On Selected Crypto Protocols

- Wireless Sensor Networks
- Remotely Keyed Encryption
- Authenticated Encryption

Petr Švenda

xsvenda@fi.muni.cz

http://www.buslab.org

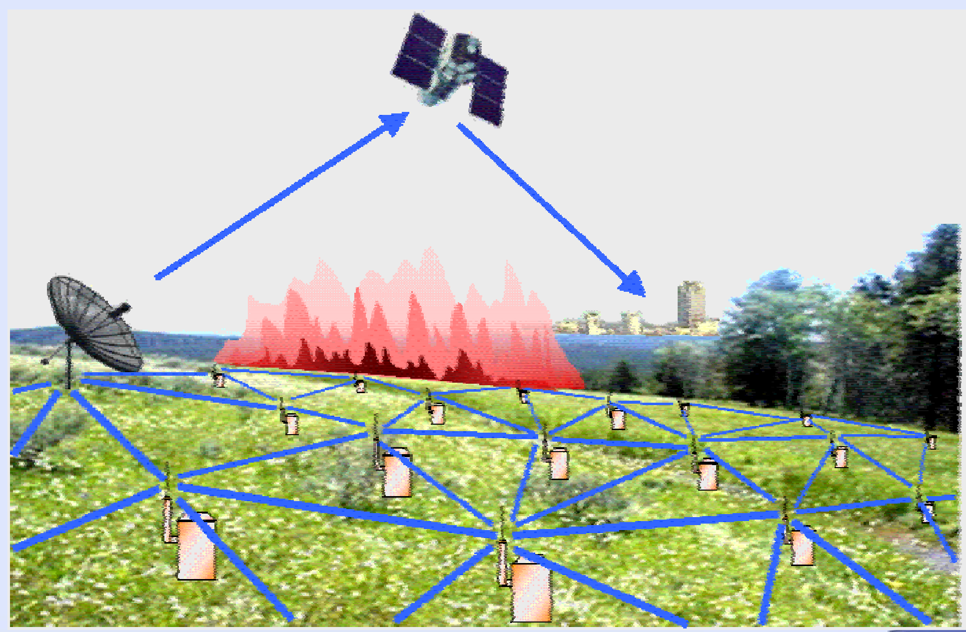
- **Key Distribution Protocols for Wireless Sensor Networks**

Overview

- Wireless Sensor Networks (WSN)
  - introduction
  - security goals, threads
- Key Distribution Protocols for WSN
  - specifics of WSN environment
  - common key distribution approaches
  - randomized keys pre-distribution
  - plaintext key distribution (Key infection)

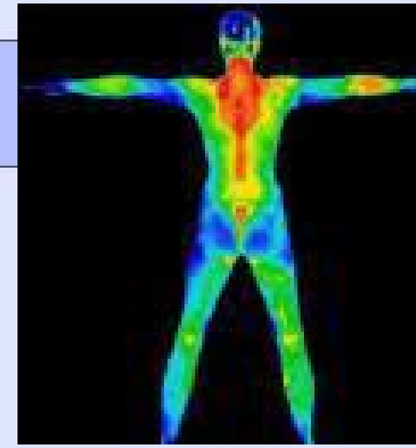
Wireless Sensor Networks

- Powerful base station(s)
- Network of nodes
  - sensing environmental conditions
  - RF transceivers
  - battery powered
  - no tamper resistance
  - number of 10^3 - 10^6
- Network topology
  - covering large areas
  - ad-hoc position/neighbours
  - distributed, multi-hop



Applications

Traffic control



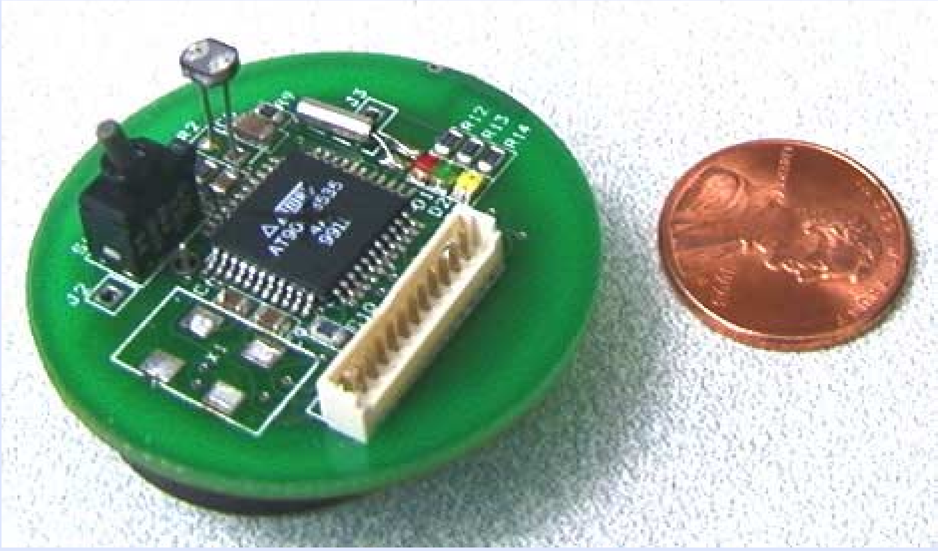
Medical monitoring

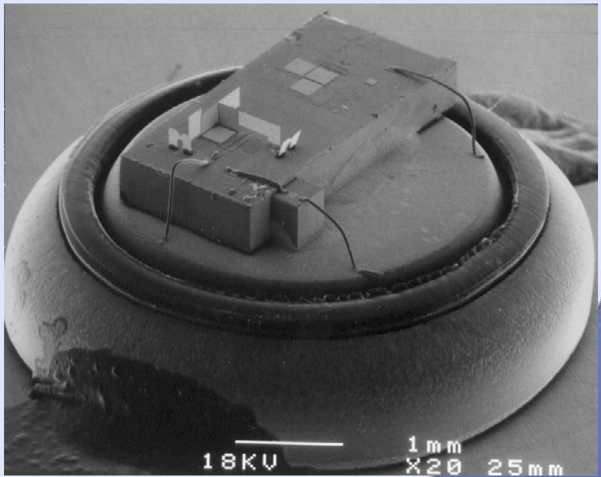
Wild fire detection

Battlefield management

Node hardware platform

- Berkeley Mote
  - 8-bit RISC processor
  - 4MHz clock
  - 512 B RAM
  - 8KB flash memory
    - OS code space: 3500 bytes
    - available code space: 4500 bytes
  - 10Kbps radio
- Berkeley's Smart Dust project
  - goal: node size < $1mm^3$
  - micro mirrors + laser beam
    - Micro-Electro-Mechanical Systems (MEMS)





Security goals

- Secure routing
- Message CIA
  - confidentiality, integrity authenticity
- Key & node revocation
- Network reinforcement
  - repeated deployment of sensors
- Node authentication

- Resiliency
  - redundancy
    - battery, utility failure
  - robustness
    - packet routing, active attack
  - node capture
    - tolerant to % compromised
    - perfect n.c. resilience - no other key but captured gets compromised

Threats

- Eavesdropping
- Message injection
- Message modification
- Message replay
- Impersonation
  - clones
- DoS
  - secure routing
  - malicious nodes
  - jamming
  - battery exhaustion
- Traffic analysis
- Side-channel analysis

Key distribution protocols (KDP)

- Common KDP schemes inappropriate as WSNs have
  - restricted resources (memory, power, CPU)
  - limited neighbours/network topology knowledge
  - small (or none) tamper resistance

- Basic protocol requirements:
  - support for large number of parties (10^3 - 10^6)
  - resource efficient
  - robust
    - single nodes compromise inevitable (no tamper resistance)
    - physical damage, battery exhaustion of single nodes
  - (trusted) base-station (BS) involvement problematic
    - single point of failure
    - strong data flow around BS (non-uniform power exhaustion)

Bootstrapping protocol phases

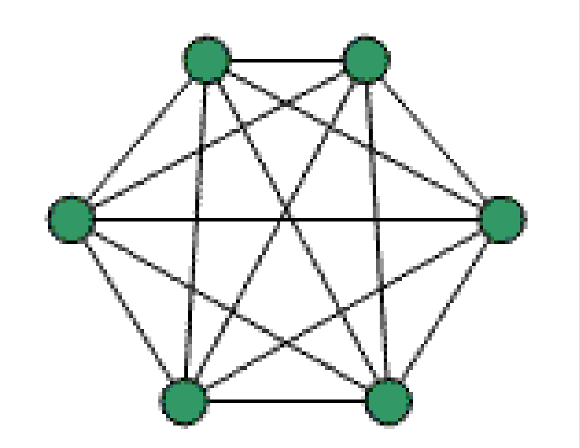
1. **Pre-deployment initialisation**
2. **Physical deployment**
3. **Neighbours discovery**
4. **Key setup**
   - Key discovery
   - Key exchange (plaintext key exchange)
5. **Key update (optionally)**
   - Secrecy amplification
   - Multi-path key reinforcement
6. **Message routing**

Global master key

- Single symmetric key shared by all nodes
  - used for initial link key exchange and than (ordinarily) erased
  - what if node gets broken? (landing failure …)
- Advantages:
  - minimal storage requirements
  - resistance against DoS (fast MAC computation)
- Disadvantages:
  - no node capture resilience
  - no nodes can be added later

Pairwise keys ((n-1) scheme)

- Unique key between each two nodes
- Each node must store (n-1) keys
- Advantages:
  - perfect resiliency to node capture
  - node-to-node authentication
- Disadvantages:
  - high production costs
  - high memory requirements
  - no re-deployed/re-enforcement later

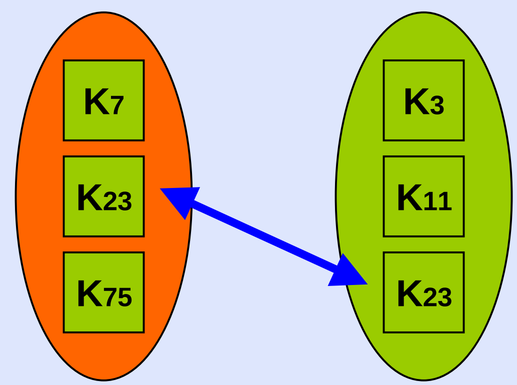


Public key cryptography

- Key pair for each node, signed by BS
- Advantages:
  - perfect node capture resilience
  - fully scalable, revocation possible
- Disadvantages:
  - need for high performance hardware
  - high memory/time/power requirements
  - battery exhaustion attack possible
    - high number of key establishment requests
- PK crypto doesn’t bring much compared to symmetric one (works better in centralised environments)

Random pre-distribution (EG)

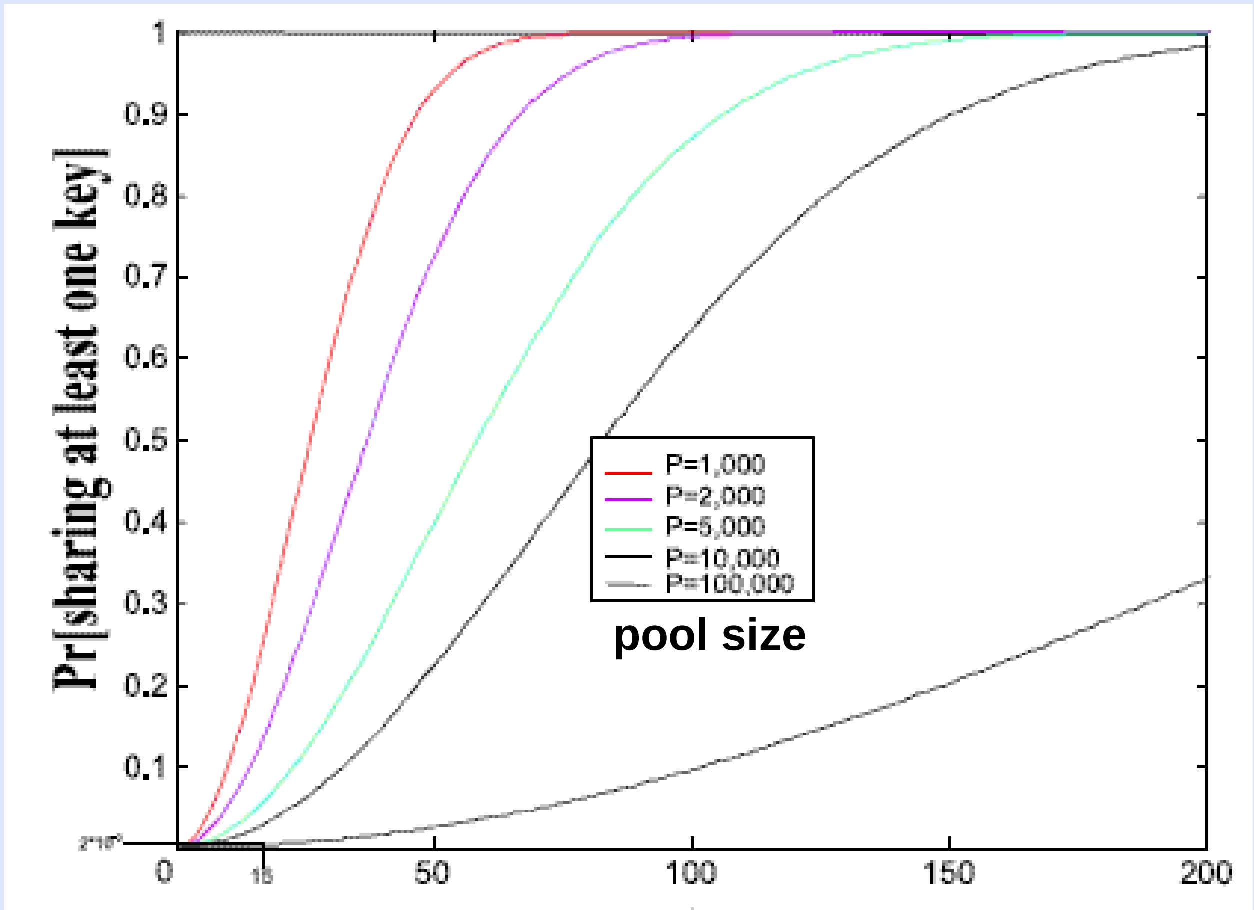
- Idea (Eschenauer, Gligor - 2002):
  - two neighbours share pre-distributed key only with a certain probability *p* <(<<) *1*
  - basically, we need a connected network, not link keys
- Pre-deployment phase
  - large key pool S, each key with a unique ID
  - each node obtains random subset of *m* keys (no replacement)



- Key setup
  - neighbours use a common link key if such exists

Random pre-distribution (cont.)

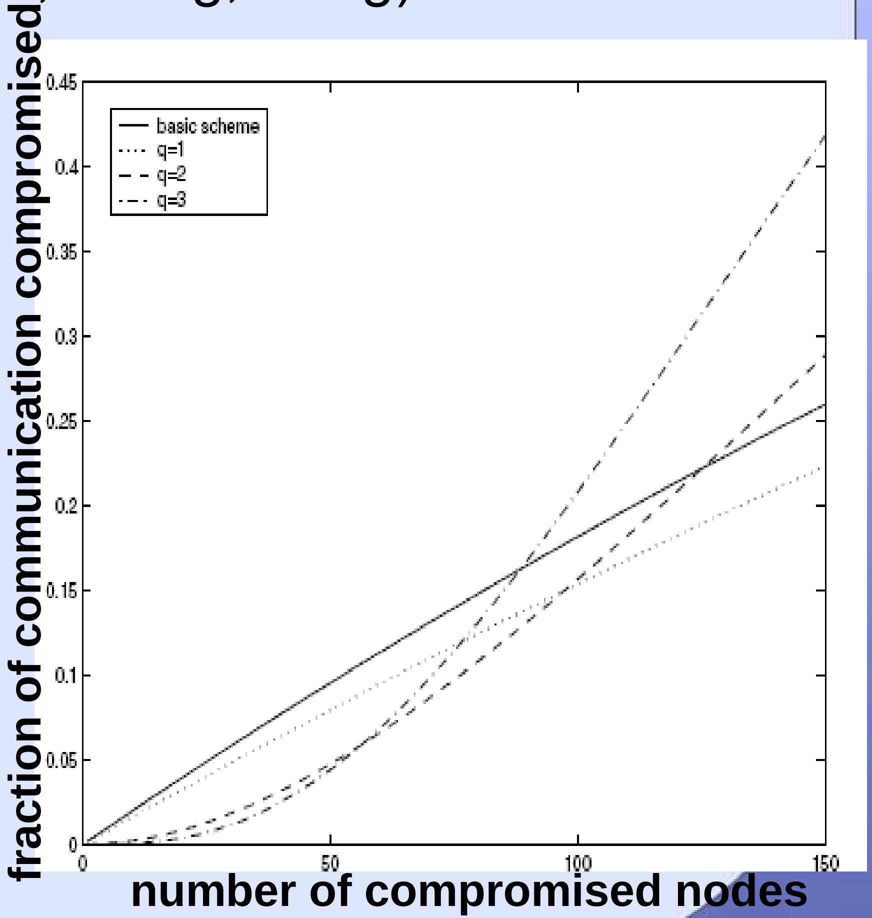
- Probability, that two neighbors shares at least one key
  - key pool P
  - ring size m



key ring size m

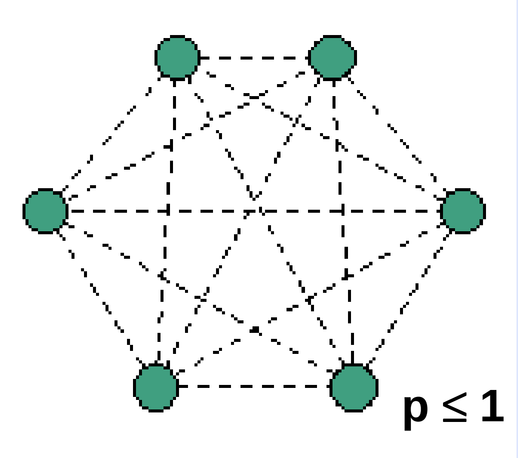
Random pre-distribution (q-EG)

- Variation of the previous scheme (Chan, Perrig, Song)
  - (EG ~ 1-EG)
- q ≥ 1 common keys required
  - K' = hash(K_1| ... | K_q)
- Node capture resilience should be improved *BUT:*
  - to keep link probability *p* same:
    - ring size m must be increased
    - pool size S must be decreased
  - …thus increasing # of compromised keys per node
- Search for function(p,m,|S|) optimum



Random pairwise key scheme

- Idea (Chan, Perrig, Song - 2003):
  - two neighbours share pre-distributed pairwise key with $p \leq 1$
- Pre-deployment phase:
  - pairwise keys for *m* randomly chosen nodes
  - key between given two nodes is predetermined or not exists (not looking for random one)
- Properties:
  - perfect resilience to node capture
  - node-to-node authentication
  - limited network size (n = m / p)

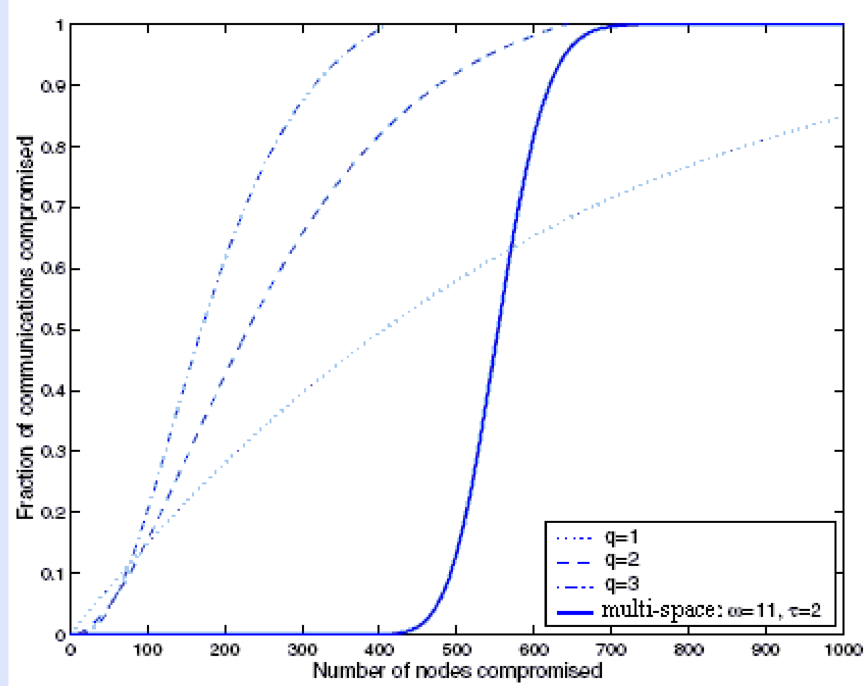


Single space pairwise keys

- Blom’s pairwise key pre-distribution schemes
- Each two nodes can compute unique pairwise key from their public and private values
- Less memory costing than (n-1) scheme
  - $\lambda + 1$ elements ($\sim \lambda + 1$ keys)
- Perfectly secure until λ nodes captured
  - but totally compromised when $> \lambda$ captured
- Still inconvenient for WSN
  - linear dependency between memory and security

Multi-space pairwise keys

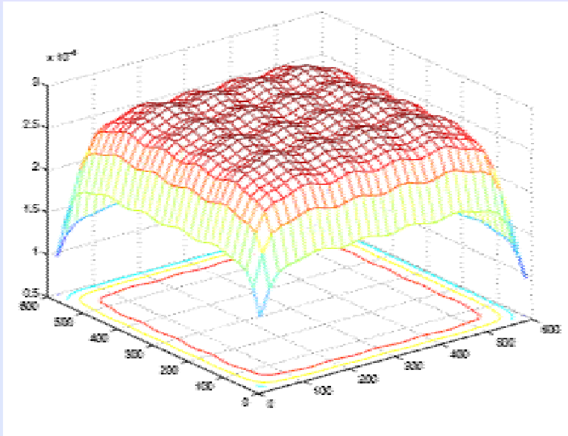
- (Du, Deng, Han, Varshney - 2003)
- Combination of Single-space + EG
  - key pool S contains Blom's key spaces
  - random subset for each node
  - pairwise key is constructed from shared Blom’s space
- More resilient than EG until treshold reached

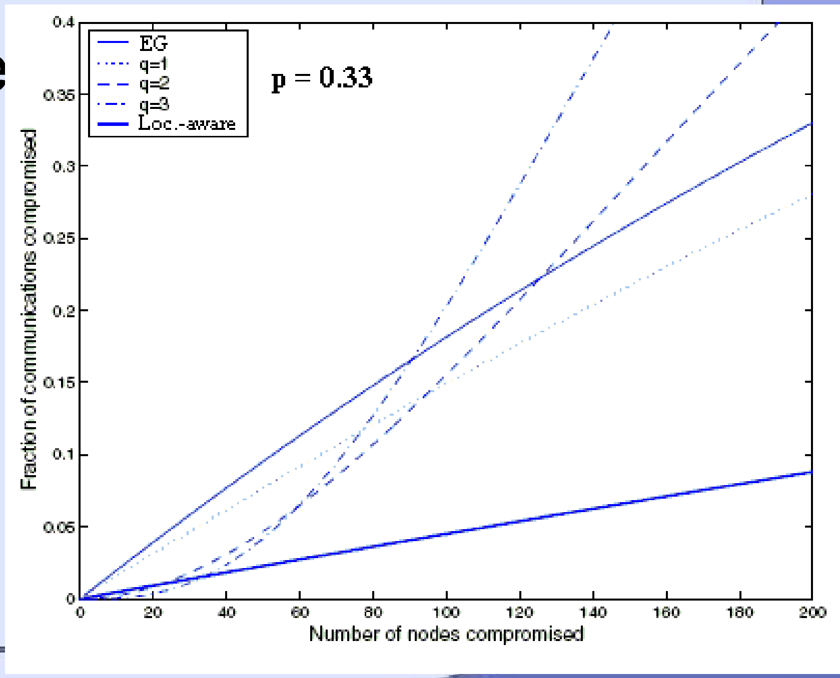


(a) $m = 200, p_{actual} = 0.33$

Location aware pre-deployment

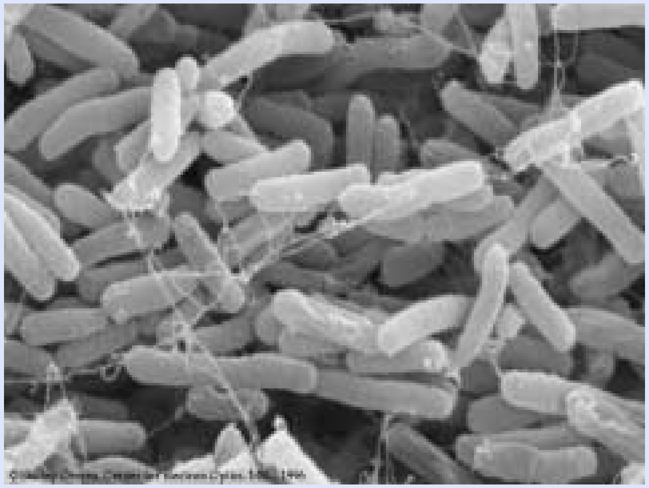
- Limited location knowledge can be available (Du et.al.)
  - same deployment “barrel”…
- Deployment area grid
  - nodes from near “cells” are more probable to be communication neighbours
- One of previous schemes is performed “locally” for group of probable neighbour nodes





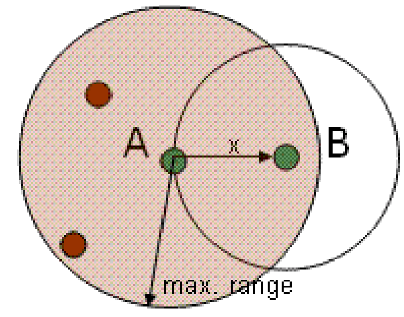
Key Infection - motivation

- More realistic attacker model
  - not able to eavesdrop the whole network
  - only a certain number of attacker’s (black) nodes
- Atomic data from sensors are not sensitive
  - the real value is in aggregates
  - we don’t try to secure all nodes but just majority
- No keys are hardwired in notes
  - low production costs
  - no danger in pre-deployment phases
- key material is distributed by ‘contact’, same as natural infection does

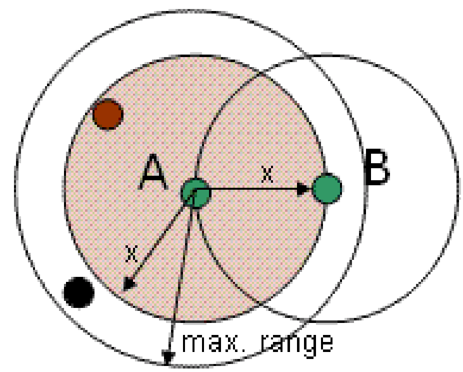


Key Infection - principle

- Restricted attacker's model
  - black/white ratio << 1
  - sensitive period - just after deployment
- Plaintext key exchange with neighbours
  - keys established after deployment
  - after any network re-deployment
- Transmission modes
  - Maximum screaming
    - max. transmission power being used
  - Whispering
    - power is gradually increased until a neighbour reached



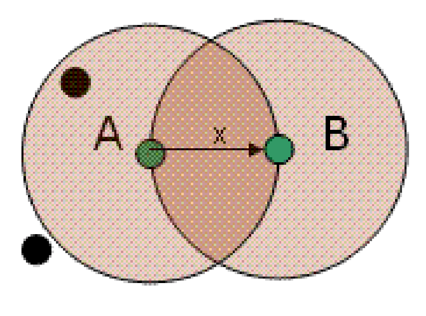
maximum screaming



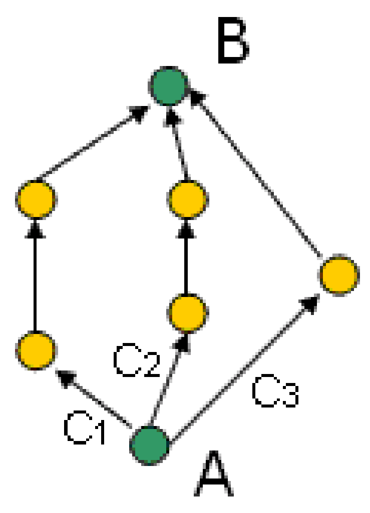
whispering

Secrecy amplification

- Mutual whispering
  - directional basic whispering
  - $K = K_{AB} \oplus K_{BA}$
- Multipath key establishment
  - key update throw values $C_1, ..., C_n$ along different paths
  - $K' = K \oplus C_1 \oplus ... \oplus C_n$
  - attacker must eavesdrop all paths

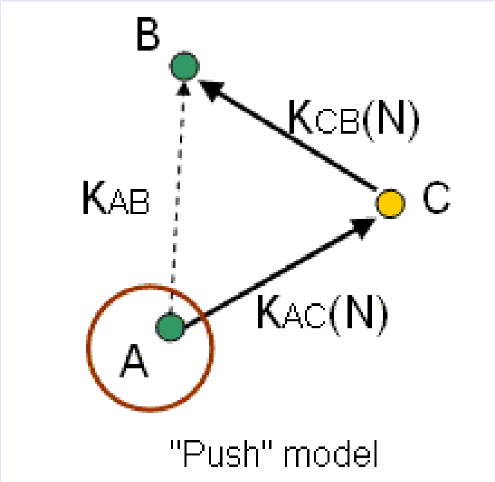


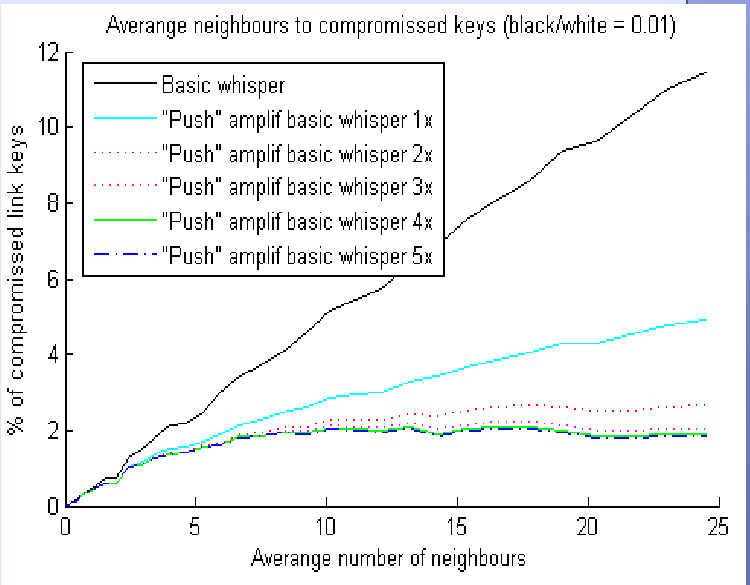
mutual whispering



Multi-hop key establishment

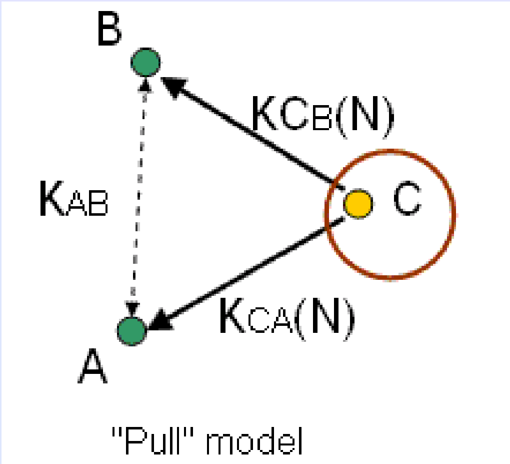
- Neighbours involved in key update
  - 2-hop scheme: A, B participants, C mediator
  - mediators immediately forget temporary values
- “Push” model (Ross, Perrig, Chan)
  - initialised by participant A
  - A asks mediator C to amplify K_{AB} by re-transmitting a number N
  - $K'_{AB} = H(K_{AB} \mid\mid N)$
  - both A and B update link key to K'_{AB}



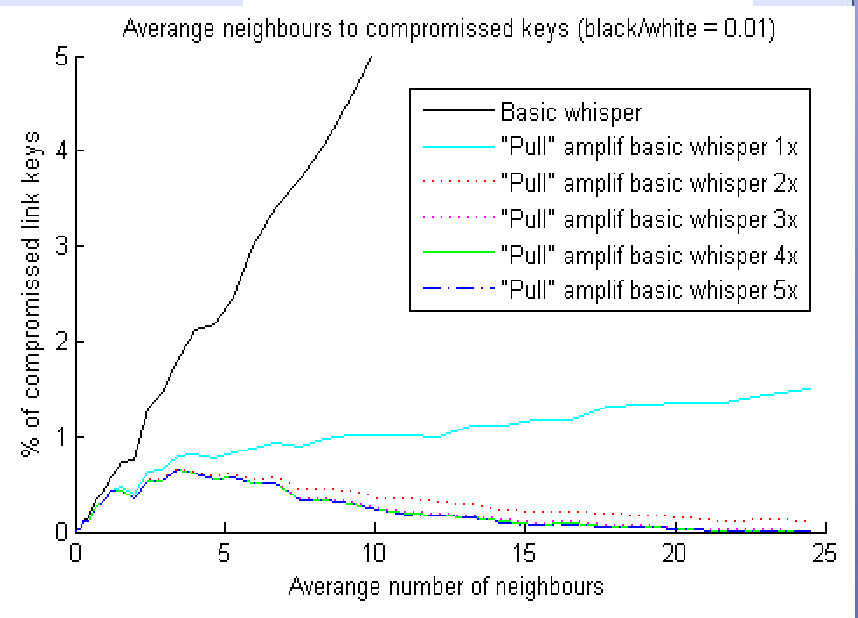


"Pull" model – our initial idea

- "Pull" model
  - initialised by mediator C
  - C decides to amplify K_{AB} for A and B by sending N
  - $K'_{AB} = H(K_{AB} || N)$
  - both A and B update link key K_{AB}
  - can be performed continually
    - ~ 3x amplification gives substantial improvements

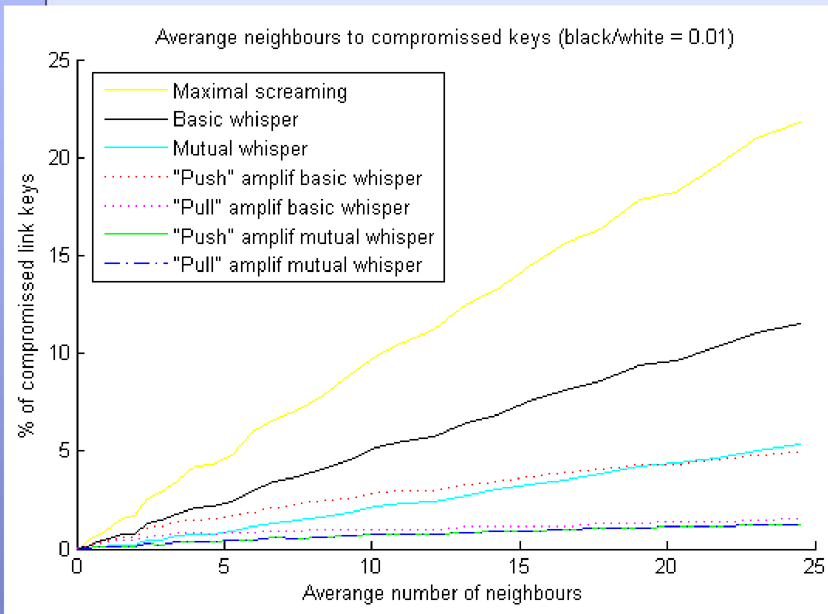


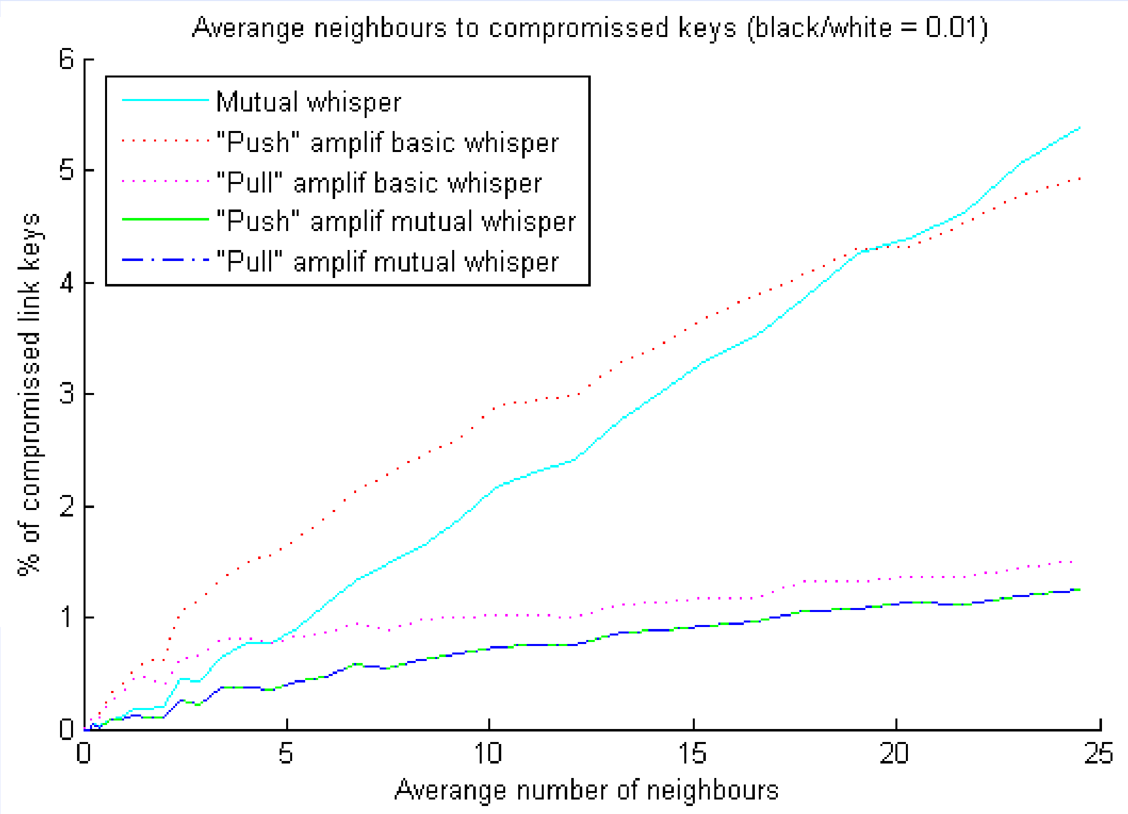
"Pull" model



Key infection - comparison

- average neighbours to compromised keys

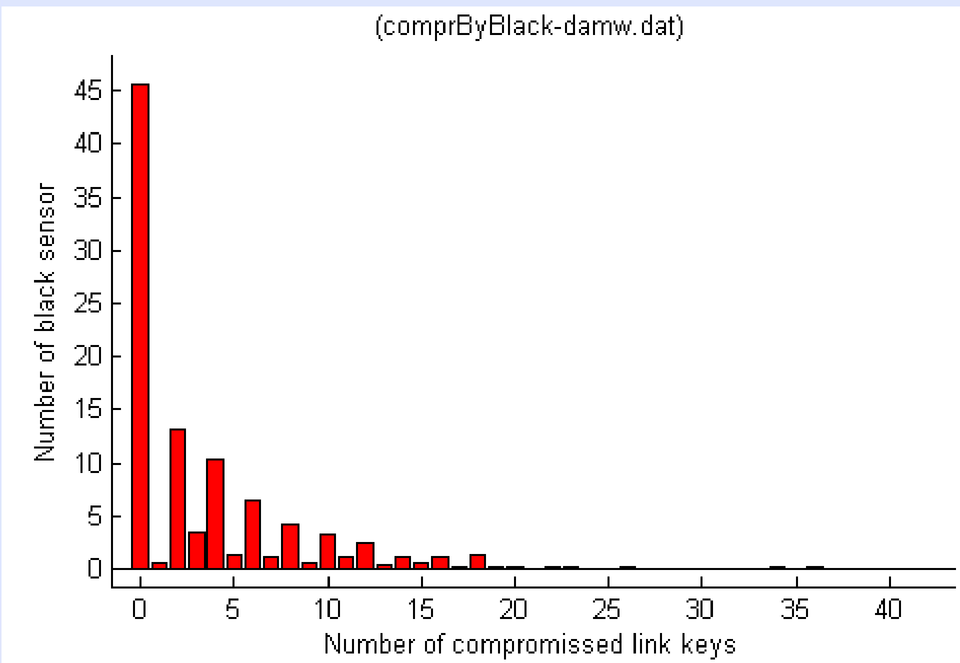


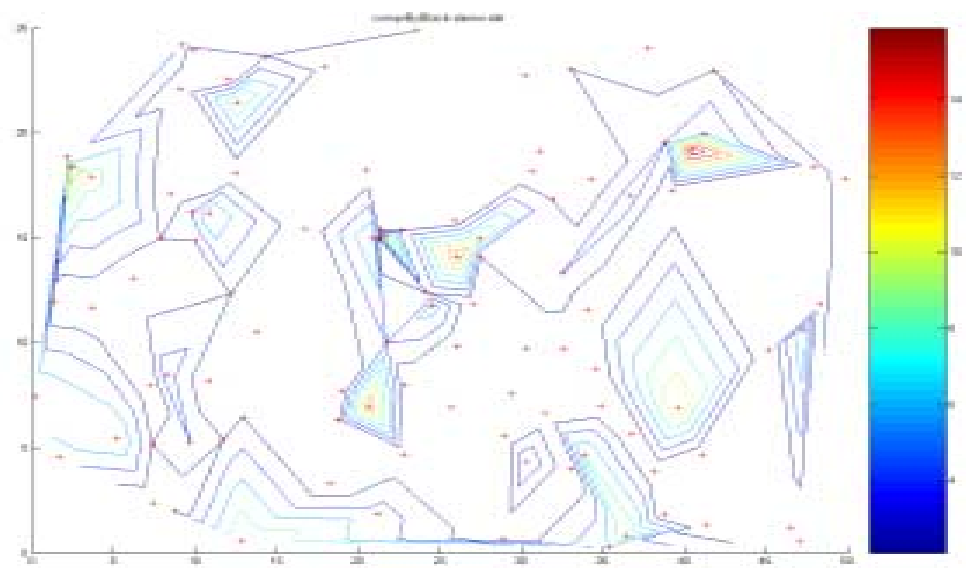


- *(mutual + “Push”) is equal to (whisper + “Pull”)*

Key infection properties II.

- Link keys are not compromised regularly
  - highly insecure areas
  - most areas are more secure than average

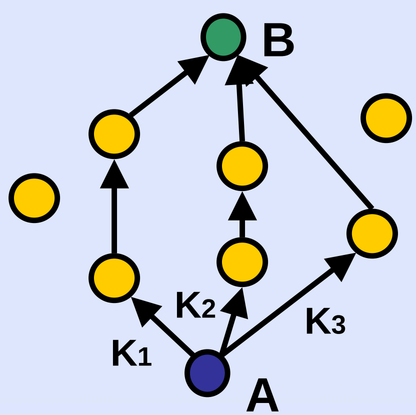


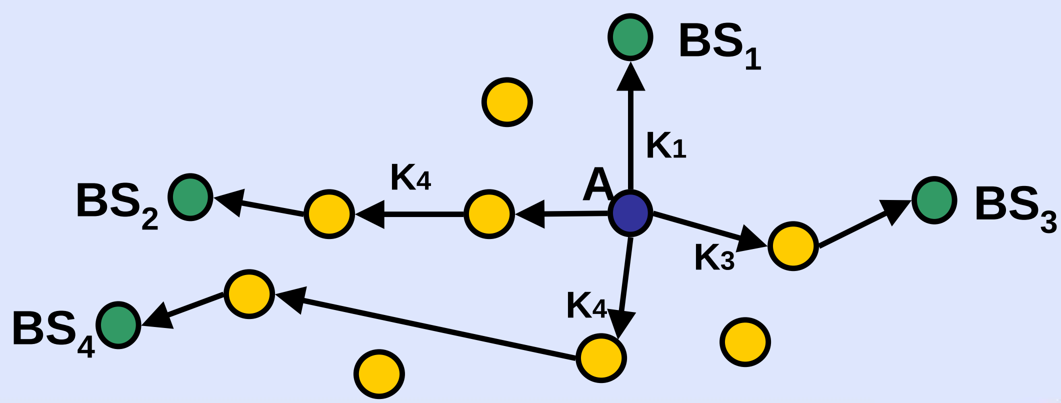


Compromised link key distribution (black/white = 0,01)
“Pull” amplification

Multi-path message routing

- Based on non-regular keys compromise
- Message is encrypted by multiple keys
  - more different BS or paths to one needed
  - each key is send along different path
  - attacker must eavesdrop all paths

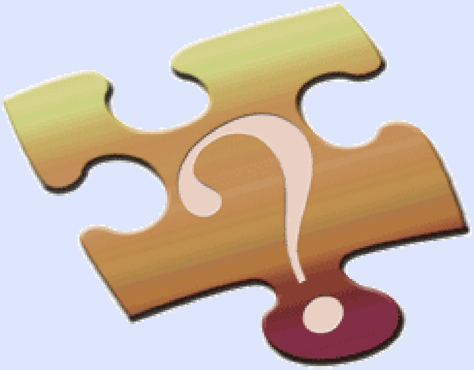




Conclusions

- WSN are expected to be in wide use
  - security is important for most scenarios
- Common KDP approaches inappropriate
  - restricted resources (memory, CPU, …)
- Resources efficient and intrusion tolerant schemes need to be developed
  - randomised approaches
  - plaintext key exchange
  - multipath message delivery

Questions to WSN?



- Remotely Keyed Encryption

Overview

- Untrusted PC and smart card
  - high-speed encryption
  - classic approaches
- Remotely Keyed Encryption (RKE)
  - attacker models
  - protocol goals
  - selected modes
  - performance comparison

Classical approaches

- Smart card as secure carrier
  - key is stored on card, loaded to PC before encryption, then erased
  - high speed encryption (>>MB/sec)
  - attacker with access to PC during encryption will obtain the key
- Smart card as encryption device
  - key never leaves card, PC sends data to encrypt
  - low speed encryption (~kB/sec)
  - attacker must attack smart card

RKE – requirements, idea

- Requirements:
  - high speed encryption
  - key never leaves smart card
  - encryption/decryption is possible only when smart card is present
- Idea: use on-card encryption, but move heavy work to PC in secure way
  - Remotely Keyed Encryption (Blaze 1996)

RKE call diagram

1. Initial request V_1, file (in)dependent

Process request, *save State*

2. Response R_1 depends on V_1 and key

Encrypt file with R_1

3. Request V_2 depends on encrypted file

4. Response R_2 depends on key, V_2 and State

Modify file with R_2

Attacker models

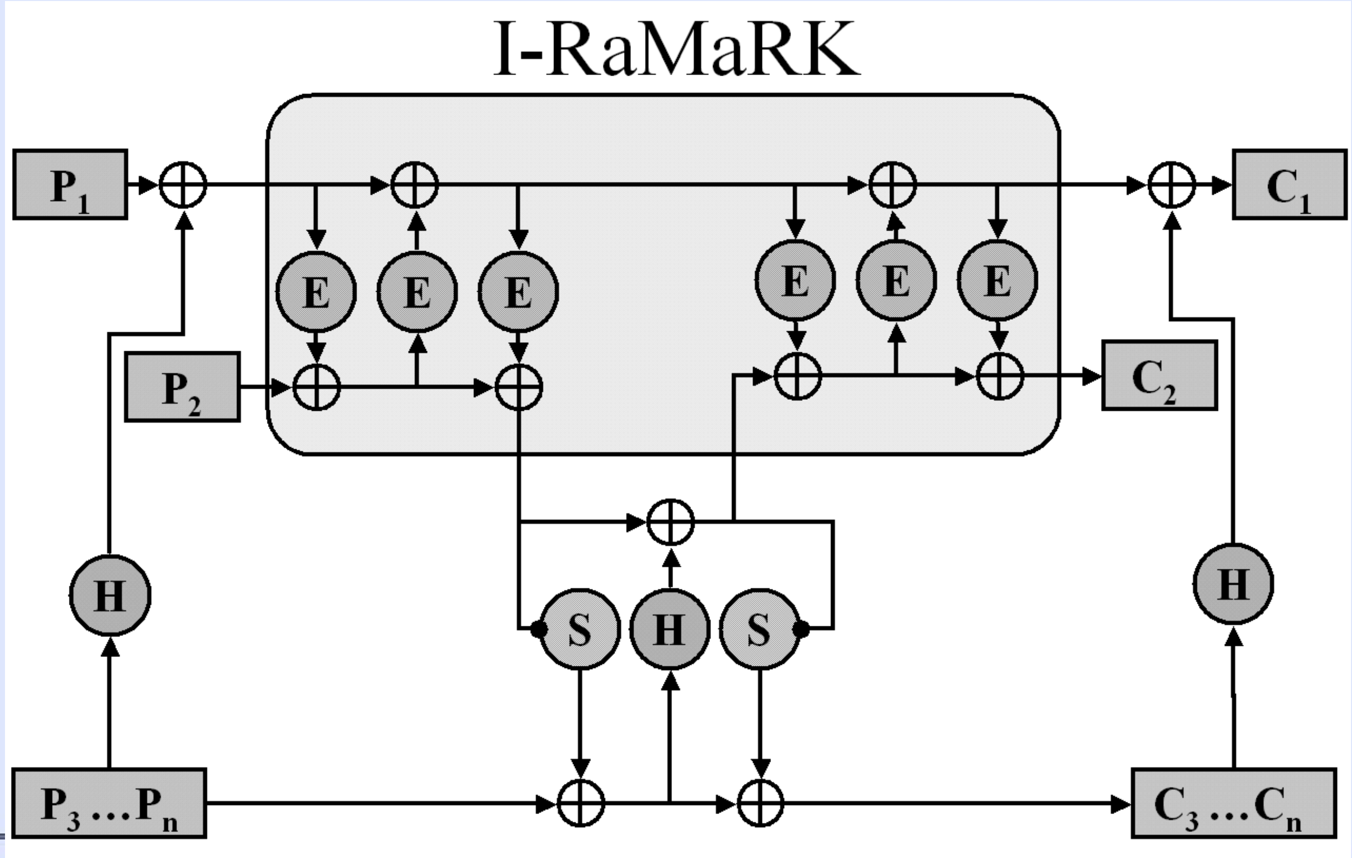
- Basic model (Blaze 96)
  - attacker have no access to SC
  - cannot create own requests
  - attacker completely control PC (ops, values)
- Strong BFN model (BFN 98)
  - attacker had access to SC for limited time
  - was able to create own request (database)
  - no access now

Strong attacker model goals

- Inversion secure
  - attacker with access to decryption engine is not able to perform encryption and vice versa
- Pseudorandom indistinguishable
  - encrypted text is indistinguishable from random string
- Forgery secure
  - attacker is to able to encrypt/decrypt messages different from used requests

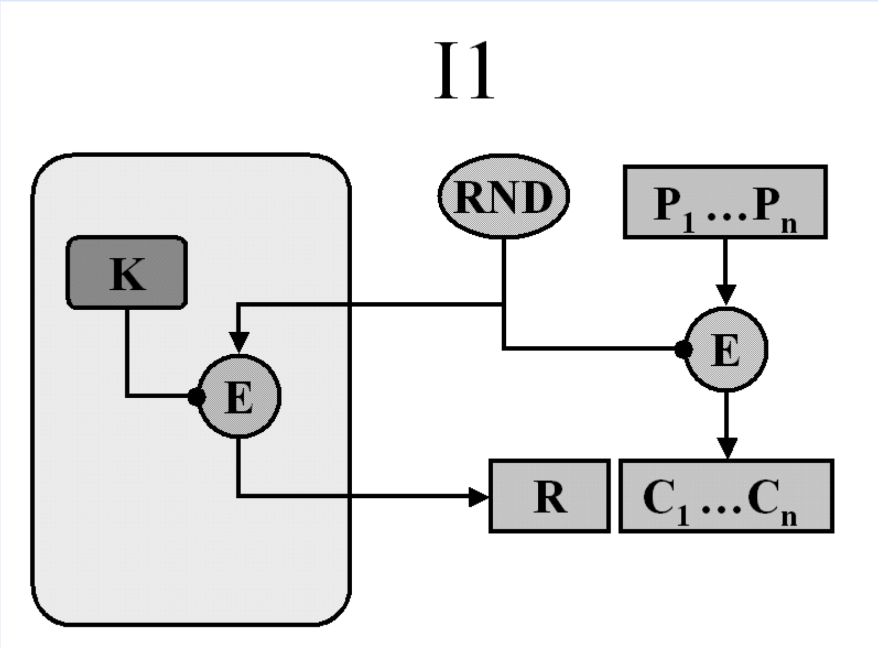
I-RaMaRK

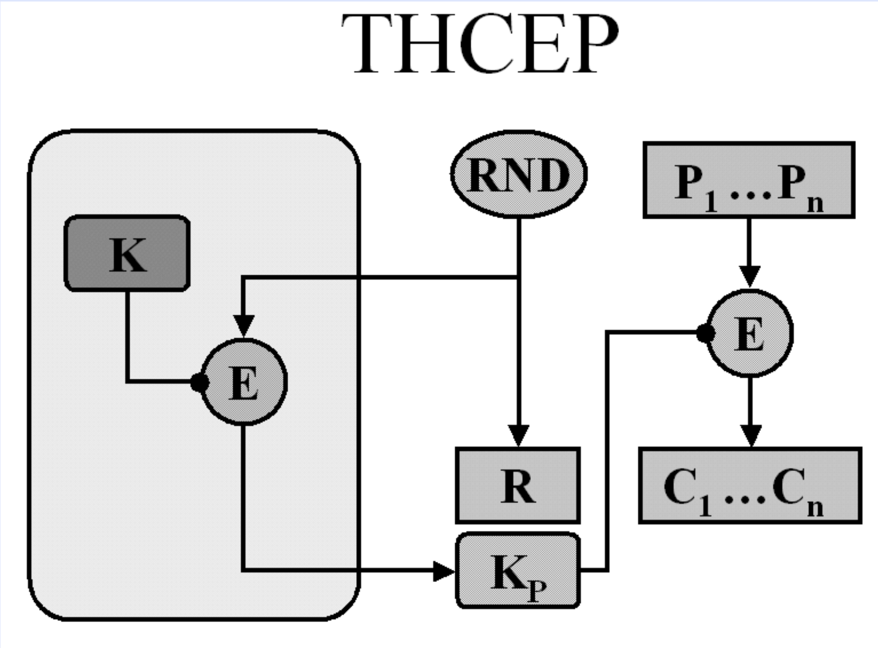
- First secure mode for RKE (strong model)
- Requires 2 APDU messages



I1 and THCEP

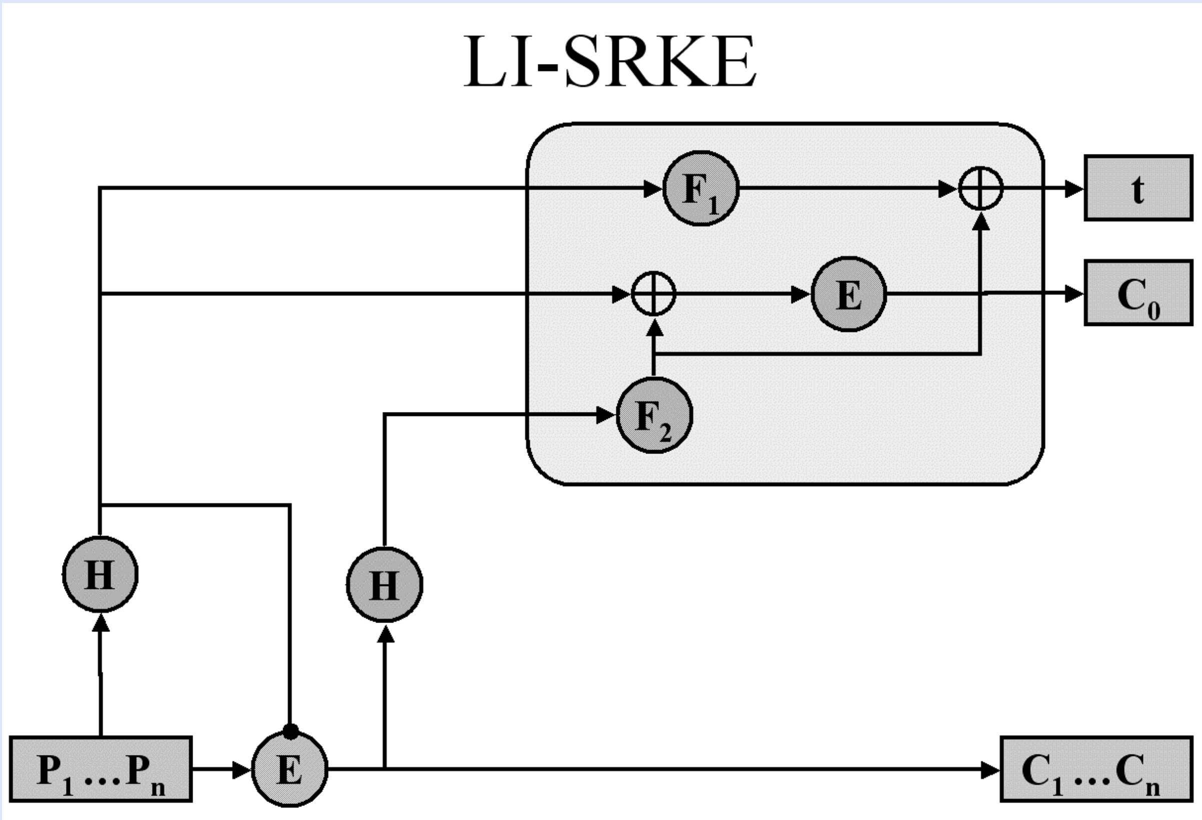
- Fast modes for basic attacker model
  - not inversion/forgery secure, key independent of file
- Requires only 1 APDU message





Length-Increasing RKE

- 1 APDU mode for strong attacker model
  - randomization nonce must be used

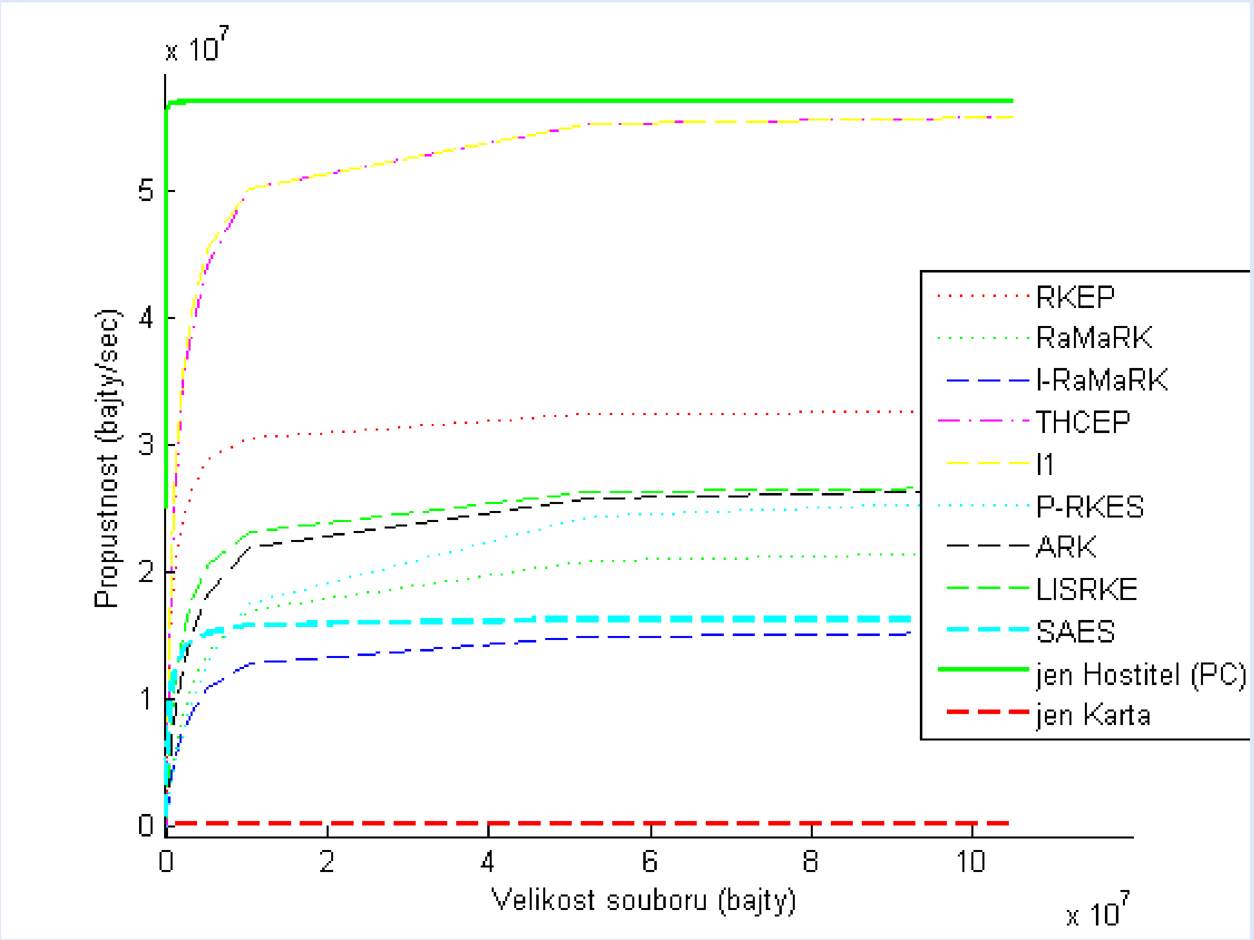


Modes history

- RKES (96) – basic model, broken
- RaMaRK (97) – basic model, broken
- P-RKES (98) – strong model, 2 APDU
- ARK (99) – strong model, 2 APDU
- SAES (99) – strong model, 2 APDU
- THCEP, I1 (00) – basic model, 1 APDU
- I-RaMaRK (00) – strong model, 2 APDU
- LI-SRKE (00) – strong model, 1 APDU

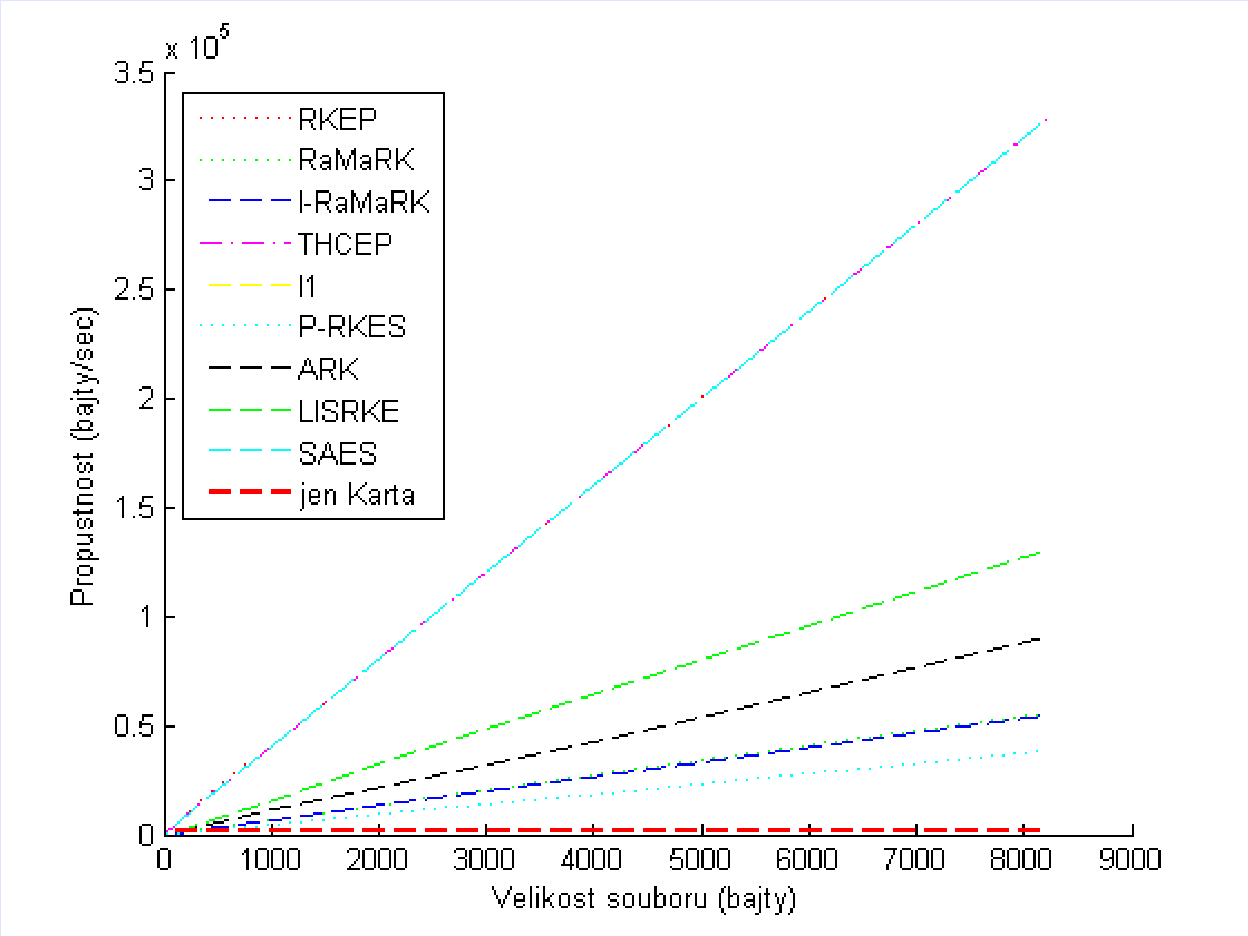
Performance comparison

- all modes, 32B-100MB (B/s)



Performance comparison (cont.)

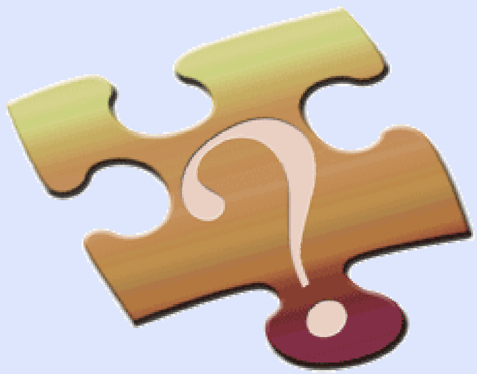
- selected modes, 32B-8kB (B/s)



Conclusions

- Secure high speed encryption
  - movie decryption
  - file disk encryption
- Key never leaves smart card
- Most work moved to untrusted host
- Modified attacker model
  - basic and strong model
  - temporal access to smart card

Questions to RKE?



❍ Block Cipher Modes for Authenticated Encryption

Overview

- Message confidentiality, integrity, privacy
- Encryption, MAC, composition
  - classic modes for block ciphers
  - types of compositions
- Authenticated encryption modes (AE)
  - usage scenarios
  - important features
  - selected modes

Confidentiality, integrity, privacy

- Message confidentiality [encryption]
  - attacker is not able to obtain info about plaintext
- Message integrity [MAC]
  - attacker is not able to modify message without being detected (PTX, CTX)
- Message privacy [encryption]
  - attacker is not able to distinguish between encrypted message and random string
  - same message is encrypted each time differently

Encryption and MAC composition

- Modes for block ciphers (CBC, CTR, CBC-MAC)
- Compositions (encryption + MAC)
  - encrypt-and-mac [$E_{Ke,Km}(M) = E_{Ke}(M) \mid T_{Km}(M)$]
    - can fail with privacy and authenticity
  - mac-then-encrypt [$E_{Ke,Km}(M) = E_{Ke}(M \mid T_{Km}(M))$]
    - can fail with authenticity
  - encrypt-then-mac [$E_{Ke,Km}(M) = E_{Ke}(M) \| T_{Km}(E_{Ke}(M)$]
    - always provides privacy and authenticity
- Paralelizability issue
- Authenticated-encryption modes (AE)
  - special block cipher modes for composed process

Usage scenarios

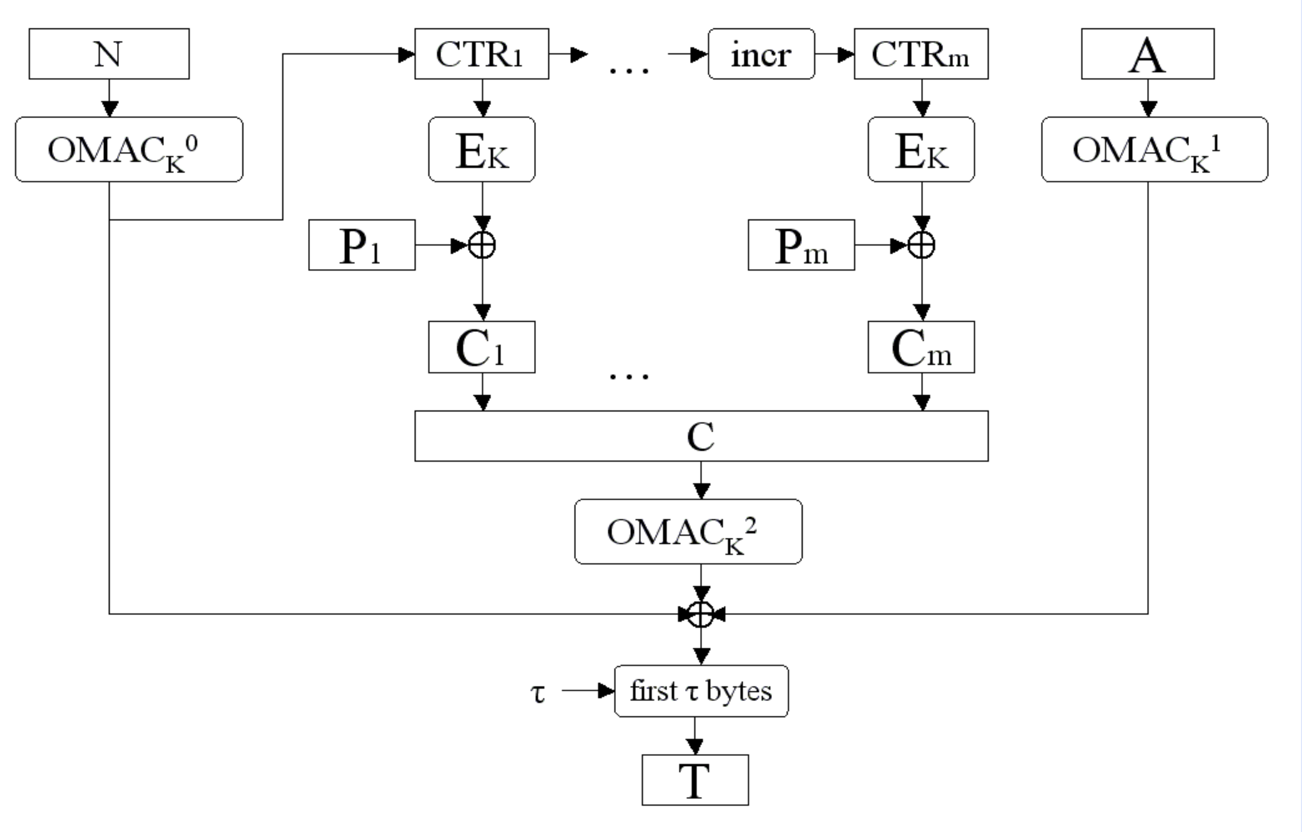
- Powerful, parallelizable environments
  - hardware accelerators
- Powerful, but almost serial environments
  - personal computer, PDA
- Restricted environments
  - smart card, cellular phone
- Different scenarios have different needs

Important features for AE modes

- Provable security
- Performance, paralelizability, memory req.
  - important for high-speed encryption, SC
- Patent
  - first AE modes were patented
- Associated data authentication
  - authentication of non-encrypted part
- Online, incremental MAC, number of keys, endian dependency …

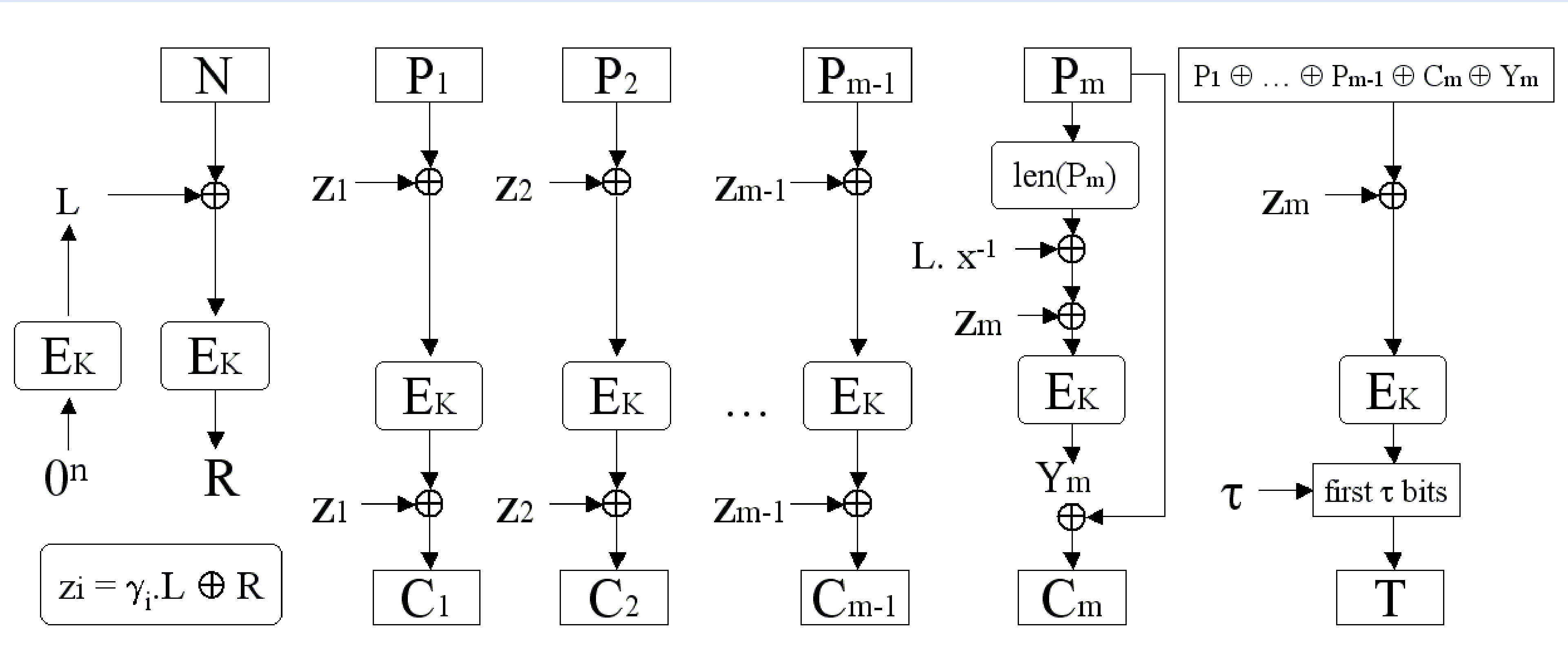
EAX mode

- Encrypt-than-mac composition
- Provable secure, unpatented



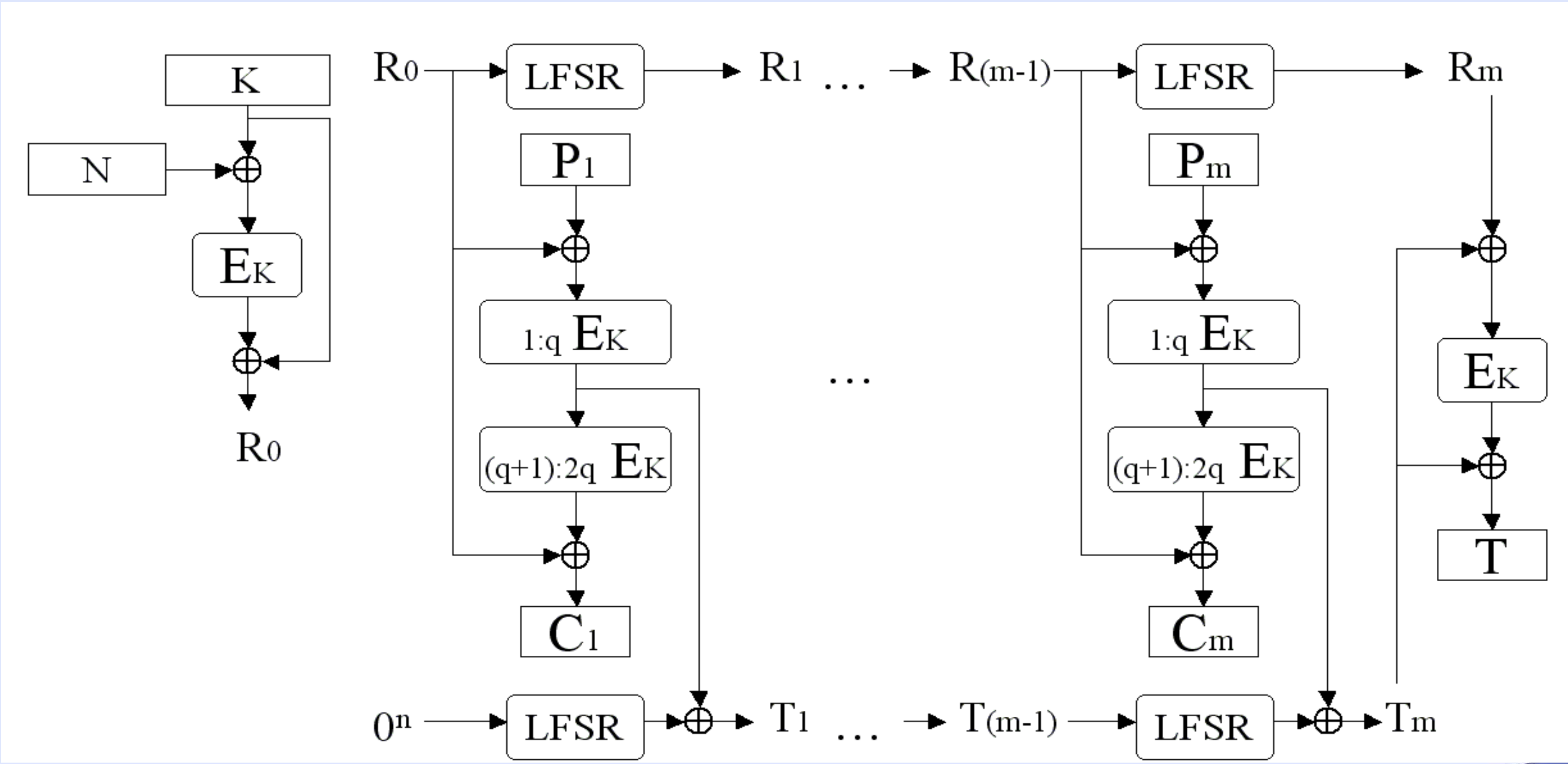
Offset CodeBook mode (OCB)

- Memory efficient, fast mode
- Provable secure, but patented



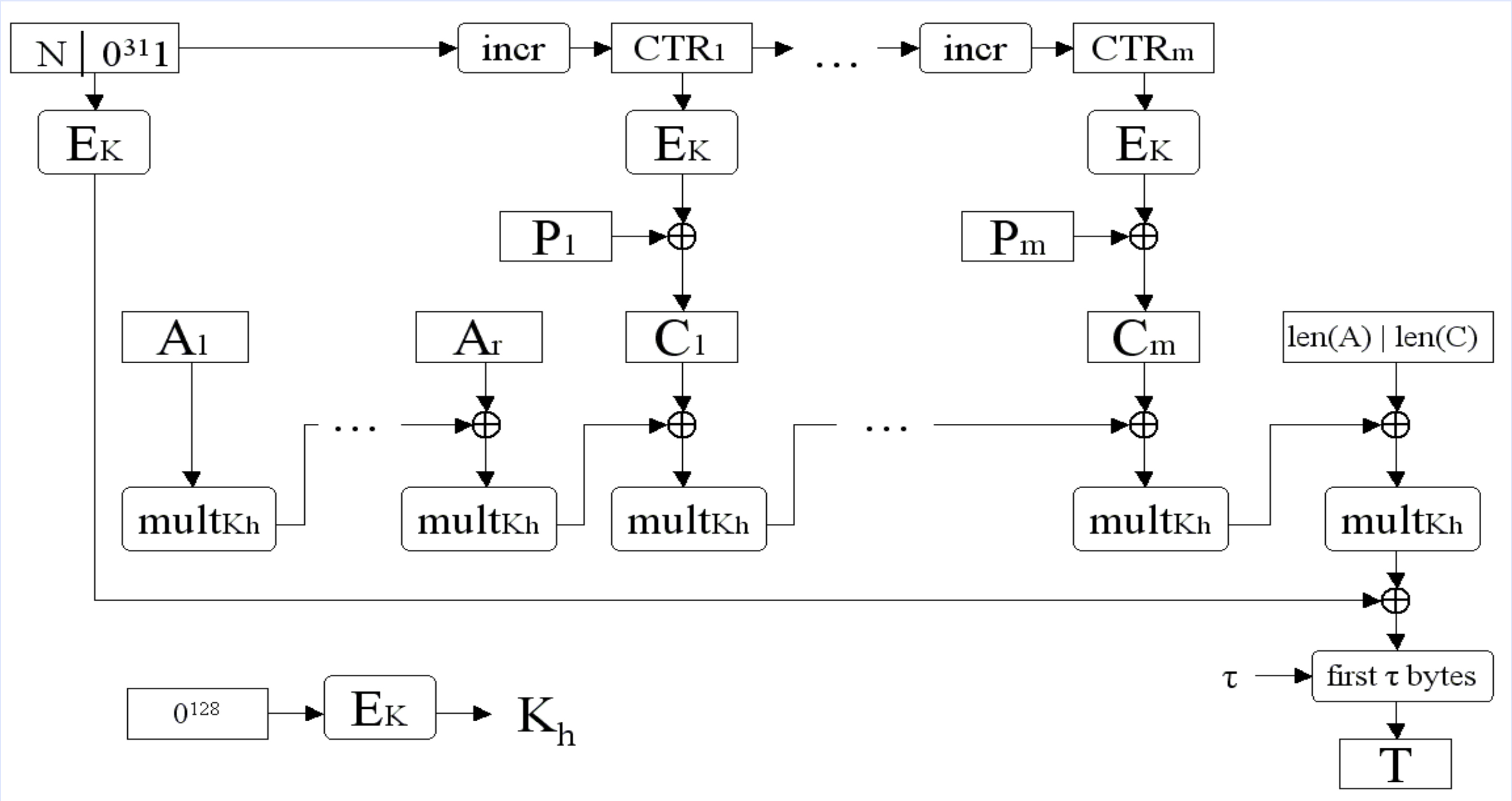
Cipher-State mode (CS)

- Memory efficient, fast mode, unpatented
- Not provable secure (inner state of cipher)



Galois/Counter Mode (GCM)

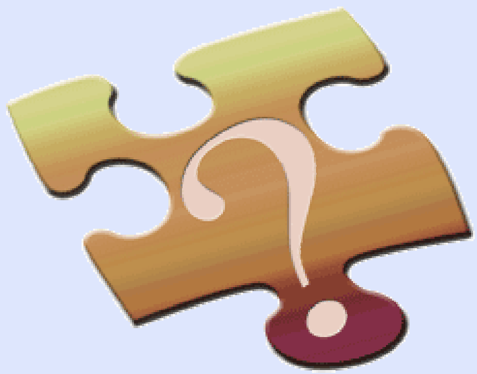
- Need pre-computed table (4kB-64kB)
- fast mode, provable secure, unpatented



Conclusions

- Composition of ENC and MAC can fail
  - encrypt-then-mac provable secure
  - specially designed composed modes
- Most promising mode is patented (OCB)
  - fast alternative GCM, CS
- Suitable mode depends on usage
  - paralelizability, memory
  - specific needs (online, incremental MAC)

Questions to AE?



Thank you.