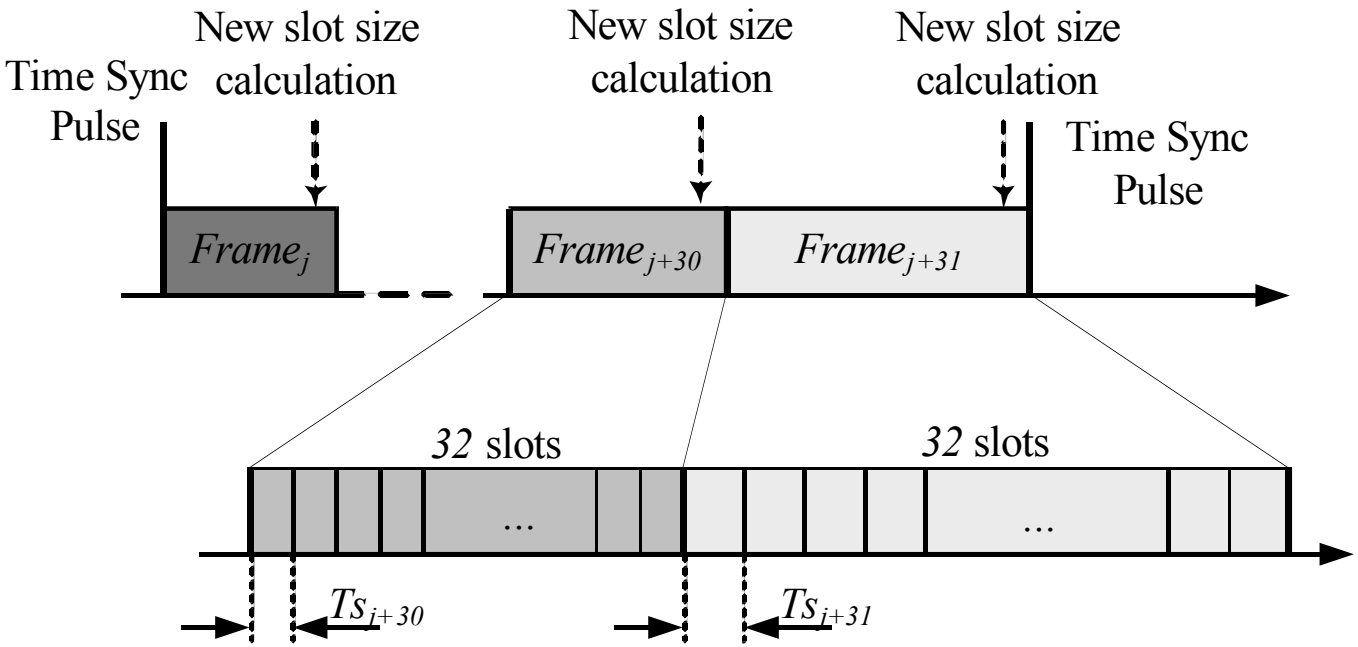
Implicit Schedule Conflict Resolution (2)



Slot size randomization

- Slot size as function of output (K_{slot} is a local key)

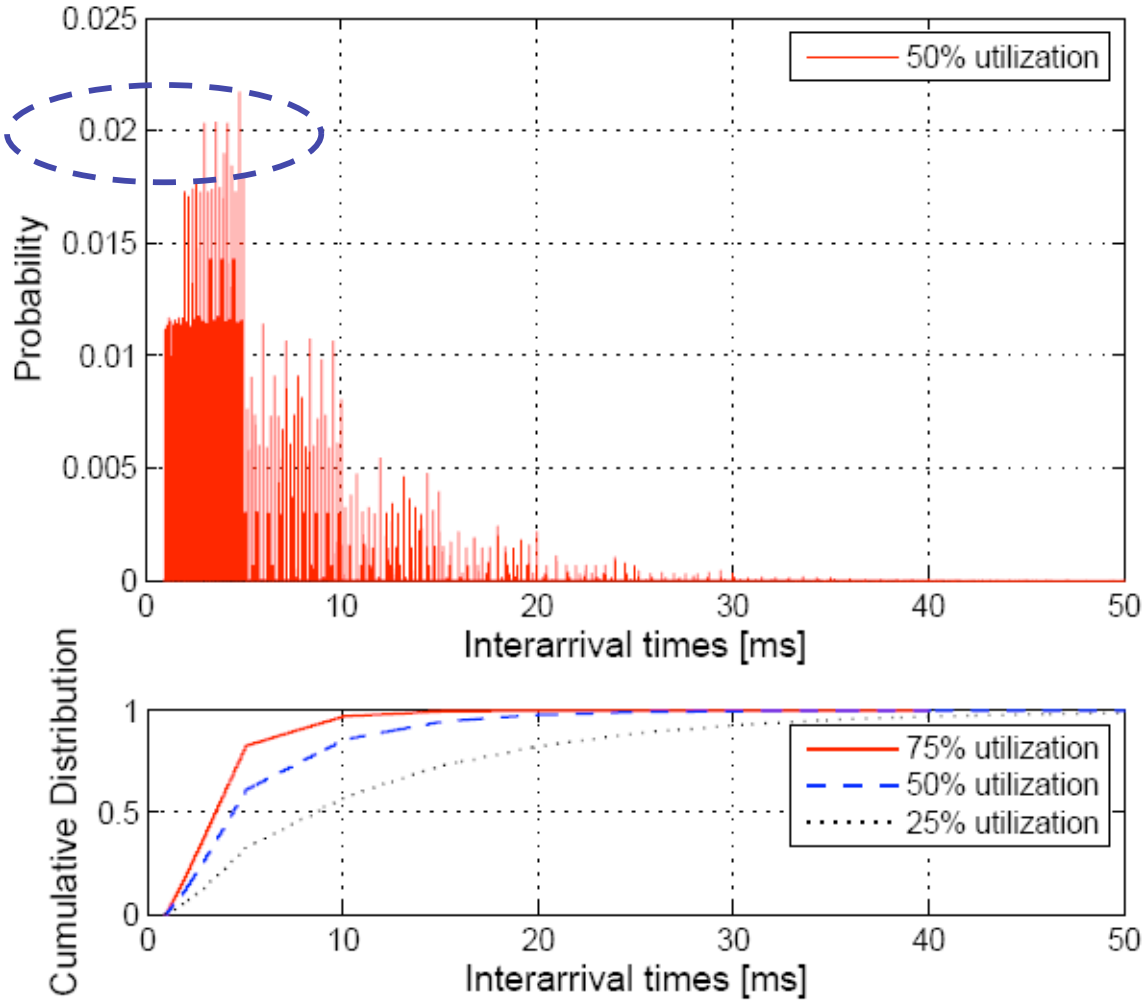
$$HMAC(cnt, K_{slot})$$



Demo

WisperNet-Time: Performance Analysis

Effect of Channel Utilization

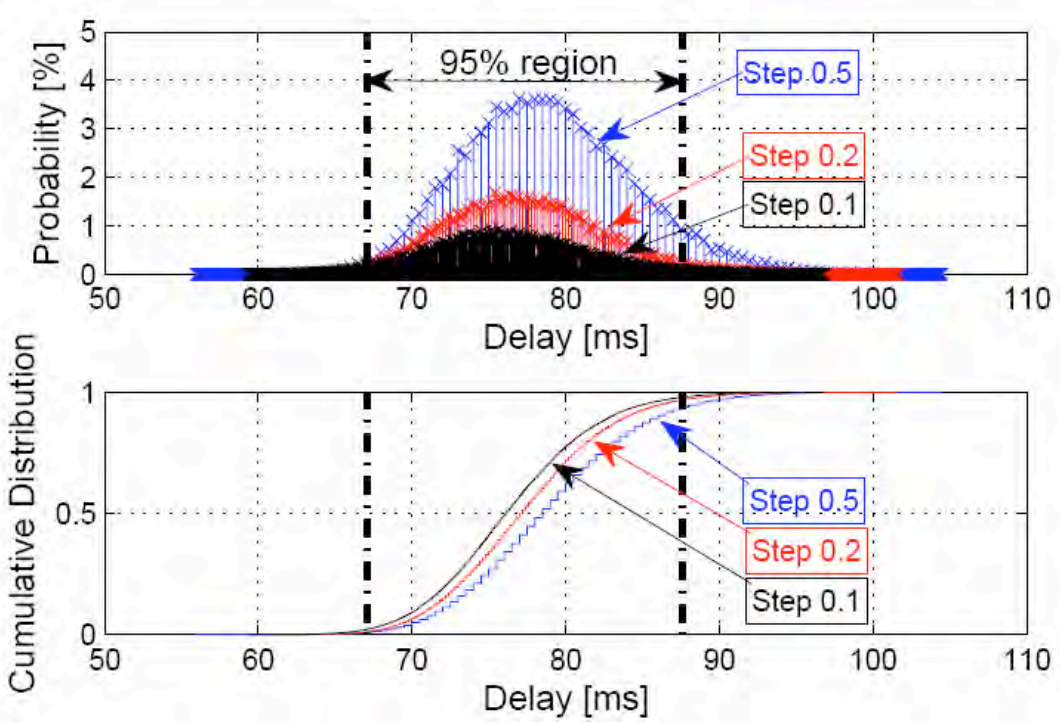


PDF

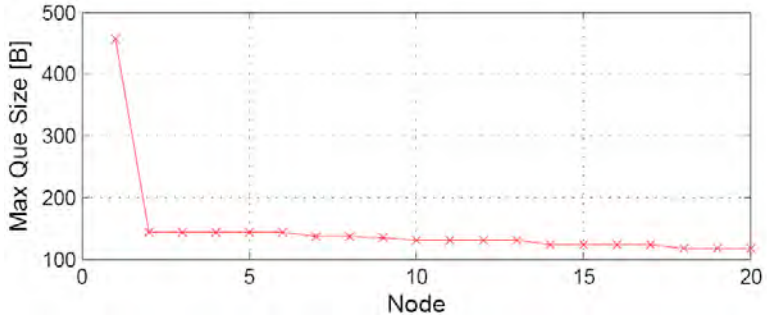
CDF

Effect of Slot size Randomization

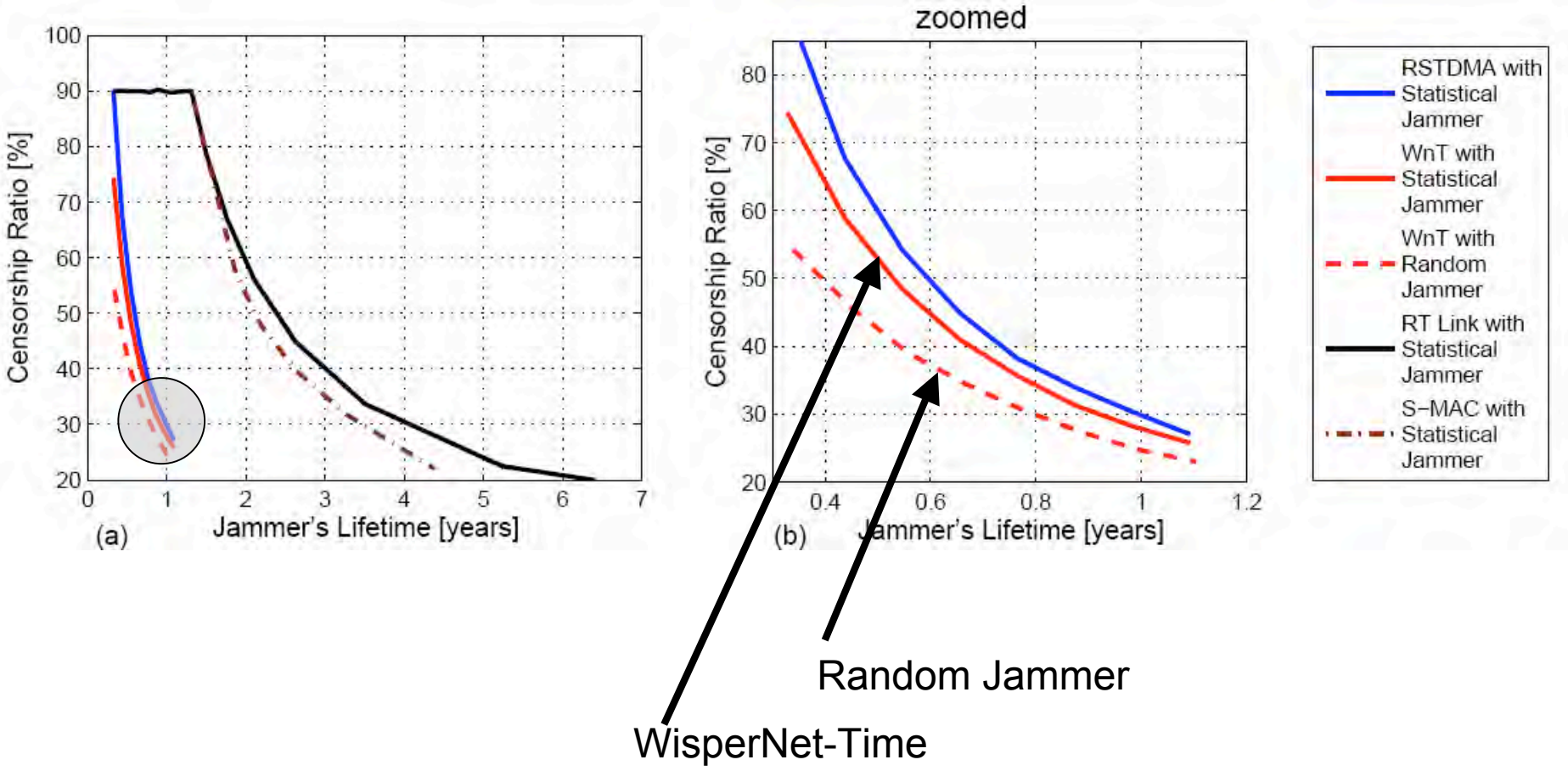
End-to-end delay



Memory requirements



Jammer's Lifetime and Censorship Ratio



WisperNet-Space: Coordinated Spatial Adaptation

- Each link $e = e(u, v)$, associated with *k* weights,

$$w_j(u, v), \quad j = 1,2... \ k$$

- For a tree *T*, the aggregate weight is defined as

$$W_j(T) = \sum_{e \in T} w_j(e), \quad j = 1,2...k$$

WisperNet-Space – Network's reliability

- Reliability of a path *P* $\prod_{e \in P} PDR(e)$

- Maximizing reliability equivalent to maximizing

$$\ln\left(\prod_{e \in P} PDR(e)\right) = \sum_{e \in P} \ln(PDR(e)).$$

WisperNet-Space – Network's reliability consideration

- RW for some link $e = e(u, v)$ is defined as

$$w_r(u,v) = \left|\ln(PDR(u,v)\right|$$

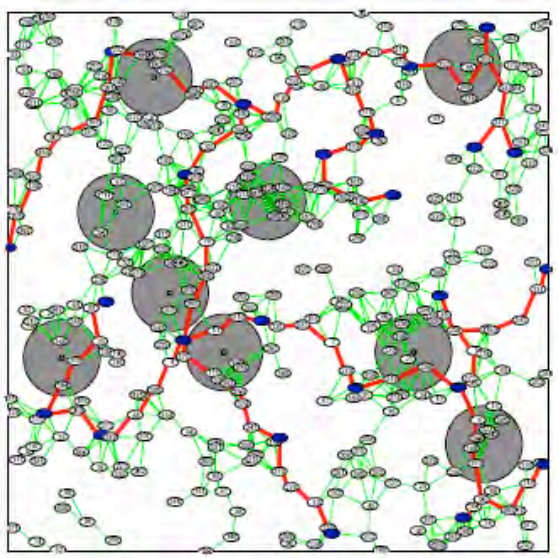
- To defend against mobile jammers, use leaky integrator

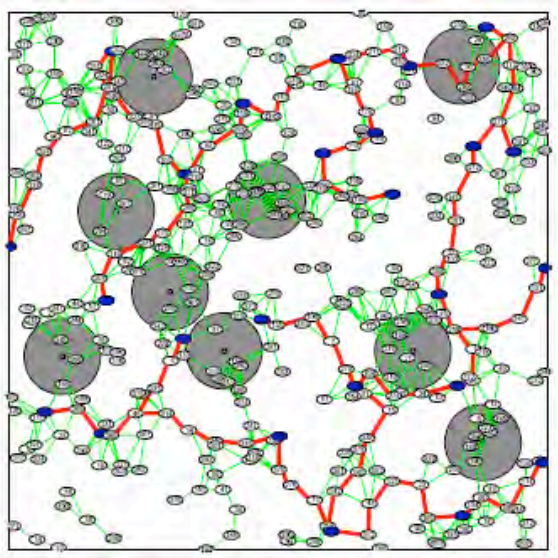
$$w_r(u,v) = \begin{cases} \left|\ln(PDR(u,v)\right|, (u,v) \in T \\ \rho \cdot w_r(u,v), (u,v) \notin T \end{cases}$$

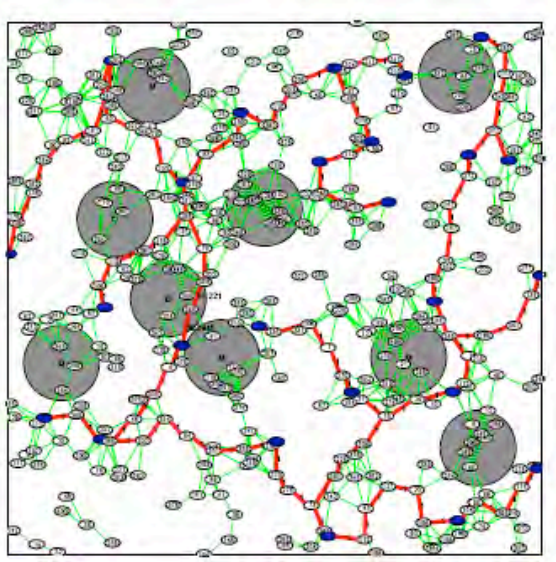
WisperNet-Space: Performance analysis

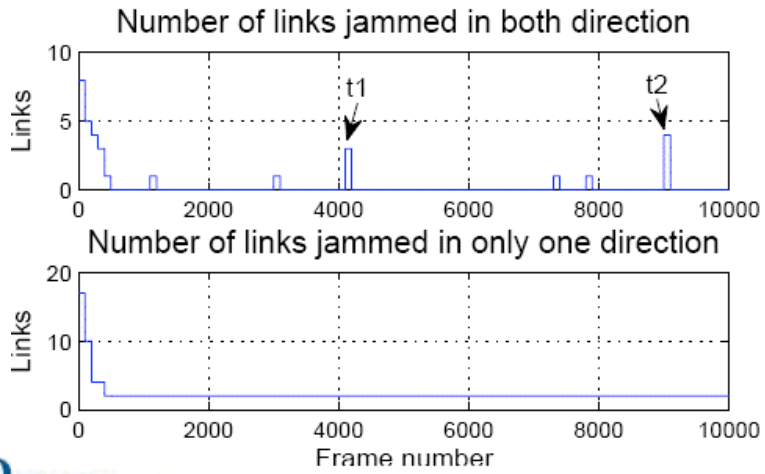
- 400 randomly distributed nodes in 4*km* x 4*km* square
- 9 randomly distributed jamming nodes, with link 50% utilization
- Topology changes are performed once in 100 frames

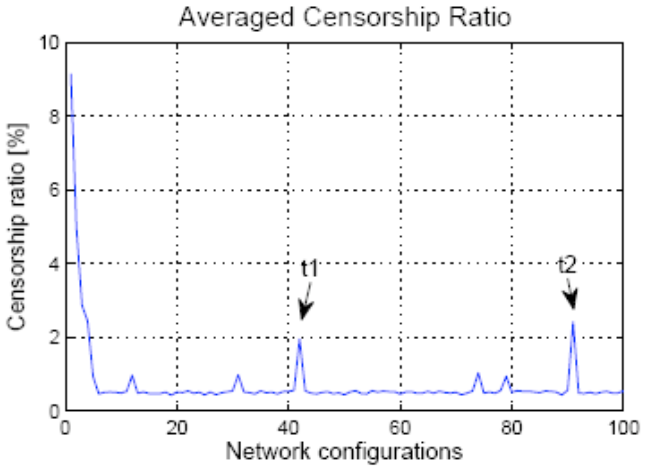
WisperNet-Space: Performance analysis









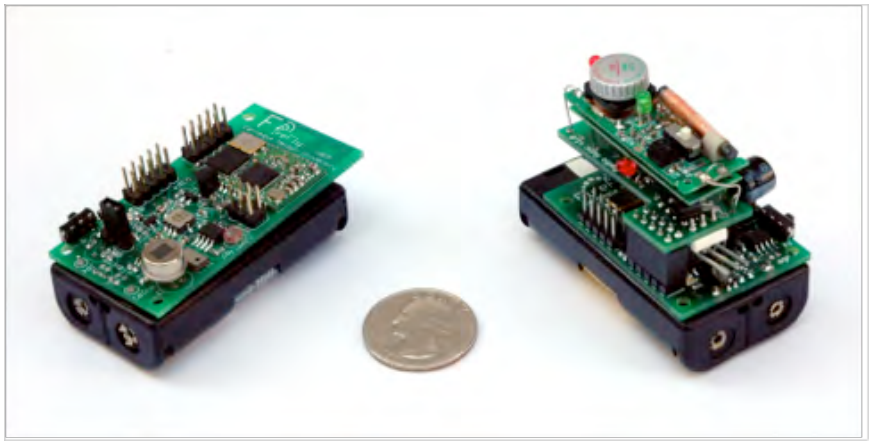


Penn Engineering

mLAB Real-Time & Embedded

Implementation: Platform

- Implemented on FireFly nodes



Implementation

- Implemented on
  - 8-bit microcontroller ATMEGA32L, running on 8MHz
  - 16-bit microcontroller MSP430F22x, running on 16MHz
- Schedule calculation – 276B of Flash, 400B of RAM
- SHA1-HMAC required only 3 additional 160-bit buffers
- SHA1-HMAC required 12.5ms on the TI MSP430F22x

Thank You

Penn Engineering

mLAB Real-Time & Embedded