

Realizing Robust User Authentication in Sensor Networks

Zinaida Benenson

joint work with **Nils Gedicke, Ossi Raivio**

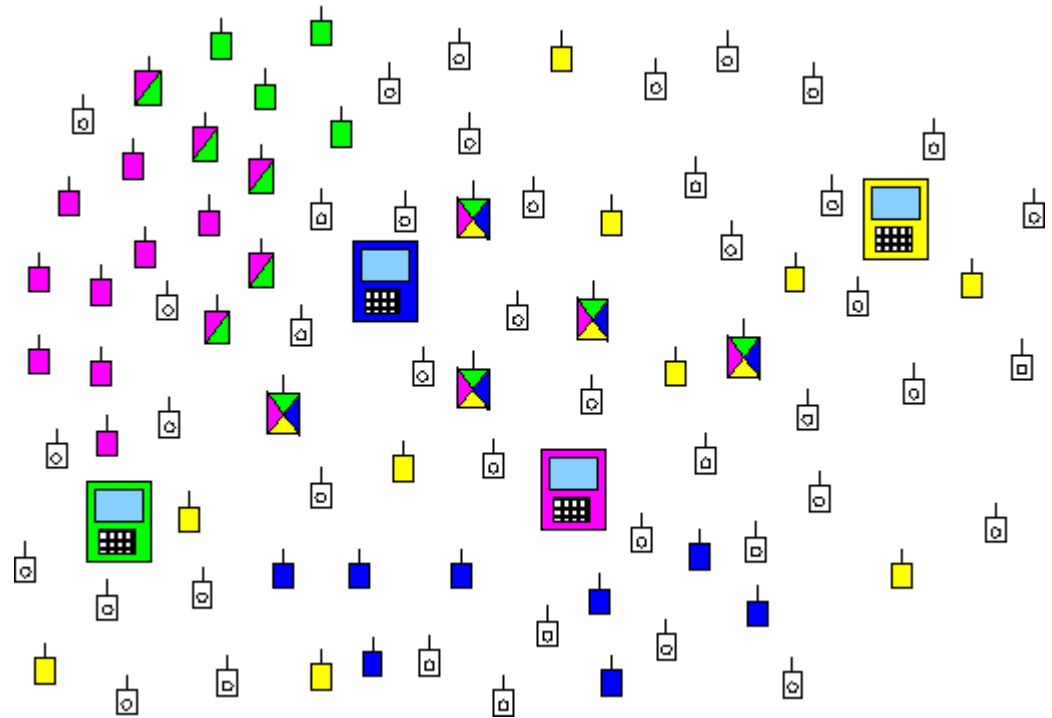
RWTH Aachen University
Chair of Computer Science 4
Communication and Distributed Systems

Our Scenario

let's be futuristic ;-)

WSN:

- large geographic area
- lots of users
- services:
 - user → WSN: queries
 - WSN → user: data
- users pay for services



query: secret + **authenticated** + fresh

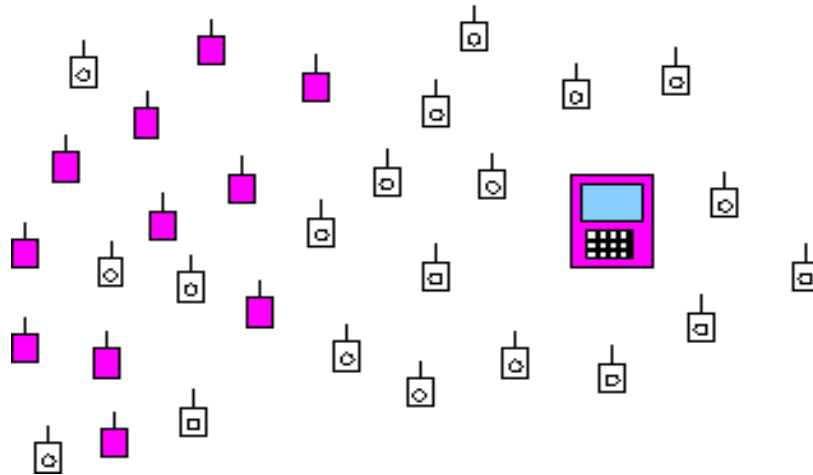
answer: secret + authenticated + fresh

Authenticated Querying: Definition

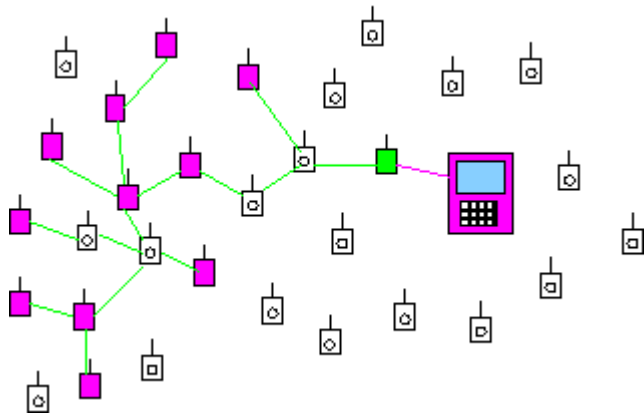
S_q := set of all sensors which must process q

WSN enables **authenticated querying**:

- *Safety*: If a sensor $s \in S_q$ processes a query q , then q was posted by a legitimate user U
- *Liveness*: Any query q posted by a legitimate user U is processed by at least all sensors $s \in S_q$



Authenticated Querying: 1st attempt

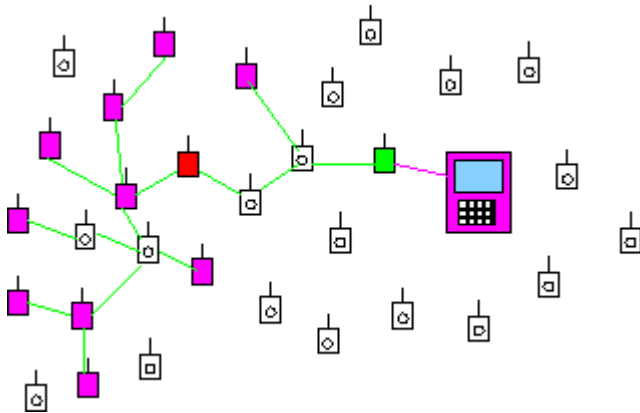


U ↔ **sink**: mutual authentication, session key
sink ↔ **sensors**: symm. key **k**

Impact of node capture (1)

node capture?!

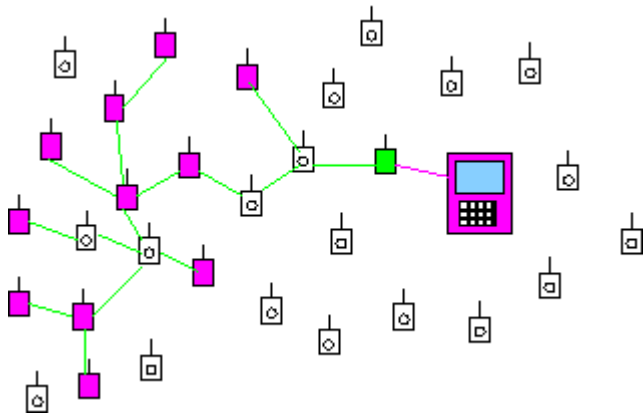
gaining full control by a physical attack



$U \leftrightarrow$ sink: mutual authentication, session key
sink \leftrightarrow sensors: symm. key k

\Rightarrow **any** sensor which knows k can send queries

Authenticated Querying: 2nd attempt



U ↔ sink: mutual authentication, session key

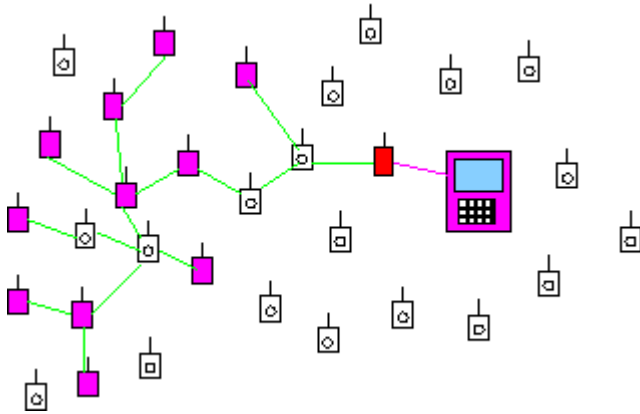
sink ↔ sensors: **authenticated multicast**

- μ TESLA (Perrig et al. 2001)
- inexpensive digital signatures (Seys et al. 2005)

⇒ **only sink** can send queries

Impact of node capture (2)

capture arbitrary node = capture the sink



U ↔ sink: mutual authentication, session key

sink ↔ sensors: **authenticated multicast**

- μ TESLA [Perrig et al. 2001]
- inexpensive digital signatures [Seys et al. 2005]

⇒ any node can be the sink

⇒ any node can send arbitrary queries

cannot rely on a single node to start a query

Authenticated Querying: Idea

withstand capture of up to t nodes

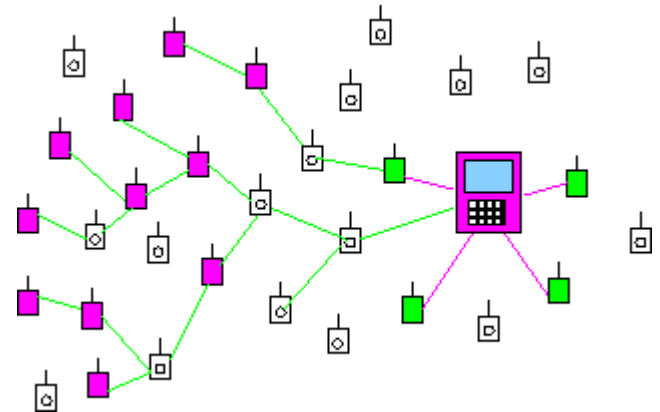
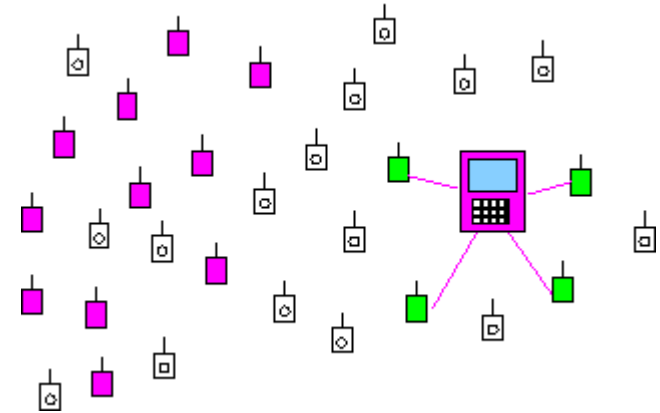
1. Robust secure channel setup user \leftrightarrow WSN

- mutually authenticated key establishment
- lots of users \Rightarrow public key cryptography (cannot store symm. keys of all users)
- SSL-like protocols

2. Authenticated query forwarding

- append additional authentication information to the queries
- other sensors can verify that the user authenticated to at least $t+1$ sensors

$t = 3$

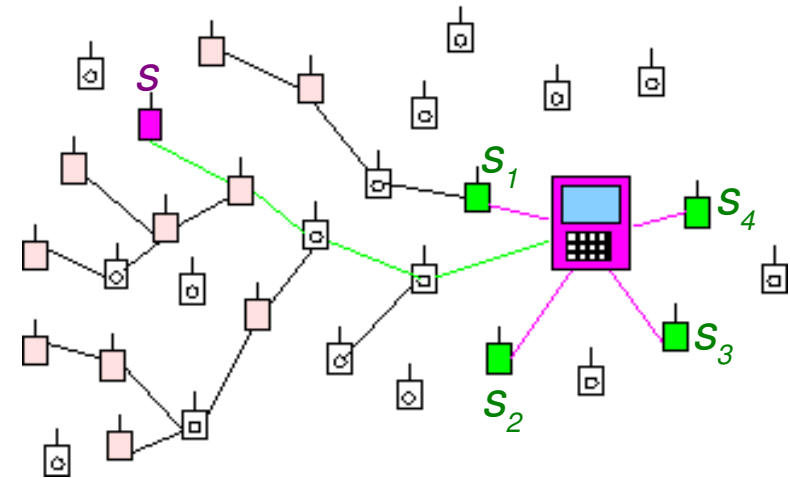


Authenticated Query Forwarding: Example 1

authenticated query for a single sensor s

$t = 3$

- $U \rightarrow s_1, s_2, s_3, s_4 : q$
- $U \leftarrow s_i : m_{i,s} = \text{MAC}(k_{i,s}, q)$
- $U \rightarrow s : m_{1,s} | m_{2,s} | m_{3,s} | m_{4,s} | q$



Authenticated Query Forwarding: Example 2

using interleaved message authentication

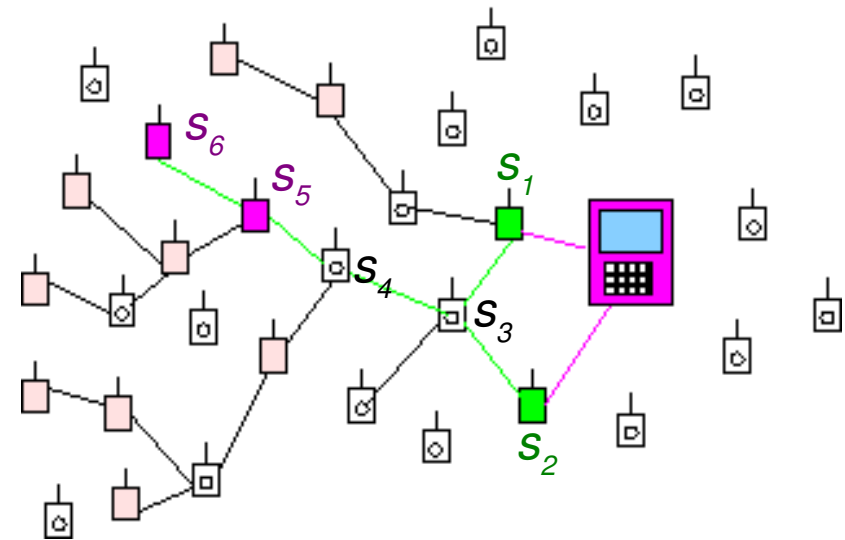
[Vogt 2004, Zhu et al. 2004]

$t = 1$

query considered legitimate
if it is approved by at least two nodes

- $U \rightarrow s_1, s_2 : q$
- $s_1 \rightarrow s_3 : m_{1,3} | m_{1,4} | q$
- $s_2 \rightarrow s_3 : m_{2,3} | m_{2,4} | q$
- $s_3 \rightarrow s_4 : m_{2,4} | m_{3,4} | m_{3,5} | q$
- $s_4 \rightarrow s_5 : m_{3,5} | m_{4,5} | m_{4,6} | q$
- $s_5 \rightarrow s_6 : m_{4,6} | m_{5,6} | m_{5,x} | q$

...



Robust User Authentication

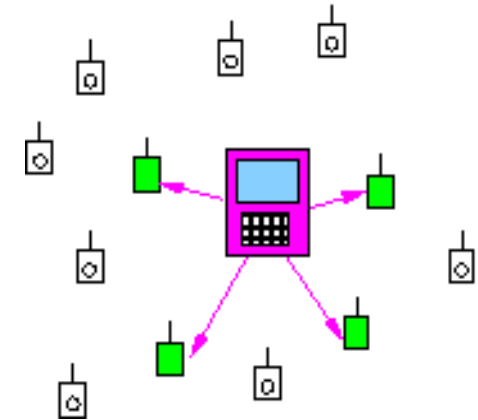
Implementation: user authenticates to n sensor nodes

EccM library for elliptic curve cryptography
[D. Malan et al. 2004]

TelosB motes

Why?

- public key cryptography feasible?
- use EccM library to implement an **advanced** crypto protocol?
- parallel protocol execution with several motes?



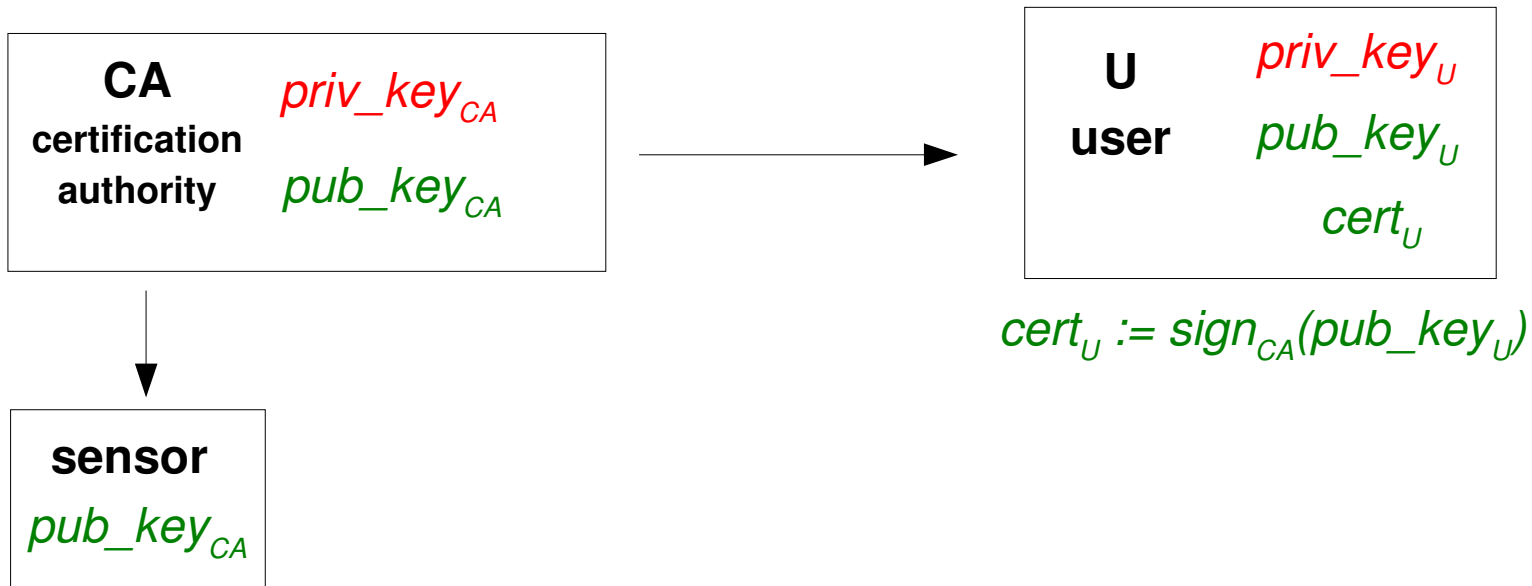
Choice of a Public Key Cryptosystem

	private key operations: <ul style="list-style-type: none">• signing• (decryption)	public key operations: <ul style="list-style-type: none">• signature verification• (encryption)
RSA	very slow	very fast
ECC	fast	fast, but twice slower than ECC private key operations

target **mutual** authentication \Rightarrow **ECC**

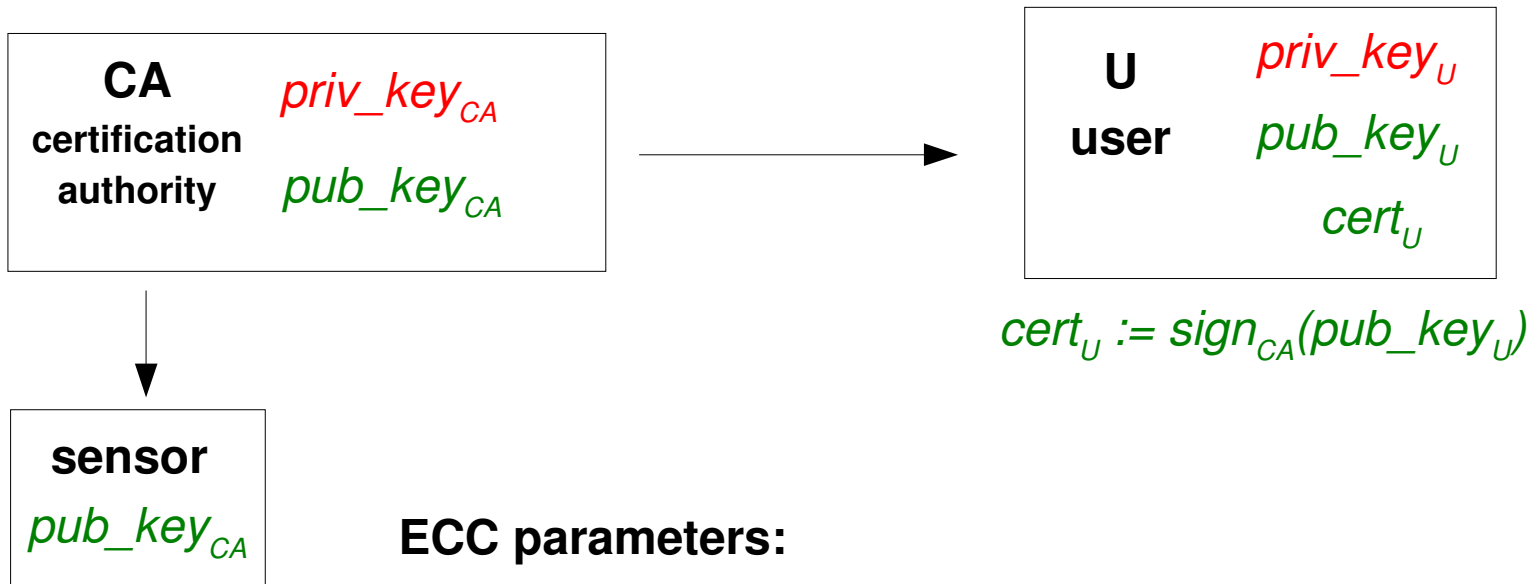
System Setup and Assumptions

PKI for ECC



System Setup and Assumptions

PKI for ECC



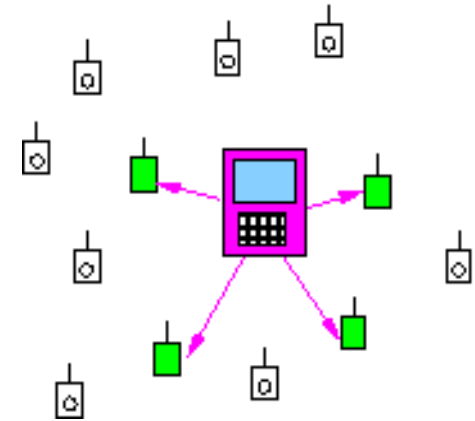
ECC parameters:

- private key length: 163 bit
- public key: 2x163 bit
- Nyberg-Rueppel signature on a 163-bit value: 2x163
- User's certificate = signed public key: 4x163 bit

System Setup and Assumptions

System parameters:

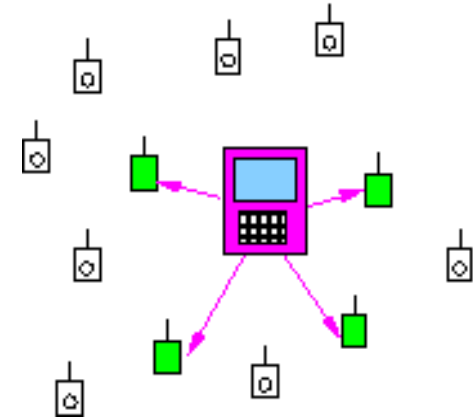
- n sensors in the comm. range of the user
- t sensors can be captured
- m sensors required to authenticate the user
- $t < m \leq n - t$



User Authentication Protocol

standard authentication using digital signatures

1. $U \rightarrow WSN: (U, cert_U)$
2. $\forall 1 \leq i \leq m \ s_i \rightarrow U: (s_i, nonce_i)$
random backoff $(0, 1, \dots, 255) \text{ ms}$
3. $U \rightarrow s_i : proof = sign_U(h(U, s_i, nonce_i))$
 $h = \text{hash function}$
4. $\forall 1 \leq i \leq m$
 $s_i: verify(cert_U) = pub_key_U$
 $s_i: verify(proof = ? sign_U(h(U, s_i, nonce_i)))$



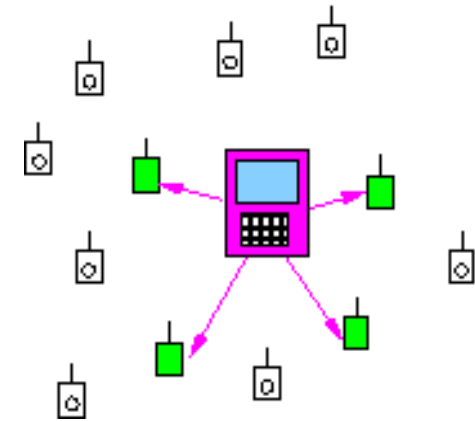
Experimental Results

protocol memory usage: 45.5 kB ROM, 2 kB RAM

implemented Nyberg-Rueppel signature using EccM library

EccM on MICA2: ≈ 34 sec for point multiplication

EccM on TelosB: ≈ 1 min (!) for point multiplication



Experimental Results

protocol memory usage: 45.5 kB ROM, 2 kB RAM

implemented Nyberg-Rueppel signature using EccM library

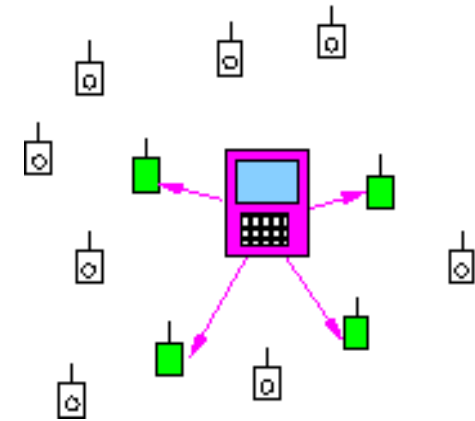
EccM on MICA2: ≈ 34 sec for point multiplication

EccM on TelosB: ≈ 1 min (!) for point multiplication

each sensor has to do:

- certificate verification = 4 point multiplications
- signed nonce verification = 2 point multiplications

⇒ protocol execution time over 6 min!



Experimental Results

protocol memory usage: 45.5 kB ROM, 2 kB RAM

implemented Nyberg-Rueppel signature using EccM library

EccM on MICA2: ≈ 34 sec for point multiplication

EccM on TelosB: ≈ 1 min (!) for point multiplication

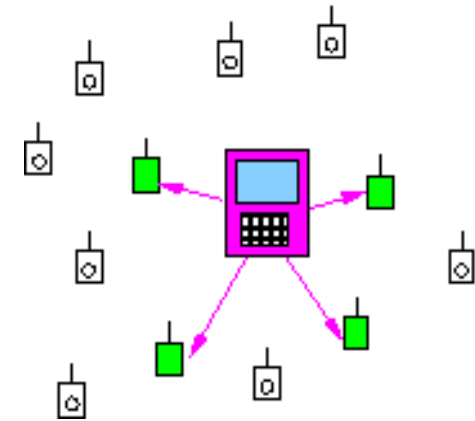
each sensor has to do:

- certificate verification = 4 point multiplications
- signed nonce verification = 2 point multiplications

⇒ protocol execution time over 6 min!

but see **Sizzle** [Gupta et al. 2005]:

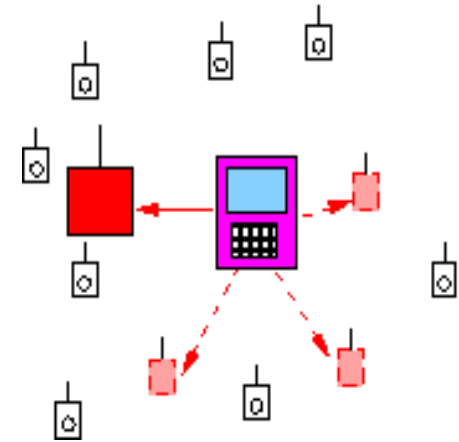
ECC-based SSL handshake on MICA2 takes **4 sec!** (assembler)



Security Analysis: Node Impersonation

unilateral **user authentication** => **single** sensor can impersonate the other $m-1$:

- can give the user false „authenticated“ positive
 - ⇒ take care of authenticated answers!
 - ⇒ DoS on the user
- ⇒ **multilateral authentication needed**

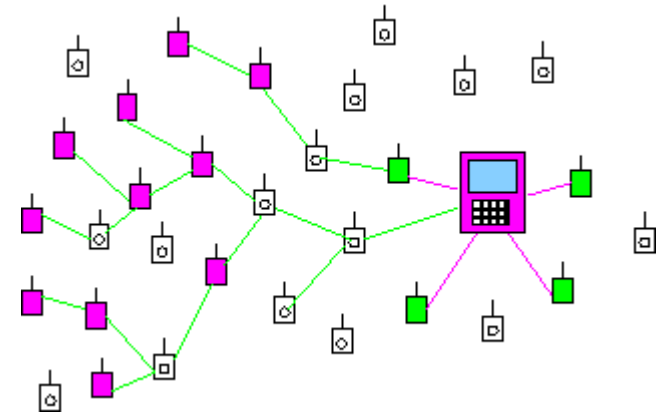
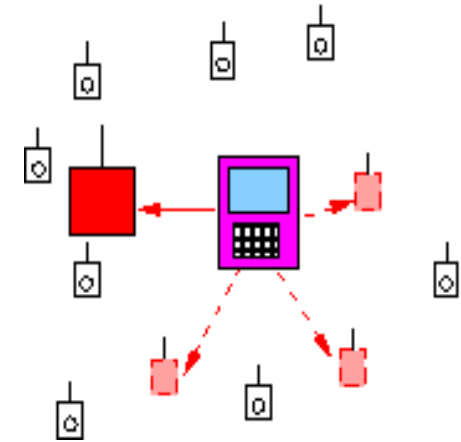


Security Analysis: Node Impersonation

legitimate user + captured sensor(s)

unilateral **user authentication** => **single** sensor
can impersonate the other $m-1$:

- can give the user false „authenticated“ positive
 - ⇒ take care of authenticated answers!
 - ⇒ DoS on the user
- ⇒ multilateral authentication needed
- can forward **fake query** into the WSN
prohibited by **authenticated query forwarding**

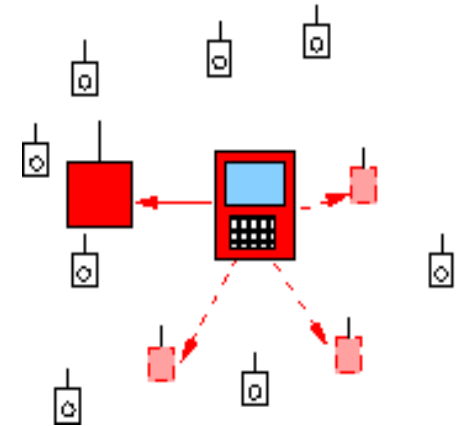


Security Analysis: User Impersonation

illegitimate user (= adversary) + captured sensor(s)

tries to forward **illegitimate query** into the WSN

⇒ prohibited by **authenticated query forwarding**

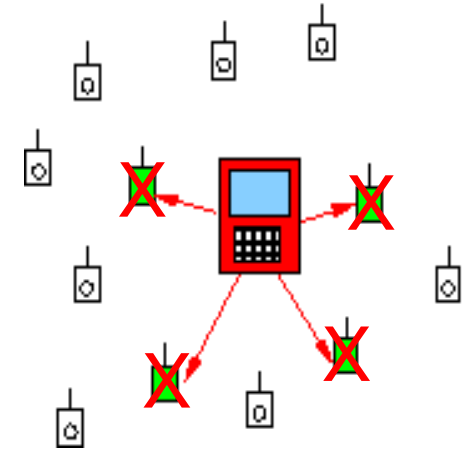


Security Analysis: Denial of Service

In the last protocol step sensors:

- verify user's certificate
 - verify signed nonce
- ⇒ **resource-demanding**

send bogus certificate and bogus signed nonce



Security Analysis: Denial of Service

In the last protocol step sensors:

- verify user's certificate
- verify signed nonce

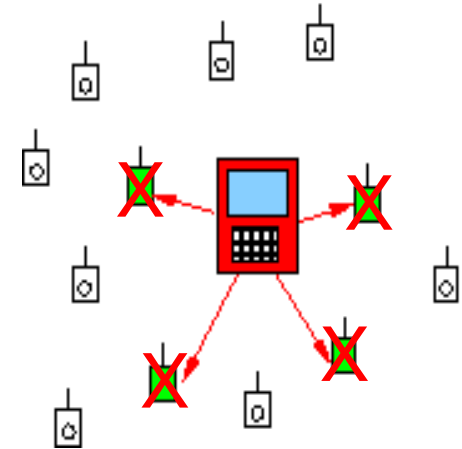
⇒ **resource-demanding**

send bogus certificate and bogus signed nonce

weak defense: certificate with redundancy

⇒ semantic check possible

but: certificate **replay attacks!**



Security Analysis: Denial of Service

In the last protocol step sensors:

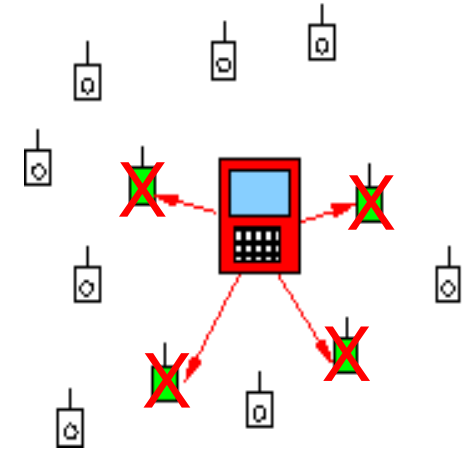
- verify user's certificate
- verify signed nonce

⇒ **resource-demanding**

send bogus certificate and bogus signed nonce

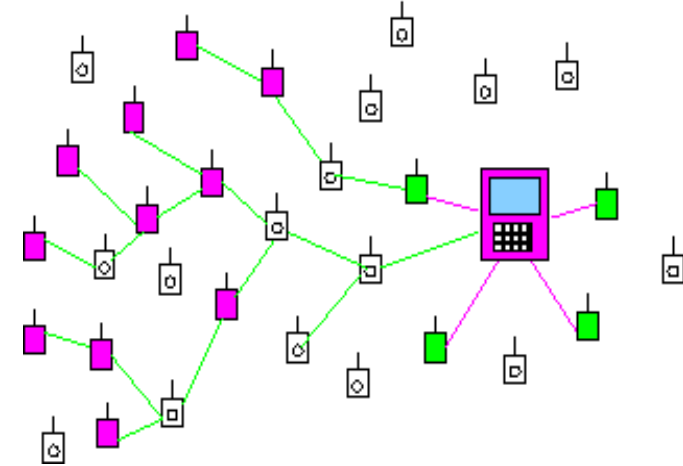
strong defense: no purely technical means!

- limit gain from DoS (data replication)
- intrusion detection (quick notification)
- deployment of additional sensors in affected area
- forewarn sensors about ongoing attack



Conclusions and Outlook

- identified a new problem: **authenticated querying** in WSNs
- proposed a **hybrid solution**: sensors in user's range used as a **gate** (interpreter):
 - public key crypto world of the user
 - ↕
 - symm. crypto world of the WSN
- implemented a user authentication protocol robust against node capture attacks
- disappointing performance due to underlying EccM library
- however, with Sizzle or other efficient implementations idea becomes applicable



References

- [Gupta et al. 2005] V. Gupta, M. Millard, S. Fung, Y. Zhu, N. Gura, N. Eberle, and S. C. Shantz. Sizzle: A standards-based end-to-end security architecture for the embedded Internet. PerComp 2005
- [Malan et al. 2004] D.J. Malan, M. Welsh, M.D. Smith. A public-key infrastructure for key distribution in TinyOS based on elliptic curve cryptography. 1st IEEE International Conference on Sensor and Ad Hoc Communications and Networks 2004
- [Perrig et al. 2001] A. Perrig, R. Szewczyk, J. D. Tygar, V. Wen, D. E. Culler. SPINS: security protocols for sensor networks. *Wireless Networks*, Vol. 8, Issue 5.
- [Seys et al. 2005] Efficient Cooperative Signatures: A Novel Authentication Scheme for Sensor Networks. 2nd International Conference on Security in Pervasive Computing 2005
- [Vogt 2004] H. Vogt. Exploring Message Authentication in Sensor Networks. 1st European Workshop on Security in Ad Hoc and Sensor Networks (ESAS 2004)
- [Zhu et al. 2004] S. Zhu, S. Setia, S. Jajodia, P. Ning. An Interleaved Hop-by-Hop Authentication Scheme for Filtering of Injected False Data in Sensor Networks. IEEE Symposium on Security and Privacy, Oakland, California, May 2004.