Cryptography and Security in Wireless Sensor Networks

Pyrgelis Apostolos
pyrgelis@ceid.upatras.gr

Department of Computer Engineering and Informatics
University of Patras, Greece

FRONTS 2nd Winterschool
Braunschweig, Germany

Outline

Cryptography

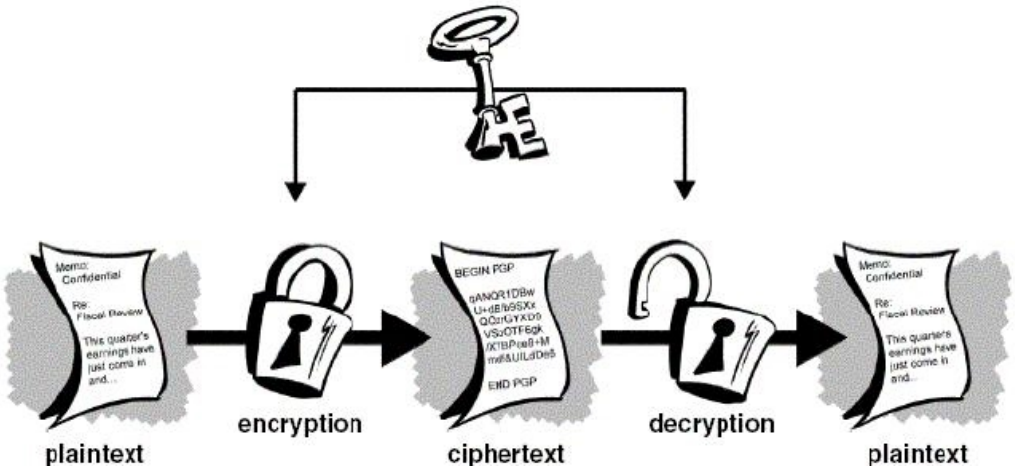
Definition

Cryptography is the study of mathematical techniques related to aspects of information security such as confidentiality, data integrity, entity authentication and data origin authentication

- A research field for scientists, mathematicians and engineers
- Important role in securing commercial and goverment applications including communications, payment systems, access and identification solutions

Cryptography Categories

- Cryptography can also be defined as the conversion of data (with use of cryptographic keys) into a scrambled code that can be deciphered and sent across a public or private network



- Cryptography is divided into two categories
  1. Symmetric-key Cryptography: In a symmetric-key algorithm both parties use the same key for encryption and decryption (DES,AES)
  2. Public-key Cryptography: Asymmetric cryptography algorithms use different keys for encryption and decryption.Each node in the network has a pair of keys, the private key and the public key (RSA, Diffie-Hellman, ECC)

Symmetric-key vs Public-key Cryptography

Symmetric-key Cryptography

- Symmetric-key ciphers have high rates of data throughput (Mbytes/sec) and relatively short keys
- Key must remain secret at both ends and must change frequently, many key pairs to be managed in large networks

Public-key Cryptography

- Only the private key must be kept secret, a private/public key pair may remain unchanged for considerable periods of time, efficient digital signature mechanisms, smaller number of necessary keys in large networks
- Much slower throughput rates than symmetric-key cryptography and larger key sizes

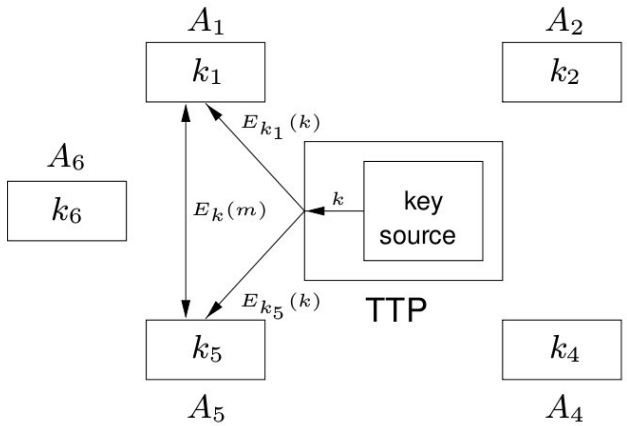
Comparison Summary

- Symmetric-key and public-key encryption have a number of complementary advantages
- Cryptographic schemes exploit the strengths of each
  - The long term nature of the public/private keys of the public-key cryptography
  - The performance efficiencies of the symmetric-key cryptography
- Public-key cryptography facilitates key management and efficient signatures (particularly non-repudiation)
- Symmetric-key cryptography is efficient for encryption algorithms and some data integrity applications

Key establishment and management

- Key establishment is any process whereby a shared secret key becomes available to two or more parties, for subsequent cryptographic use (key agreement, key transport)
- Key management (KM) is the set of processes and mechanisms which support key establishment and the maintenance of ongoing keying relationships between parties, including replacing older keys with new ones

- KM through symmetric-key techniques (easy to add/remove entities, TTP which stores n secret keys)
- KM through public-key techniques (no TTP, a public file with the nodes public keys, authentication problems and need for public key certification)



Diffie-Hellman Key Agreement

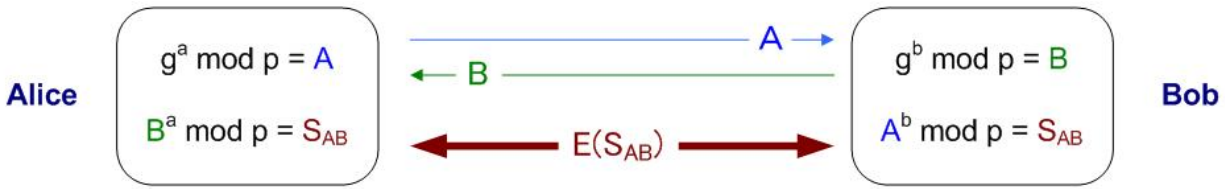
- A cryptographic protocol that allows two parties that have no prior knowledge of each other to establish a shared secret key ($g^{ab} mod p$) over an insecure communications channel

Alice

1) Picks a secret number "a"
2) Calculate "A" using the following
(values of g and p are predetermined)
$g^a \bmod p = A$
3) Send "A" to Bob
4) Receive "B" from Bob
5) Calculate the shared secret using the following
$B^a \bmod p = S_{AB}$

Bob

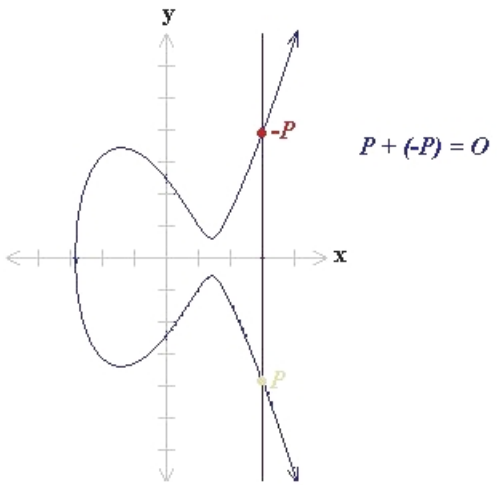
1) Picks a secret number "b"
2) Calculate "B" using the following
(values of g and p are predetermined)
$g^b \bmod p = B$
3) Send "B" to Alice
4) Receive "A" from Alice
5) Calculate the shared secret using the following
$A^b \bmod p = S_{AB}$

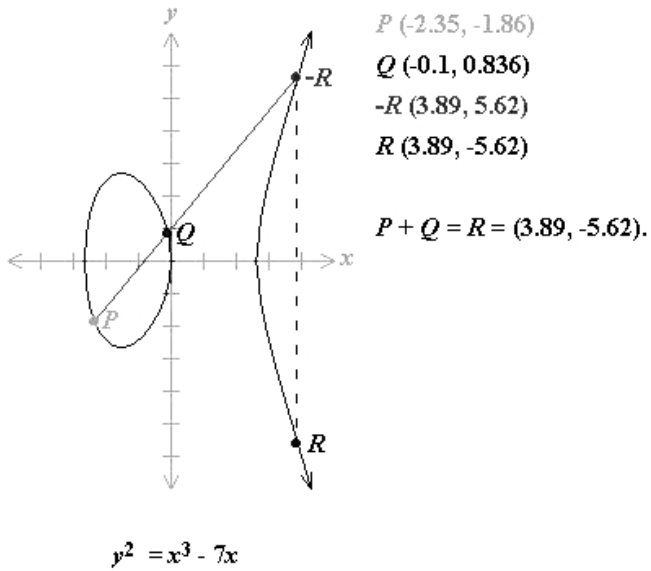


- Its security is based on the DLP : given an element g in a finite group G and another element $h \in G$, find an integer x such that $g^x = h$
- Authentication issues (Man in the Middle Attack)

Elliptic Curve Cryptography (1/2)

- Public-key cryptosystem introduced by Victor Miller and Neal Koblitz in the year 1985
- An elliptic curve E is defined as the set of solutions $(x, y) \in Z_p x Z_p$ that satisfy the equation $y^2 \equiv x^3 + ax + b(modp)$ along with the point at infinity O
- $a, b \in Z_p$ are constants such that $4a^3 + 27b^2 \neq 0(modp)$ and $p > 3$
- The set of points on the curve with coordinates in a finite field along with the point of infinity O form groups with respect to addition operation
- $P + O = O + P = P$ for all $P \in E$
- $P + Q = Q + P$ and $(P + Q) + R = P + (Q + R)$ where $P, Q, R \in E$





Elliptic Curve Cryptography (2/2)

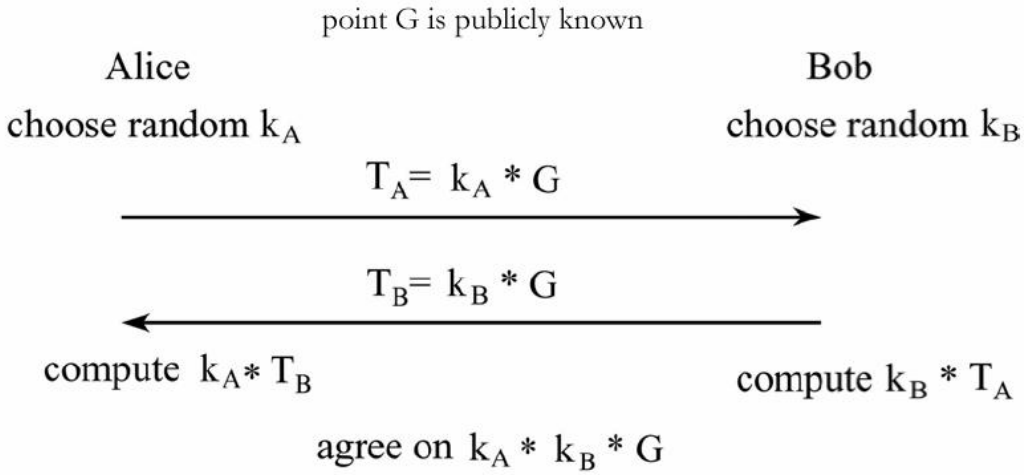
- For computation of a multiple kP of an elliptic curve point P where $k > 0$ binary method (double and add) is used.For example $6P = 2(2P + P)$
- Its security is based on the ECDLP on the EC group: given points P and Q on the elliptic curve, find a least positive integer k that $Q = kP$
- Several known protocols have been adapted to elliptic curves (ECDH, ECDSA)

- Main Advantage: Smaller key sizes than other public-key systems (RSA) for achieving the same level of security (performance advantages)

Symmetric	ECC	DH/DSA/RSA
80	163	1024
128	283	3072
192	409	7680
256	571	15,360

Elliptic Curve Diffie Hellman

- Alice and Bob want to exchange a key.They carefully chose an elliptic curve E and a public base point $G(x, y)$ on the curve
- Alice chooses her private key, a random integer k_A and Bob chooses a random integer k_B.The random integers are kept private
- Alice computes her public key, a new point on the elliptic curve by performing scalar multiplication $T_A = k_A G$ and sends it to Bob who simultaneously computes his public key $T_B = k_B G$



- Alice receives T_B and computes the shared secret, a new point on elliptic curve $K = k_A T_B = k_A k_B G$.Similarly, Bob takes T_A and computes $K = k_B T_A = k_B k_A G$

ECIES (1/2)

- Agreement on a an elliptic curve E, a public base point $G(x, y)$ on the curve and a MAC scheme
- Encryption of message M with receiver's public key $Q = dG$
  - Select a random integer k and compute public key $R = kG = (x_R, y_R)$
  - Compute shared secret $P = kQ = (x_P, y_P)$ and $z = x_P$
  - Use z on KDF to generate keying data K
  - Use len octets of K as encryption key EK and maclen octets of K as mac key MK
  - Use symmetric encryption scheme to encrypt the message M to EM with key EK
  - Use mac scheme with key MK to produce a tag D for EM
  - Output $C = R|EM|D$

ECIES (2/2)

- Decryption of ciphertext C with receiver's private key d
  - Obtain the elliptic curve point $R = (x_R, y_R)$
  - Compute shared secret $P = dR = dkG = kQ = P = (x_P, y_P)$ and $z = x_P$
  - Use z on KDF to generate keying data K
  - Use len octets of K as encryption key EK and maclen octets of K as mac key MK
  - Use mac scheme with key MK to verify that D is the tag on EM
  - Use symmetric encryption scheme to decrypt EM using key EK and recover initial message M

Outline

Network Security

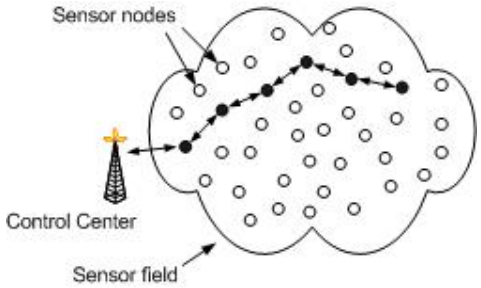
Definition

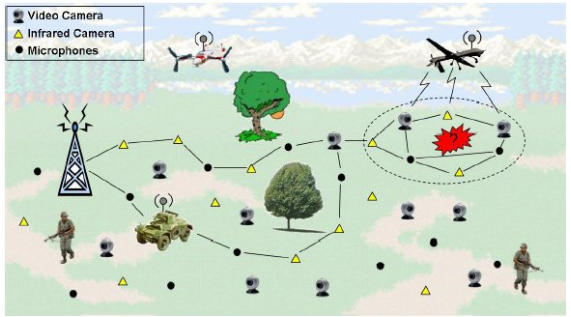
The protection of a computer network and its services from unauthorized modification, destruction, or disclosure.

- Distribution of secret information
- Efficiency of communication protocols
- Cryptographic issues
- Network attacks and coutermeasures

Security Challenges in WSN

- The broadcast nature of the wireless communication renders a WSN susceptible to link attacks ranging from passive eavesdropping to message replay and message distortion
- The network deployment in hostile environments (e.g. battlefield, forest) with relatively poor physical protection





- The limitations in energy, computational power and memory of the tiny sensors
- The extremely large number of interacting devices in a sensor network
- The dynamic nature of WSN (frequent changes in both its topology and its membership)

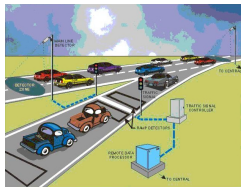
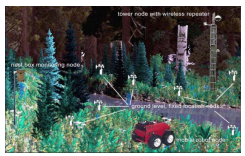
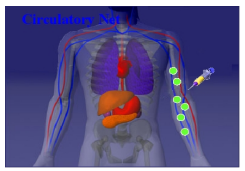
Typical WSN Applications

Huge range of possible applications depending on the sensor type (thermal, acoustic, seismic etc) :

- Monitor and Control
  (Habitat, Environmental, Ecosystem, Agricultural, Structural, Traffic, Manufacturing, Health)
- Security and Surveillance
  (Border and Perimeter control, Target tracking, Intrusion detection)

Security and Privacy issues are raised.



Security Principles

- Data Confidentiality: Ensuring that only authorized sensor nodes can access the content of the messages
- Data Authentication: Ensuring that the data is originated from the correct source
- Data Integrity: Ensuring that any received data has not been altered in trasmit by unauthorized parties
- Data Freshness: Ensuring that no old messages have been replayed
- Availability: Ensuring that services offered by whole WSN or by a single sensor node must be available whenever required

Security Principles

- Data Confidentiality: Ensuring that only authorized sensor nodes can access the content of the messages
- Data Authentication: Ensuring that the data is originated from the correct source
- Data Integrity: Ensuring that any received data has not been altered in trasmit by unauthorized parties
- Data Freshness: Ensuring that no old messages have been replayed
- Availability: Ensuring that services offered by whole WSN or by a single sensor node must be available whenever required

Security Principles

- Data Confidentiality: Ensuring that only authorized sensor nodes can access the content of the messages
- Data Authentication: Ensuring that the data is originated from the correct source
- Data Integrity: Ensuring that any received data has not been altered in trasmit by unauthorized parties
- Data Freshness: Ensuring that no old messages have been replayed
- Availability: Ensuring that services offered by whole WSN or by a single sensor node must be available whenever required

Security Principles

- Data Confidentiality: Ensuring that only authorized sensor nodes can access the content of the messages
- Data Authentication: Ensuring that the data is originated from the correct source
- Data Integrity: Ensuring that any received data has not been altered in trasmit by unauthorized parties
- Data Freshness: Ensuring that no old messages have been replayed
- Availability: Ensuring that services offered by whole WSN or by a single sensor node must be available whenever required

Security Principles

- Data Confidentiality: Ensuring that only authorized sensor nodes can access the content of the messages
- Data Authentication: Ensuring that the data is originated from the correct source
- Data Integrity: Ensuring that any received data has not been altered in trasmit by unauthorized parties
- Data Freshness: Ensuring that no old messages have been replayed
- Availability: Ensuring that services offered by whole WSN or by a single sensor node must be available whenever required

Security Principles

- Data Confidentiality: Ensuring that only authorized sensor nodes can access the content of the messages
- Data Authentication: Ensuring that the data is originated from the correct source
- Data Integrity: Ensuring that any received data has not been altered in trasmit by unauthorized parties
- Data Freshness: Ensuring that no old messages have been replayed
- Availability: Ensuring that services offered by whole WSN or by a single sensor node must be available whenever required

WSN Adversary (1/2)

Definition

A person or another entity that attempts to cause harm to the network, for example, by unauthorized access or denial of service.

He can be :

- Passive: Only monitors the communication channel.Threatens the confidentiality of data.
- Active: Attempts to delete, add or alter the transmission on the channel.Threatens data integrity, authentication and confidentiality.

WSN Adversary (1/2)

Definition

A person or another entity that attempts to cause harm to the network, for example, by unauthorized access or denial of service.

He can be :

- Passive: Only monitors the communication channel.Threatens the confidentiality of data.
- Active: Attempts to delete, add or alter the transmission on the channel.Threatens data integrity, authentication and confidentiality.

WSN Adversary (2/2)

Definition

A person or another entity that attempts to cause harm to the network, for example, by unauthorized access or denial of service.

He can be :

- Mote-Class Attacker: Has access to a few nodes with similar capabilities to those deployed in the network.
- Laptop-Class Attacker: Has access to more powerful devices like a laptop.Has advantages over legitimate nodes like greater battery power, more capable cpu and high-power antenna.
- Insider: Has compromised some authorized nodes of the network (stolen key material, run malicious code).
- Outsider: Has no special access to the network.

WSN Adversary (2/2)

Definition

A person or another entity that attempts to cause harm to the network, for example, by unauthorized access or denial of service.

He can be :

- Mote-Class Attacker: Has access to a few nodes with similar capabilities to those deployed in the network.
- Laptop-Class Attacker: Has access to more powerful devices like a laptop.Has advantages over legitimate nodes like greater battery power, more capable cpu and high-power antenna.
- Insider: Has compromised some authorized nodes of the network (stolen key material, run malicious code).
- Outsider: Has no special access to the network.

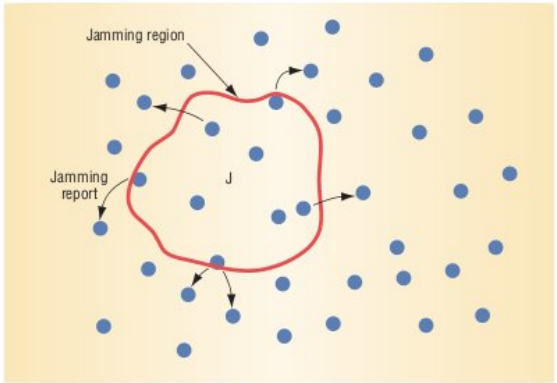
Attacks and Countermeasures in WSN (1/5)

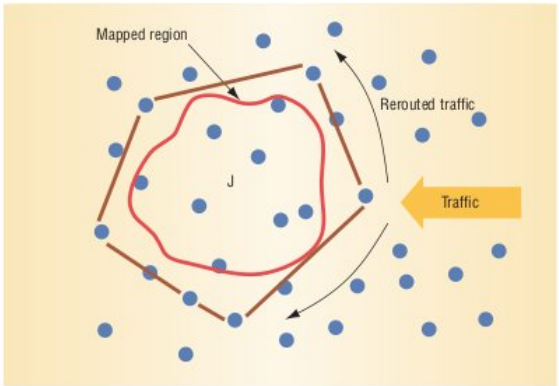
Definition

The denial of service attack (DoS) is any event that diminishes or eliminates a network's capacity to perform its expected function

- Physical Layer
  - Jamming: Interference with the radio frequencies a network's nodes are using
  - Tampering: Physical compromise of nodes

Solutions: spread spectrum communication, jamming reports, accurate and complete design of the node physical package

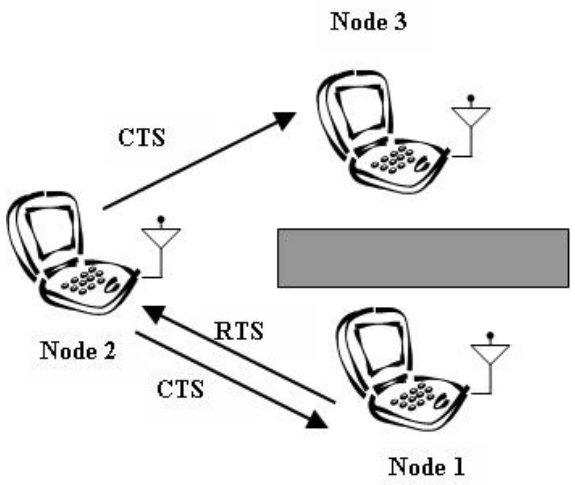




Attacks and Countermeasures in WSN (2/5)

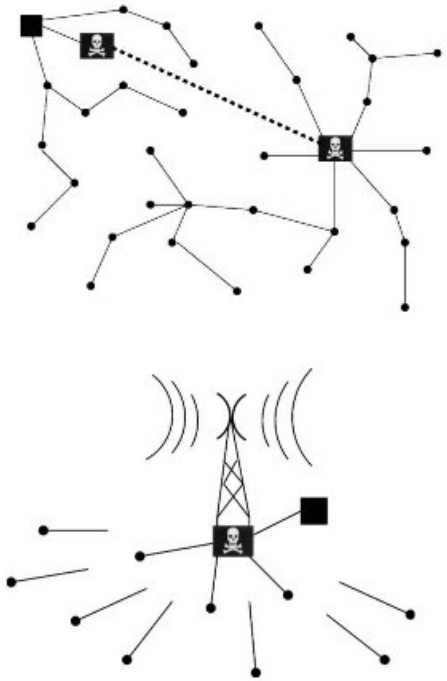
- Data Link Layer
  - Collision: Altering of transmission octets to disrupt the packets (checksum mismatch,back off in some MAC protocols)
  - Exhaustion: Collisions and back off in MAC protocols result in re-transmissions which result to the exhaustion of battery resources
  - Unfairness: Degrading service by causing users of a real-time MAC protocol to miss their deadlines

Solutions: Error correcting codes, collision detection techniques, TDM, rate limiting



Attacks and Countermeasures in WSN (3/5)

- Network Layer
  - Selective Forwarding: Malicious nodes refuse to forward certain messages and simply drop them
  - Sinkhole: The adversary attracts the surrounding nodes with unfaithful routing information
  - Sybil attack: A single node presents multiple identities to other nodes
  - Wormhole: The adversary tunnels the traffic received in a part of the network to another
  - HELLO flood: A laptop-class attacker broadcasts information with enough transmission power convincing every node in the network that he is his neighbor

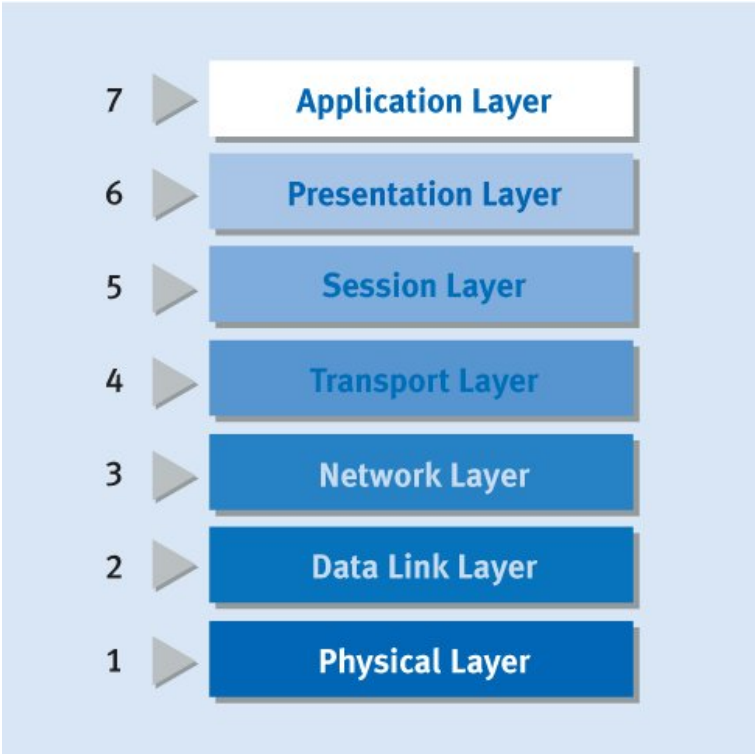


Solutions:Link layer encryption and authentication, multipath routing, identity verification, authenticated broadcast

Attacks and Countermeasures in WSN (4/5)

- Transport Layer
  - Flooding: The adversary sends many connection establishment requests to the victim (memory and resource exhaustion)
  - Desynchronization: The adversary repeatedly forces messages which carry sequence numbers to one or both endpoints (request for retransmission of missed frames)

Solutions: Connection-less protocols, packet authentication including all control fields in the transport protocol header



Attacks and Countermeasures in WSN (5/5)

- Summary of Attacks and Countermeasures in WSN
  - Need for physical network protection (not always possible)
  - Cryptography can provide link layer encryption and authentication mechanisms (MAC) but this is not enough
  - End to end security mechanisms are impractical
  - Careful protocol design (routing, localization, data aggregation) with respect to security principles and attacker models
  - Consideration of energy issues when adapting countermeasures

Key Distribution in WSN - Properties

Key distribution mechanisms should support the security requirements mentioned before plus

- Scalability: support of large networks and flexibility against the increase of their size
- Efficiency: consideration of storage, processing and communication limitations on sensor nodes
  - Storage Complexity: amount of memory required to store security credentials
  - Processing Complexity: amount of processor cycles required to establish a key
  - Communication Complexity: number of messages exchanged during a key generation process
- Key Connectivity: probability that two (or more) sensor nodes store the same key or keying material
- Resilience: resistance against node capture (higher resilience means lower number of compromised links)

Key Distribution in WSN

Basic problems

- Pair-wise keying: Establishment of a key used to secure unicast communication between a pair of sensor nodes over single or multi-hop wireless link
- Group-wise keying: Establishment of a key used to secure multicast communication among a group of sensor nodes over single or multi-hop wireless link

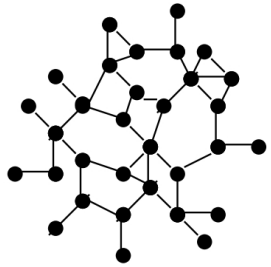
Approaches

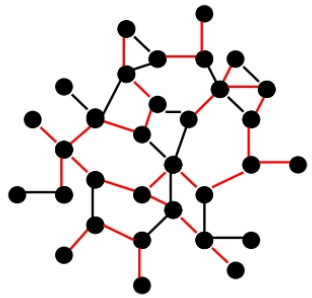
- Probabilistic: key-chains are randomly selected from a key-pool and distributed to sensor nodes
- Deterministic: deterministic processes are used to design the key-pool and the key-chains to provide better key connectivity
- Hybrid: combination of the above to improve scalability and resilience

Mechanisms

- Pre-distribution (safety??)
- Dynamic key generation

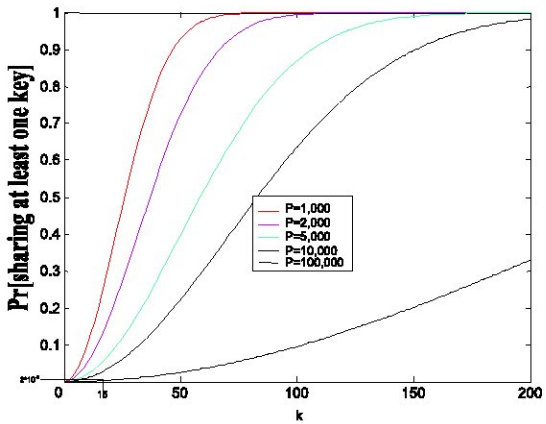
Pair-wise keying





- L.Eschenauer and V.Gligor, *"A key-management scheme for distributed sensor networks" –ACM CSS 2002*
- Random pair-wise key pre-distribution
- A set of keys randomly chosen from a key pool
- Reservoir of P keys
- $k(<< P)$ keys pre-distributed in each sensor
- Probability for any 2 sensors to have a common key:

$$p = 1 - \frac{((P-k)!)^2}{P!\,(P-2k)!}$$



- 3 phases: key pre-distribution, shared-key discovery, path-key establishment

Group-key establishment

- The creation of the shared secret must be done very carefully
- Some protocols based on the GDH: each member generates a nonce N_i and contributes to the shared key $K_{group} = g^{N_1 \ldots N_n}$
- Should guarantee:
  - computational group key secrecy: it is computational infeasible for any passive adversary to discover any group key
  - key indepedence: an attacker knowing proper subset of group keys cannot discover any other group keys
  - forward/backward secrecy: any subset of group keys can not be used to discover previous/subsequent keys
- The key should change every time the group changes
- Should be able to handle membership events (join events, leave events, group merge events, group partition events)

ECC in WSN (1/2)

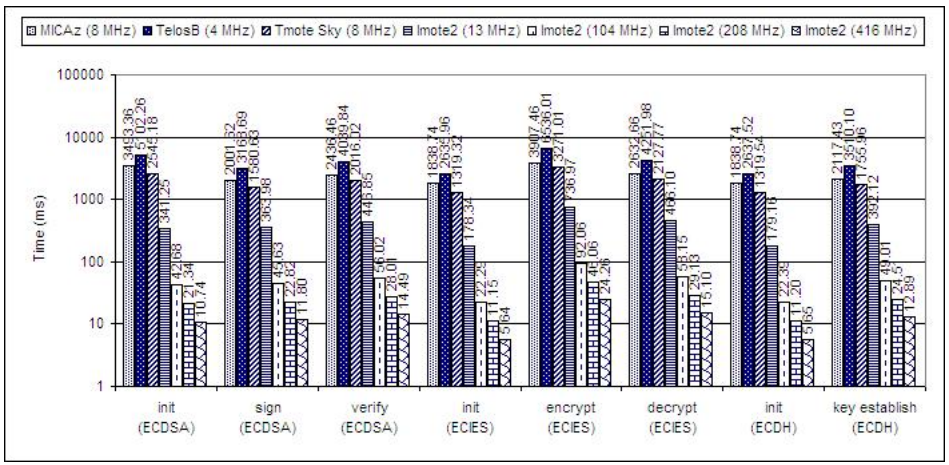
- D.Malan, M.Welsh and M.Smith- *"Implementing Public Key Infrastructure for sensor networks"*
- The first implementation of ECC (over F_{2^p}) for sensor networks on the 8-bit, 7.38 MHz MICA2 mote
- Use a 163-bit key for distribution of the 80-bit TinySec keys

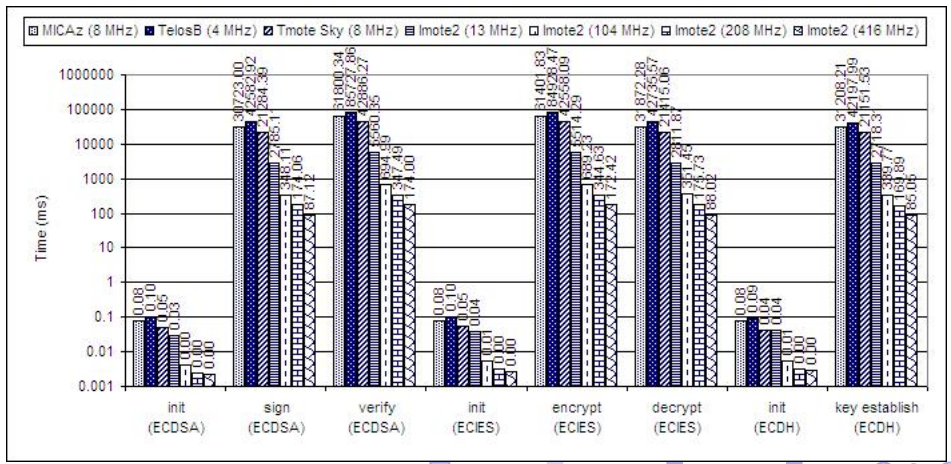
	Private-Key Generation	Public-Key Generation
Total Time	0.229 sec	34.161 sec
Total CPU Utilization	1.690×10^6 cycles	2.512×10^8 cycles
Total Energy	0.00549 Joules	0.816 Joules

- Generation of private keys in 0.229 seconds
- Generation of public keys in 34 seconds using 1Kb of SRAM and 34Kb of ROM
- Preloaded on the sensors the curve E approved by NIST with equation $y^2 + xy \equiv x^3 + x^2 + 1$, its order $0x4000000000000000020108a2e0cc0d99f8a5ef$ and the base point $G = (G_x, G_y)$ where $G_x = 0x2fe13c0537bbc11acaa07d793de4e6d5e5c94eee8$ and $G_y = 0x289070fb05d38ff58321f2e800536d538ccdaa3d9$

ECC in WSN (2/2)

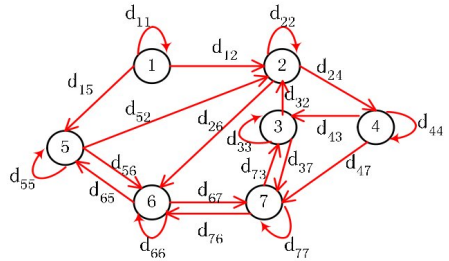
- An Liu and Peng Ning- *"TinyECC: A configurable library for elliptic curve cryptography in wireless sensor networks" (IPSN 2008)*
- Implementation of ECC over F_p with elliptic curve parameters recommended by SECG (128-bit, 160-bit and 192-bit)
- Implementation on TinyOS for portability. Platforms MICAz, TelosB, Tmote Sky and Imote2
- Implementation of ECIES, ECDSA and ECDH
- Optimization techniques for large numbers operations (inline assembly) and elliptic curve operations



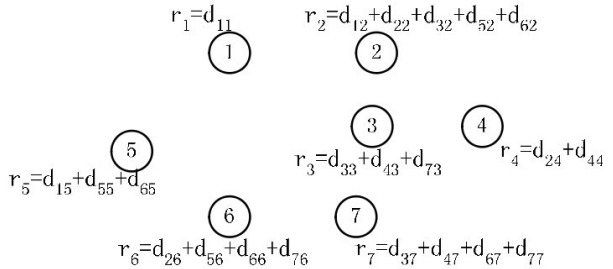


Privacy in WSN

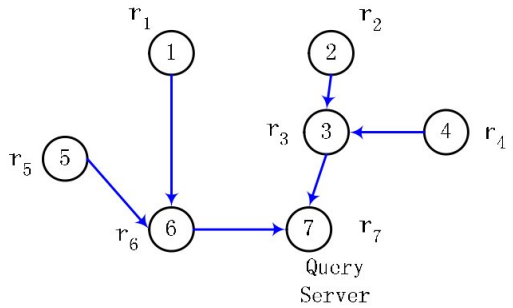
- Protecting sensor information
- He, Liu, Nguyen, Nahrstedt, Abdelzaher- *"Privacy Preserving Data Aggregation in Wireless Sensor Networks"*, *INFOCOM 2007*
- WSN may be placed in homes, hospitals, companies buildings.Need for a way to collect the aggregated sensor readings while preserving data privacy
- SMART: Slice Mix Aggregate.Based on pair-wise key pre-distribution
- Three phases: Slicing, Mixing, Aggregation



(a) Slicing ($J = 3, h = 1$):$d_{ij} (i \neq j)$ is encrypted and transmitted from node i to j, where $j \notin S_i$. d_{ii} is the data piece kept at node i.



(b) Mixing: Each node i decrypts all data pieces received and sums them up including the one kept at itself (d_{ii}) as r_i.



(c) Aggregation (No encryption is needed)

Conclusions

- Building secure sensor networks is of paramount importance but it is quite difficult
- Privacy issues should be considered depending on the application
- A single solution is highly vulnerable
- Cryptography alone is not enough
  - Public-key cryptography is computationally expensive
  - ECC provides performance benefits (should be combined with symmetric-key techniques)
- Multi-level Approach: only viable solution is to combine different techniques for securing the system
  - design protocols (routing schemes, data aggregation,time synchronization, localization) with respect to security principles
  - use carefully the key management methods and mechanisms
  - the combination of multiple attacking angles increases the overall achieved security

Outline

pMP

- Multi-precision arithmetic is necessary in embedded systems (cryptography, data aggregation)
- A variety of software libraries that implement big-number operations (GNUMP)
- Most of them rely on dynamic memory allocation to represent big-numbers and carry out the operations
- Optimized assembly code is used for performance speedups
- Very difficult to port such implementations to sensor platforms
- pMP: implementation of big number operations without use of dynamic memory allocation
- Basic operations for elliptic curve cryptography over F_{2^p}

The Crypto Concept

The Crypto Concept

```
template<typename OsModel>
class CryptoConcept {
   typedef ... Os;
   typedef ... node_id_t;
   typedef ... block_data_t;

   void set_os( OsModel* os );

   void enable( void );
   void disable( void );

   void key_setup(node_id_t,block_data_t* key);
   void encrypt(block_data_t* input,block_data_t* output,int8_t length);
   void decrypt(block_data_t* input,block_data_t* output,int8_t length);
};
```

Routing Concept

The Routing Concept

```
template<typename OsModel, typename Radio = OsModel::Radio>
class RoutingAlgorithmConcept {

   void set_os( OsModel* os );

   void enable( void );
   void disable( void );

   template
   < class Callee, void (Callee::*Method)
                     (node_id_t, size_t, data_t*) >
   int reg_recv_callback( T *obj_pnt );

   void unreg_recv_callback( int );

   int send( node_id_t receiver, size_t len, data_t* data );
};
```

Routing + Crypto Combination

- Combination of any routing algorithm with any crypto algorithm
- Not a single change in their code

The Secure Routing Concept

```
template<typename OsModel,typename Radio = OsModel::Radio,typename crypto,typename Routing>
class SecRoutingConcept {
  void set_os( OsModel* os );

  void enable( void );
  void disable( void );

 template
  < class Callee, void (Callee::*Method)
                    (node_id_t, size_t, data_t*) >
  int reg_recv_callback( T *obj_pnt );

  void unreg_recv_callback( int );

  void send( node_id_t receiver, size_t len, block_data_t *data );
};
```

A SecRouting Example

- The file "crypto.h"

A crypto algorithm

```
template<typename OsModel_P>
   class crypto
   {
   void enable( void );
   void disable( void );
   ...
   }
template<typename OsModel_P>
   void
   crypto<OsModel_P>::
   ECIES_encrypt(uint8_t * a,uint8_t * b,int8_t length )
   {...}

template<typename OsModel_P>
   void
   crypto<OsModel_P>::
   ECIES_decrypt(uint8_t * a,uint8_t * b,int8_t length)
   {...}
```

The SecRouting Class

- The file "sec_routing.h"

The Enable Function

```
template<typename OsModel, typename Radio,typename crypto,typename Routing>
void
SecRouting<OsModel,Radio ,crypto, Routing>::
enable( void )
  {

  routing.enable();
  routing.reg_recv_callback<self_type,&self_type::receive>(this);
  crypto.enable();

  }
```

The Send Function

```
template<typename OsModel,typename Radio,typename crypto,typename Routing>
void
SecRouting<OsModel, Radio,crypto, Routing>::
send( node_id_t receiver, size_t len, block_data_t *data )
  {

  crypto.ECIES_encrypt(data,buffer,len);
  routing.send(receiver,len,buffer);

  }
```

A SecRouting Example Application

Dsdv + Crypto

```
#include "algorithms/routing/dsdv_routing.h"
#include "algorithms/crypto/crypto.h"
#include "algorithms/secrouting/sec_routing.h"

typedef wiselib::iSenseOsModel Os;
typedef wiselib::StaticArrayRoutingTable<Os, Os::Radio, 8,
 wiselib::DsdvRoutingTableValue<Os, Os::Radio> > DsdvRoutingTable;
typedef wiselib::DsdvRouting<Os, DsdvRoutingTable> dsdv_routing_t;
typedef wiselib::crypto<Os> crypto_t;
typedef wiselib::SecRouting<Os,Os::Radio,crypto_t,dsdv_routing_t> secrouting_t;
....
void
iSenseDemoApplication::
boot(void)
{
secrouting_.set_os( &os() );
secrouting_.enable();
secrouting_.reg_recv_callback<iSenseDemoApplication,
   &iSenseDemoApplication::receive_routing_message>(this);
}
....
void
iSenseDemoApplication::
execute( void* userdata )
{
...
secrouting_.send();
...
}
```

The End

Thank you very much!