Security in Sensor and Ad-Hoc Networks

Virgil D. Gligor

gligor@umd.edu

Electrical and Computer Engineering Department

University of Maryland

College Park, Maryland 20742

FOSAD 2004

Bertinoro, Italy

September 5-11, 2004

Outline

- Sensor Networks
  - Security Requirements
  - Ex. Computational Constraints, Lightweight Cryptographic Primitives
- Key Distribution Schemes
  - Impractical Approaches
  - Basic Scheme for Key Pre-distribution and its Extensions
  - Random Pair-wise Key Pre-distribution
  - Multiple Key Spaces
- Key (and Node) Revocation
  - Centralized
  - Distributed
- Trust Establishment in Mobile Ad-Hoc Networks (MANETs)
  - Trust Establishment Scenarios
- Research Areas

Sensor Networks

- Similar to "traditional" embedded wireless networks
  - arrays of sensors
    - battery powered, limited computational and communication capabilities
    - intermittent wireless communication
  - base stations: data collection nodes, control nodes (possibly mobile)

- Important differences
  - scale
    - 10,000 as opposed to 100
  - ad-hoc deployment
    - by scattering sensor nodes on a large area (e.g., via aerial vehicles)
    - incremental addition and deletion of nodes after deployment
  - potentially hostile environments
    - sensor nodes monitoring, capture, replication, insertion, and input manipulation

Security Requirements

- secure node-to-node communication
  - both for already-deployed and for incrementally added nodes
  - no *a priori* knowledge of node neighbors
  - scalable security mechanisms and protocols
- resistance to DoS (e.g., battery depletion) attacks
  - minimal computational, storage, and communication resources
  - lightweight cryptographic primitives
- selective revocation of (captured) keys and nodes
- resilience to node capture
  - minimal number of *non-captured* nodes affected
- resilience to insertion of illegitimate nodes in network

Examples of Computational Constraints

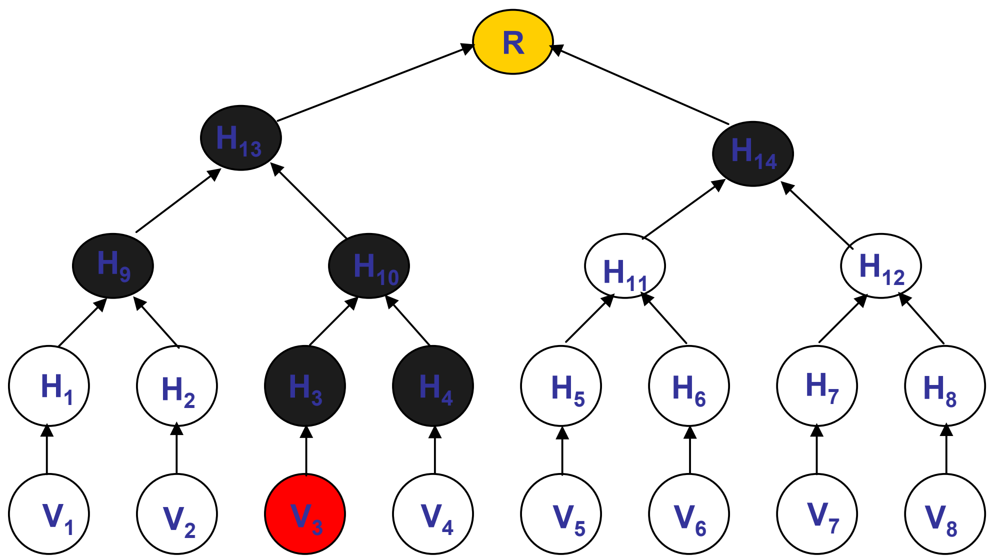
- Wide range but *limited* processing capabilities: > 10 x
  - Atmel Atmega 128L -> … MC68328 "DragonBall" … -> MIPS R4000
    (8 bit, 4 Mhz, 4KB SRAM) (32 bit, 16 MHz) (64 bit, 80 MHz)

- Traditional asymmetric cryptosystems are impractical *(in "this range")*
  - Encryption/Signatures - MC68328 "DragonBall" [CKM**2000**]
    - 1024-bit RSA encryption/signature vs. 1024 bit AES encryption (42/840 mJ vs. 0.104mJ)
  - Communication: ~ 0.5 of Computing Energy - Sensoria WINS NG RF
    - 1024-bit block over 900m at 10Kbps and 10 mW - 21.5 mJ
    - lower energy consumption for transmission on smaller distances
  - ECC encryption/signature: much better, but not good enough
    - same order as RSA encryption (at high end)
  - Vulnerability to DoS attacks

Examples of Lightweight Cryptographic Primitives

- ***Hash Functions (one-way, collision-resistant)***
  - ***- 5 - 7x faster than symmetric (block) encryption***
  - ***- 3 - 5 orders of magnitude faster than public-key signatures***

- ***Hash trees*** **(lightweight, if *no. of leaves* is small)**



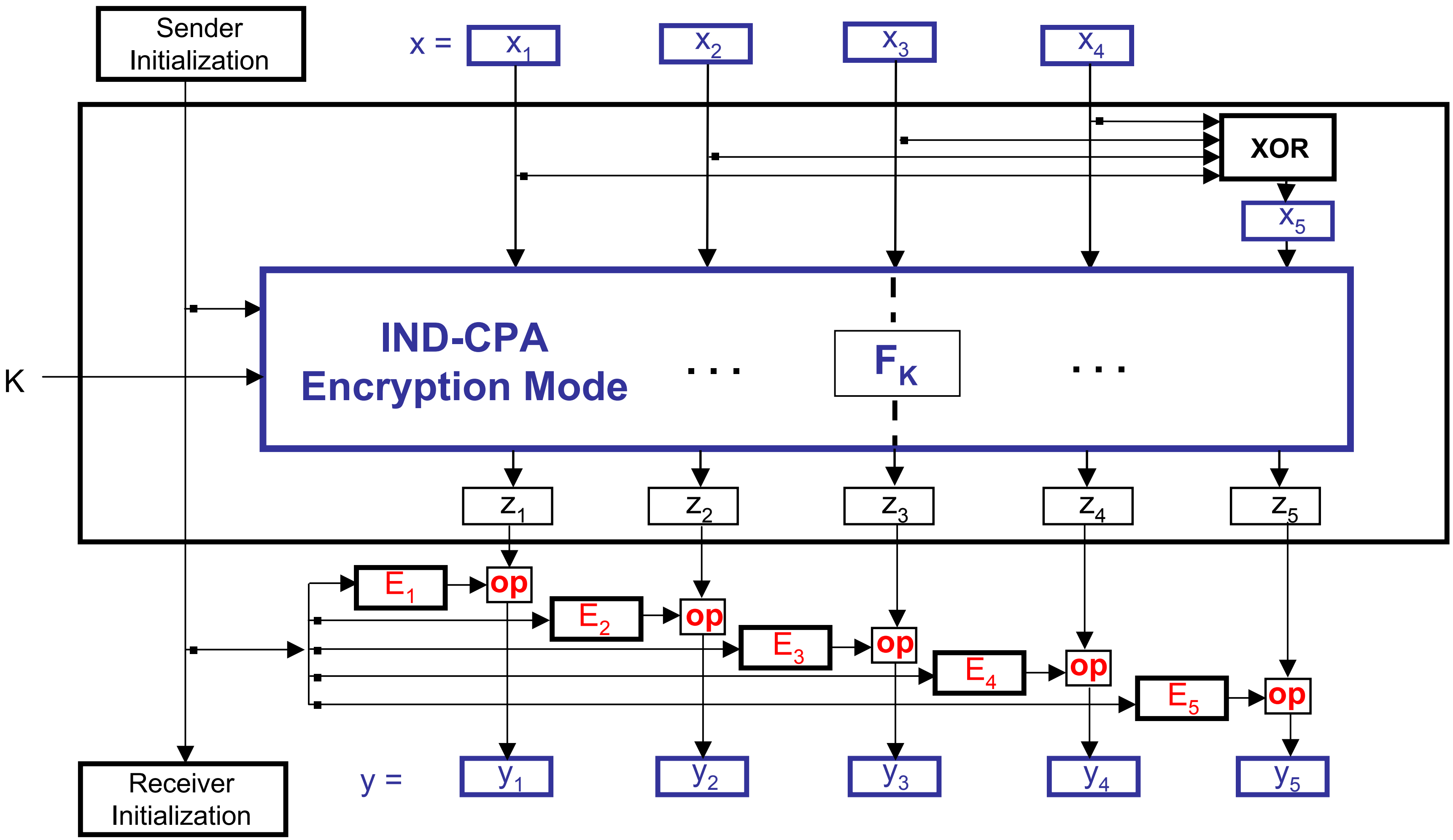
- ***Random Polynomials of degree t*** **(lightweight, if *t* is small)**

  $q(x) = a_o + a_1x + a_2x^2 + \ldots + a_{t-1}x^{t-1}$, a_i = *secret, random values* in $[0, \ell\text{-}1]$

  $hash(q(x)) = hash(a_o|a_1|\ a_2|\ \ldots|\ a_{t-1})$

Examples of Lightweight Cryptographic Primitives

- **Authenticated Encryption (AE) in 1 pass - 1 crypto primitive**



XCBC-XOR [GD 00], IACBC [Jutla00]

Lightweight E_i, op? Under What Conditions?

1. IND-CPA encryption mode: processes block x_i and inputs result to block cipher (SPRP) F_K

2. **"op"** has an inverse

3. Elements **E_i** are **unpredictable, $1 \leq i \leq n_m+1$,** *and*
   E^p_i op^{-1} E^q_j are **unpredictable,** where **$(p, i) \neq (q, j)$**
   and messages **p,q** are encrypted with **same key K**

4. Additional mechanisms for length control, padding

Examples

op = mod +/- **; $E_i = r_0 \times i$;** **($E_0 = r_0$; $E_i = E_{i-1} + r_0$)** [GD00]

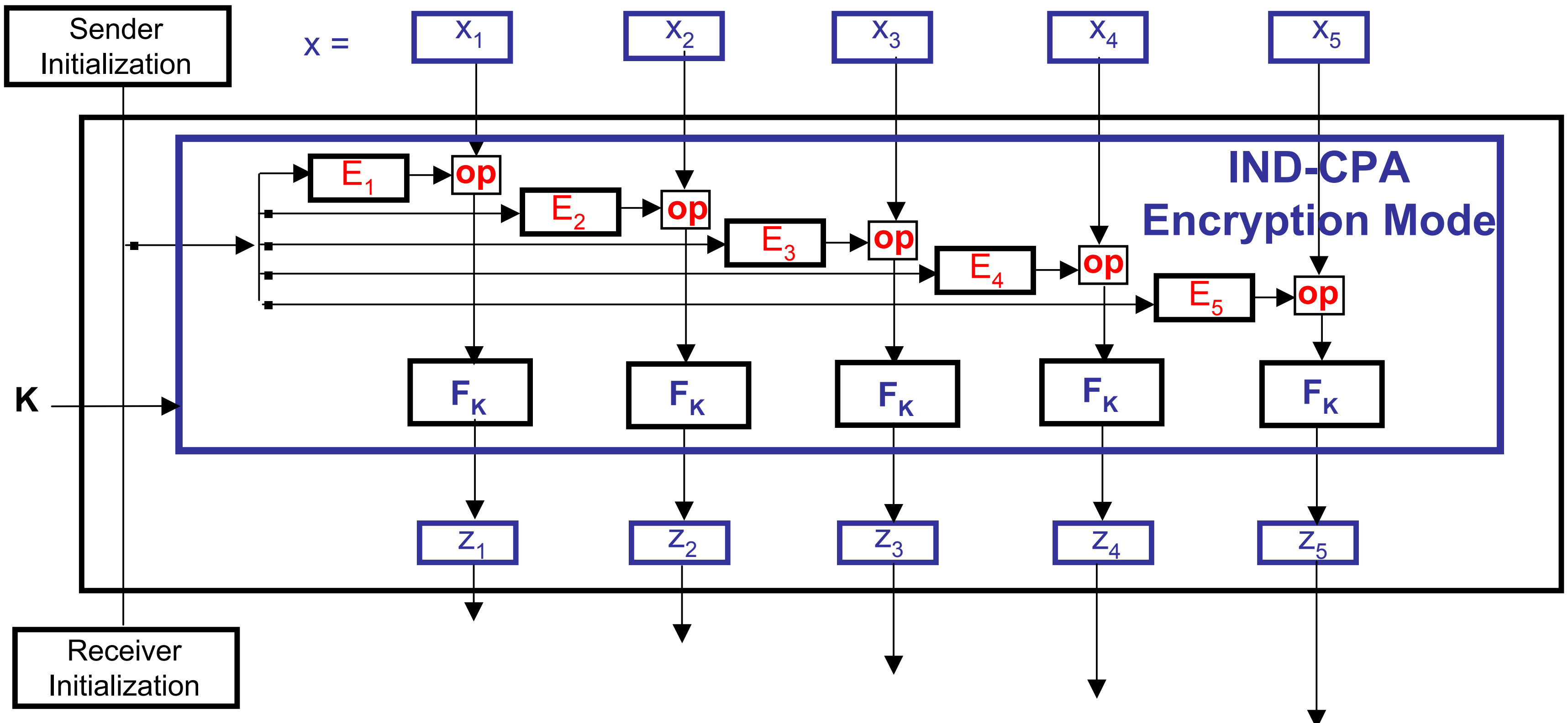
op = xor **; $E_i = r_0 \times i + r_1$ mod *p* (pairwise indep.)** [Jutla00]

… and others [Rogaway01]

***Optimal AE*: n+1 cipher ops; latency in || mode: 1 cipher op.**

Parallel AE in 1 pass - 1 crypto primitive

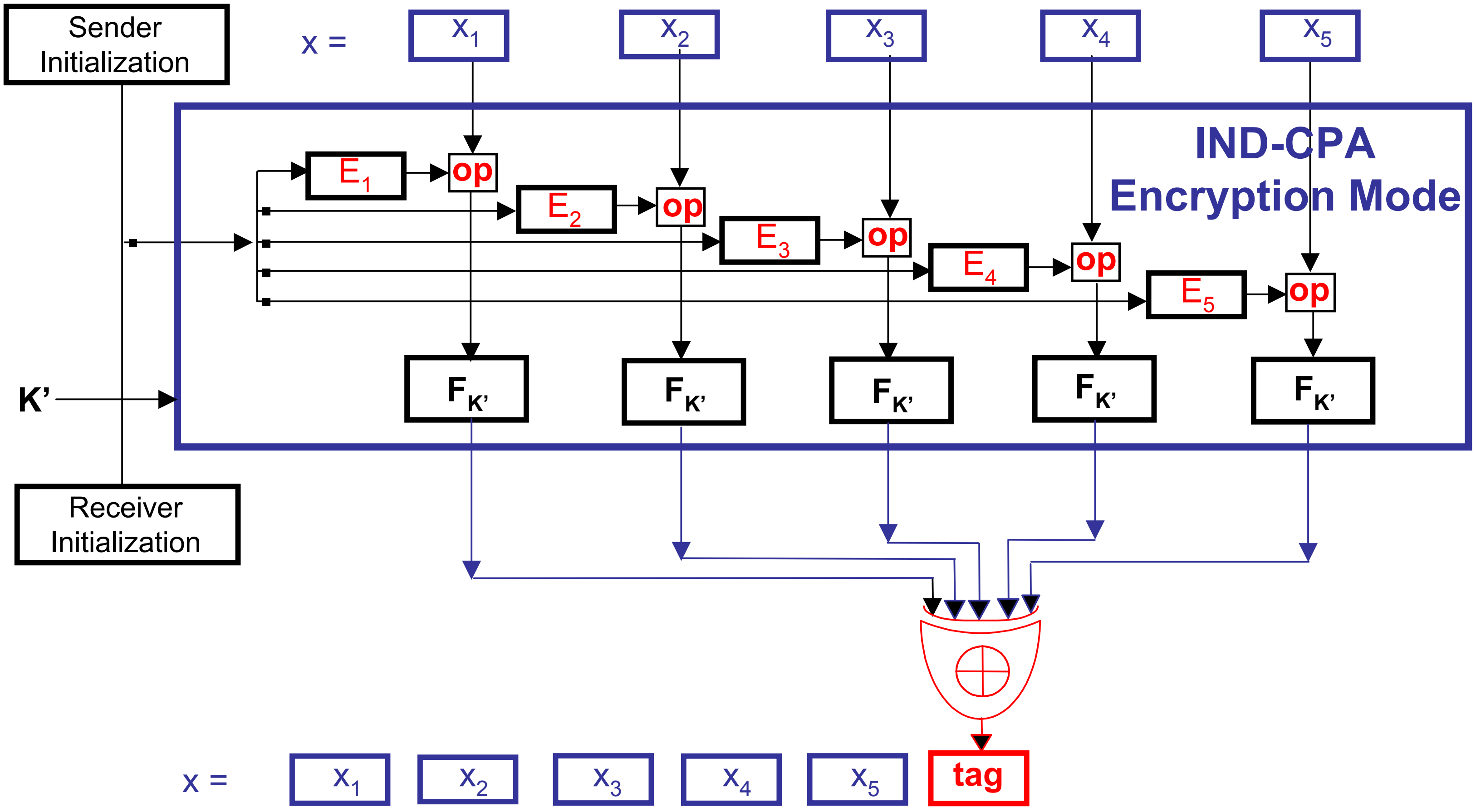
Same hardware for input (viz., IAPM [Jutla00], XECB-XOR [GD00])



... can lead to an IND-CPA encryption mode, further minimize hardware footprint, and also provide ...

Parallel MAC

… a (parallel) MAC w/ an extra XOR gate (viz., [G98, GD00])



Design of AE in 1 pass -1 crypto primitive: a *very* Dangerous Exercise ...

1. Clark Weissman: use *CBC* with *MDC = Cyclic Redundancy Code (CRC)*
   - proposed at 1977 DES Conference at NBS
   - stronger scheme broken by S. Stubblebine and V. Gligor (IEEE Security and Privacy 1992)

2. Carl Campbell: use *Infinite Garble Extension* (IGE) mode with *MDC = constant appended to message*
   - proposed at 1977 DES Conference at NBS
   - IGE was reinvented *at least* three times since 1977
   - broken by V. Gligor and P. Donescu 1999

3. V. Gligor and B. Lindsay: use *CBC* with *MDC = any redundancy code*
   - Object Migration and Authentication, IEEE TSE Nov, 1979 (and IBM Research Report 1978)
   - instances of it were known to be broken by 1981 (see below)

4. US Dept. of Commerce, NBS Proposed Standard (1981): use *CBC* with *MDC = XOR*
   - withdrawn in 1981; see example of integrity breaks above

Design of AE in 1 pass -1 crypto primitive: a *very* Dangerous Exercise ...

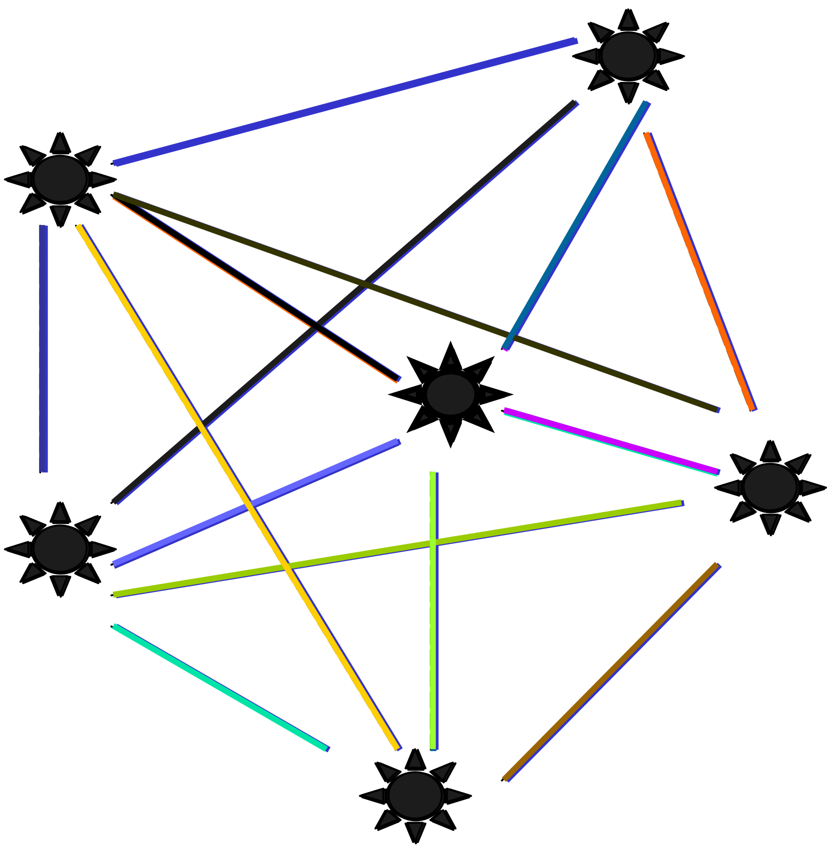
5. MIT Kerberos v.4: use *PCBC* with *MDC = constant appended to last block*
- proposed at 1987 - 1989
  - broken by J. Kohl at CRYPTO '89

6. MIT Kerberos v.5 - confounder (i.e., unpredictable block) prepended to message data
  - CRC-32 is computed over the counfounded data and inserted into message before encryption
  - proposed in 1991 Kerberos v.5 specs. (used within US DoD ?)
  - broken by S. Stubblebine and V. Gligor (IEEE Security and Privacy 1992)

7. V. Gligor and P. Donescu: use *iaPCBC* with *MDC = unpredictable constant appended as the last block of message (not the XOR version)*
  - proposed at the 1999 Security Protocols Workshop, Cambridge, UK.
  - actually the proposal had MDC = XOR
  - broken by the "twofish gang" (D, Whiting, D. Wagner, N. Ferguson, J.Kelsey); and by C. Jutla

8. US DoD, NSA: Use *Dual Counter Mode* with *MDC = XOR*
  - proposed August 1, 2001 and withdrawn August 9, 2001
  - broken by P. Donescu, V.D. Gligor, D. Wagner, and independently by P. Rogaway

Key-Distribution Schemes



- Impractical schemes
  - Key Distribution Center (KDC)
  - Unique mission key
  - Pair-wise key sharing
  - Public-Key schemes
- New key distribution schemes
  - Basic Scheme and Extensions
  - Random Pair-wise Scheme
  - Multiple Key Spaces

Impractical Key-Distribution Schemes

- Key exchange/distribution based on a trusted KDC is *impractical* :

  not scalable - large communication (multi-hop) overhead
  - contention at nodes closest to KDC
  - multiple KDC and/or communication paths to KDCs may be necessary

  KDC becomes attractive attack target

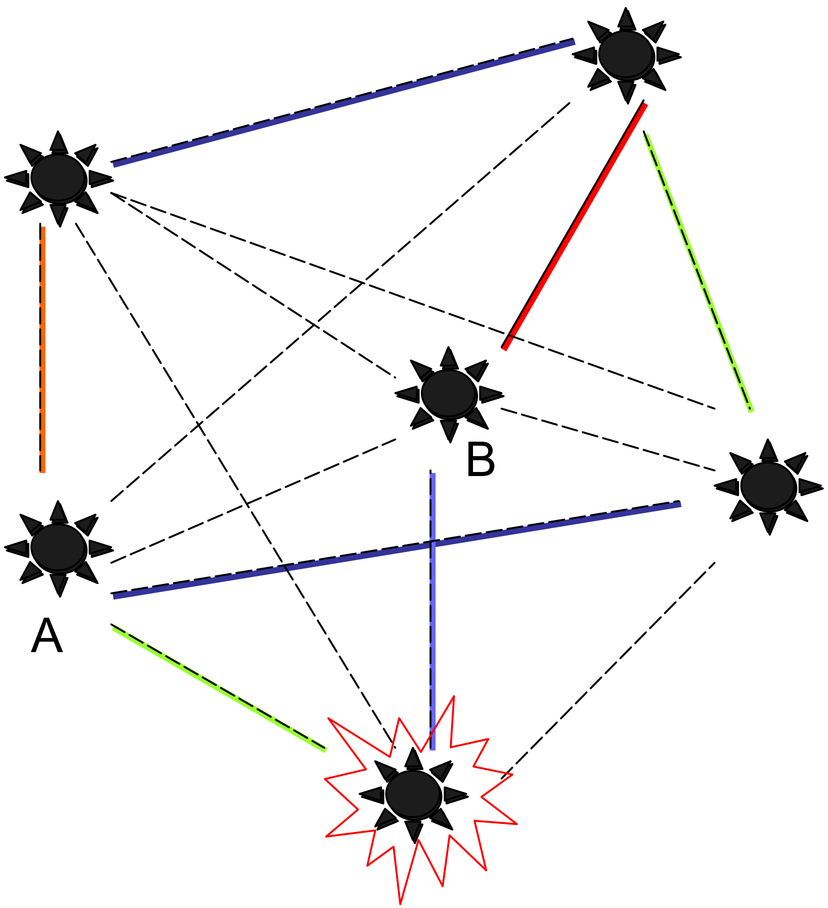
- Key *pre-distribution* is only practical solution (to date)… However,

  single mission key: same key for all communication links
  - capture of *any* sensor node may compromise the entire SN
  - erasure of mission key after *link*-keys setup => no incremental node addition

  pair-wise: storage of n-1 keys in each sensor node
  - not scalable: memory cost unrealistic at current state of technology
  - incremental addition and re-keying complex and expensive
  - full-connectivity is *not* usable/required for SN

  public-key schemes: used sparingly (i.e., only for symmetric key distribution)
  - vulnerability to DoS Attacks
  - extra hardware; not resilient to node-insertion attacks

Key (Pre)distribution - Basic Scheme [EG02]

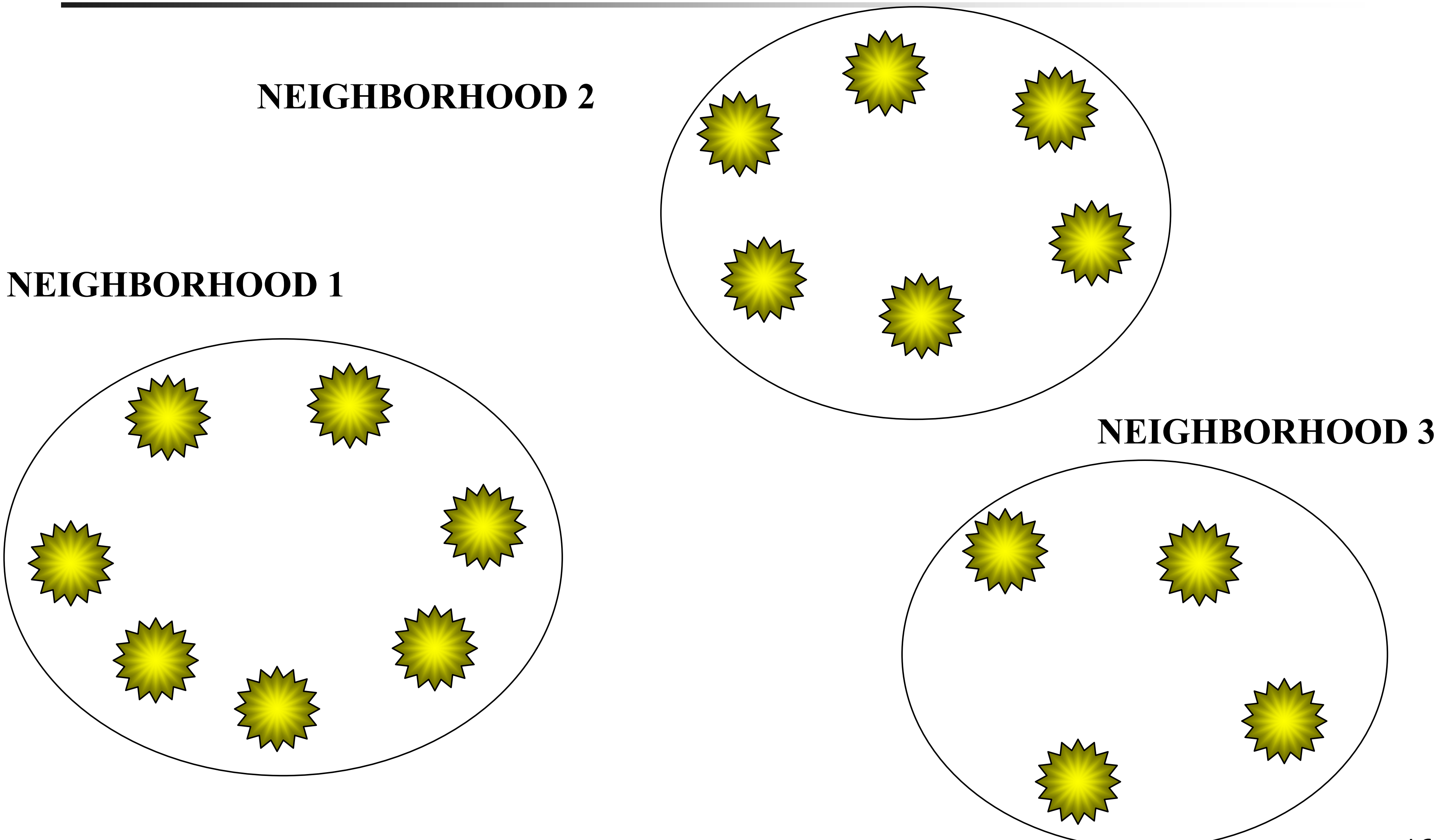


- Probabilistic Key (Pre)Distribution
  - key pre-distribution
    - generation of a *large pool* of *P* keys
    - random drawing of *k* keys out of *P* w/o replacement
    - loading of the *key ring* into each sensor
  - shared-key discovery
    - upon initialization every node discovers its neighbors with which it shares keys
  - path-key establishment (- - -)
    - assigns a *path-key* to neighbors w/o shared key
    - multiple disjoint paths exist between two nodes
      - example (A,B)

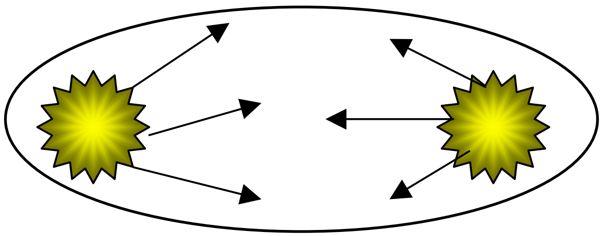
- Consequences
  - **node-to-node authentication ?**
  - **key revocation ? scope ?**
  - **node-capture detection ?**
  - **resilience to node capture ? insertion ?**
  - **network extension**

node capture

Basic Scheme



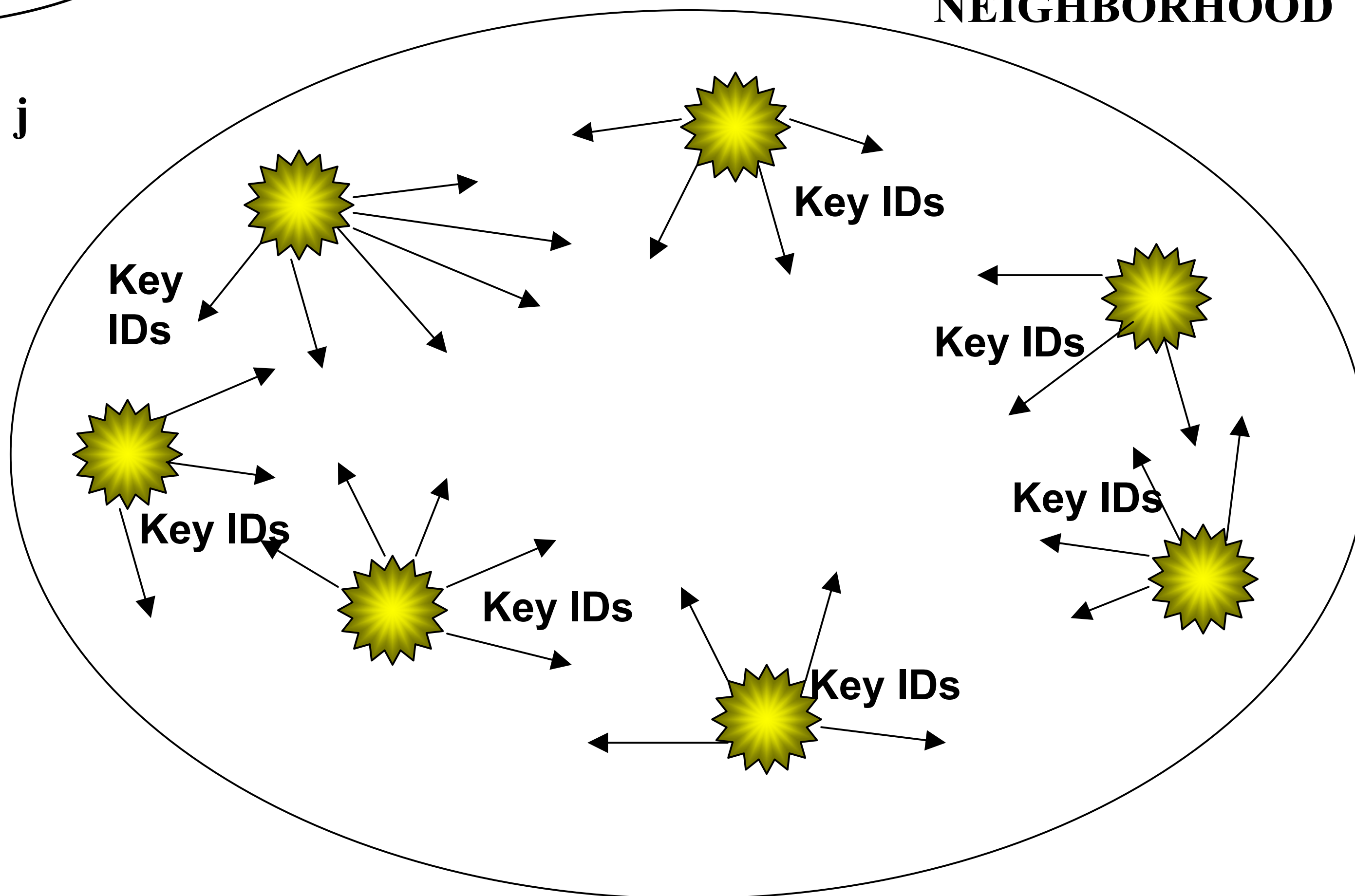
Basic Scheme: Shared-Key Discovery



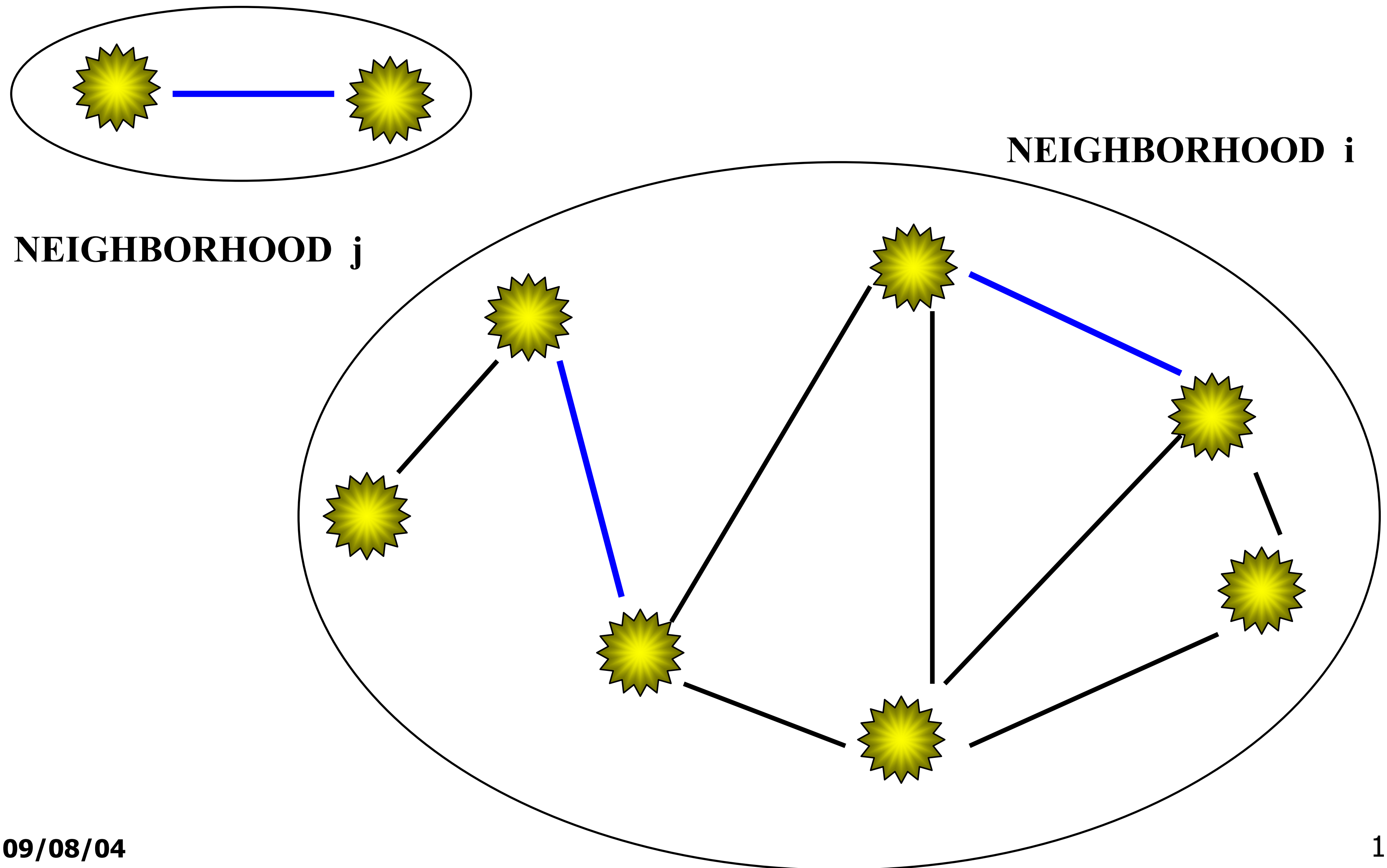
Each node sends out Key IDs (or < a, E(K_i, a)> list) and each node discovers its neighbors

NEIGHBORHOOD i

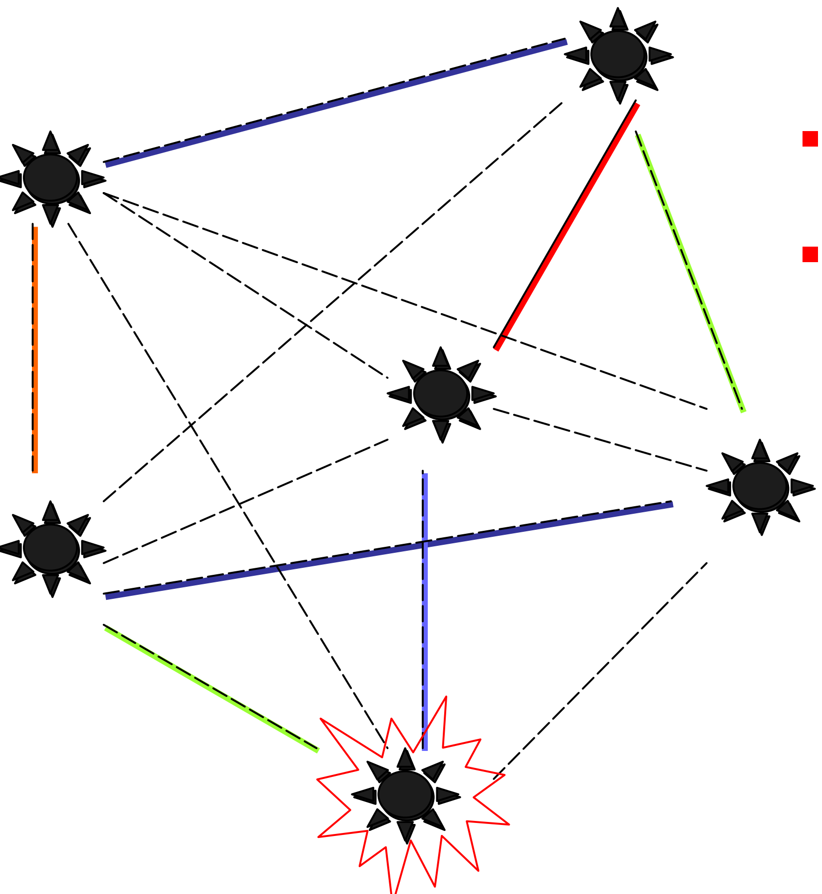
NEIGHBORHOOD j



Shared-Key Discovery (ctnd.)



Basic Scheme - Analysis



Probabilistic key sharing

Questions

- **Q1**: Given ***n*** nodes, find what ***k***, ***P*** should be such that the SN is connected ?
- **Q2**: Given wireless connectivity constraints (no. of nodes ***n′*** << ***n*** in a neighborhood, direct link connectivity), find ***k***, ***P*** ?

node capture

Analysis

- **Q1**: Given ***n***, find ***k***, ***P*** such that the SN is connected ?

- *Suppose* the SN is a Random Graph *G(**n**,**p**)*
  n = no. of nodes, ***p*** = probability of a link *i~j*
  - Erdös – Rényi (1960)
    - if $p = \frac{\ln n}{n} + \frac{c}{n}$
    - with *c* any real constant, then

$$\lim_{n \to \infty} \Pr[G(n,p) connected] = e^{-e^{-c}}$$

- **Example:**
  - Given ***n*** = 10,000 and *desired* **Pr[*G(n,p) connected*]** = .99999,
    find *c* = 11.5, ***p*** = 2* 10^{-3} and ***d*** = 2* 10^{-3} * 9999 ≈20

Analysis *(cont.)*

- **Probabilistic key sharing with constraints**
  - **Q2**: Given wireless connectivity constraints, find ***k, P***
    - communication range *limits neighborhood* to $n' << n$
    - *choose* $d' \geq d$ direct (*one-hop*) links in the neighborhood
  - p' = Pr[link i~j] = Pr[at least one key shared between nodes i and j]

$$p' = \frac{d'}{n'} >> p$$

- **Example ctnd.**
  - $d' = d = 20$
  - Let n' = 40, p' = 20/(40-1) . P = 100,000 => k = 250.
  - Let n' = 60, p' = 20/(60-1) . P = 100,000 => k = 200.

- **Tradeoff**

  $d' > d$ (lower energy consumption) vs. k (more memory) | n', P

Example - Summary

- Parameters
  - n = 10,000
  - neighborhood connectivity constraints n' = 40 nodes, d' = 20
  - Pr[Graph is connected] chosen to be 0.99999
- Analysis
  - $\boldsymbol{c} = 11.5 \Rightarrow \boldsymbol{p} = (\ln(10{,}000)+11.5)/10{,}000 = 2*10^{-3}$, $\boldsymbol{d} = p*(n-1) = 20$
  - constraints: $\boldsymbol{d'} = \boldsymbol{d}$, $\boldsymbol{n'}$ = 40 **=>** $\boldsymbol{p'} = d/(n'-1) = 0.5$
  - if pool size $\boldsymbol{P}$ = 100,000 keys
    - each node needs to have a key ring of size $\boldsymbol{k}$*=250 keys*
    - **64-bit keys** => **2KB** memory (**80 KB** for *pair-wise* scheme)

A Consequence of Basic Scheme

- *Source* Authentication => *all* nodes are trusted
  - $K_{i,j}$ = *hash* (k_{ij} || ID_i || ID_j), where $ID_i > ID_j$, is "unique"
- Node-Capture Detection
  - redundant sensor coverage; data cross-correlation ?
  - grand challenge problem
- Centralized Revocation (=/=> node-to-node authentication)
  - A controller node broadcasts signed list of *k* key *identifiers* to be revoked
    - disables all connectivity of the compromised node
    - affects other nodes on a small part of their key ring

All-trusted-node assumption for Source Authentication =>

Node-Capture Detection + Revocation

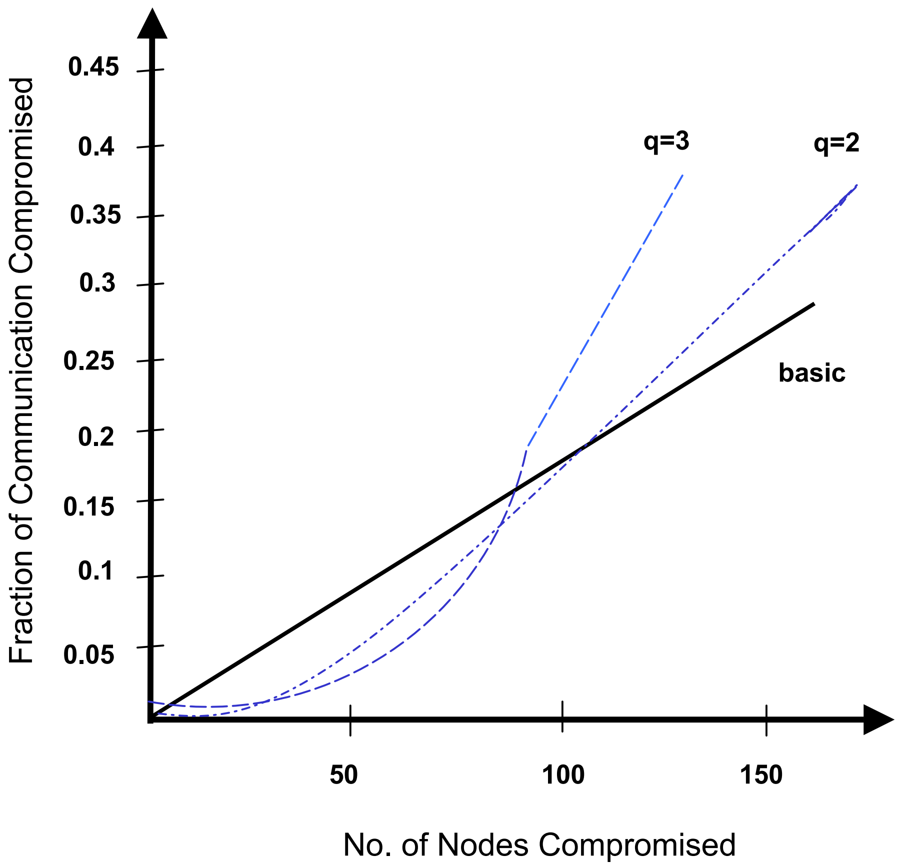
- Resilience (w/o node shielding)
  - Capture of a key ring affects links ***k*** * *no. links*/***P*** links

Extensions of Basic Scheme

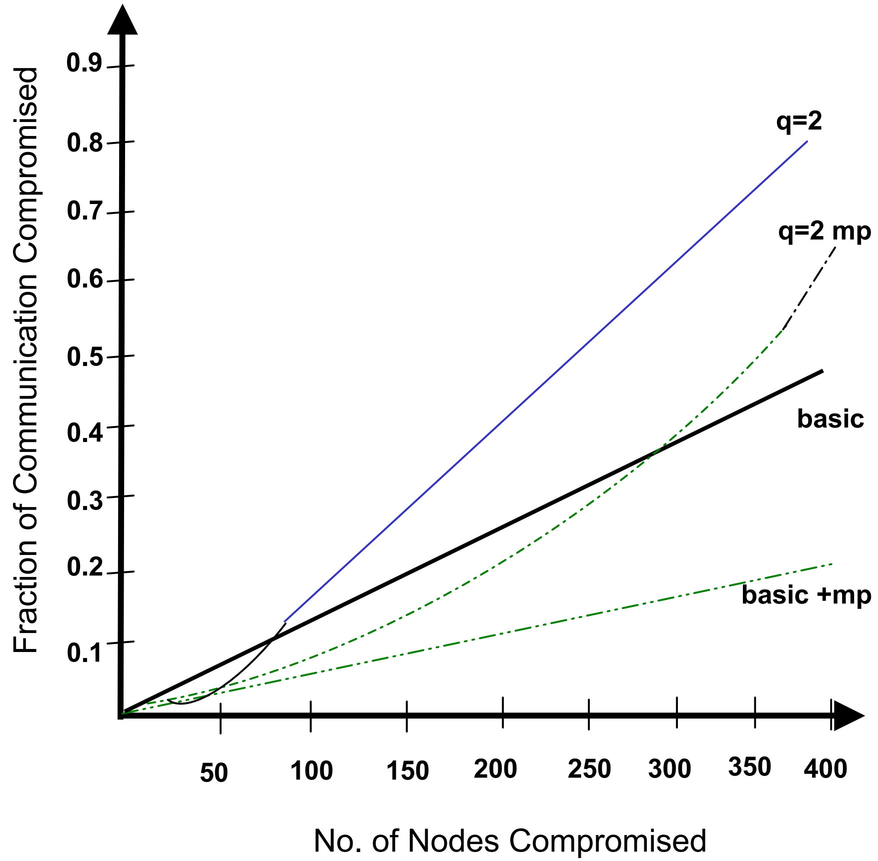
- *q-composite key* extension of Basic Scheme [CPS03]
- MOTIVATION
  - Improve Resilience to Node Capture
    - fraction of compromised communication; network size
    - multipath key reinforcement
  - Node-to-Node (not Source) Authentication
    - nodes need *not* trust each other
- IDEA
  - decrease pool size P s.t. $\geq q$ keys are shared between any two nodes
    - $K_{i,j} = \text{hash}\,(k^1_{ij} \,||\, k^2_{ij} \,||\ldots||\, k^q_{ij})$ is "unique"
  - j disjoint node paths between A and B; $v_1 \ldots v_j$ path keys
    - $K_{A,B} = v_1$ *xor* …. *xor* v_j
  - less vulnerable to node capture than Basic Scheme up to *threshold*, more after

Extensions of Basic Scheme (ctnd.)

q-composite key



+ multipath key



k = 200, p = 0.33

Random Pairwise Key Pre-distribution [CPS03]

- MOTIVATION
  - Node-to-node authentication
  - Resilience to capture and resilience to replication (*without* node shielding)
  - Distributed Revocation
  - Resistance to node replication
  - Comparable network size ?
- IDEA
  - For every *possible node* (ID), pick **k** random neighbors (IDs)
  - Generate **k** pair-wise shared keys
  - Scatter nodes and discover neighbors; *multi-hop extension*
  - Distributed revocation via *threshold voting* scheme.

    vote authentication (e.g., session, source, replication detection, count integrity)
    - policy (e.g., session start/end times, revocation quotas)
  - Replication detection: limit **d** for every node, integrity of *neighbor counts*

Multiple Key Spaces - Motivation

- Single Key-Space Schemes for *Group Keying* [Blundo *et al.* '91]
  - random bivariate *t*-degree polynomial over finite field F_q, q= prime, |q| ≈ key length,

$$f(x,y) = \sum_{i,j=0}^{t} a_{ij}x^i y^j, \text{ with property } f(x,y) = f(y,x).$$

  - for each sensor i, pre-distribute polynomial share f(i,y) in *(t+1)logq* space;
  - sensors *i* and *j* compute shared key k_{ij} = f(i,j);
    - sensor *i* evaluates f(i,y) at point *j*, and sensor *j* evaluates f(j,y) at point *i*;
  - unconditionally secure but *resiliency limited to a threshold of t captured nodes*
  - limited scalability for SN
    - storage cost per node is exponential in group size
    - computation intensive for |q| = 64 bits in 8-bit processors (e.g., ATMEL Atmega 128) even for relatively *small t*
      - 27 - 64 multiplication operations per two 64-bit integers
      - 16 multiplication operations for 64 bit x 16 bit integers

- Other similar ideas for *Group Keying* exist [Blom '84]

- Multiple Key Spaces: improve scalability and resiliency by combining Probabilistic Approach of Basic Scheme with Group Keying Schemes [LN03, DDHV03]

Multiple Key Spaces - Example [LN03]

1. **Set-up**

   a) Generate Pool ***F*** of Random, bivariate, *t*-degree polynomials (with given property) over finite field F_q, where q is a prime. Each polynomial has a unique ID.

   b) For each sensor node *i*, pick a subset of polynomials $F_i \subseteq \boldsymbol{F}$ at random and install the polynomial shares in node *i*.

2. **Shared-key discovery**

   broadcast list of polynomial IDs to neighbors; or broadcast <a, E_{Kv}>, v =1,…,| F_i | and Kv is a potential key neighbor nodes may have

3. **Path-key discovery**

   a) source node broadcasts two lists of polynomial IDs
      - lists of polynomial IDs of the source and destination nodes

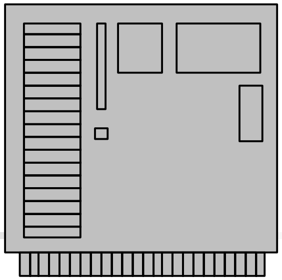
   b) if intermediate recipient finds ID matches with source and destination nodes, it
      - broadcasts two encrypted copies of newly generated path-key each encrypted with shared key of intermediary and source/dest.

   c) repeat the process among intermediaries until a path is found within a certain range.

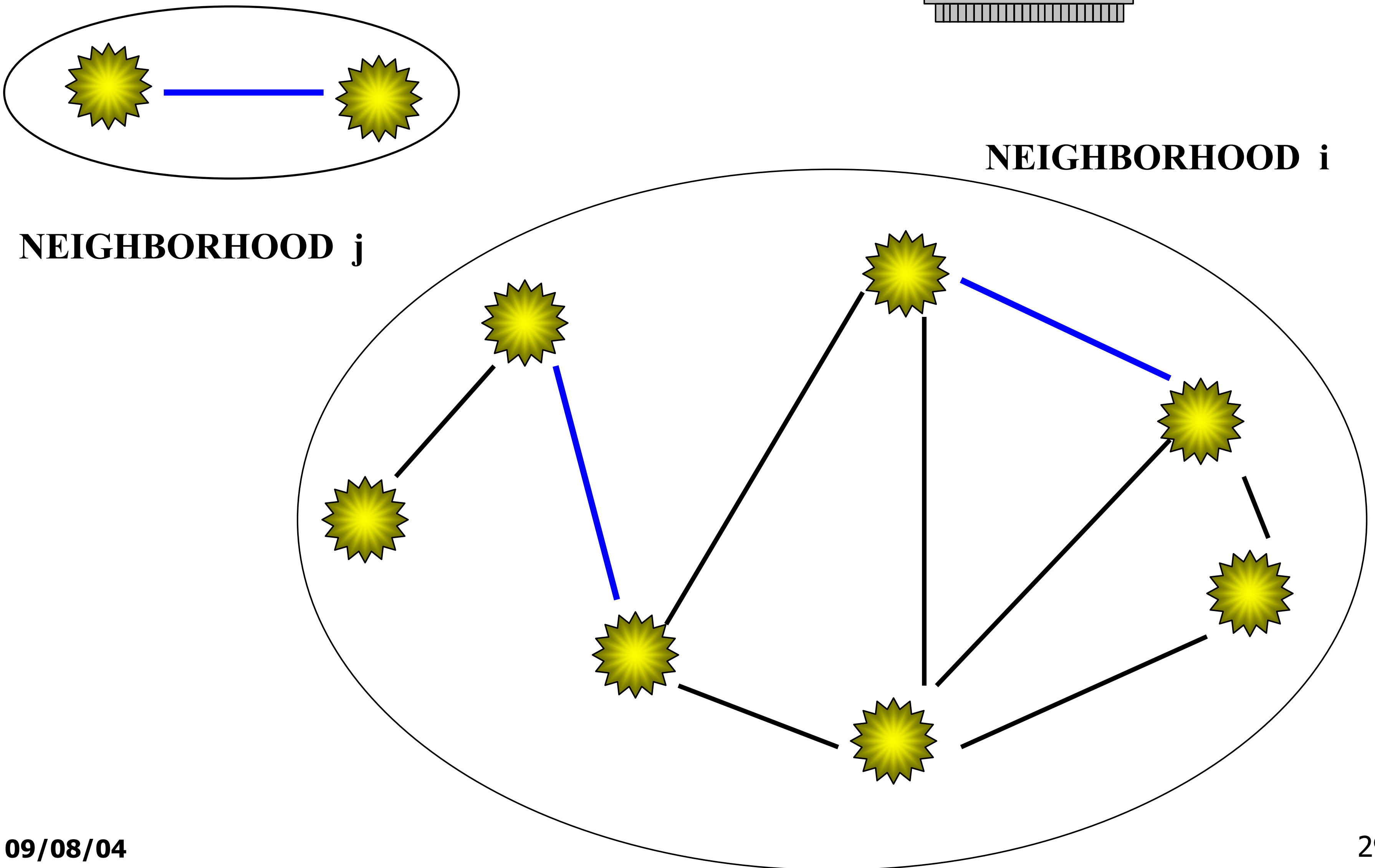
Generalization: *t*=0 => *Basic Scheme*; |***F***|=1 => Single Key-Space for *Group Keying* [Blundo'91]

Other multiple-key-space schemes have been proposed [DDHV03] based on [Blom'84]

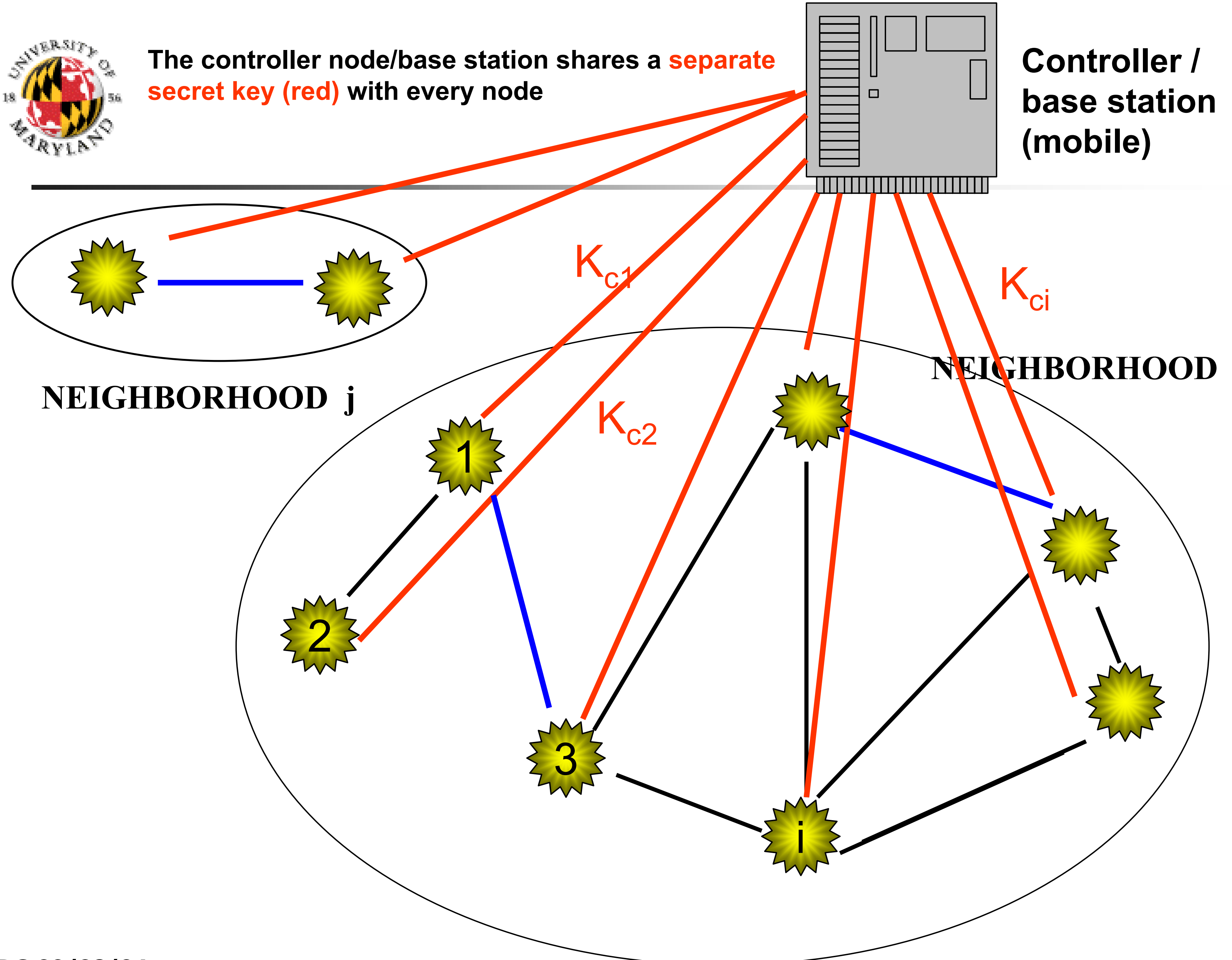
Basic Scheme: Centralized Revocation



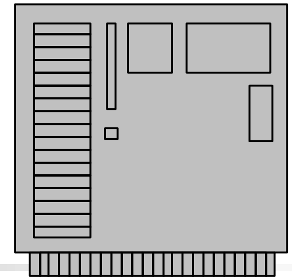
Controller / base station (mobile)

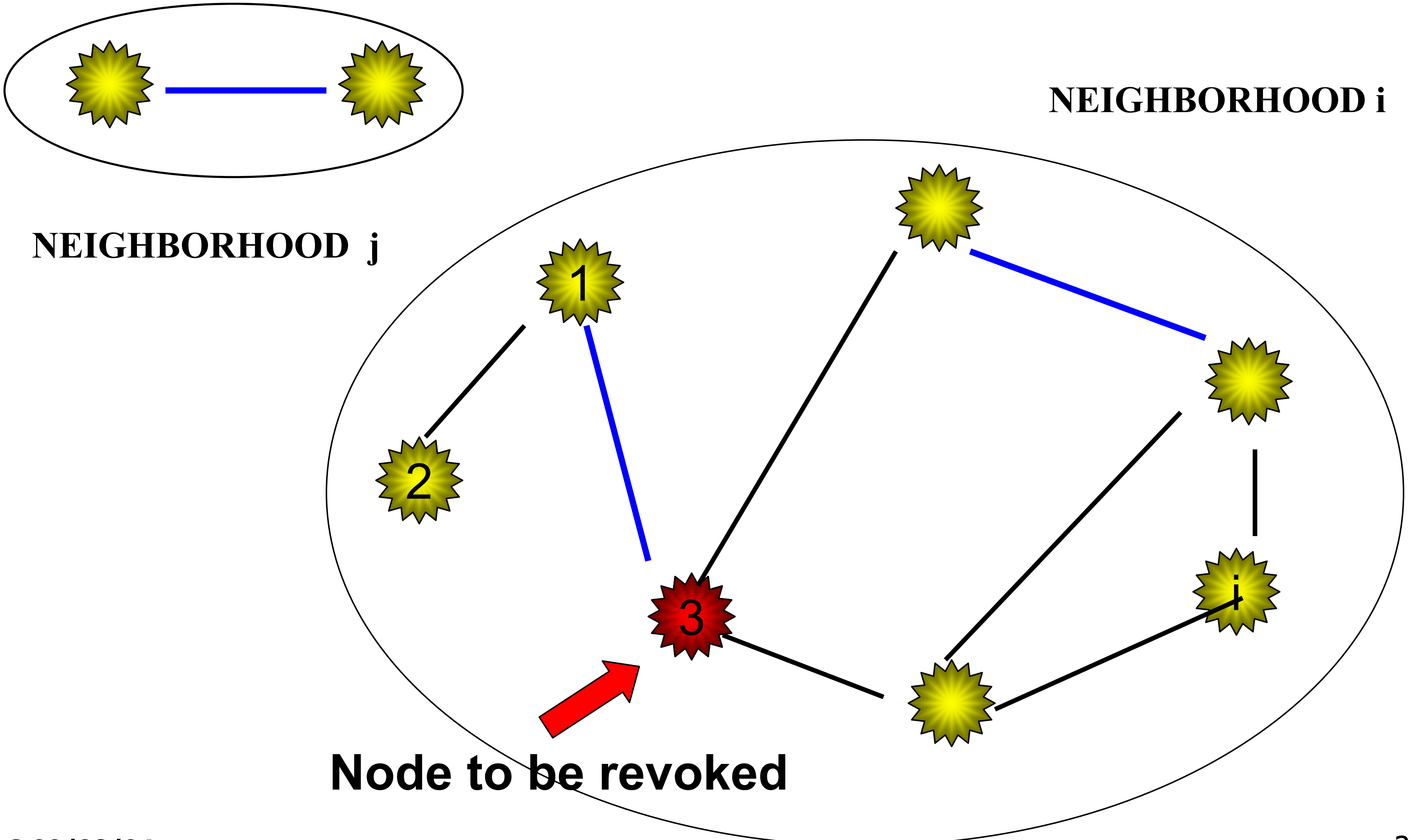


The controller node/base station shares a separate secret key (red) with every node

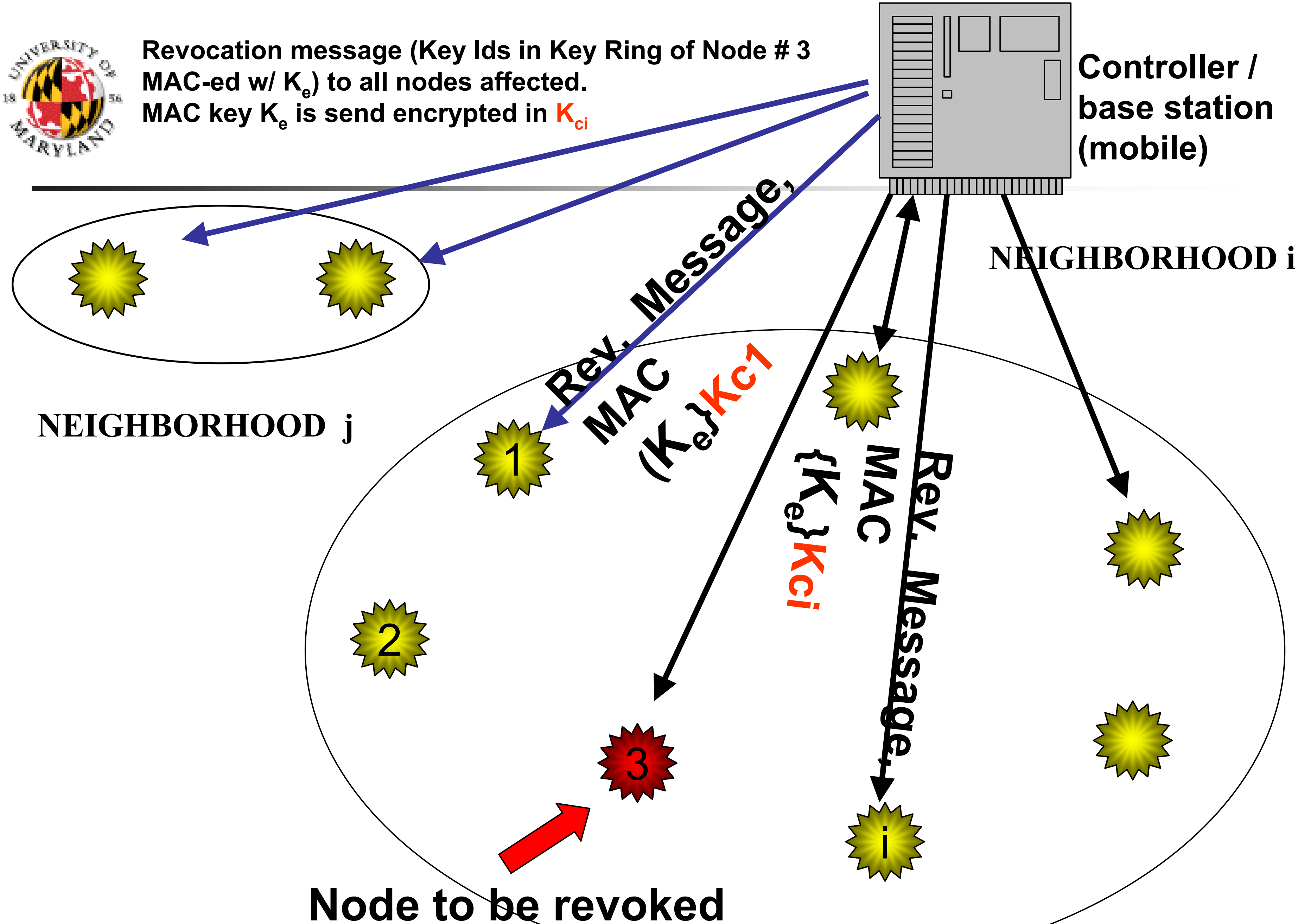


Revocation Scope

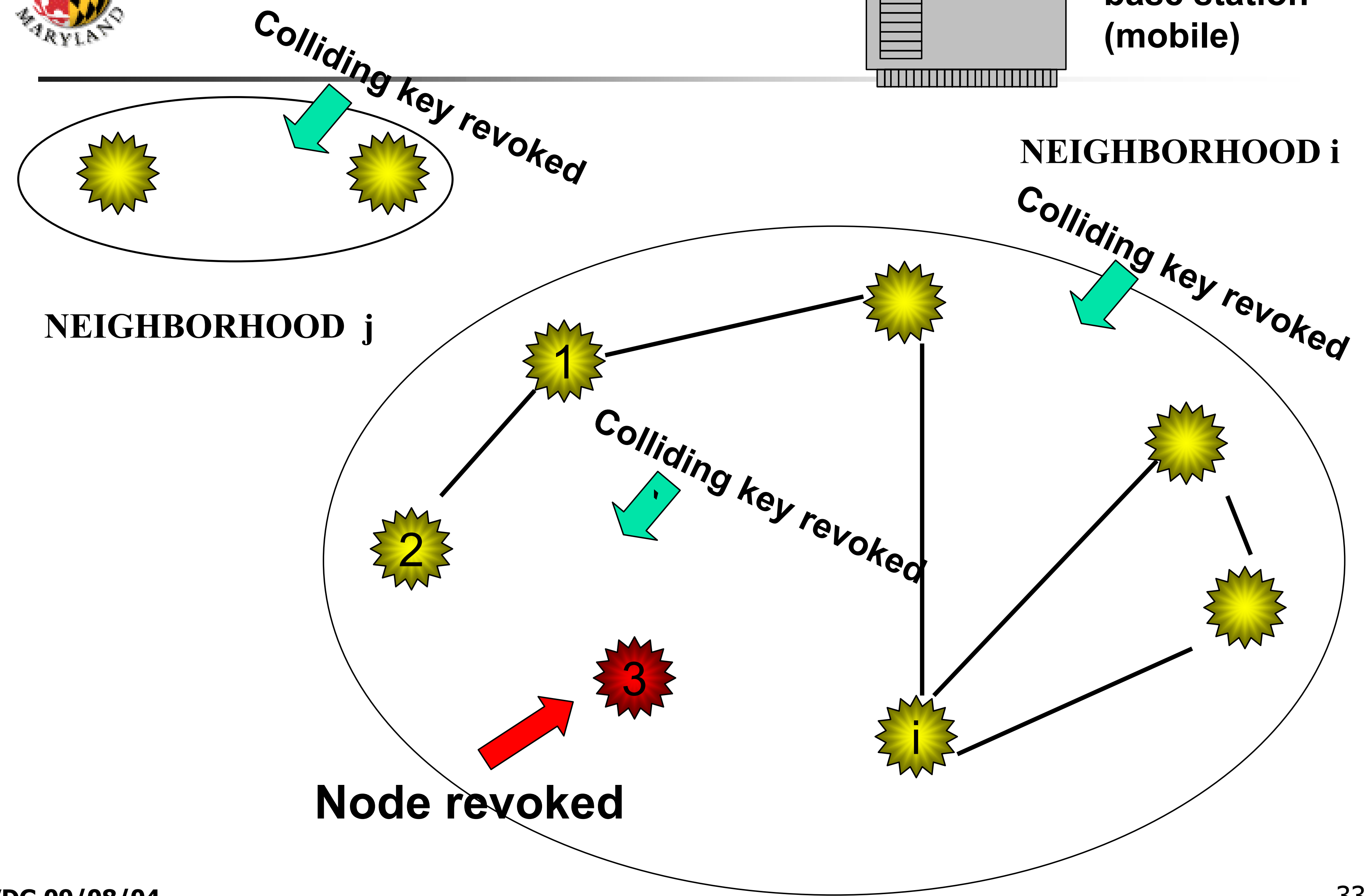




Revocation message (Key Ids in Key Ring of Node # 3 MAC-ed w/ K_e) to all nodes affected.
MAC key K_e is send encrypted in K_{ci}



Nodes delete revoked keys (Key IDs)



Summary: Centralized Revocation

Advantages

Revocation policy is *uniformly enforced* and *non-circumventable* *(e.g., adversary cannot execute the rev. protocol)*

***Node-to-node* message authenticity *not* required**

Minimal memory size (e.g., for multi-node revocation)

Disadvantages

Slow (e,g., slower than distributed revocation)

Controller needs global network reach

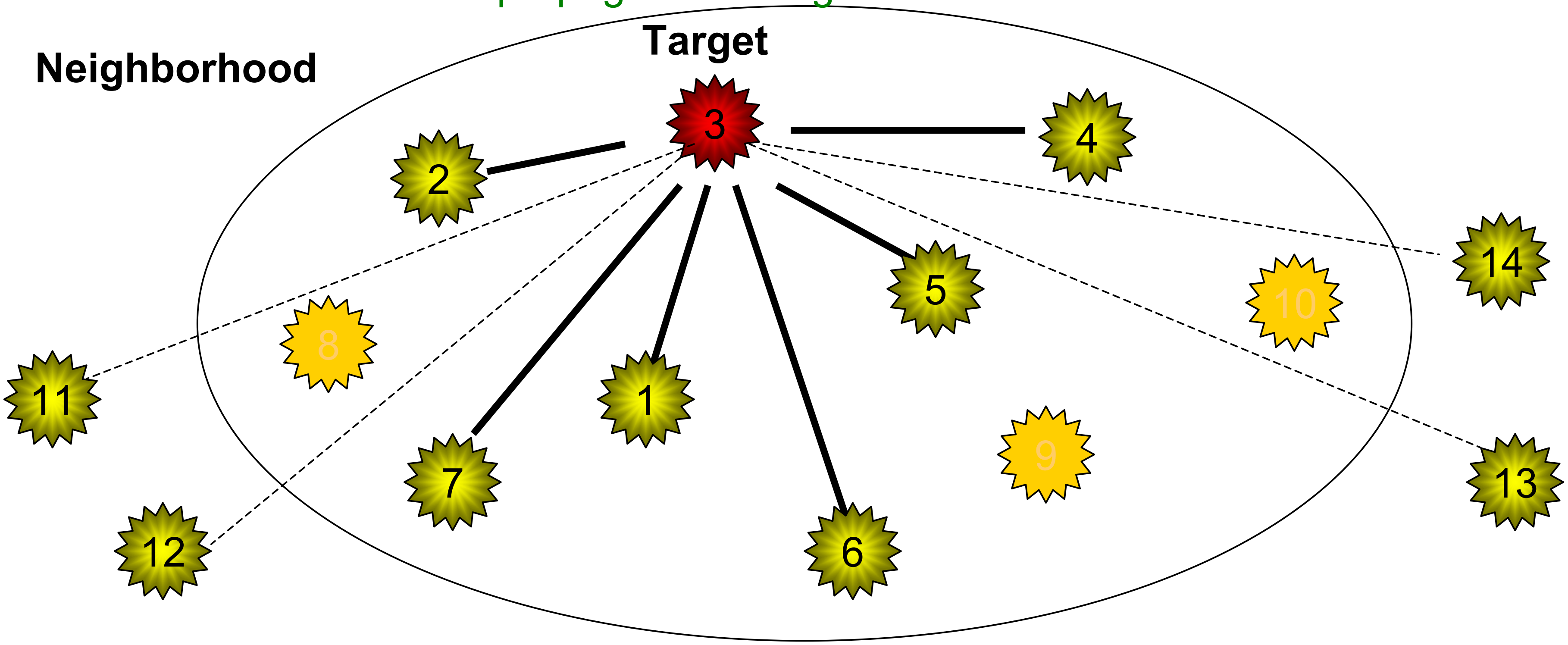
Single point of failure

Distributed Revocation [CMGP04]

Policy:

1. Local neighbors of a revocation target make the revocation decision
   - threshold-based decision [CPS03]
     - ***t*** votes to revoke (***t*** > node degree, ***d***) => delete keys shared with target

2. Revocation decision is propagated throughout the network



Distributed Revocation

Advantages

- Faster than centralized scheme
- Only inexpensive neighborhood comm. required
- No single point of failure

Disadvantages

- Need for *Vote* (not just node-tonode message) *Authenticity*
- More complex (e.g., adversary may be a protocol participant)
- Revocation Policy Agreements

Adversary Goals

1. Capture sensor nodes that collude to subvert revocation policy

Examples:

- block the decision by exhausting resources of legitimate neighbors
  - exhaust votes, revocation sessions by casting forged votes
- refuse to carry out protocol steps

2. Capture enough neighbors and revoke uncompromised nodes

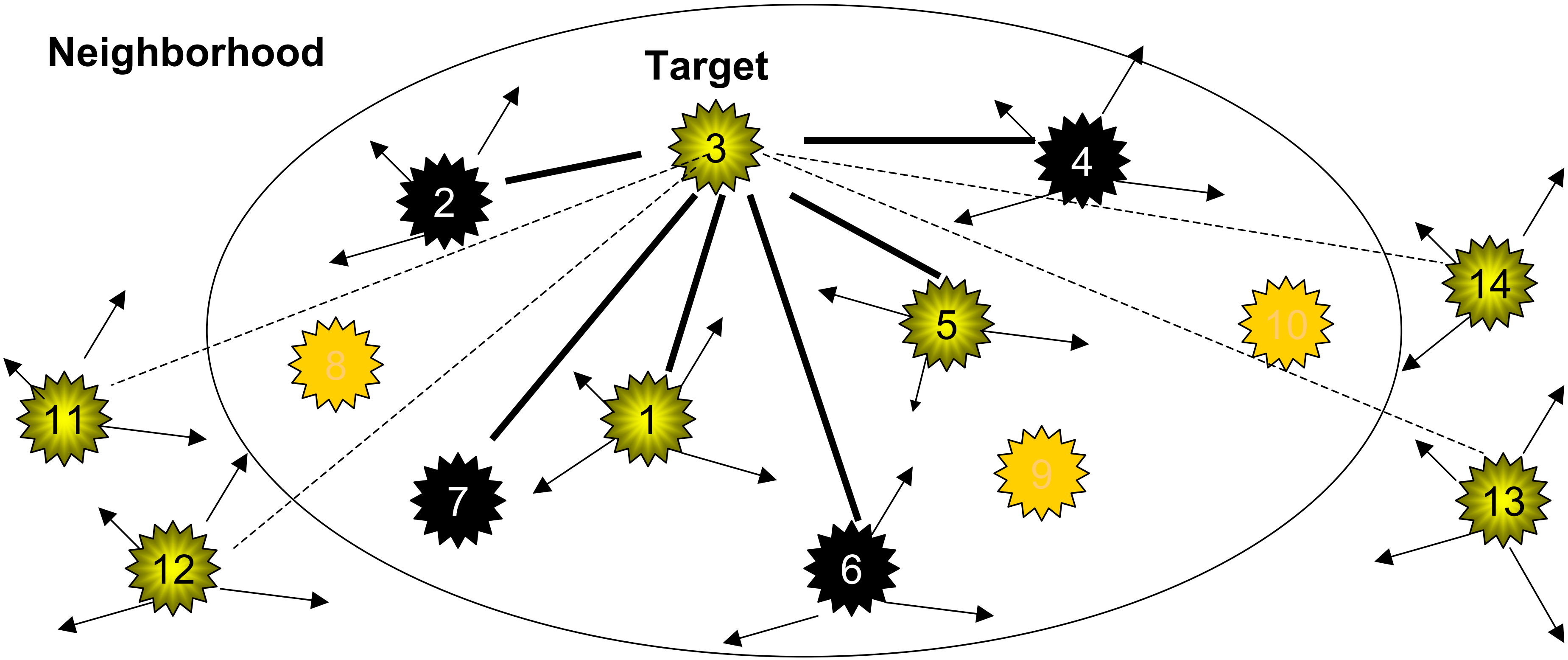
=> ***emergent property: secure communication paths disappear***

Note: Goals are Different from those of a Byzantine Adversary

- reach - not prevent - consensus on (albeit, malicious) revocation
- different bounds for revocation consensus (i.e., ***t*** vs. ***2d/3*** legitimate nodes)

Node Revocation by an Adversary

Example: t = 4, nodes 2,4,6,7 are compromised



Distributed Revocation - Protocol Properties

A. Correctness

1. Complete Revocation

If a *compromised node* is detected by ***t*** or more *uncompromised neighbors*, then the node is revoked from the entire network permanently

2. Sound Revocation

If a node is revoked from the network, then at least ***t*** *nodes* must have agreed on its revocation

3. Bounded-Time Revocation Completion

Revocation decision and execution occur within a *bounded time* from the sending of the *first revocation vote*

4. Unitary Revocation

Revocations of nodes are *unitary* (all-or-nothing, everywhere-or-nowhere) in the network

B. Security of Emergent Property

1. Resistance to Revocation Attack

If ***c*** nodes are compromised, then they can only revoke at most ***ac*** other nodes, where $a \ll m/t$ is a constant and ***m*** is the maximum number of neighbors (at key distribution)

2. Revocation Attack Detection

Revocation attacks are detected *centrally* by a base station in bounded time

Adversary Model and Protocol Assumptions

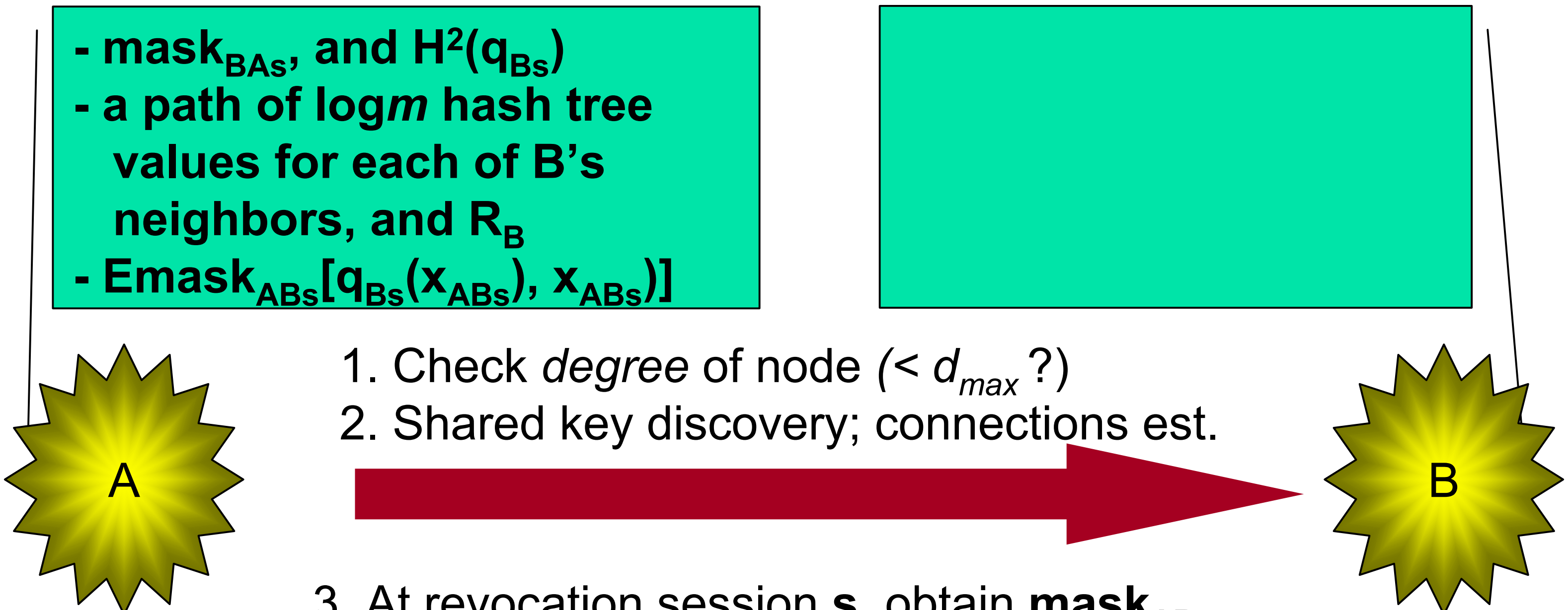
A. *Adversary Model*

1. Universal Communication Presence
2. Can Compromise *any* Node it Chooses
3. Can Force Collaboration among Compromised Nodes
4. *Cannot* block or significantly *delay* communication

B. *Protocol Assumptions*

1. Network is quiescent during deployment of new nodes
2. Locality of Compromised Nodes
3. Minimum Node Degree > *t*
4. Revocation Sessions are Relatively Rare and Cannot be Exhausted

Distributed Node Revocation: Protocol Summary



3. At revocation session **s**, obtain $\mathbf{mask_{ABs}}$
4. Unmask *Vote*= ($\mathbf{q_{Bs}(x_{ABs}), x_{ABs}}$) with $\mathbf{mask_{ABs}}$ key
5. A casts *Vote* against B
6. All B's neighbors check validity of the ***t*** cast votes in session s using the stored **hash tree values vs.** $\mathbf{R_B}$
7. All B's compute the revocation polynomial, and $\mathbf{H(q_{Bs})}$ and broadcast $\mathbf{H(q_{Bs})}$ in the network.
8. Each of B's **m** neighbors check $\mathbf{hash(H(q_{Bs})) = H^2(q_{Bs})}$ and revoke keys shared with B

Research Areas

- Resilience to node capture
  - good engineering: limited node shielding => fast key erasure
- Node-Capture Detection
  - complexity mitigated by limited node shielding
- Distributed Revocation
  - Needs robust, distributed consensus. Revocation control ?
  - Needs Policies: when do we really want to revoke the keys of a node ?
- Non-Random Scattering of Sensors ?
  - optimizations ? new basis for deployment ?
- Evaluation of Key Distribution and Revocation Schemes (2003 ->)
  - Tradeoffs ?

    e.g., communication/computation (e.g., energy) *vs*. storage size *vs*. network size *vs*. resilience

Trsust Establishment in MANETs

- Security in Mobile Ad-Hoc Networks (MANETs)
- Trust Establishment in MANETs
  - Three scenarios
- Research areas

Mobile Ad-Hoc Networks (MANETs)

Ad-hoc = > no designated infrastructure prior to deployment

- no predetermined access points or topology, no allocation of nodes to administrative services
  - no dedicated router nodes, name servers, certification authorities, etc.
- no distinction between trusted and untrusted nodes
  - no physical and administrative protection of trusted nodes
  - nodes are subject to capture

- Mobile => topology changes dynamically

- Wireless => connectivity among nodes is not guaranteed
  - broadcast to one-hop neighbors is inexpensive
  - limited power and energy traded-off for connectivity

…. are very different from Mobile IP v6

Example of Trust Relations

- *Trust:* a *relation* among entities (e.g., domains, principals, components)
  - established by evidence evaluation using specified metrics, and
  - required by
    - *specified policies* (e.g., by administrative procedures, business practice, law)
    - *specified design goals* (e.g., composition correctness via use of layering, abstraction)

Example: An Authentication-Trust Relation

"A *accepts* CA_B's *signature* on X's *PK certificate*"

Basis for *A's acceptance of* CA_B's *signature :* off-line *evaluation* of *evidence*

- CA_B's authentication of X is done using "*acceptable*" mechanisms and policies (i.e., A *trusts*^{AU} CA_B)
- CA_B's registration database (including X's registration) is protected using "*acceptable*" mechanisms and policies (i.e., A *trusts the Registration DBMS*)
- CA_B's server is managed using "*acceptable*" administrative, physical and personnel policies (i.e., A *trusts* CA_B's administrators)

What Do We Mean By Trust Establishment?

Trust establishment (in general):

- *application of an* ***evaluation metric*** *to a body of* ***evidence****,*
- ***on-*** *or* ***off-line****, on* ***short-*** *or* ***long-terms****, and*
- *where the evidence may include* ***already established trust relations****.*

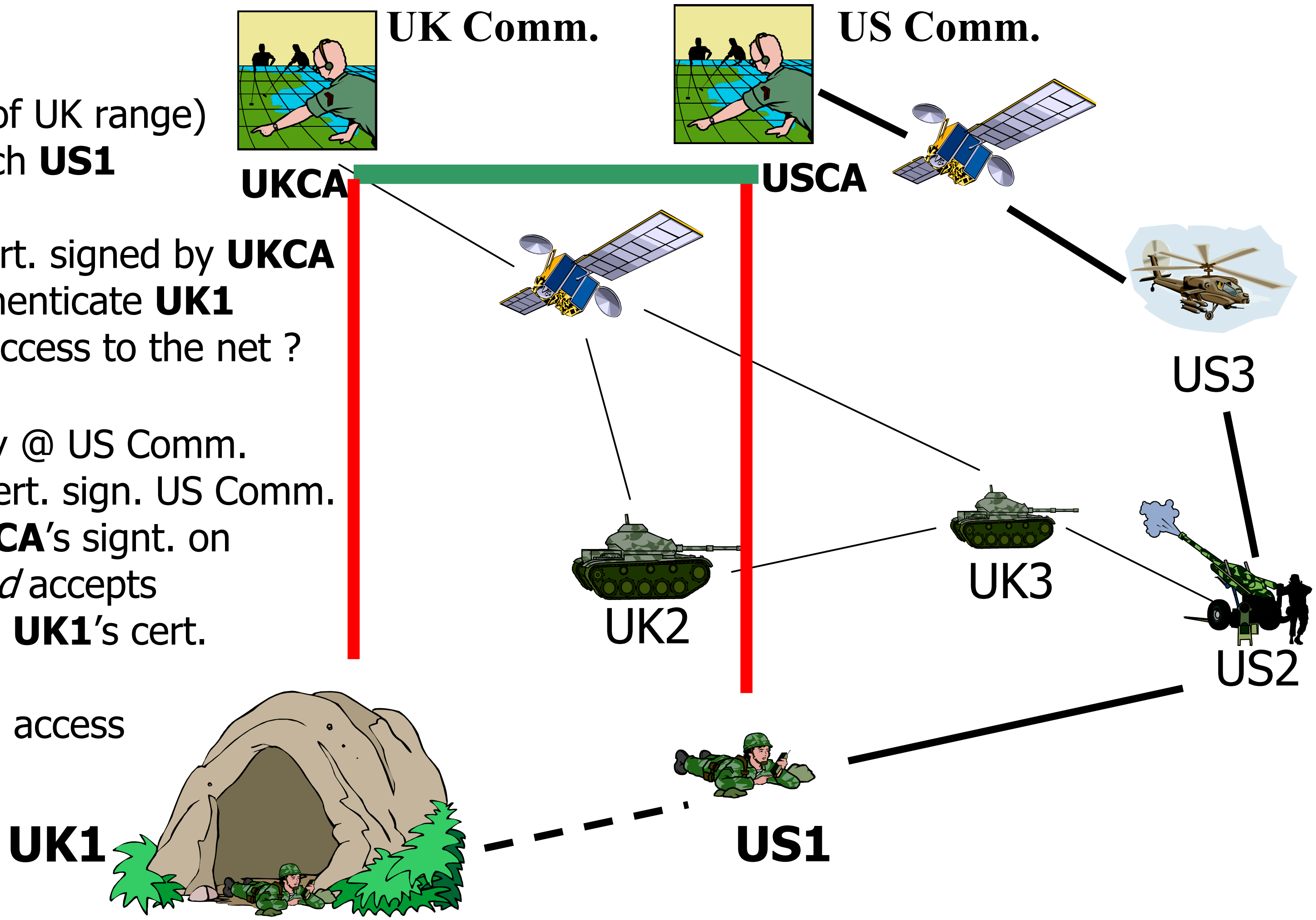
Old Focus: The Internet...

Scenario 1:
UK1 is lost (out of UK range) and can only reach **US1**

UK1 b-casts a cert. signed by **UKCA**
• Could **US1** authenticate **UK1** and grant him access to the net ?

- **US1** -> Directory @ US Comm.
- **US1** <- **UKCA** cert. sign. US Comm.
- **US1** accepts **USCA**'s signt. on **UKCA**'s cert. *and* accepts **UKCA**'s signt. on **UK1**'s cert.

• **US1** grants **UK1** access



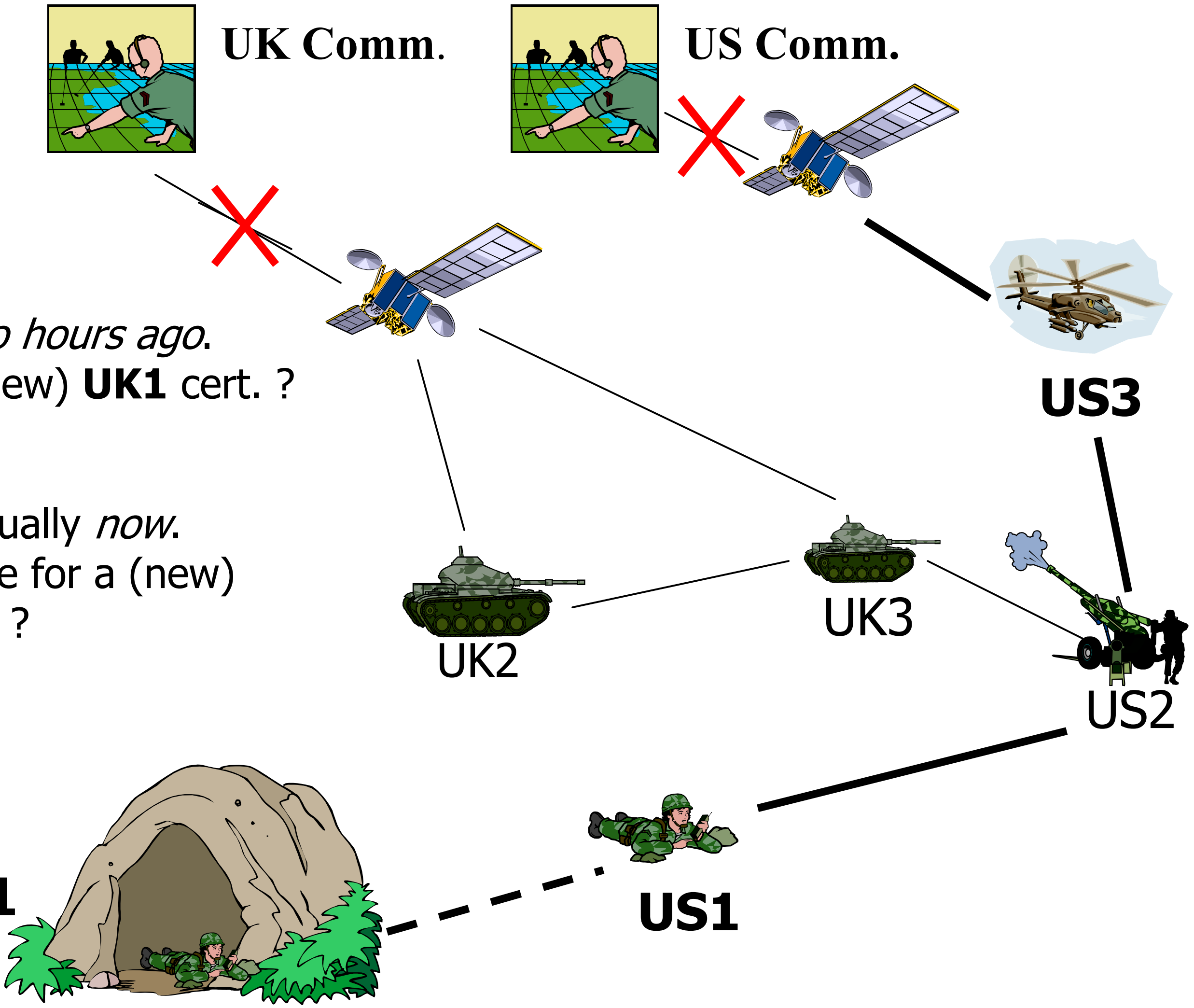
All trust relations (- - and evaluated evidence) are *available* and are used.

... vs. New Focus: MANETs

Scenario 2:
What if satellite links die ?
Or if **UK1**'s certificate expires ?

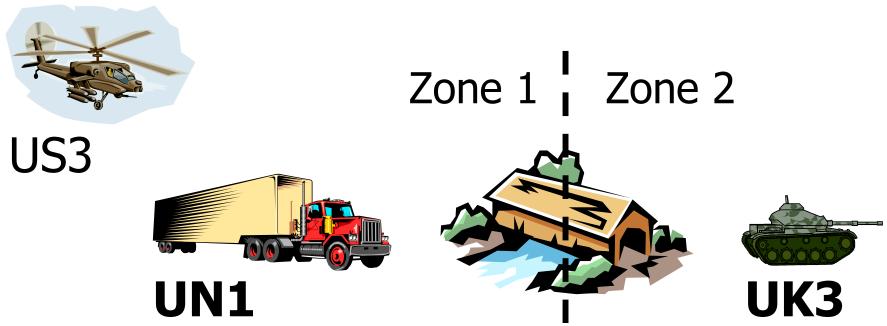
- **Fact 1**: **US3** located **UK1** *two hours ago*.
  - Should **US3** have issued a (new) **UK1** cert. ?

- **Fact 2**: **US1** locates **UK1** visually *now*.
  - Should **US1** issue a certificate for a (new) **UK1**'s key? What about **US3** ?



Need 1: Dynamic, proactive, generation of trust evidence on-line

… MANETs *(cont.)*



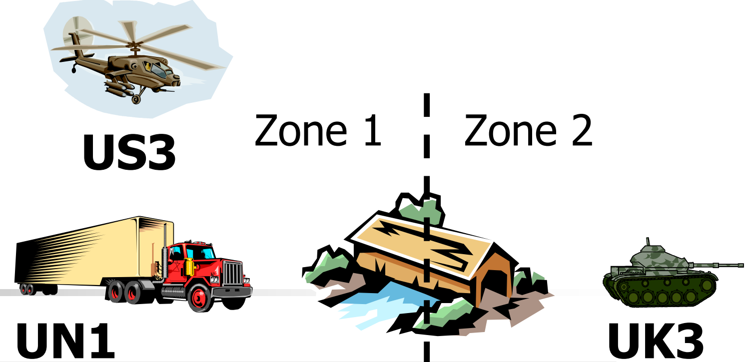
Scenario 3:

- **UN1** needs a "zone report" before entering Zone 2 and sends a request to **UK3**
- **UK3** negotiates with **UN1** the *types* of credentials needed for a "zone report"

UK3's policy for providing "zone reports":

(**Role** = UK/US mil.$\vee$ UN convoy) with conf.= high $\wedge$ (**location**={neighbors}) with conf.= medium

... MANETs *(cont.)*



- **UN1's request presents credentials**

Cert(**Role**=UNConvoy)$_{\mathbf{USCA}}$; ~~Cert(**Location**/GPS=zone2)$_{\mathbf{GPS1}}$~~; Cert(**Location**/Visual=zone2)$_{\mathbf{US3}}$

Fact 3: **UK3**'s trust relations **UKCA** for **Role**; **GPS1**, **UAV1**, and **UK1** for **Location**

Fact 4: Directory Server @ UK Comm. and **UK1** are *out of* **UK3**'s *range*

UK3's *metric* for confidence evaluation of *location evidence*

- Type(source) = GPS and source trusted -> conf.= low
  = UAV and source trusted -> conf.= low
- Type(src1) = UAV
  $\wedge$ Type(src2) = GPS and src1 and src2 trusted -> conf.= medium
- Type(source) = Visual and source trusted -> conf.= high
- Other -> conf.= null

UK3's *metric* for confidence evaluation of *role evidence*

- Type(source) = CA and source trusted -> conf.= high
- Other -> conf.= null

UK3 must *collect & evaluate evidence* re: **USCA**, **US3** *via net* search

Should UK3 return a "zone report" to UN1 ?

Research Areas

- Dynamic, proactive, generation of trust evidence
- Methods for trust-evidence distribution / revocation
  - Characteristics
    - *"Nothing but net": no distribution / rev. infrastructure but the network itself*
      - evidence may be stored anywhere in the network
      - producer may be unreachable at time of evidence use
    - *It is not just a request routing problem ...*
      - A principal may need more than one answer per request
        - Ideally should collect all the evidence that has been generated
        E.g: REQUEST(Alice/Location) should return more than one answer
      - A principal may *not* know what to look for
        - should handle wildcard requests; e.g: REQUEST(Alice/*)

Research Areas (ctnd.)

Evaluation metrics for of trust evidence (on-line)

- accept uncertainty, and negative evidence
- “weed-out” false evidence

Early work: limited types of evidence and mostly off-line generated

- R. Yahalom, B. Klein and T. Beth [1993]
- T. Beth, M. Borcherding, and B. Klein [1994]
- Ueli Maurer [1996, 2000]
- M. K. Reiter and S. G. Stubblebine [1997]
- etc.