Enhancements of Security in Wireless Sensor Networks

Brian D. Caravantes

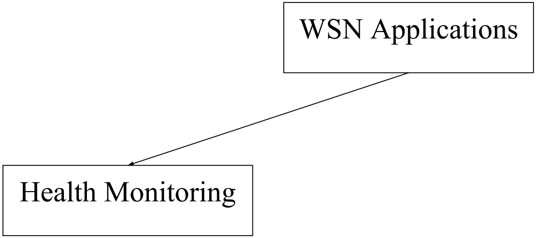
Division of Science and Mathematics
University of Minnesota, Morris
Morris, Minnesota, USA

What are Wireless Sensor Networks?

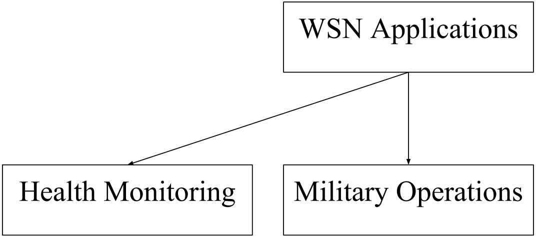
Wireless sensor networks (WSNs)

- Large number of sensor nodes that have computation and communication capabilities.
- Nodes gather and send the data in various settings
- Use the data locally or allows accessed by some user in real-time.

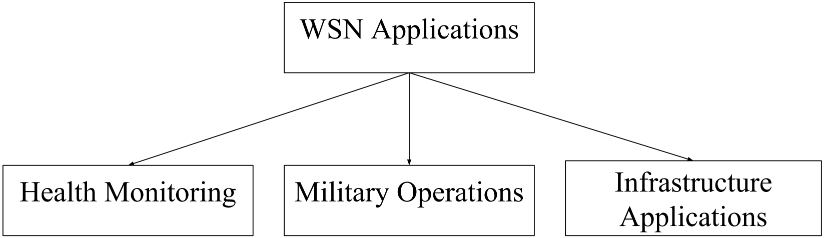
Why should we even care?



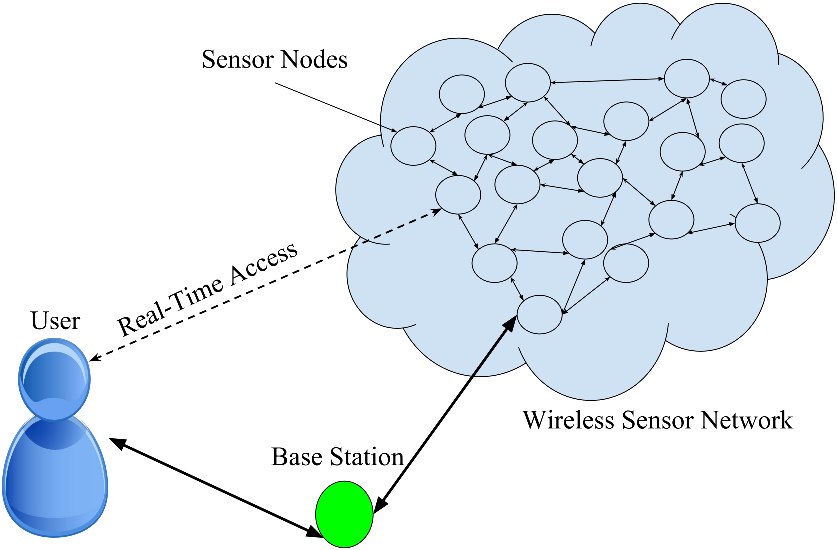
Why should we even care?



Why should we even care?



Why should we even care?



The Security Problem

- Since WSNs consist of such a **large network** of interconnected components thus making it hard to keep each node secure.
- WSNs face **resource restrictions**:
  - storage capacity
  - processing power
  - battery power
- Security Requirements: Data integrity, Confidentiality and Availability

Outline

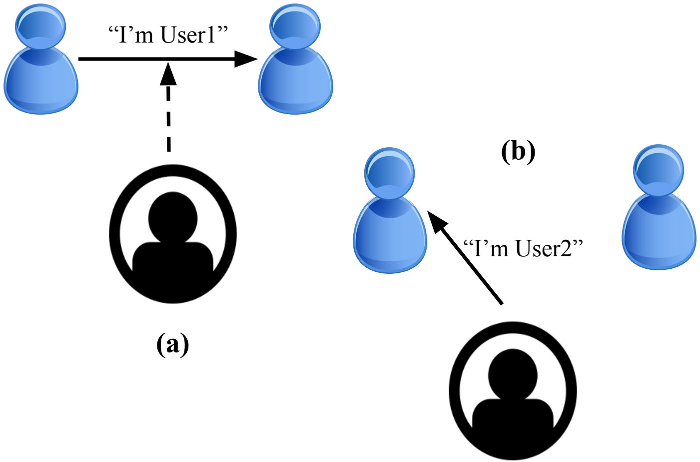
1. Background
2. 4 Phase AA (Anonymous Authentication) Scheme
3. Security Analysis
4. Conclusions

Outline

1. Background
   - Common Attacks on WSNs
   - Cryptography used in WSNs
2. 4 Phase AA (Anonymous Authentication) Scheme
3. Security Analysis
4. Conclusions

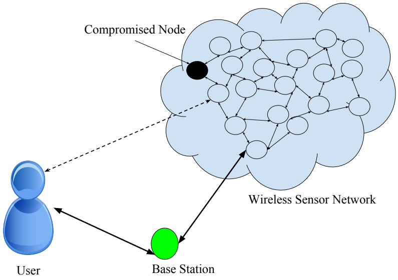
Impersonation Attack

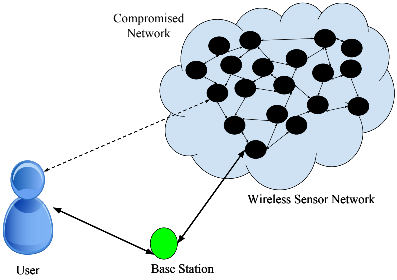
The goal is to access unauthorized information acting as a certain party.



Node Capture Attack

Goal is to get control of any number of nodes, isn't difficult for them to add, remove, or alter information about some node.





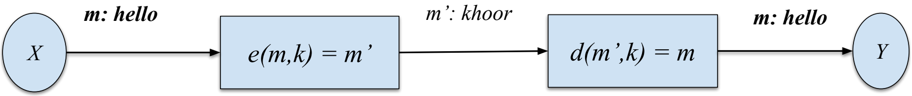
Cryptography used in WSNs

Private Key Cryptography

- Goal is to send messages between parties without any adversary listening
- Uses a shared secret key that allows both parties to encrypt and decrypt a message

Private Key (Symmetric) Cryptography

Key: Data that is used to encrypt and decrypt information.
Encryption and **Decryption:** functions that take in a message and key as parameters and output specific message



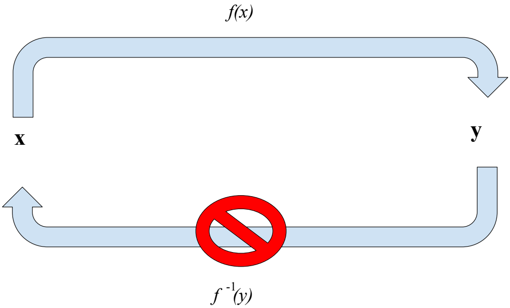
Each letter is equal to its third letter down the alphabet
(a → d, b → e, c → f)

One way hash function

- Denoted by *h(.)* where . are various parameters
- Hash Function: function that can be used to map data of arbitrary size to data of fixed size.
- "One way": Easy to compute in one direction but hard to compute in the other direction.

One way hash function

- Denoted by *h(.)* where . are various parameters
- Hash Function: function that can be used to map data of arbitrary size to data of fixed size.
- "One way": Easy to compute in one direction but hard to compute in the other direction.

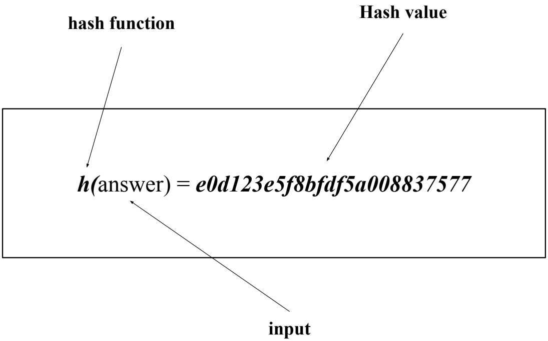


One way hash function

- Returning hash value from function is used to verify some type of information.

One way hash function

- Returning hash value from function is used to verify some type of information.



XOR (Exclusive-OR Operator)

- Denoted by $\oplus$
- Means "one or the other but NOT both"
- Example: 11010 $\oplus$ 10110 = 01100

XOR Truth Table

a	b	a$\oplus$b
0	1	1
1	1	0
0	0	0
1	0	1

XOR (Exclusive-OR Operator)

- Denoted by $\oplus$
- Means "one or the other but NOT both"
- Example: 11010 $\oplus$ 10110 = 01100
- Property:

k $\oplus$ k$\oplus$ m =m $\oplus$ k $\oplus$ k = m

XOR Truth Table

a	b	a$\oplus$b
0	1	1
1	1	0
0	0	0
1	0	1

Concatenation

- Joining multiple given elements
- Denoted by ||
- Example: 11 || 00 = 1100

Outline

1 Background

2 4 Phase AA (Anonymous Authentication) Scheme

3 Security Analysis

4 Conclusions

4 Phase AA Scheme

- Protocol proposed by Gope et al
- **Two factor authentication**: requires more than just username and password for authentication but also some information that only the user has/knows such as some **physical token**

4 Phase AA Scheme

What does this protocol do?

4 Phase AA Scheme

What does this protocol do?

- Provides a secure way for user to register with and authenticate with desired sensor node in network.

4 Phase AA Scheme

What does this protocol do?

- Provides a secure way for user to register with and authenticate with desired sensor node in network.
- Also provides secure way of changing password securely and introducing new new nodes into the network.

4 Phase AA Scheme

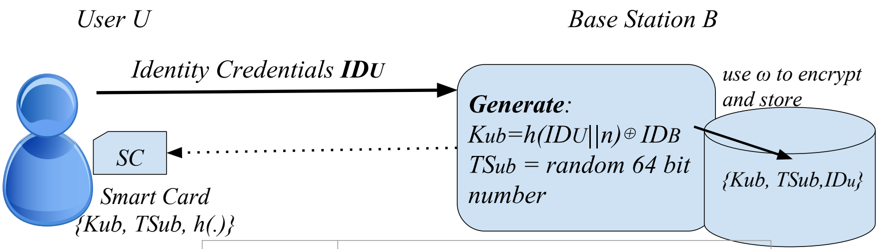
Phases:

1. Phase 1: Registration Phase
2. Phase 2: Anonymous Authentication and Key Exchange
3. Phase 3: Password Renewal Phase
4. Phase 4: Dynamic Node Addition Phase

Phase 1: Registration Phase

- Purpose: sets up the underlying foundation for the construction of the proposed authentication scheme.

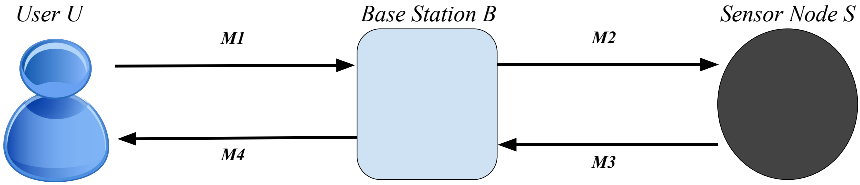
Phase 1: Registration Phase



Kub	Shared Key between U and B
TSub	Transaction Sequence number
n	Randomly generated number only known to U
IDB	Identity of Base Station on hardware
ω	Private Key of B

Phase 2: Anonymous Authentication and Key Exchange

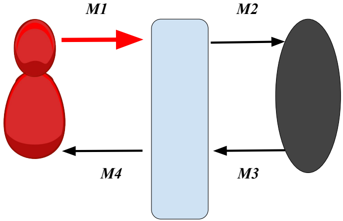
- Purpose: Achieve secure authentication among the user, base station, and the sensor node.



Phase 2: Step 1

1. **Generate:** random number N_U

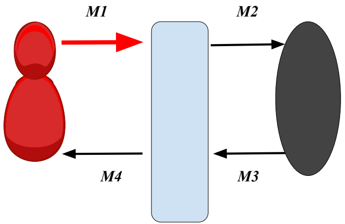
Smart Card has:
K_{ub}: Shared key between user and base station
TS_{ub}: Transaction number
ID_U: User identity



Phase 2: Step 1

1. **Generate:** random number N_U
2. **Derive:**
   - One Time Alias Identity $AID_U = h(ID_U||K_{ub}||N_U||S)$

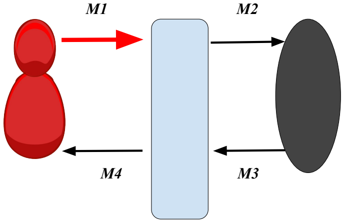
Smart Card has:
K_{ub}: Shared key between user and base station
TS_{ub}: Transaction number
ID_U: User identity



Phase 2: Step 1

1. **Generate:** random number N_U
2. **Derive:**
   - One Time Alias Identity $AID_U = h(ID_U||K_{ub}||N_U||S)$
   - $N_x = K_{ub} \oplus N_U$

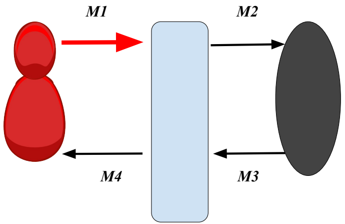
Smart Card has:
K_{ub}: Shared key between user and base station
TS_{ub}: Transaction number
ID_U: User identity



Phase 2: Step 1

1. **Generate:** random number N_U
2. **Derive:**
   - One Time Alias Identity $AID_U = h(ID_U||K_{ub}||N_U||S)$
   - $N_x = K_{ub} \oplus N_U$
   - $V_1 = h(AID_U||K_{ub}||N_x||S)$

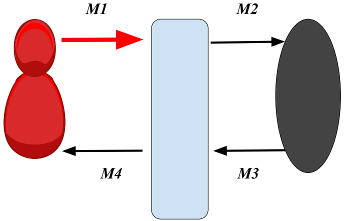
Smart Card has:
K_{ub}: Shared key between user and base station
TS_{ub}: Transaction number
ID_U: User identity



Phase 2: Step 1

1. **Generate:** random number N_U
2. **Derive:**
   - One Time Alias Identity $AID_U = h(ID_U || K_{ub} || N_U || S)$
   - $N_x = K_{ub} \oplus N_U$
   - $V_1 = h(AID_U || K_{ub} || N_x || S)$
3. **Create M1**

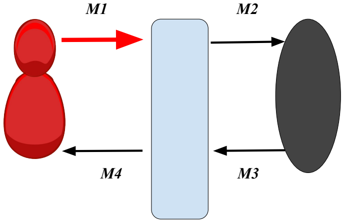
Smart Card has:
K_{ub}: Shared key between user and base station
TS_{ub}: Transaction number
ID_U: User identity



Phase 2: Step 1

1. **Generate:** random number N_U
2. **Derive:**
   - One Time Alias Identity $AID_U = h(ID_U || K_{ub} || N_U || S)$
   - $N_x = K_{ub} \oplus N_U$
   - $V_1 = h(AID_U || K_{ub} || N_x || S)$
3. **Create M1**
   - M1: $\{AID_U, N_x, V_1, S, TS_{ub}\}$

Smart Card has:
K_{ub}: Shared key between user and base station
TS_{ub}: Transaction number
ID_U: User identity



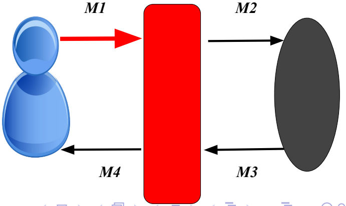
Phase 2: Step 1

Upon receiving M1: $\{AID_U, N_x, V_1, S, TS_{ub}\}$

B:

1. **Checks:** TS_{ub} valid?

Base station has:
K_{ub}: Shared key between user and base station
TS_{ub}: Transaction number
ID_U: User identity
ID_B Base station identity
V_1: Validation message
AID_U: One-time alias identity

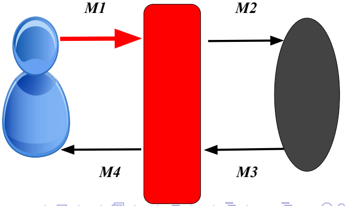


Phase 2: Step 1

Upon receiving M1: $\{AID_U, N_x, V_1, S, TS_{ub}\}$
B:

1. **Checks:** TS_{ub} valid?
   - Will check if TS_{ub} in **M1** is equal to TS_{ub} stored in B's database

Base station has:
K_{ub}: Shared key between user and base station
TS_{ub}: Transaction number
ID_U: User identity
ID_B Base station identity
V_1: Validation message
AID_U: One-time alias identity

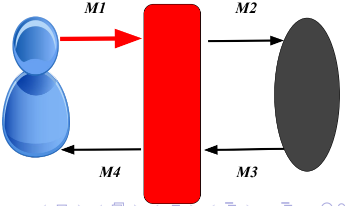


Phase 2: Step 1

Upon receiving M1: $\{AID_U, N_x, V_1, S, TS_{ub}\}$
B:

1. **Checks:** TS_{ub} valid?
   - Will check if TS_{ub} in **M1** is equal to TS_{ub} stored in B's database
2. **Decrypt:** ID_U and shared key K_{ub} with validated TS_{ub} and private key ω

Base station has:
K_{ub}: Shared key between user and base station
TS_{ub}: Transaction number
ID_U: User identity
ID_B Base station identity
V_1: Validation message
AID_U: One-time alias identity

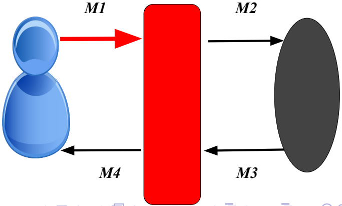


Phase 2: Step 1

Upon receiving M1: $\{AID_U, N_x, V_1, S, TS_{ub}\}$
B:

1. **Checks:** TS_{ub} valid?
   - Will check if TS_{ub} in **M1** is equal to TS_{ub} stored in B's database
2. **Decrypt:** ID_U and shared key K_{ub} with validated TS_{ub} and private key ω
   - Now B knows who the user is.

Base station has:
K_{ub}: Shared key between user and base station
TS_{ub}: Transaction number
ID_U: User identity
ID_B Base station identity
V_1: Validation message
AID_U: One-time alias identity



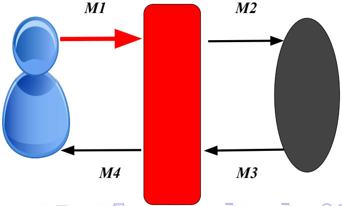
Phase 2: Step 1

Upon receiving M1: $\{AID_U, N_x, V_1, S, TS_{ub}\}$

B:

1. **Checks:** TS_{ub} valid?
   - Will check if TS_{ub} in **M1** is equal to TS_{ub} stored in B's database
2. **Decrypt:** ID_U and shared key K_{ub} with validated TS_{ub} and private key ω
   - Now B knows who the user is.
3. **Check if:**
   - $V_1 =$

Base station has:
K_{ub}: Shared key between user and base station
TS_{ub}: Transaction number
ID_U: User identity
ID_B Base station identity
V_1: Validation message
AID_U: One-time alias identity

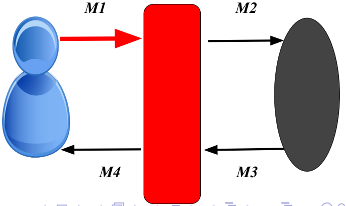


Phase 2: Step 1

Upon receiving M1: $\{AID_U, N_x, V_1, S, TS_{ub}\}$
B:

1. **Checks:** TS_{ub} valid?
   - Will check if TS_{ub} in **M1** is equal to TS_{ub} stored in B's database
2. **Decrypt:** ID_U and shared key K_{ub} with validated TS_{ub} and private key ω
   - Now B knows who the user is.
3. **Check if:**
   - $V_1 = h(AID_U || K_{ub} || N_x || S)$

Base station has:
K_{ub}: Shared key between user and base station
TS_{ub}: Transaction number
ID_U: User identity
ID_B Base station identity
V_1: Validation message
AID_U: One-time alias identity

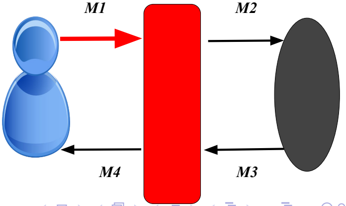


Phase 2: Step 1

Upon receiving M1: $\{AID_U, N_x, V_1, S, TS_{ub}\}$
B:

1. **Checks:** TS_{ub} valid?
   - Will check if TS_{ub} in **M1** is equal to TS_{ub} stored in B's database
2. **Decrypt:** ID_U and shared key K_{ub} with validated TS_{ub} and private key ω
   - Now B knows who the user is.
3. **Check if:**
   - $V_1 = h(AID_U || K_{ub} || N_x || S)$
   - $AID_U =$

Base station has:
K_{ub}: Shared key between user and base station
TS_{ub}: Transaction number
ID_U: User identity
ID_B Base station identity
V_1: Validation message
AID_U: One-time alias identity

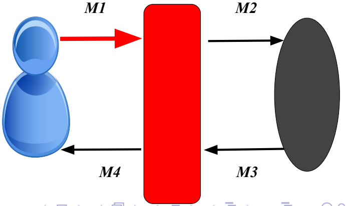


Phase 2: Step 1

Upon receiving M1: $\{AID_U, N_x, V_1, S, TS_{ub}\}$
B:

1. **Checks:** TS_{ub} valid?
   - Will check if TS_{ub} in **M1** is equal to TS_{ub} stored in B's database
2. **Decrypt:** ID_U and shared key K_{ub} with validated TS_{ub} and private key ω
   - Now B knows who the user is.
3. **Check if:**
   - $V_1 = h(AID_U || K_{ub} || N_x || S)$
   - $AID_U = h(ID_U || K_{ub} || N_U || S)$

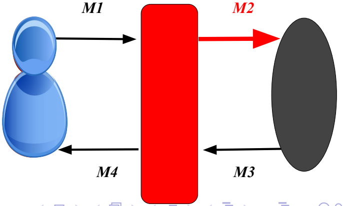
Base station has:
K_{ub}: Shared key between user and base station
TS_{ub}: Transaction number
ID_U: User identity
ID_B Base station identity
V_1: Validation message
AID_U: One-time alias identity



Phase 2: Step 2

1. **Generate:**
   - Session Key *SK* (randomly)

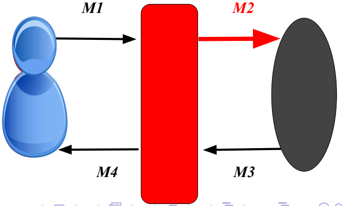
Base station has:
K_{ub}: Shared key between base station and sensor node
TS_{ub}: Transaction number
ID_U: User identity
ID_B Base station identity
V_1: Validation message
AID_U: One-time alias identity



Phase 2: Step 2

1. **Generate:**
   - Session Key SK (randomly)
   - Time Stamp T

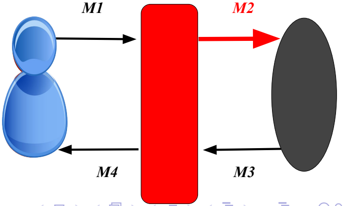
Base station has:
K_{ub}: Shared key between base station and sensor node
TS_{ub}: Transaction number
ID_U: User identity
ID_B Base station identity
V_1: Validation message
AID_U: One-time alias identity



Phase 2: Step 2

1. **Generate:**
   - Session Key SK (randomly)
   - Time Stamp T
2. **Encrypt:** SK such that:
   - $SK' = h(K_{bs}) \oplus SK$

Base station has:
K_{ub}: Shared key between base station and sensor node
TS_{ub}: Transaction number
ID_U: User identity
ID_B Base station identity
V_1: Validation message
AID_U: One-time alias identity



Phase 2: Step 2

1. **Generate:**
   - Session Key SK (randomly)
   - Time Stamp T
2. **Encrypt:** SK such that:
   - $SK' = h(K_{bs}) \oplus SK$
3. **Compute:**
   - $V_2 = h(AID_U || SK' || T || K_{bs})$

Base station has:

K_{ub}: Shared key between base station and sensor node

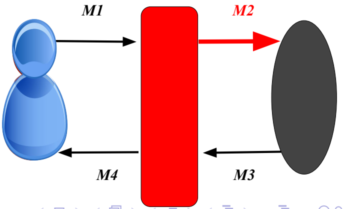
TS_{ub}: Transaction number

ID_U: User identity

ID_B Base station identity

V_1: Validation message

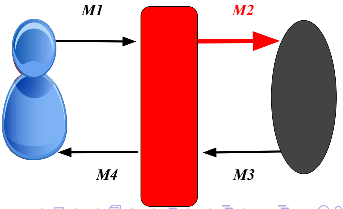
AID_U: One-time alias identity



Phase 2: Step 2

1. **Generate:**
   - Session Key *SK* (randomly)
   - Time Stamp *T*
2. **Encrypt:** *SK* such that:
   - $SK' = h(K_{bs}) \oplus SK$
3. **Compute:**
   - $V_2 = h(AID_U||SK'||T||K_{bs})$
4. **Create M2:**

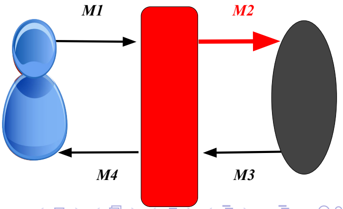
Base station has:
K_{ub}: Shared key between base station and sensor node
TS_{ub}: Transaction number
ID_U: User identity
ID_B Base station identity
V_1: Validation message
AID_U: One-time alias identity



Phase 2: Step 2

1. **Generate:**
   - Session Key SK (randomly)
   - Time Stamp T
2. **Encrypt:** SK such that:
   - $SK' = h(K_{bs}) \oplus SK$
3. **Compute:**
   - $V_2 = h(AID_U||SK'||T||K_{bs})$
4. **Create M2:**
   - $M2$: $\{AID_U, SK', T, K_{bs}\}$

Base station has:
K_{ub}: Shared key between base station and sensor node
TS_{ub}: Transaction number
ID_U: User identity
ID_B Base station identity
V_1: Validation message
AID_U: One-time alias identity



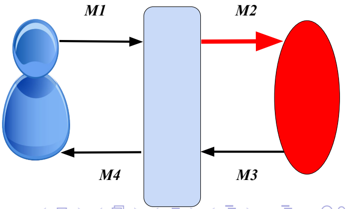
Phase 2: Step 2

Sensor node receives:

$M2$: $\{AID_U, SK', T, K_{bs}\}$

1. **Verify if:**
   - T Valid

Sensor node has:
K_{ub}: Shared key between user and base station
SK': Encrypted session key
ID_U: User identity
ID_B Base station identity
V_2: Validation message
AID_U: One-time alias identity



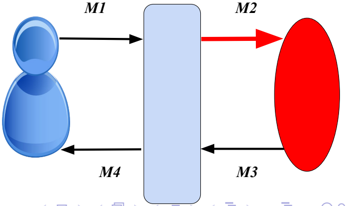
Phase 2: Step 2

Sensor node receives:

$M2$: $\{AID_U, SK', T, K_{bs}\}$

1. **Verify if:**
   - T Valid
   - $V_2 =$

Sensor node has:
K_{ub}: Shared key between user and base station
SK': Encrypted session key
ID_U: User identity
ID_B Base station identity
V_2: Validation message
AID_U: One-time alias identity



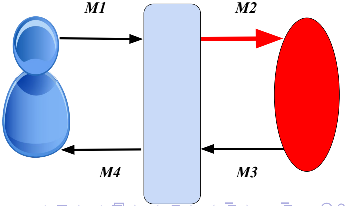
Phase 2: Step 2

Sensor node receives:

$M2$: $\{AID_U, SK', T, K_{bs}\}$

1. **Verify if:**
   - T Valid
   - $V_2 = h(AID_U||SK'||T||V_2)$

Sensor node has:
K_{ub}: Shared key between user and base station
SK': Encrypted session key
ID_U: User identity
ID_B Base station identity
V_2: Validation message
AID_U: One-time alias identity



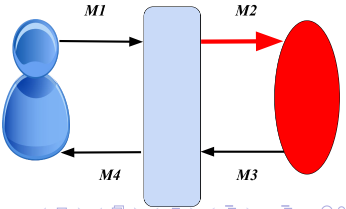
Phase 2: Step 2

Sensor node receives:

$M2$: $\{AID_U, SK', T, K_{bs}\}$

1. **Verify if:**
   - T Valid
   - $V_2 = h(AID_U||SK'||T||V_2)$
2. **Decrypt:** SK' into SK

Sensor node has:
K_{ub}: Shared key between user and base station
SK': Encrypted session key
ID_U: User identity
ID_B Base station identity
V_2: Validation message
AID_U: One-time alias identity



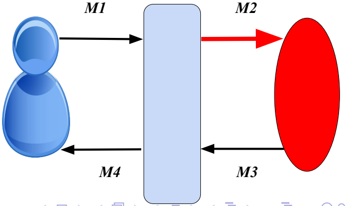
Phase 2: Step 2

Sensor node receives:

$M2$: $\{AID_U, SK', T, K_{bs}\}$

1. **Verify if:**
   - T Valid
   - $V_2 = h(AID_U || SK' || T || V_2)$
2. **Decrypt:** SK' into SK
   - $SK = h(K_{bs}) \oplus SK'$

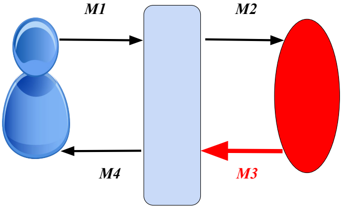
Sensor node has:
K_{ub}: Shared key between user and base station
SK': Encrypted session key
ID_U: User identity
ID_B Base station identity
V_2: Validation message
AID_U: One-time alias identity



Phase 2: Step 3

1. **Generate:** New timestamp T'

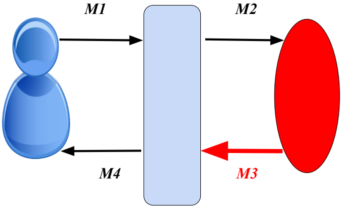
Sensor node has:
K_{bs}: Shared key between base station and sensor
SK': Encrypted session key
S: Sensor node identity
AID_U: One-time alias identity



Phase 2: Step 3

1. **Generate:** New timestamp T'
2. **Compute:**
   - $V_3 = h(SK'||K_{bs}||S||T')$

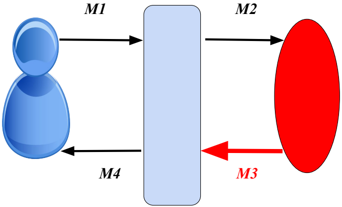
Sensor node has:
K_{bs}: Shared key between base station and sensor
SK': Encrypted session key
S: Sensor node identity
AID_U: One-time alias identity



Phase 2: Step 3

1. **Generate:** New timestamp T'
2. **Compute:**
   - $V_3 = h(SK'||K_{bs}||S||T')$
3. **Create M3:**

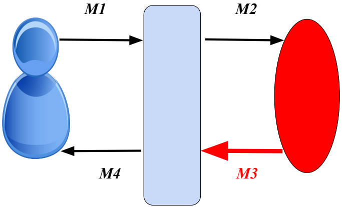
Sensor node has:
K_{bs}: Shared key between base station and sensor
SK': Encrypted session key
S: Sensor node identity
AID_U: One-time alias identity



Phase 2: Step 3

1. **Generate:** New timestamp T'
2. **Compute:**
   - $V_3 = h(SK'||K_{bs}||S||T')$
3. **Create M3:**
   - $M3$: $\{T', S, V_3\}$

Sensor node has:
K_{bs}: Shared key between base station and sensor
SK': Encrypted session key
S: Sensor node identity
AID_U: One-time alias identity

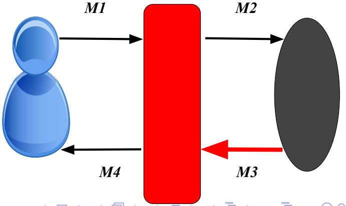


Phase 2: Step 3

B: receives $M3$: $\{T', S, V_3\}$

1. **Check:** T'

Base station has:
K_{bs}: Shared key between base station and sensor
K_{ub}: Shared key between user and base station
N_U: Randomly generated number by user
ID_U: User identity

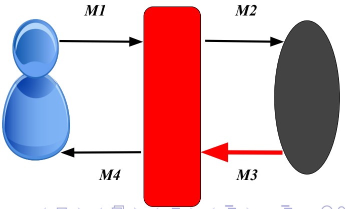


Phase 2: Step 3

B: receives $M3 : \{T', S, V_3\}$

1. **Check:** T'
2. **Verify if:** $V_3 =$

Base station has:
K_{bs}: Shared key between base station and sensor
K_{ub}: Shared key between user and base station
N_U: Randomly generated number by user
ID_U: User identity

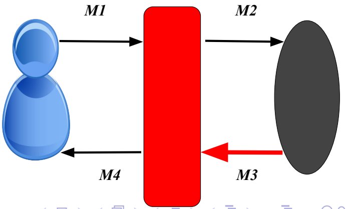


Phase 2: Step 3

B: receives $M3 : \{T', S, V_3\}$

1. **Check:** T'
2. **Verify if:** $V_3 = h(SK'||K_{bs}||S||T')$

Base station has:
K_{bs}: Shared key between base station and sensor
K_{ub}: Shared key between user and base station
N_U: Randomly generated number by user
ID_U: User identity



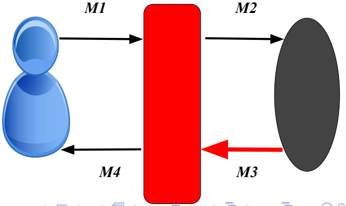
Phase 2: Step 3

B: receives $M3 : \{T', S, V_3\}$

1. **Check:** T'
2. **Verify if:** $V_3 = h(SK' || K_{bs} || S || T')$
3. **Encrypt:**
   - TS_{ub} so that

     $TS = h(K_{ub} || ID_U || N_U) \oplus TS_{ub}$

Base station has:
K_{bs}: Shared key between base station and sensor
K_{ub}: Shared key between user and base station
N_U: Randomly generated number by user
ID_U: User identity



Phase 2: Step 3

B: receives $M3 : \{T', S, V_3\}$

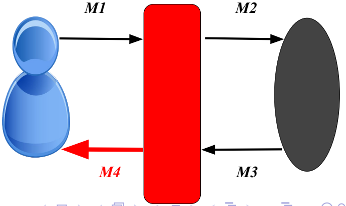
1. **Check:** T'
2. **Verify if:** $V_3 = h(SK'||K_{bs}||S||T')$
3. **Encrypt:**
   - TS_{ub} so that

     $TS = \text{h}(K_{ub}||ID_U||N_U) \oplus TS_{ub}$
   - $SK'' = h(K_{ub}||ID_U||N_U) \oplus TS_{ub}$

Phase 2: Step 4

1. **Compute:**
   - $V_4 = h(SK''||K_{ub}||N_U||TS)$

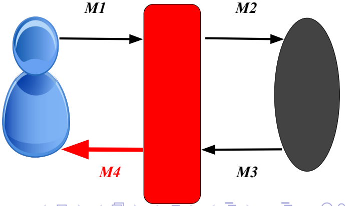
Base station has:
K_{ub}: Shared key between user and base station
SK'': Encrypted session key
ID_U: User identity
TS: Encrypted transaction number
N_U: Randomly generated number by user



Phase 2: Step 4

1. **Compute:**
   - $V_4 = h(SK''||K_{ub}||N_U||TS)$
2. **Create M4:**

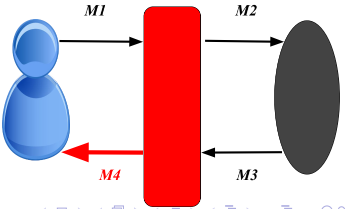
Base station has:
K_{ub}: Shared key between user and base station
SK'': Encrypted session key
ID_U: User identity
TS: Encrypted transaction number
N_U: Randomly generated number by user



Phase 2: Step 4

1. **Compute:**
   - $V_4 = h(SK''||K_{ub}||N_U||TS)$
2. **Create M4:**
   - $M4$: $\{SK'', TS, V_4\}$

Base station has:
K_{ub}: Shared key between user and base station
SK'': Encrypted session key
ID_U: User identity
TS: Encrypted transaction number
N_U: Randomly generated number by user

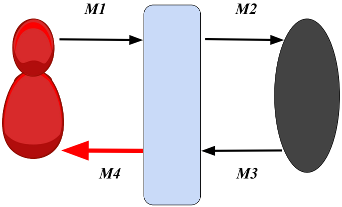


Phase 2: Step 4

SC recieves $M4$: $\{SK'', TS, V_4\}$

1. **Verify:** V_4 =

Smart Card has:
K_{ub}: Shared key between user and base station
TS_{ub}: Transaction number
ID_U: User identity
N_U: Randomly generated number by user

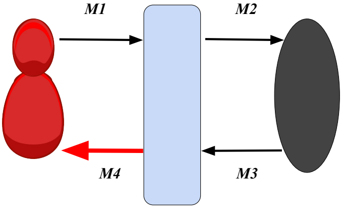


Phase 2: Step 4

SC recieves $M4$: $\{SK'', TS, V_4\}$

1. **Verify:** $V_4 = h(SK''||K_{bs}||N_U||TS)$**??**

Smart Card has:
K_{ub}: Shared key between user and base station
TS_{ub}: Transaction number
ID_U: User identity
N_U: Randomly generated number by user

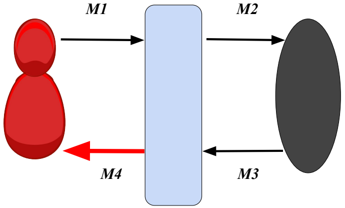


Phase 2: Step 4

SC recieves *M4*: $\{SK'', TS, V_4\}$

1. **Verify:** $V_4 = h(SK''||K_{bs}||N_U||TS)$**??**
2. **Decrypt:** TS
   - $TS_{ub} = h(K_{ub}||ID_U||N_U) \oplus TS$

Smart Card has:
K_{ub}: Shared key between user and base station
TS_{ub}: Transaction number
ID_U: User identity
N_U: Randomly generated number by user

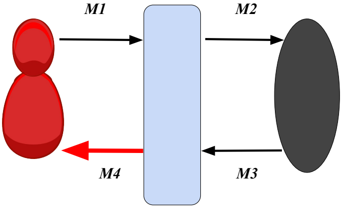


Phase 2: Step 4

SC recieves *M*4: $\{SK'', TS, V_4\}$

1. **Verify:** $V_4 = h(SK''||K_{bs}||N_U||TS)$**??**
2. **Decrypt:** *TS*
   - $TS_{ub} = h(K_{ub}||ID_U||N_U) \oplus TS$
   - *SC* stores TS_{ub} in its memory for future authentication.

Smart Card has:
K_{ub}: Shared key between user and base station
TS_{ub}: Transaction number
ID_U: User identity
N_U: Randomly generated number by user



Phase 2: Step 4

SC recieves $M4$: $\{SK'', TS, V_4\}$

1. **Verify:** $V_4 = h(SK''||K_{bs}||N_U||TS)$**??**
2. **Decrypt:** TS
    - $TS_{ub} = h(K_{ub}||ID_U||N_U) \oplus TS$
    - SC stores TS_{ub} in its memory for future authentication.

User is now authenticated and free to communicate with said node S.

Phase 3 and Phase 4

Phase 3: Password Renewel Phase

- Purpose: Provide secure manner for user to change password on smart card
- Unlike most AA schemes the user need not communicate with B and is free to change his/her password on the smart card SC

Phase 4: Dynamic Node Addition Phase

- Purpose: Provide a secure manner of adding new nodes into the network

Outline

1 Background

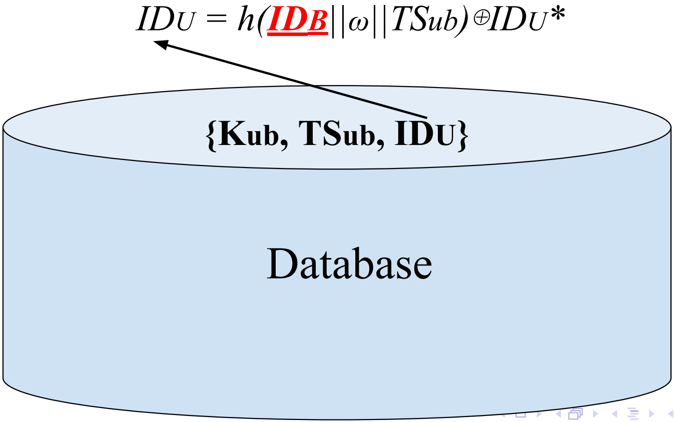
2 4 Phase AA (Anonymous Authentication) Scheme

3 Security Analysis
- Resilience Against Key Compromise Impersonation Attack
- Resilience Against Node Capturing Attack

4 Conclusions

Resilience Against Key Compromise Impersonation Attack

Suppose an adversary *A* has obtained the servers' (*B*) secret key ω and the encoded parameters from the servers database.

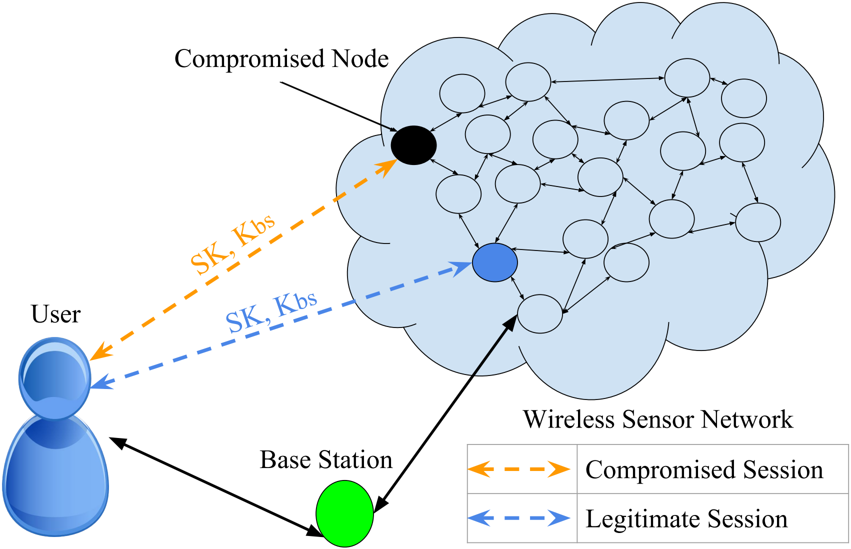


Resilience Against Node Capturing Attack

How do captured nodes affect entire network?

Suppose a compromised node has acquired the private key ω and session key SK.

Resilience Against Node Capturing Attack



Outline

1. Background
2. 4 Phase AA (Anonymous Authentication) Scheme
3. Security Analysis
4. Conclusions

How does this protocol enhance security in WSNs?

- Data Integrity: One of the main ways data remains unaltered through transmission is defense against node capture attack.

How does this protocol enhance security in WSNs?

- Data Integrity: One of the main ways data remains unaltered through transmission is defense against node capture attack.
- Confidentiality: Since the protocol defends against an impersonation of the user then we can say their identity will stay secure.

How does this protocol enhance security in WSNs?

- Data Integrity: One of the main ways data remains unaltered through transmission is defense against node capture attack.
- Confidentiality: Since the protocol defends against an impersonation of the user then we can say their identity will stay secure.
- Availability: Once the user is securely authenticated through our protocol any authorized user can acquire their data in real-time.

Acknowledgments

Thank you to Nic McPhee and Elena Machkasova for feedback and help during process. As well as peers that gave me feedback.

Thanks for coming!

Questions?