Data Security in Unattended Wireless Sensor Network

Roberto Di Pietro
UNESCO Chair in Data Privacy

Di Ma
UCI

Prof. Luigi Mancini
Università di Roma “La Sapienza”

Claudio Soriente
UCI

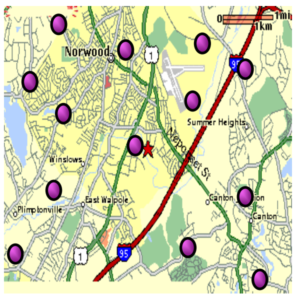
Angelo Spognardi
INRIA – Rhone Alpes

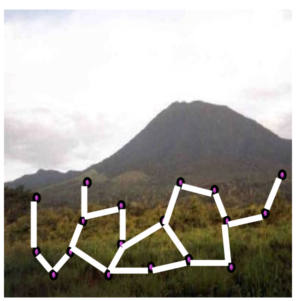
Prof. Gene Tsudik
UCI

Agenda

1. Introduction to UWSN
2. ADV model
3. POSH
   a. Preliminaries and assumptions
   b. The protocol
   c. Analysis
4. Conclusions

A "Typical" Wireless Sensor Network





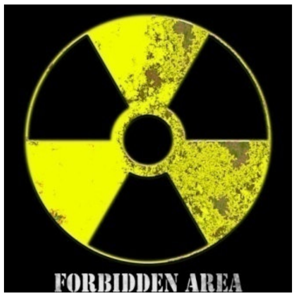
Many real, alleged and imagined applications

- Networking
  - Sensor-to-sink communication (opt. sink-to-sensors)
- Collection method
  - Periodic collection
    or
  - Event driven
    or
  - Query based = on-demand
- Online Sink
  - Real-time off-loading of data

<u>Unattended</u> Wireless Sensor Network (UWSN)

- Nodes operate in hostile environment
  - Initial deployment might be ad-hoc
- No ever-present sink
  - Itinerant
- Periodic data sensing (on-demand, event-driven– N/A)
  - Nodes might retain data for a long time
  - Data might be valuable
- Nodes are left on their own
  - Adversary roams around
  - **Challenge: Data Security in UWSNs**

Examples



- WSN deployed in a recalcitrant country to monitor any potential nuclear activity

- Underground WSN monitoring sound and vibration produced by troop movements or border crossings

- Anti-poaching WSN

New kind of Adversary (ADV)

- Previous adversaries would corrupt a fixed threshold of the nodes in the network
  - Security protocols were aimed at attack detection
  - The online sink can then mitigate the attack
    - Excluding compromised nodes

- Our adversary is ***MOBILE***
  - Roams the network and compromises different sets of sensors
    - Given enough time it can subvert the whole network
  - The sink is offline: real-time detection does not help
    - Adv can reach its goal and leave with impunity

Does this sound familiar?

- ADV shares many feature with the well known Crypto Mobile Adversary
  - Ostrovsky &Yung: How to Withstand Mobile Virus Attacks, PODC'91
  - Proactive Cryptography: Decryption and Signatures
  - Adversary aimed at learning some shared secret
- Now the problem is different
  - No such secret to hide
  - Less resources (power, storage, …)
  - Brand new solutions required

UWSN Mobile Adversary

ADV defined by: goal / operation / visibility

Goal:
- Search-and-erase
- Search-and-replace
- Curious

Operation:
- Reactive
- Proactive

Visibility:
- Stealthy
- Visible

The journey so far…

- Search-and-erase
  - No Crypto
    - Nodes collaborate to hide data location

      Catch Me (if you can): Data Survival in Unattended Sensor Networks (IEEE PerCom'08)
  - Crypto-enabled sensors
    - Design and evaluation of cryptographic protocol to protect target data

      in submission…

- Search-and-replace
  - Collaborative authentication
  - ongoing work…

- Curious
  - Co-operative self healing

    POSH (IEEE SRDS'08)

POSH
Proactive co-Operative Self Healing
in Unattended Wireless Sensor Networks

Motivation

- Curious adversary aims at reading sensor-collected data

- Encryption does not help
  - Symmetric keys are exposed with node compromise
  - w/ Public Key encryption, the adversary can GUESS the cleartext
    - Randomized encryption helps but only with a TRNG
      - Not currently available (nor foreseeable)

- Sensor-collected data can be partitioned based on compromise
  - Before Compromise (1)
    - Requires Forward Secure Encryption Scheme
  - During Compromise (2)
    - Not much can be done!
  - After compromise (3)
    - Requires Backward Secure Encryption Scheme

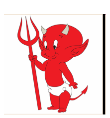
Can we protect category (1) and (3) data?

Forward Secrecy

- Even if ADV learns current key, it is not able to derive PREVIOUS round keys
- Based on per-round key evolution
  - At the end of round r, the next round key is computed through a one-way function (and the current round key is securely erased)
    - $K^{r+1}=H(K^r)$
- Suitable UWSNs
  - But after compromise, ADV can mimic key evolution process
  - Anyway we will use it…

$K^1 \rightarrow K^2 \rightarrow K^3 \rightarrow K^4 \rightarrow K^5 \rightarrow K^6 \rightarrow K^7 \rightarrow \ldots$

Sensor compromised at round 4 and then released



$\leftarrow K^4 \rightarrow K^5 \rightarrow K^6 \rightarrow K^7 \rightarrow \ldots$

Backward Secrecy

- Even if ADV learns current key, it is not able to derive FUTURE round keys

- Based on per-round key evolution
  - In the literature so far, it requires an online trusted authority

- Not suitable for UWSNs
  - The sink is offline
  - Sensor can not act as a trusted authority for their peers as any sensor can be easily compromised

$K^1 \rightarrow K^2 \rightarrow K^3 \rightarrow K^4 \rightarrow K^5 \rightarrow K^6 \rightarrow K^7 \rightarrow \ldots$

Sensor compromised at round 4 and then released

$K^1 \leftarrow K^2 \leftarrow K^3 \leftarrow K^4 \rightarrow$

Key Insulated schemes

- Encryption Schemes that are both BACKWARD and FORWARD secure are known as KEY INSULATED schemes
  - Unfortunately no such scheme is currently available for UWSNs
  - Require online trusted third party
  - Expensive computation

$K^1 \rightarrow K^2 \rightarrow K^3 \rightarrow K^4 \rightarrow K^5 \rightarrow K^6 \rightarrow K^7 \rightarrow \ldots$

Sensor compromised at round 4 and then released

$\leftarrow K^4 \rightarrow$

POSH: Main Idea

- Forward secrecy is achieved through key evolution

- Backward secrecy is achieved through sensor cooperation
  - A sensor can securely regenerate a key unknown to ADV, if it obtains at least one *contribution* from a non compromised peer sensor

Network Assumptions 1/2

- **Periodic data collection**
  - Time divided in equal and fixed collection rounds and each of the ***n*** sensor collects a single data unit per round
- **Unattended Operation**
  - An itinerant sink periodically visits the UWSN to collect sensed data.
  - ***v*** is the maximum number of collection rounds between successive sink visits.
- **Communication**
  - The UWSN is always connected
  - Any two sensors can communicate either directly or through peers

Network Assumptions 2/2

- **Storage**
  - Each sensor has enough storage for O(v) data units
- **Cryptographic Capabilities**
  - Cryptographic hashing
  - Symmetric key encryption (unique secret key shared with the sink)
  - Pseudo-Random Number Generator (PRNG) (unique secret seed shared with the sink)
- **Re-initialization**
  - At each visit, the sink re-initializes the sensors (secrets refreshing)
    - New secret key
    - New secret seed
    - Empty storage

Adversarial model 1/2

- **Goal**
  - ADV's main goal is to learn from nodes as many secrets as possible (keys or other keying material).
- **Compromise Power**
  - ADV can compromise at most $0 < k < n/2$ sensors at any round.
  - It reads all storage/memory and listens to all communication of each compromised sensor.
- **Periodic Operation**
  - At the end of each compromise round, ADV picks a subset of up to k sensors to compromise in the following round.
  - At the start of each round, the adversary atomically releases the subset from the previous round and compromises the new subset.

Adversarial model 2/2

- **Topology Knowledge**
  - ADV knows the entire topology of the UWSN.
- **Minimal Disruption**
  - ADV does not interfere with sensors’ behavior, in order to remain undetected
- **Defense Awareness**
  - ADV is fully aware of any scheme or algorithm used by the UWSN.

POSH algorithm

Contributions to be sent

Generic node protocol run (**round i**):

Contributed nodes

1. Generate t random values $\{R_{i_1}, \ldots, R_{i_t}\}$
2. Select $\{s_{i_1}, \ldots, s_{i_t}\} \leftarrow_R \{s_1, \ldots, s_{i-1}, s_{i+1}, \ldots, s_n\}$
3. Send R_{i_j} to s_{i_j}, $1 \leq j \leq t$
4. Receive contributions $\{c_{i_1}, \ldots, c_{i_{t'}}\}$
5. Sensing, encryption, authentication...
6. Compute $K_i^{r+1} = H(K_i^r || c_{i_1} || \ldots || c_{i_{t'}})$
7. Erase K_i^r

Normal operating activities

Key refresh

$\{s_1, \ldots, s_n\}$ = set of sensors in the network
K_i^r = key used by s_i at round r
$\{s_1, \ldots, s_n\}$ = set of sensors in the network
K_i^r = key used by s_i at round r

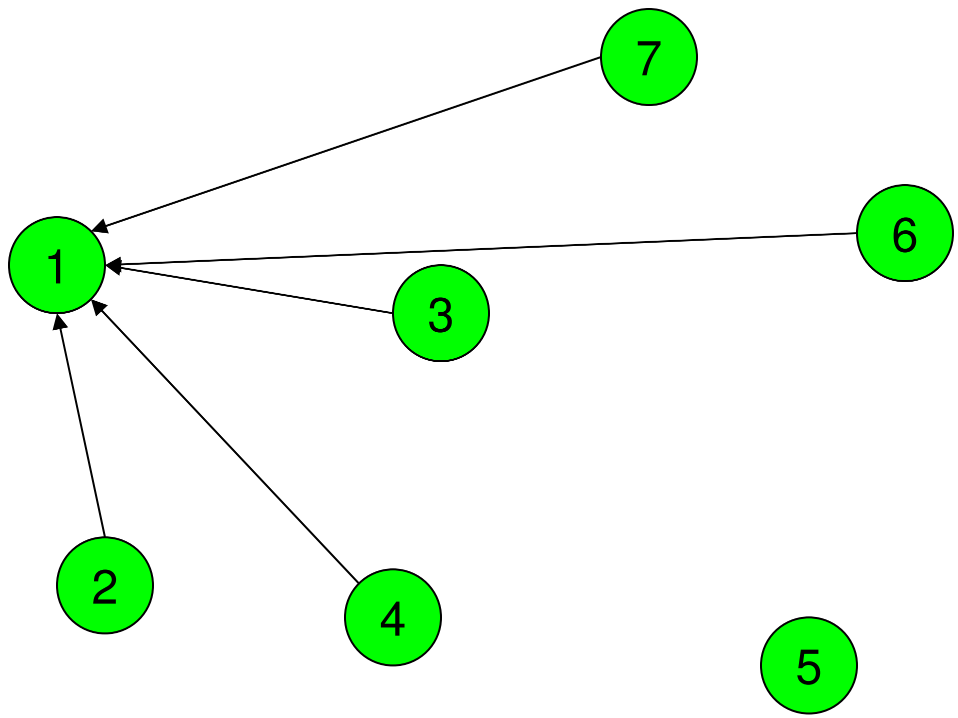
Analysis (aka Sensor Coloring)

Starting from round 1, ADV compromises k sensors per round:

- Red sensors (R^r)
  - currently controlled by ADV
- Yellow sensors (Y^r)
  - have been compromised in some previous round and their current keys are known to ADV
- Green sensors (G^r)
  - Either they have never been compromised
  - **Or** they have recovered through POSH

Example

r = 4



Sensor 1

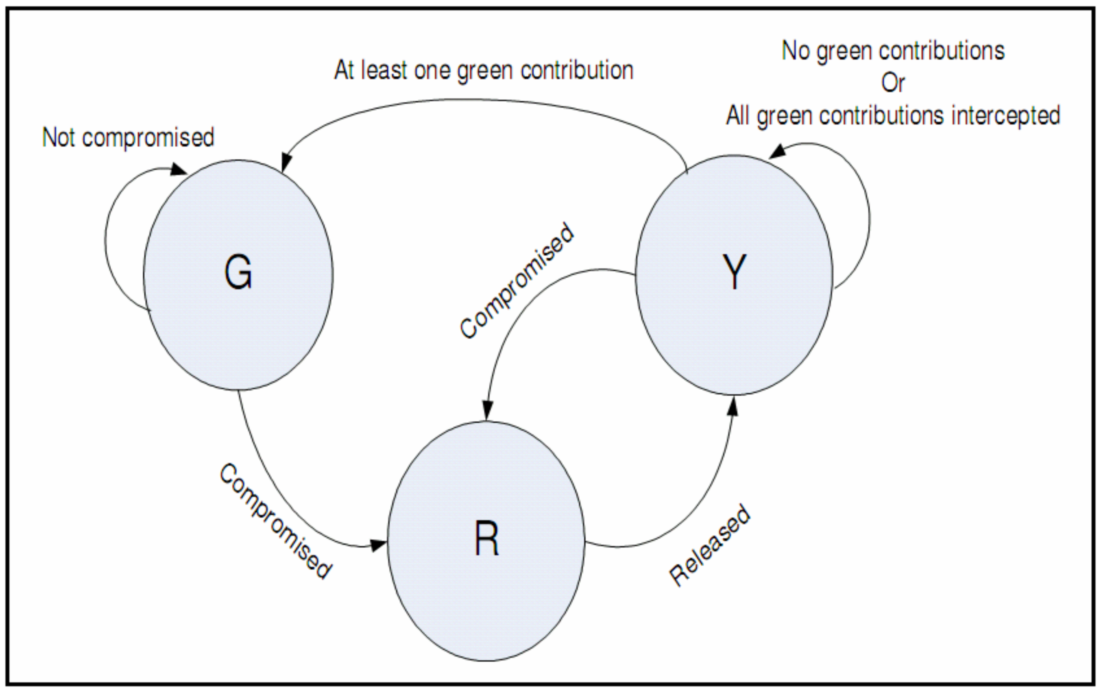
K^1

$K^2 = H(K^1 \mid\mid c_3 \mid\mid c_6)$

$K^3 = H(K^2 \mid\mid c_2)$

$K^4 = H(K^2 \mid\mid c_4 \mid\mid c_7)$

Sensor transition diagram

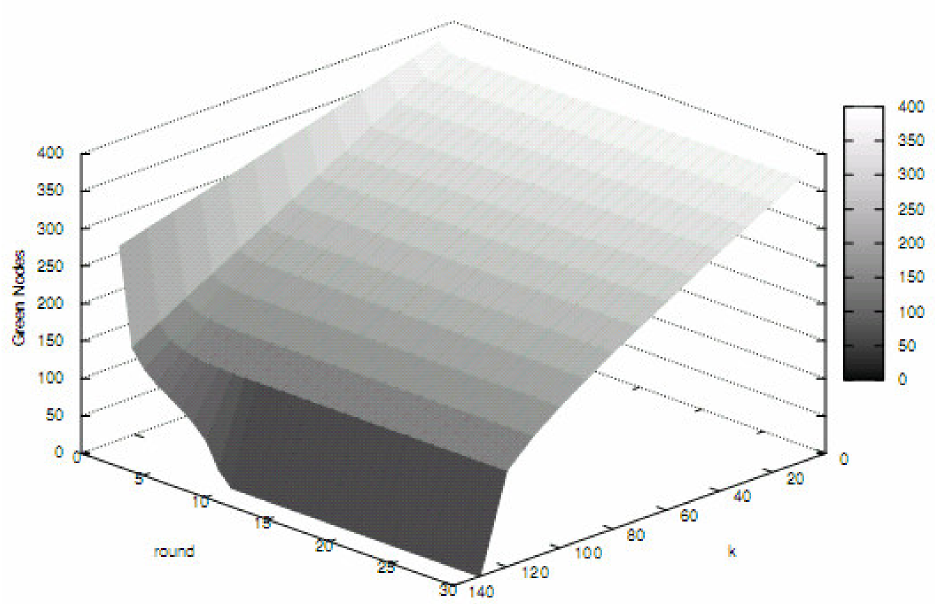


- |R|=k
- ADV's goal it to maximize |Y|+|R|
- Network goal: |G|=n-2k

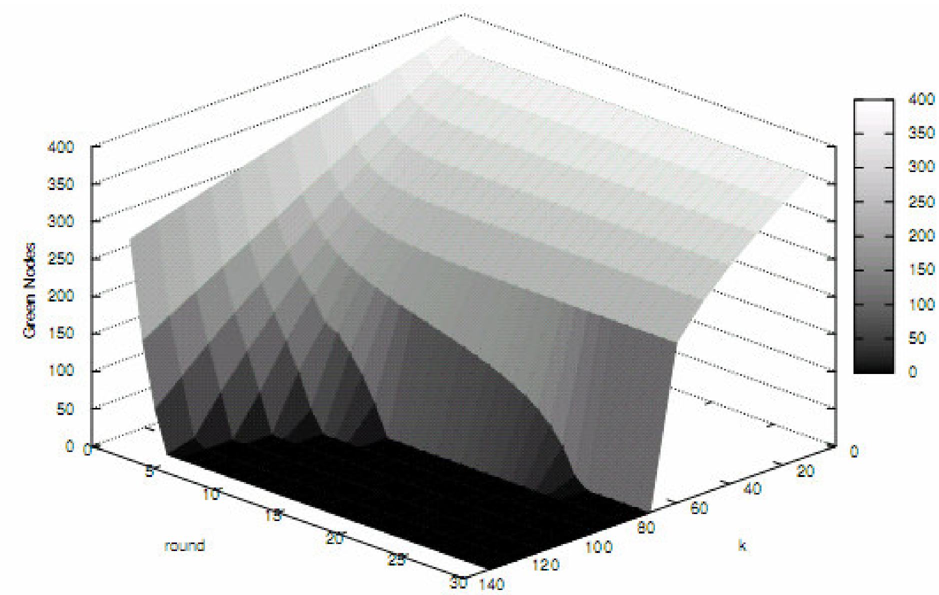
Two kinds of ADV

- INF-ADV is always aware of G
  - Unrealistic but very powerful
  - Used as benchmark
- RR-ADV moves through set of nodes in a round-robin fashion
  - Time based heuristic…nodes in Y for a long time could have moved G
  - Realistic but possibly weak
    - Might choose to compromise a yellow sensor

Results (|G| against INF-ADV)



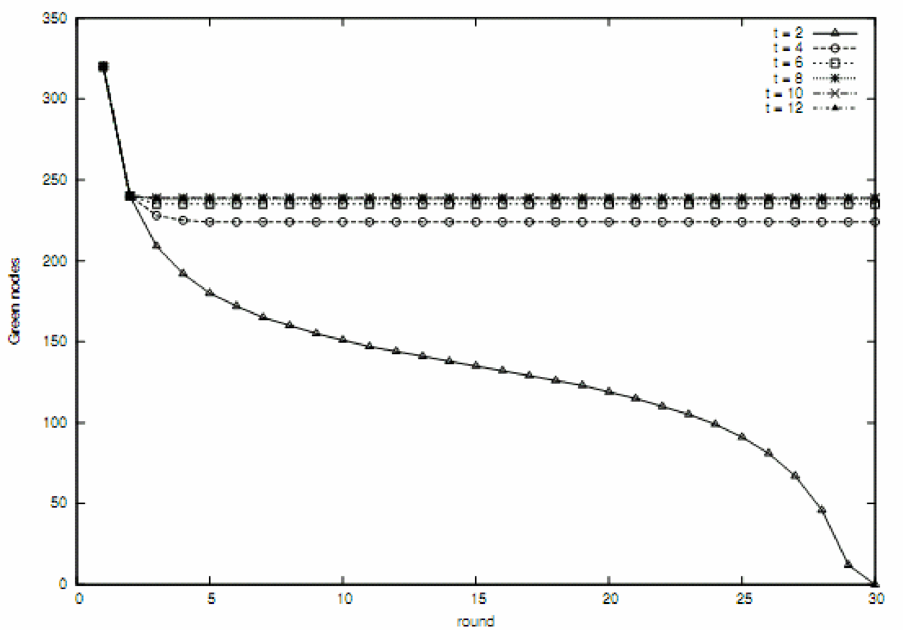
(a) $n = 400, t = 6, p = 0.2$



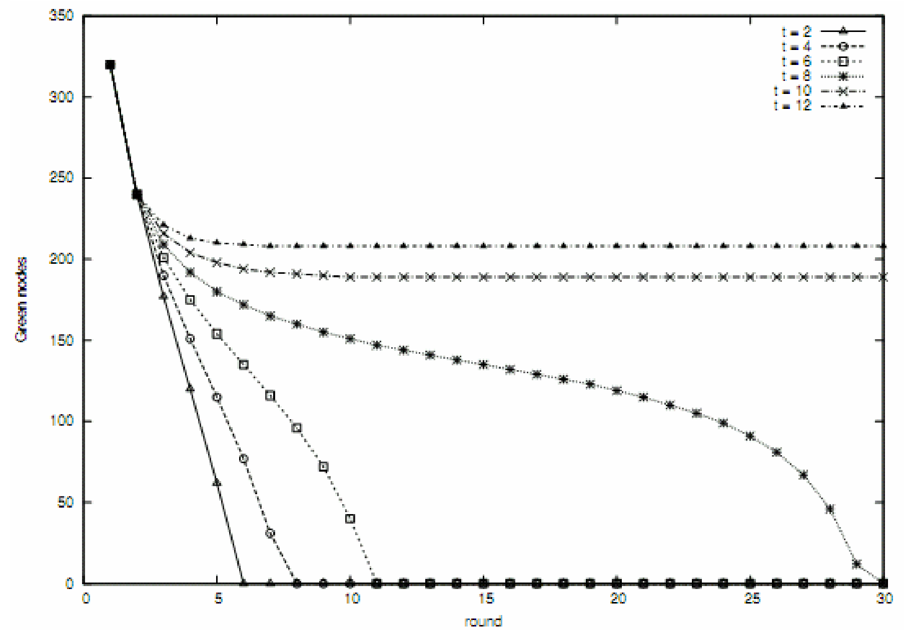
(b) $n = 400, t = 6, p = 0.8$

- p = ADV eavesdropping prob.
- t = 6 results in each sensor receiving at least one contribution on the average
- Threshold phenomena:
  - e.g. for p=0.2, |G| remains stable for k/n < 80/400
  - That is 20% per round!!!

Effect of “t”



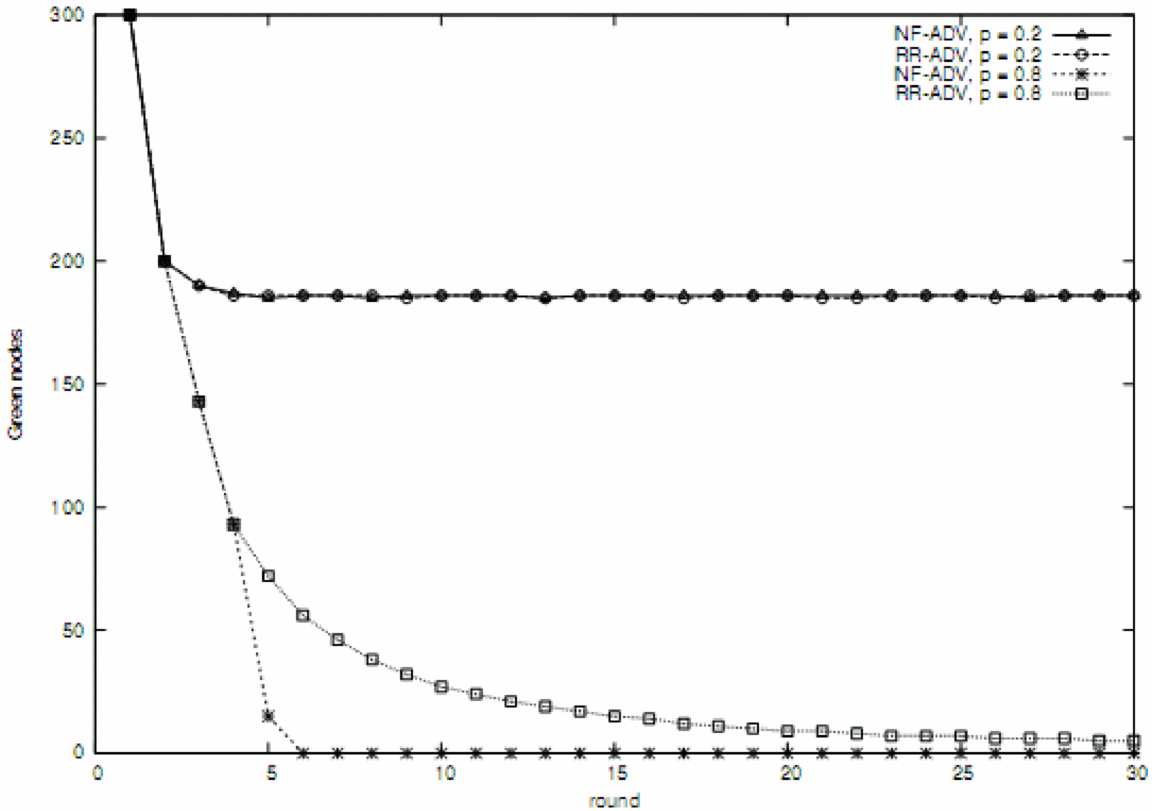
(a) $n = 400, k = 80, p = 0.2$



(b) $n = 400, k = 80, p = 0.8$

- Increasing t when |G| ~ n-2k does not help
  - Further, messages are expensive!

INF-ADV vs RR-ADV



No difference if |G| is close to its optimal value

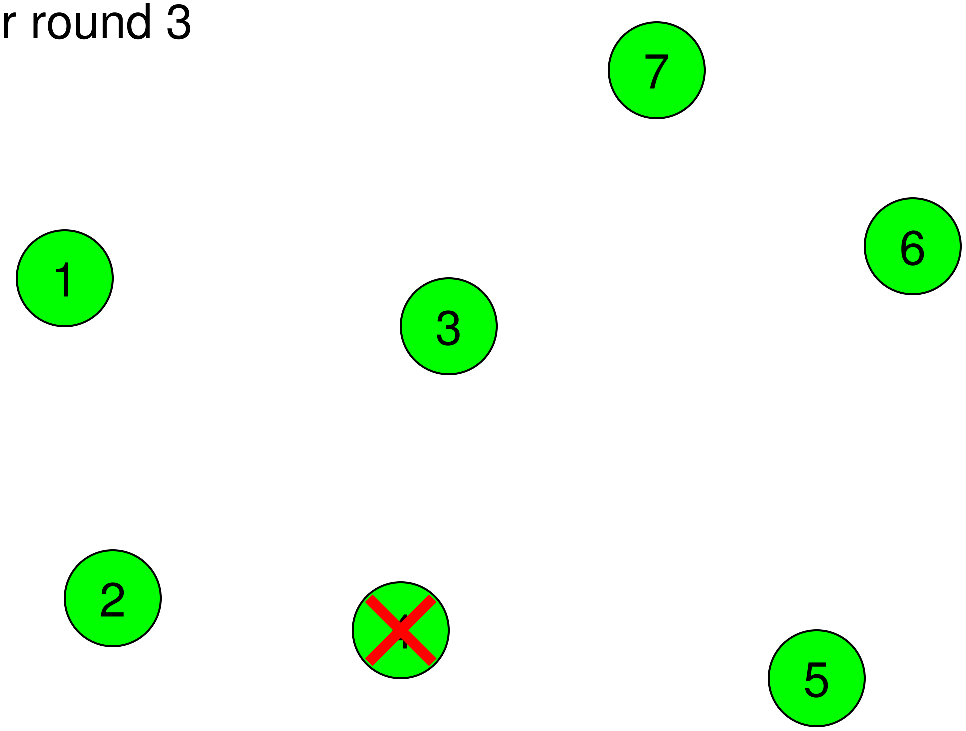
$$n = 400,\ k = 100,\ t = 6$$

Dealing w/ real world

- Message delivery failure
  - Sink synchronization
  - Sensor must store the ID of their contributors
- Sensor failure
  - If storage becomes unavailable key sensor history cannot be reconstructed
  - Other sensors might depend on the failed one
- Publik Key Crypto
  - Encrypt round key under the sink PK
    - Use round key for everything else

Example

Sensor 4 fails after round 3



Sensor 1

K^1

$K^2=H(K^1 \,||\, c_3 \,||\, c_6)$

$K^3=H(K^2 \,||\, c_2)$

$K^4=H(K^2 \,||\, c_4 \,||\, c_7\,)$

K^4 requires sensors 4 and 7

K^1 is shared

Sensor 1 will have contribute to other peers…

Sink

K^1

$K^2=H(K^1 \,||\, c_3 \,||\, c_6)$

$K^3=H(K^2 \,||\, c_2)$

$K^4=H(K^2 \,||\, ? \,||\, c_7\,)$

Conclusion

- UWSN is a new, exciting field that calls for innovative security solutions
- No crypto no means no security
- But….
- Crypto helps!
- Role of randomization in UWSN not completely characterized yet

References

- Catch Me (If You Can): Data Survival in Unattended Sensor Networks
  - R. Di Pietro, L.V. Mancini, C. Soriente, A. Spognardi, G. Tsudik
  - IEEE PerCom'08
- POSH: Proactive co-Operative Self-Healing in Unattended Wireless Sensor Networks
  - R. Di Pietro, D. Ma, C. Soriente, G. Tsudik
  - IEEE SRDS'08
- DISH: distributed self-healing (in unattended sensor networks).
  - D. Ma and G. Tsudik.
  - Cryptology ePrint Archive, Report 2008/158, 2008
- Playing Hide-and-Seek with a Focused Mobile Adversary: Maximizing Data Survival in Unattended Sensor Networks
  - R. Di Pietro, L.V. Mancini, C. Soriente, A. Spognardi, G. Tsudik
  - Cryptology ePrint Archive, Report 2008/292, 2008