Module SPC:

Security,Protocoals and Countermeasures in Wireless Sensor Networks

DRAFT

W. Li (TSU) and R. Verma (UH)

Lecture Contents

Part 1. WSN Security

Part 2. WSN Security Protocol

Part 3. WSN Security Countermeasures

Part 1: WSN Security

1.1. Overview

- WSN security: Too many problems… A number of solutions… Enough?
- Survey Paper: outlines security issues, discusses some existing solutions, and suggests possible research directions
- Issues include:
  - key establishment
  - secrecy
  - authentication
  - privacy
  - denial-of-service attacks → More info in a later set of slides
  - secure routing → More info in a later set of slides
  - node capture
- Also discuses some sample security services for wireless sensor networks

1.2. Problems Applying Traditional Network Security Techniques

- Sensor devices are limited in their energy, computation, and communication capabilities
- Sensor nodes are often deployed in open areas, thus allowing physical attack
- Sensor networks closely interact with their physical environments and with people, posing new security problems

1.3. Key Establishment and Trust

- Sensor devices have limited computational power, making public-key cryptographic primitives too expensive in terms of system overhead.
- Simplest solution is a network-wide shared key
  - **problem**: if even a single node were compromised, the secret key would be revealed, and decryption of all network traffic would be possible
- Slightly better solution:
  - use a single shared key to establish a set of link keys, one per pair of communicating nodes, then erase the network-wide key
  - problem: does not allow addition of new nodes after initial deployment

1.3. Key Establishment and Trust

- Bootstrapping keys using a trusted base station
  - Each node needs to share only a single key with the base station and set up keys with other nodes through the base station
  - The base station becomes a single point of failure
    - Utilize tamper-resistant packaging for the base station, reducing the threat of physical attack
    - Most existing work assumes base station is safe – Good assumption???

1.4 Random-key pre-distribution protocols

- Large pool of symmetric keys is chosen
- Random subset of the pool is distributed to each sensor node
- To communicate, two nodes search their pools for a common key
  - If they find one, they use it to establish a session key
  - Not every pair of nodes shares a common key, but if the key-establishment probability is sufficiently high, nodes can securely communicate with sufficiently many nodes to obtain a connected network
- No need to include a central trusted base station
- Disadvantage: Attackers who compromised sufficiently many nodes could also reconstruct the complete key pool and break the scheme

1.5 Secrecy and Authentication

- We need cryptography as protection against eavesdropping, injection, and modification of packets
- Trade-offs when incorporating cryptography into sensor networks:
  - end-to-end cryptography achieves a high level of security but requires that keys be set up among all end points and be incompatible with passive participation and local broadcast
  - link-layer cryptography with a network-wide shared key simplifies key setup and supports passive participation and local broadcast, but intermediate nodes might eavesdrop or alter messages

1.6. Hardware vs. Software Cryptography

- Hardware solutions are generally more efficient, but also more costly ($)
- University of California, Berkeley, implementation of TinySec incurs only an additional 5%–10%. performance overhead using software-only methods
  - Most of the overhead is due to increases in packet size
  - Cryptographic calculations have little effect on latency or throughput, since they can overlap with data transfer
  - Hardware reduces only the computational costs, not packet size
- Thus, software-only techniques are sufficient (or reasonable to be more careful)

1.7. Privacy

- Issues
  - Employers might spy on their employees
  - Shop owners might spy on customers
  - Neighbours might spy on each other
  - Law enforcement agencies might spy on public places
- Technological improvements will only worsen the problem
  - Devices will get smaller and easier to conceal
  - Devices will get cheaper, thus surveillance will be more affordable

1.7. Privacy

- Sensor networks raise new threats that are qualitatively different from what private citizens worldwide faced before
  - Sensor networks allow data collection, coordinated analysis, and automated event correlation
  - Networked systems of sensors can enable routine tracking of people and vehicles over long periods of time
  - EZ Pass + OnStar == Big Brother?
- Suggested ways of approaching solution include a mix of:
  - Societal norms
  - New laws
  - Technological responses

1.8. Robustness to Denial of Service

- Simple form: Radio jamming
- Sophisticated form: Transmit while a neighbor is also transmitting or continuously generating a request-to-send signal
- Possible solution (when the jamming affects only a portion of the network):
  - Detect the jamming
  - Map the affected region
  - Route around the jammed area

1.9. Secure Routing

- Proper routing and forwarding are essential for communication in sensor networks
- Injection attacks
  - Transmit malicious routing information into the network resulting in routing inconsistencies
  - Authentication might guard against injection attacks, but some routing protocols are vulnerable to replay by the attacker of legitimate routing messages
- Sensor network routing protocols are particularly susceptible to node-capture attacks
  - Compromise of a single node could be enough to take over the entire network or prevent any communication within it

1.10. Resilience to Node Capture

- In traditional computing, physical security is often taken for granted
- Sensor nodes, by contrast, are likely to be placed in open locations
  - Attacker might capture sensor nodes
  - Extract cryptographic secrets
  - Modify programs/Replace them with malicious nodes
- Tamper-resistant packaging may be one defense, but it's expensive

1.11. Algorithmic Solutions to Node Capture

- Attempt to build networks that operate correctly even in the presence of nodes that might behave in an arbitrarily malicious way
  - Replicate state across the network and use majority voting to detect inconsistencies
  - Gather redundant views of the environment and crosscheck them for consistency
- Most challenging problems in sensor network security
  - We are far from a complete solution

1.12. Network Security Services

- So far, we've explored low-level security primitives for securing sensor networks.
- Now, we consider high-level security mechanisms.
  - Secure group management
  - Intrusion detection
  - Secure data aggregation

1.13. Secure Group Management

- Protocols for group management are required to
  - securely admit new group members
  - support secure group communication
- Outcome of group computation must be authenticated to ensure it comes from a valid group
- Any solution must also be efficient in terms of time and energy

1.14. Intrusion detection

- In wired networks, traffic and computation are typically monitored and analyzed for anomalies at various concentration points
  - expensive in terms of the network's memory and energy consumption
  - hurts bandwidth constraints
- Wireless sensor networks require a solution that is fully distributed and inexpensive in terms of communication, energy, and memory requirements
- In order to look for anomalies, applications and typical threat models must be understood
- It is particularly important for researchers and practitioners to understand how cooperating adversaries might attack the system
- The use of secure groups may be a promising approach for decentralized intrusion detection

1.15. Secure Data Aggregation

- One benefit of a wireless sensor network is the fine-grain sensing that large and dense sets of nodes can provide
- The sensed values must be aggregated to avoid overwhelming amounts of traffic back to the base station
- Depending on the architecture of the network, aggregation may take place in many places
  - All aggregation locations must be secured
- If the application tolerates approximate answers, powerful techniques are available
  - Randomly sampling a small fraction of nodes and checking that they have behaved properly supports detection of many different types of attacks

Part 2: WSN Security Protocol

2.1 Authentication Protocols

- Back Ground
  - Ad hoc networks, either static (like sensor networks) or mobile, poses various challenges in providing secured service
  - Authenticating nodes is a cornerstone in security
  - Authentication supports confidentiality and access control
  - Other services depend upon proper authentication of the communication entity[9].

2.1 Authentication Protocols

❑ Components of the Authentication Process

- A generic authentication process has six major phases
  - ✓ Bootstrapping – providing supplicant with a key or a password
  - ✓ Pre-authentication – Supplicant presents its credentials to authenticator
  - ✓ Credential Establishment – Supplicant's credentials is verified and it is authorized for services thereafter

2.1 Authentication Protocols

❑ Components of the Authentication Process (contd.)

- Authentication state – Communications between supplicant and the authenticator are considered authorized
- Monitoring – Supplicant's behavior is being monitored for fear of its being compromised or misbehaving
- Revoked – A compromised supplicant's authorization is revoked and its request for re-authorization is denied

2.1 Authentication Protocols

❑ Classification of Authentication Process

- In this paper [1], authors have identified three major criteria for the classification of authentication process
  - ✓ Classification Based on Authentication Function
  - ✓ Classification Based on type of Credentials
  - ✓ Classification Based on Establishment of Credentials

2.1 Authentication Protocols

❑ Classification Based on Authentication Function

- Homogeneous – All nodes in the network have the same role and responsibility with respect to the authentication operation. Nodes in the network make authentication decisions autonomously
- Heterogeneous – Nodes in the network have different roles with respect to the authentication operation. There is an underlying service in the network that aids other nodes in making authentication decisions

2.1 Authentication Protocols

❑ Classification Based on type of Credentials

- Identity-based credentials – It recognizes a unique possession owned by the supplicant that could be used to identify it with high confidence.
  - ✓ Identity based credentials can be further classified into encryption based and non-encryption based.
- Context Based Credentials – This category recognizes a unique contextual attribute of the supplicant that can be used to identify it with high confidence.
  - ✓ Contextual based credentials can be behavioral or physical.

2.1 Authentication Protocols

❑ Classification Based on Establishment of Credentials

- Pre-deployed Credential – This category assumes a pre-distribution offline phase (before deployment) where credentials are established.
- Derived Credential – This category assumes that credentials are established post-deployment.
- Post-deployment Credential – In this category the actual credentials used for authentication are derived from the initial credentials post deployment.

2.2. Authenticating Public Keys

❑ Back Ground

- In any Sensor Network the security of communication between the nodes is extremely important
- To provide proper security, communication should be encrypted and authenticated
- Symmetric key could be an attractive techniques in this issue
- However, due to the limitation on memory, this technique is not able to achieve both a perfect connectivity and a perfect resilience

2.2. Authenticating Public Keys

❑ Back Ground (*contd.*)

- The use of Public-Key Cryptography (PKC) would eliminate the above problem
- The main problem of using PKC in sensor networks is its computational complexity and communication overhead
- Various studies are being carried out [13] to optimize the PKC protocol
- In this paper[2], the authors have proposed the optimization of an essential operation in PKC: the public key authentication, by exploring network properties

2.2. Authenticating Public Keys

❑A Naive Scheme

- Nodes of the network can carry the public key of all the other nods to eliminate the public key authentication problem without any certification
- However, since the size of public keys can be large, sensor might not have enough memory to save all the public keys
- This situation can be improved by letting each node carry a one-way hash value of the public keys of other nods
- However, for a large network, even this might need a large memory size.

2.2. Authenticating Public Keys

❑A Memory Efficient Scheme

- Merkle trees [12] method can be used to solve the memory-usage problem.
- A Merkle tree can be constructed as follows:
  1) Let us consider N leaves $L_1, \ldots, L_n$, with each leaf corresponding to a sensor node
  2) Each leaf contains the bindings between the identity (id_i) and the public key (pk_i)of the corresponding node i
  3) Let us use V to denote an internal tree node, and V_{left} and V_{right} to denote V's two children
  4) Then The φ value of each node is defined as

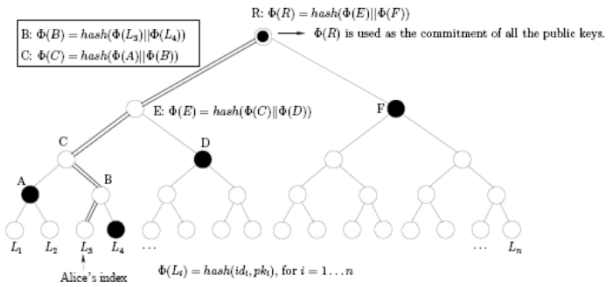
     $\varphi(L_i) = hash(id_i, pk_i),\ for\ i = 1, \ldots, N$

     $\varphi(V) = hash(\varphi(V_{left}) \,||\, \varphi(V_{right}))$, (|| means concatenation of two string)

2.2. Authenticating Public Keys

❑ A Memory Efficient Scheme *(contd.)*

- Each sensor only needs to store φ(R), where R is the root of the Merkle tree. Therefore, the memory usage is the length of one hash value



Using Merkle tree To Authenticate Public Keys

2.2. Authenticating Public Keys

❑ Communication cost

- The communication cost for authenticating public key in this scheme has been calculated as follow:

1) Let pk be Alice's public key, and L be Alice's corresponding leaf node in the tree.
2) Let λ denote the path from L to the root (not including the root), and let H represent the length of the path.
3) For each tree node v ∈ λ, Alice sends φ(v's sibling) to Bob, along with the public key pk. Use $\lambda_1, \ldots, \lambda_H$ to represent these φ values, and call these φ values the proofs.

2.2. Authenticating Public Keys

❑ Communication cost *(contd.)*

- To verify the authenticity of Alice's public key pk (assume Alice's identity is id), Bob computes hash (id, pk); he then uses the results and $\lambda_1, \ldots, \lambda_H$ to reconstruct the root of the Merkle tree R' with $\varphi(R')$. Bob will trust that the binding between id and pk is authentic only if $\varphi(R') = \varphi(R)$.
- Because the Merkle tree is a complete binary tree with N leaves, its height is $\log N$ (the base of the logarithm is assumed to be 2). Therefore, the communication costs is $L.\log N$, with L being the length of a hash value.

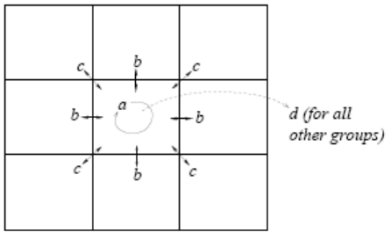
2.2. Authenticating Public Keys

❑ Minimize communication cost

- Communication cost can be further trim down by considering the fact that the nodes that are nearer to each other (neighbor nods) communicate to each other more frequently than to a distant node.
- We can also consider the nodes to be belonged to groups with two node may either be in the same group, horizontal or vertical group, diagonal group or in a non-group (considering a squire mesh deployment)
- In that case we can break down the Merkle tree into a sub-tree with height a for the nodes in same group, height b for the horizontal/ vertical group, c for the diagonal group and d for a non-group node.

2.2. Authenticating Public Keys

- Minimize communication cost



Height of Merkle Tree for nodes from different neighbor groups.

2.2. Authenticating Public Keys

- Minimize communication cost

  - If we consider the probability of two nodes to be in any of the four group as w0 for group height a, w1 for group height b, w2 for group height c and w3 for group height d, then Communication cost C can be given as

    $C = w0.a + w1.b + w2.c + w3.d$

  - However the the memory usage per node increases by

    $m = S/2^a + 4S/2^b + 4S/2^c + N/2^d$

    Where S is the number of nodes in each group and N is the number of total nodes.

2.2. Authenticating Public Keys

- Conclusion *(for this paper)*
  - The authors have shown in this paper that due to a unique property of sensor networks, public keys do not need to be authenticated in the same way as it is done in the Internet environment (i.e., using certificates); instead, public keys can be authenticated using one-way hash functions, which are much more efficient than signature verification on certificates.
  - They have conducted extensive evaluation on their scheme, where they have claimed that the results show significant savings on power consumption with a moderate memory use.

2.3 Energy Efficient Security Protocol

- Background
  - Sensors are operated by low-powered battery
  - Key challenge is to maximize the life of sensor nodes
  - Another key issue is to have secure communication between nodes and base station
  - Encryption, decryption, signing data, verifying signatures consumes extra battery power

2.3 Energy Efficient Security Protocol

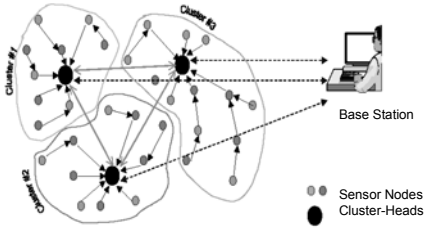
❑ Background (*cont.*)

- Asymmetric cryptographic algorithms are not suitable - limited computation, power and storage resources of nodes
- Symmetric cryptographic algorithms are first employed in "SPINS" protocol [7] for WSNs in 2002 to provide security
- It also compromises security – limited key length, limited memory space in sensor nodes (4.5 KB)
- In this paper [3], non-blocking OVSF (Orthogonal Variable Spreading Factor) codes [13] is used

2.3 Energy Efficient Security Protocol

❑ System Model

- Cluster-based sensor network is considered
- Nodes are assumed immobile
- Cluster-heads are chosen dynamically



Typical cluster-based sensor network

2.3 Energy Efficient Security Protocol

❑ Secure Data Transmission Algorithm

1) The base station will generate the session key K_b at a certain time intervals (to maintain data freshness) and broadcast to all sensor nodes when it is needed.
2) The cluster-head will send the current session key K_b to its sensor node *i* when it is requested from the node *i*.
3) After receiving the current session key, sensor node *i* will XOR the session key (K_b) with its built-in secret key K_i to compute the secret encrypted session key $K_{i,b}$.
4) Sensor node *i* will encrypt the sensed data with $K_{i,b}$ and append its ID number as well as the time stamp and then will be sent to the cluster head using NOVSF code-hopping technique.

2.3 Energy Efficient Security Protocol

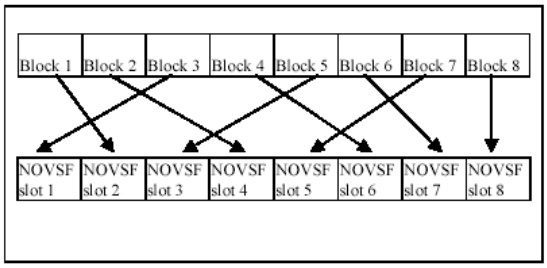
❑ Secure Data Transmission Algorithm (*Cont.*)

5) After receiving the encrypted data from sensor nodes, cluster head will append its own ID number and finally send them to higher cluster-head or the base station (Appending ID numbers will help the base station in location the origin of the data).
6) When the base station receives the encrypted data, it will decrypt the data by using the secret key $K_{i,b}$ and perform the authentication with the time stamp and the ID number.
7) If the current encryption key $K_{i,b}$ decrypt the data perfectly after a successful authentication, the transmitted message will be obtained for further process, otherwise the data will be discarded.

2.3 Energy Efficient Security Protocol

❑NOVSF Code Hopping Technique

- "Non-blocking Orthogonal Variable Spreading Factor"
- Can be implemented without utilizing additional power
- Each NOVSF code has 64 time slots to assigned Data



Mapping data blocks to NOVSF time slots, where eight blocks are available in a buffer

2.3 Energy Efficient Security Protocol

❑Implementation

- Used prototype sensor nodes of SmartDust project [6]
  - 8 bit, 4 MHz CPU
  - 10 kbps bandwidth
  - TinyOS Operating system
  - 3.5 KB OS code, 4.5 KB free space
- Consideration of Cryptographic Algorithms
  - Rinjdael AES algorithm is fast, but required 800 byte memory space
  - TEA (Tiny Encryption Algorithm) is small, and not much secured
  - DES also needs large lookup tables
  - ✓ Blowfish (mini version) needs 8 bit processor, 24 bit RAM, 1 KB ROM

2.3 Energy Efficient Security Protocol

❑Implementation (*Cont.*)

- Around 2 KB memory space is required which is acceptable for SmartDust sensor nodes
  - 1,000 bytes for Blowfish cryptographic algorithm
  - 580 bytes for MAC (Medium Access Control) operation [7]
  - 400 bytes for key setup
- No simulation or comparison results is shown

2.3 Energy Efficient Security Protocol

❑Conclusion (*of this paper*)

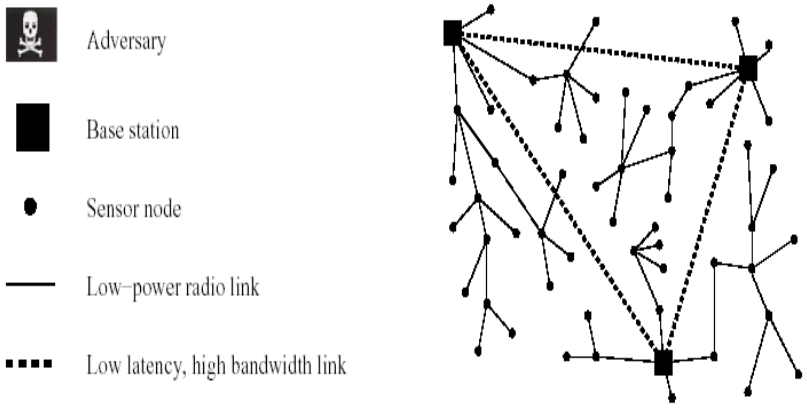
- How this protocol is energy efficient and secured –
  - ✓ Implementing *NOVSF* needs no additional power
  - ✓ Cryptographic algorithm *Blowfish* saves memory space
  - ✓ *NOVSF*s 64 time slot provides more security
  - ✓ Dynamically changing of session keys by base station
  - ✓ Appending ID# and time stamp to verify data freshness
  - ✓ Encrypting data with *Secret session keys* provides data authentication

Part 3: WSN Attack and Countermeasures

3.1 Problem Statement

- It is assumed that radio links used in wireless communication are insecure
- Attackers might have control of more than one node and extract all key materials, data and code stored
- Sensor nodes are not assumed temper resistance
- Base station is considered *trustworthy* and behave correctly

3.1 Problem Statement



A representative sensor network architecture [4]

3.1 Problem Statement (*Cont.*)

- ***Mote Attackers*** : The attackers who has get access to a few sensor nodes with similar capabilities to motes.
- ***Laptop-class Attackers*** : The attackers who has access to more powerful devices, like high-power radio transmitter or a sensitive antenna and so on. A laptop-class attacker might be able to jam the entire sensor network using its stronger transmitter.
- ***Outsider Attackers*** : The attackers who has no special access to the sensor network
- ***Inside Attackers*** : The attacker is an authorized participant in the sensor network, who has stolen the key material, code, and data from legitimate nodes.

3.1 Problem Statement (*Cont.*)

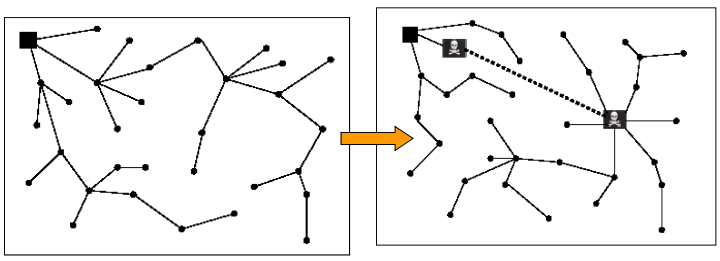
- Security issue in ad-hoc networks are similarly to sensor networks, but there are several distinctions between the two :
- Ad-hoc networks typically support routing between any pair of nodes, whereas sensor nodes may communicate in many-to-one, one-to-many as well as locally communicate with neighbors
- In most of the sensor networks nodes are not mobile, possibly embedded in walls or dispersed from an airplane in a filed.
- Ad-hoc networks may have 32-bit process, 1 MB RAM, 2 Mbps radio and a re-chargeable high powered battery. A typical sensor node has 8-bit processor, 1 KB RAM, 40 Kbps radio and a tiny battery.
- There exist a data redundancy in sensor networks as several nodes send data to the base station at correlated times.

3.2 Attacks on WSNs

- ***Spoofed, Altered, or Replayed Routing Information*** : Adversaries may be able to
  - - create routing loops, or extend or shorten routes
  - - generate false error message
  - - make partition to the network
  - - increase end-to-end delay latency.
- ***Selective Forwarding*** : Malicious nodes may refuse to forward certain messages, drop them, ensuring that they are not propagated any further.
- ***Wormholes*** : Wormholes can be used to convince two distant nodes that they are neighbors by relaying packets between the two of them.

3.2. Attacks on WSNs (*Cont.*)

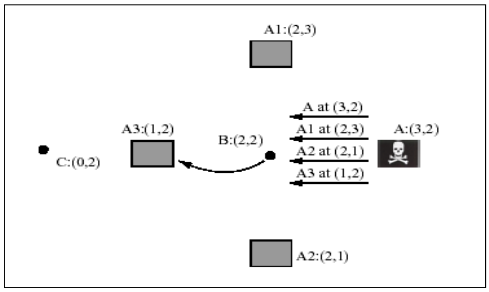
- ***Sinkhole Attacks*** : Adversary take control of all the traffics from a particular area and acts as a (*fake*) sink (i.e. base station). All neighboring nodes forward packets for a base station through the adversary.



A laptop-class adversary using a wormhole to create a sinkhole attack

3.2 Attacks on WSNs (*Cont.*)

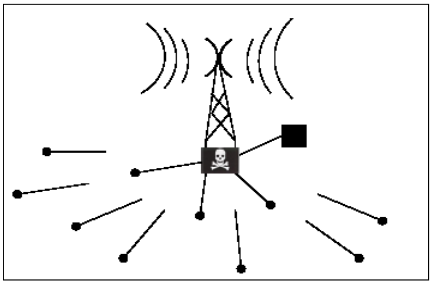
- ***The Sybil Attacks*** : In a Sybil attackIn a Sybil attack, a single node presents multiple identities to other nodes. This can reduce the effectiveness of fault-tolerant schemes. Adversary can be in more than one place at once by using this attack.



Adversary A contains multiple identities (A1, A2, A3) to capture data sending from B to C through A3

3.2 Attacks on WSNs (*Cont.*)

- ***HELLO Flood Attacks*** : A laptop-class attacker broadcasting routing or other information with large enough transmission power could convince every node in the network that the adversary is its neighbor.



HELLO Flood attack against TinyOS

3.2. Attacks on WSNs (*Cont.*)

- Shared key & link layer encryption
  - Prevent outsider attacks, e.g., Sybil attacks, selective forwarding, ACK spoofing
  - Cannot handle insider attacks
    - Wormhole, Hello flood, TinyOS beaconing
- Sybil attack
  - Every node shares a unique secret key with the base station
  - Create pairwise shared key for msg authentication
  - Limit the number of neighbors for a node
- Hello flood attack
  - Verify link bidirectionality
  - Doesn't work if adversary has very sensitive radio

3.2. Attacks on WSNs (*Cont.*)

- Wormhole, sinkhole attack
  - Cryptography may not help directly
  - Good routing protocol design
  - Geographic routing
- Geographic routing
  - Location verification
  - Use fixed topology, e.g., grid structure
- Selective forwarding
  - Multi-path routing
  - Route messages over disjoint or Braided paths
  - Dynamically pick next hop from a set of candidates
  - Measure the trustworthiness of neighbors

3.2. Attacks on WSNs (*Cont.*)

- Authenticated broadcast
  - uTESLA
- Base station floods blacklist
  - Should be authenticated
  - Adversaries must not be able to spoof

3.2 Attacks on WSNs (*Cont.*)

- A summary of different types attacks against existing sensor network routing protocols is shown below :

Protocol	Insecure?	Relevant attacks
TinyOS beaconing	√	Bogus routing information, selective forwarding, sinkholes, Sybil, wormholes, HELLO floods
Directed diffusion and its multipath variant	√	Bogus routing information, selective forwarding, sinkholes, Sybil, wormholes, HELLO floods
Geographic routing (GPSR, GEAR)	√	Bogus routing information, selective forwarding, Sybil
Minimum cost forwarding	√	Bogus routing information, selective forwarding, sinkholes, wormholes, HELLO floods
Clustering based protocols (LEACH, TEEN, PEGASIS)	√	Selective forwarding, HELLO floods
Rumor routing	√	Bogus routing information, selective forwarding, sinkholes, Sybil, wormholes
Energy conserving topology maintenance (SPAN, GAF, CEC, AFECA)	√	Bogus routing information, Sybil, HELLO floods

3.3 Countermeasures for attacks

- ***Outsider Attacks and Link Layer Security*** :
  - Can be prevented by providing link layer data encryption and authentication mechanisms using a globally shared key
  - Replay can be detected by maintaining a monotonically increasing counter with each packet, discard packets contains older value
- ***The Sybil Attacks*** :
  - Replay can be detected by maintaining a monotonically increasing counter with each packet, discard packets contains older value
  - Identity must be verified and a unique symmetric key should be shared

3.3 Countermeasures for attacks (Cont.)

- ***HELLO Flood Attacks*** :
  - Can not be countered by link layer encryption and authentication mechanism
  - Verify the bi-directionality of a link before receive any packet
  - Same measures as described in the Sybil attacks
- ***Wormhole and Sinkhole Attacks*** :
  - Difficult to defend when the two are used in combination
  - Protocols that construct topology initiated by base station are more likely to be attacked
  - Geographic protocol, that construct topology on demand and without initiating from the base station, has less risk of Wormhole or Sinkhole attack