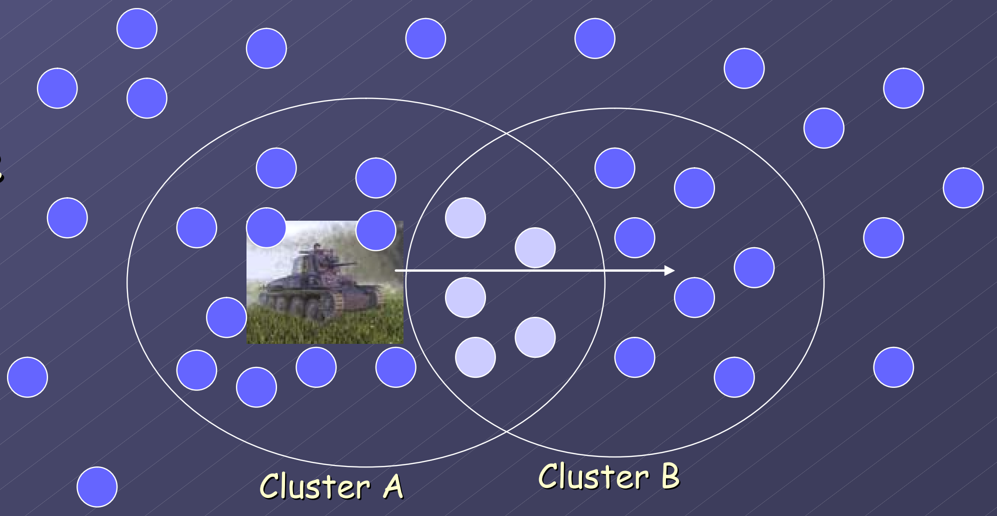
Background & Motivation

- Typical WSN deployment environment is prone to various malicious attacks
- What makes security for WSN so unique …
  - Scarce resources - energy, memory, computation
  - Ad-hoc nature and extreme dynamic environments
  - High node density (scalability)
  - Existing security solutions can not be directly applied
- Moreover, ad-hoc node cluster formation is unique
  - Unpredictable location, scope and dynamics
  - Requires short response times



Background & Motivation (cont.)

- Public key infrastructure (PKI) is a powerful and proven technology for addressing Confidentiality, Authentication and Message integrity
- However, due to resource limitations in WSN, existing PKI solutions can not be directly applied
  - Low computational capabilities
  - Limited memory space
  - Energy constraints imposed on communications

It would be highly desirable to have public key generation methodologies specifically designed and optimized for ad-hoc clusters of wireless sensor nodes

Prior Work: Random Key Pre-distribution Schemes

- Each node (i) is loaded with a small subset, C_i, of a large key chain, C, prior to deployment
  - Two nodes that wish to communicate are required to identify a common key
  - If they do not share a common key, a "key discovery" process is required
- Fundamental limitations of *random key pre-distribution*
  - Scalability – w.r.t. node memory and network size
  - Communication framework- finding nodes that share keys
  - Cryptographic robustness – inherently offers "statistical" robustness, which is always questionable

Public key distribution systems, if made feasible, overcome all of the above limitations

What is an Elliptic Curve?

In GF(p) an ordinary elliptic curve E suitable for elliptic curve cryptography is defined by the set of points (x; y) that satisfy the equation :

$$\boxed{y^2 = \left(x^3 + a \cdot x + b\right) \bmod p \quad a; b \in GF(p)}$$

Why use Elliptic Curve Cryptography (ECC)?

- Shorter key sizes (160 bit ECC cryptocomplexity equivalent to 1024 bit RSA)
- Faster calculations
- Less memory is required
- Recent work established its viability for WSN

Self Certified DH Key Generation: Fixed Key Addressing the Authentication Issue

The CA (Certifying authority) provides each user with a set of public and private keys: (U_v, X_v)

User i

ID_i , U_i

User j

ID_j , U_j

$$x_i \times [H(ID_j , U_j) * U_j + R] \quad = \quad x_i \times [H(ID_i , U_i) * U_i + R]$$

ID_v : identification of node v - scalar

U_v : user v's public key, generated by the CA - a point on the curve

X_v : user v's private key, generated by the CA - scalar

Self Certified DH Key Generation: Ephemeral Key Addressing the Authentication Issue

The CA (Certifying authority) provides each user with a set of public and private keys: (U_v, X_v)

User i

ID_i , U_i , Ev_i

User j

ID_j , U_j, Ev_j

$$Pv_i \times [H(ID_j , U_j) \times U_j + R] + (x_i + Pv_i)\, Ev_j = Pv_j \times [H(ID_i, U_i) \times U_i + R] + (x_j + Pv_j)\, Ev_i$$

ID_v: identification of node v - scalar

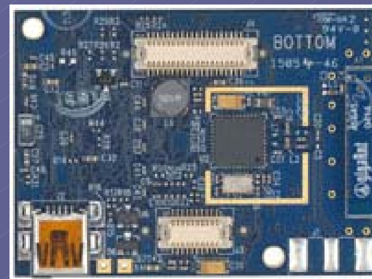
U_v : user v's public key, generated by the CA - a point on the curve

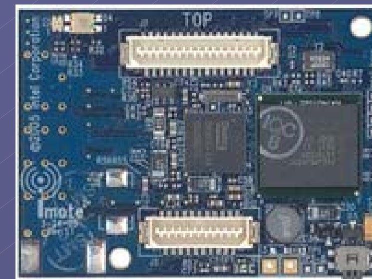
X_v : user v's private key, generated by the CA - scalar

Pv_v : a random number generated by user v - scalar

$Ev_v = Pv_v * G$

Intel Mote 2 Sensor Network Platform





- Electronic
  - 320/416/520MHz PXA271 XScale Processor (Dynamic voltage scaling)
  - Programming in NeSC
  - 32MB Flash on-board
  - 32MB SDRAM on-board
  - Mini-USB Client (slave), multiplexed with RS232 console over USB, power
  - I-Mote2 Basic Sensor connector (31+ 21 pin connector)
  - Zigbee [802.15.4] Radio (ChipCon CC2420)
  - Tri-color status LED; Power LED; battery charger LED, console LED
  - Switches: on/off slider, Hard reset, Soft reset, User programmable switch
- Mechanical
  - Size: 1.89inches x 1.42in. PCB Thickness 0.069in
  - Size: 48mm x 36mm. PCB Thickness 1.75mm

Intel Mote 2 Implementation Results (at 312 MHz)

Fixed key requires <u>one</u> on-line point by scalar multiplication

$x_i[H(ID_j , U_j) \times U_j + R] = x_i H(ID_j , U_j) \times U_j + x_i R$

Ephemeral key requires <u>two</u> on-line point by scalar multiplication

$Pv_i [H(ID_j , U_j) \times U_j + R] + (x_i+Pv_i) Ev_j =$

$Pv_i \times (H(ID_j , U_j) \times U_j + (x_i+Pv_i) (Ev_j +R) - x_i \times R$

Scalar Point Multiplication			
EccM			
Time (msec)	Voltage (v)	Current (mA)	Energy (mJ)
190	3	1.8	69
TinyEcc			
Time (msec)	Voltage (v)	Current (mA)	Energy (mJ)
42	3	1.8	22
Radio Transmission (including a 7 byte header)			
Time (msec)		Energy (mJ)	
~15		0.127	

Intel Mote 2 Implementation Results (cont.)

- ECC-based 160-bit key generations
  - Includes all computation & communication overheads
- CPU clock frequency ranges from 13 MHz to 312 MHz