Lectures 6: Security for Mobile and Wireless Computing

Ing-Ray Chen

CS 6204 Mobile Computing

Virginia Tech

Courtesy of G.G. Richard III for providing some of the slides

Protect what?

- Integrity
  - **System: performs its intended function in an unimpaired manner, free from deliberate or inadvertent unauthorized manipulation of the system.**
  - **Data: Should be possible for a receiver to verify that data has not been modified; an intruder should not be able to substitute fake data**
- Confidentiality
  - **Only intended recipient(s) should be able to read data**
- Non-repudiation
  - **Sender should not be able to falsely deny sending data.**
- Availability (Denial of Service, Distributed DoS)
  - **A third party with no access should not be able to block legitimate parties from using a resource.**

Security: Before/During/After Attack

- Prevention (before)
  - Authentication, authorization, accounting
- Detection (during)
  - Intrusion Detection
    - **Host/network**
    - **Signature/Anomaly behavior**
- Reaction (after)
  - Digital Forensics
    - **Evidence preservation**
    - **Who? What? When? From where?**
    - **Sources (files, logs, timestamp info, ISP records, …)**
  - Attack Assessment, Damage Assessment, Data Recovery

Before

- Prevention
  - Authentication: *"Are they who they claim to be?"*

    "The act of verifying a claimed identity, in the form of a pre-existing label from a mutually known name space, as the originator of a message (message authentication) or as the end-point of a channel (entity authentication)."
  - Authorization: *"Do they have permission to do it?"*

    "The act of determining if a particular right, such as access to some resource, can be granted to the presenter of a particular credential."
  - Accounting: a log or history of what happened

    "The collection of resource consumption data for the purposes of capacity and trend analysis, cost allocation, auditing, and billing. Accounting management requires that resource consumption be measured, rated, assigned, and communicated between appropriate parties."

Wireless Risks

- Wireless – all of the above concerns *plus* an increased risk of eavesdropping (and transmitting).
  - No need to tap or plug into the network. Only need to be "nearby."
  - Depending on the wireless technology, *nearby* can be line-of-sight, same room, outside a building, within a few miles
- Greatly increases threats to confidentiality, integrity, authentication, non-repudiation

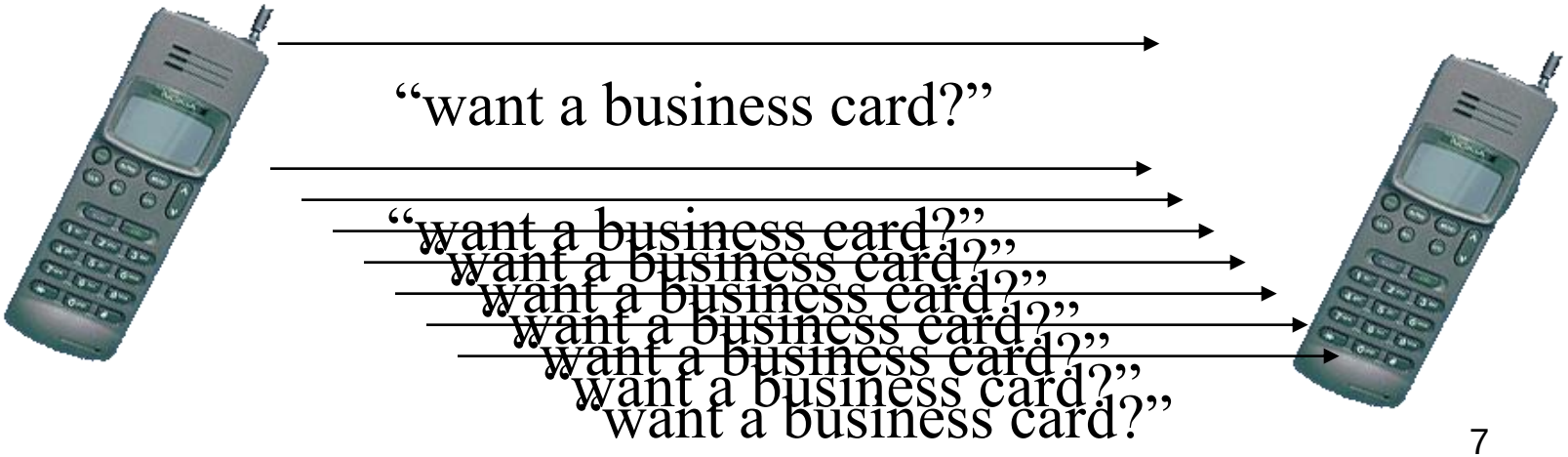
Wireless Risks (2)

These risks can allow adversaries to:

- Perform data snooping
- Hijack sessions (e.g., Man-in-the-middle)
- Commit fraud and identity theft (e.g., gathering an individual's personal information from RF-enabled cards carried on a person in their access control)

Risks: Resource Depletion

- Hardware limitations, such as low network bandwidth and limited battery power, also increases denial-of-service risk:
  - Resource depletion/exhaustion attacks



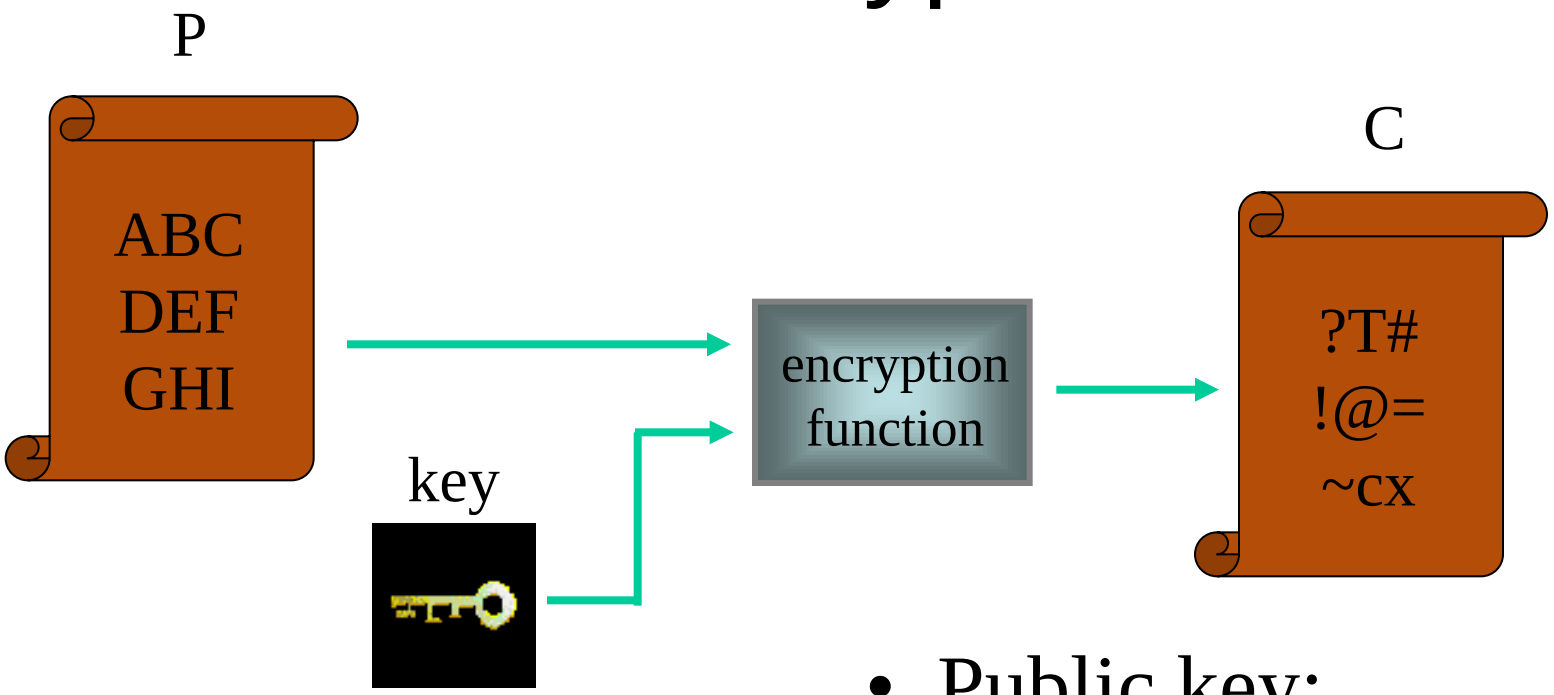
Protections

- Make it harder to intercept transmissions at the intruder's physical layer
  - Use low power, limit reception/interception range.
  - Use a technique like frequency hopping
    - But, generally want anyone to be able to join in and use the network.
    - Frequency hopping is actually used to increase number of users, *not* for protection.

Protections

- Encryption: “they” can’t decode data, so they can’t use what they do steal
- Digital signatures: prevent forging or modifying data

Encryption



- Symmetric key:

  $C = E_k(P)$,

  $P = D_k(C)$

- Public key:

  $C = E_{pub}(P)$,

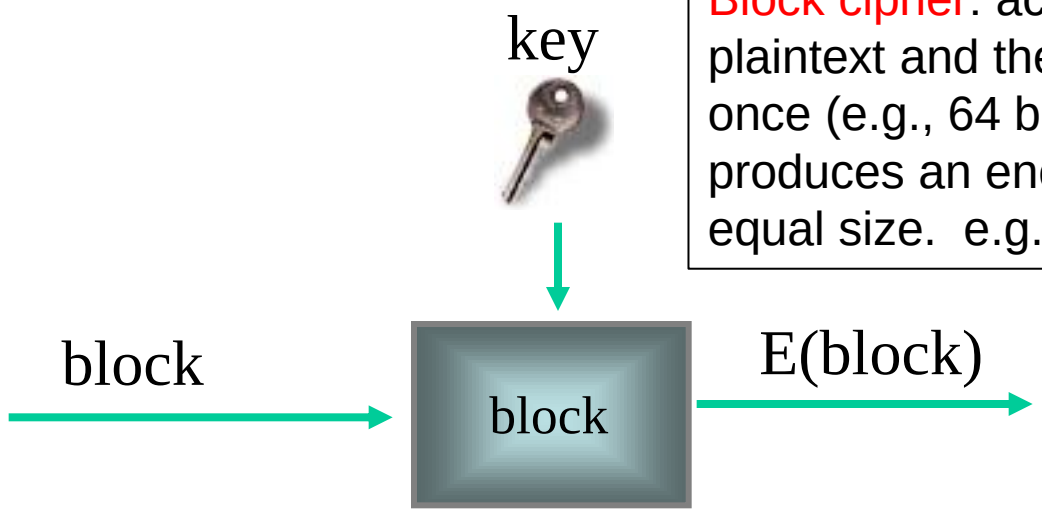
  $P = D_{priv}(C)$, also

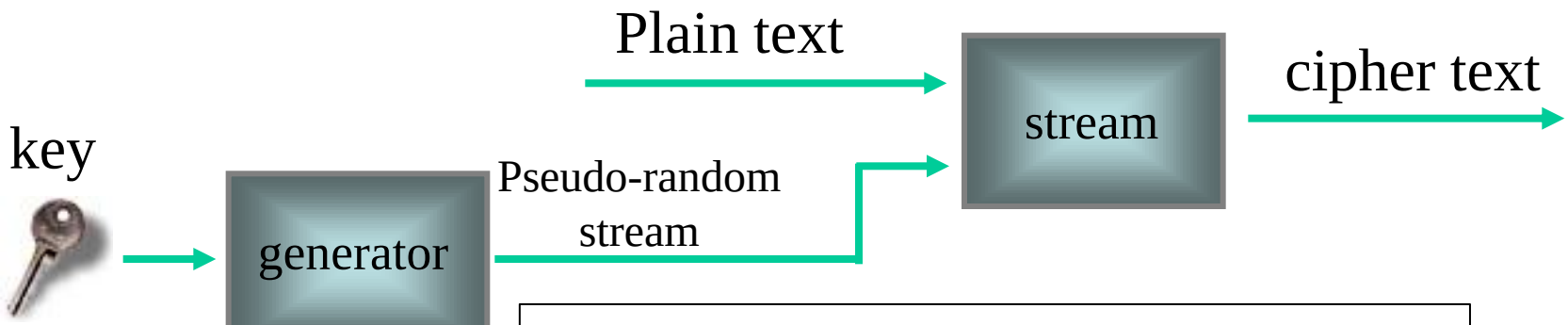
  $C' = E_{priv}(P)$,

  $P = D_{pub}(C')$

Block vs. Stream Cipher

Block cipher: accumulates a group of plaintext and then operates on it at once (e.g., 64 bits at a time) and produces an encrypted block of equal size. e.g., DES, AES, RSA





Stream cipher: operate on plaintext a single bit/byte at a time. e.g., RC4 used in WEP

Simple examples

– XOR:

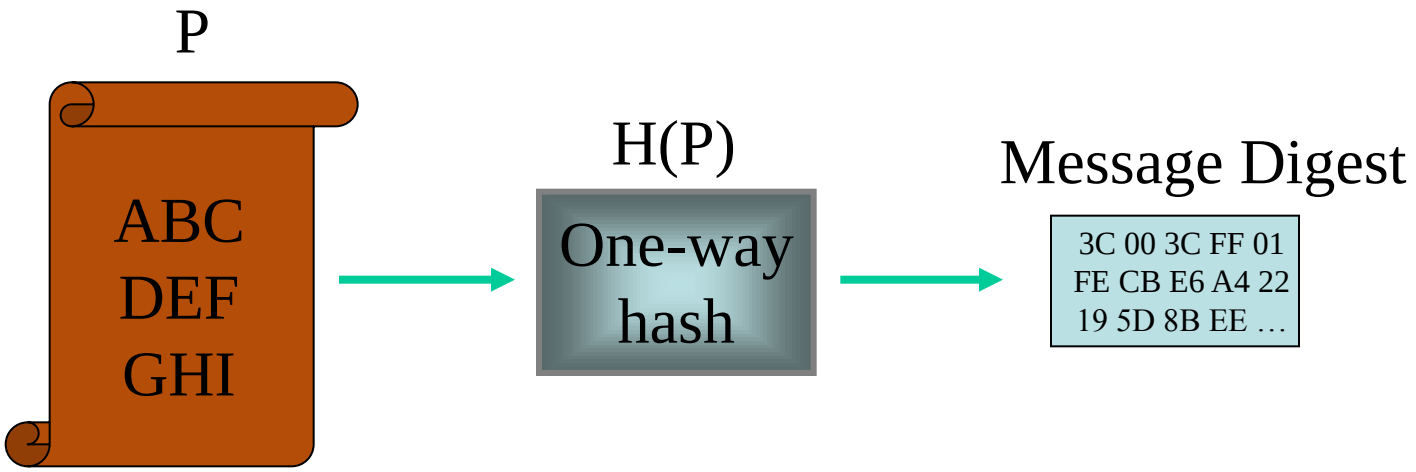
Plaintext		Key*		Ciphertext
• 1111 0000	XOR	1010 1010	=	*0101 1010*
• *0101 1010*	XOR	1010 1010	=	1111 0000

(* Note that this is really a single sample from a key-stream.)

– Rotation (trivial cipher):

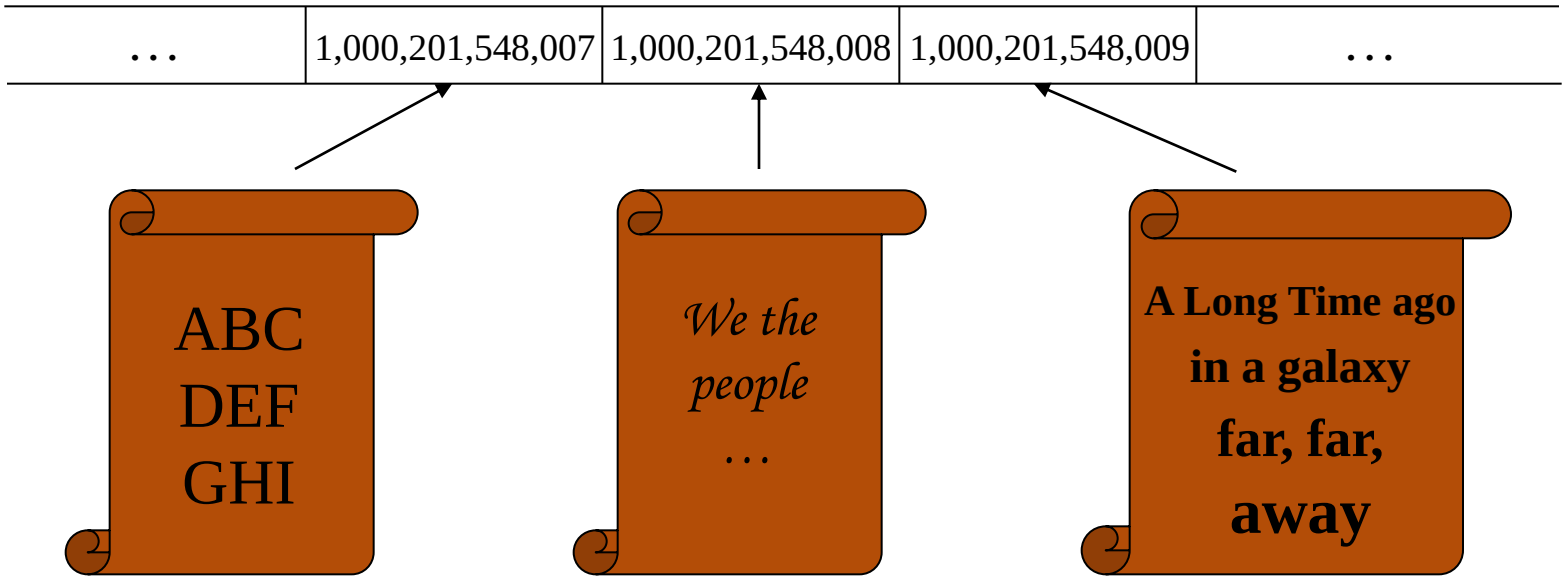
- ROT1: “HAL” → “IBM”

Message Digests and Hashes



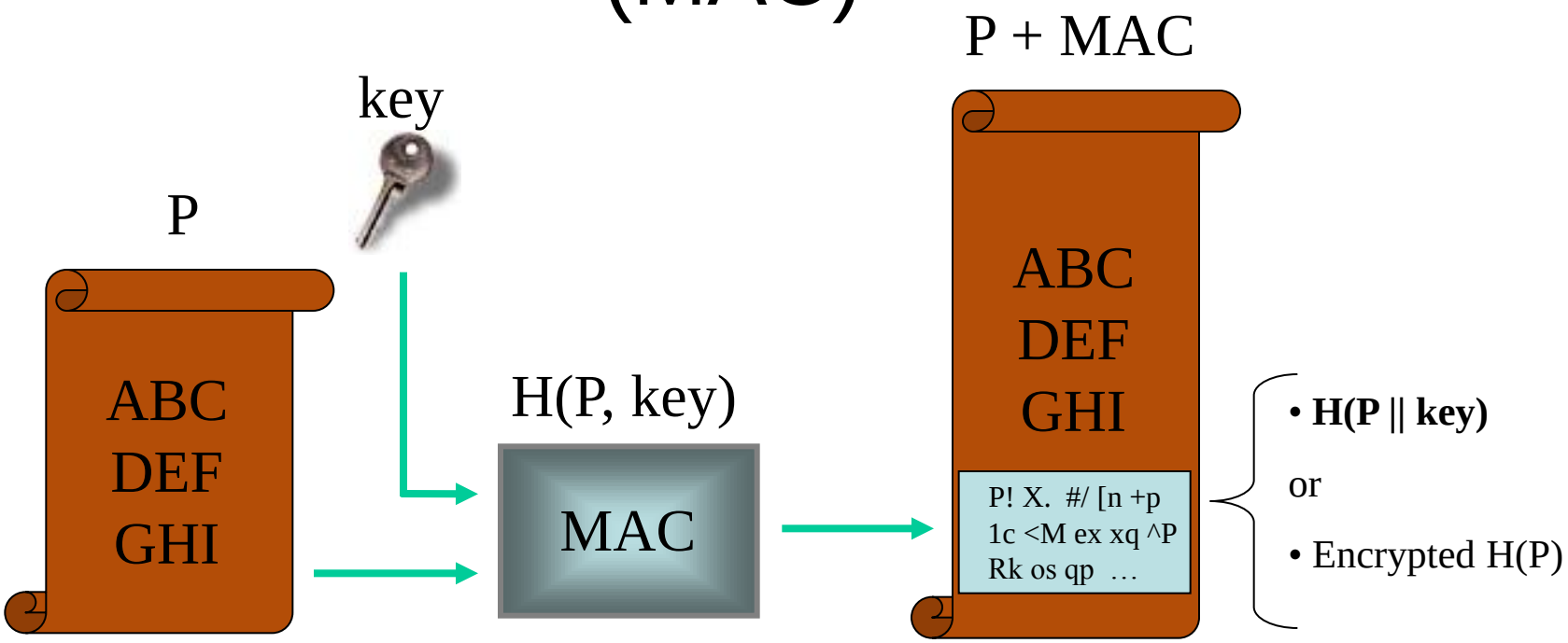
- A cryptographically secure one-way hash function produces a short sequence of bytes (e.g., 128 or 160 bits) based on the input.
- e.g., MD4, MD5, SHA

Hash Space



- Cryptographically secure: A single bit change in source changes ~½ the bits in the hash.
- Small changes in the hash come from very different sources.
- Computationally infeasible to find matching source from hash.

Message Authentication Code (MAC)



- For authenticity without secrecy: attach the MAC to the message
- MAC is a one-way hash function plus a secret key
  - Hash the concatenation of the message and key, or
  - Encrypt the hash of the message with the key

Message Overhead of Encryption

- Public key: send the message one by one using the receiver's public key
  - Message overhead is n * M, where
    - n: number of recipients
    - M: message length
- Hybrid (PGP): encrypt shared session key using public key encryption one by one and then broadcast the encrypted message with the shared key
  - Message overhead is n*K + M, where
    - K: key length of a shared session key
- If M >> k, message overhead is greatly reduced

Costs of Protections

- Encryption overhead! (more tradeoffs)
  - Poor performance
  - CPU load
  - Power consumption
  - Reduced battery life
  - Increased data size → increased transmission time

Cost of Protections

- Public Key Infrastructure (PKI)
- Certificates
  - Certificate revocation or expiration
  - Trusted 3rd party
- Shared secret key (and risk) vs. public key
- Key management
  - Key setup
  - Key exchange
- Individual vs. group keys (overhead)

Misc. Attacks

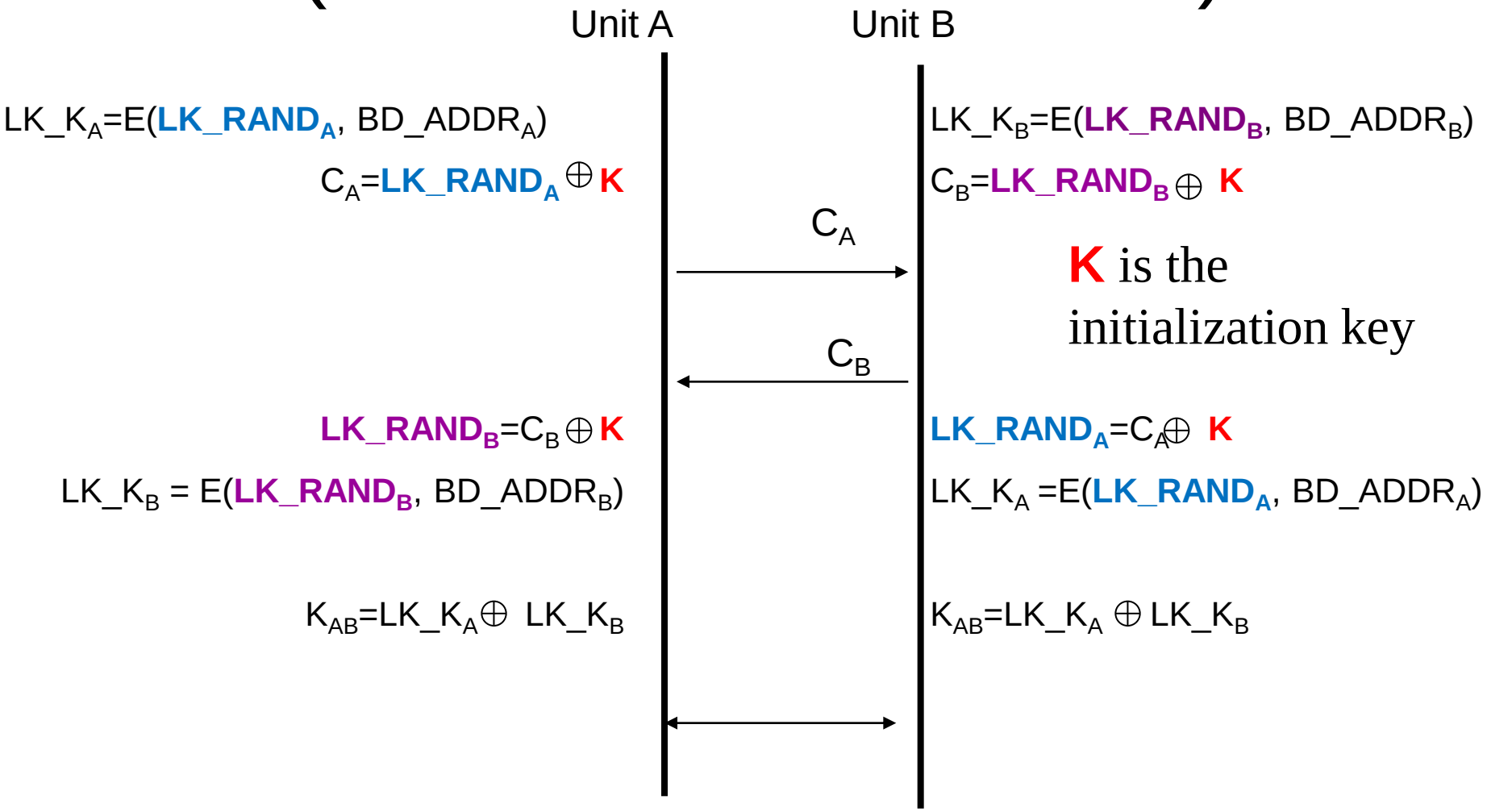
Counterattack methods:

- Man-in-the-middle attacks
  - Use authentication
- Replay attacks
  - Use sequence number or one-time unique number (called nounce) that will not be honored the second time
- Traffic analysis
  - Use encrypted communication (e.g., IPsec)

IEEE 802 Standards

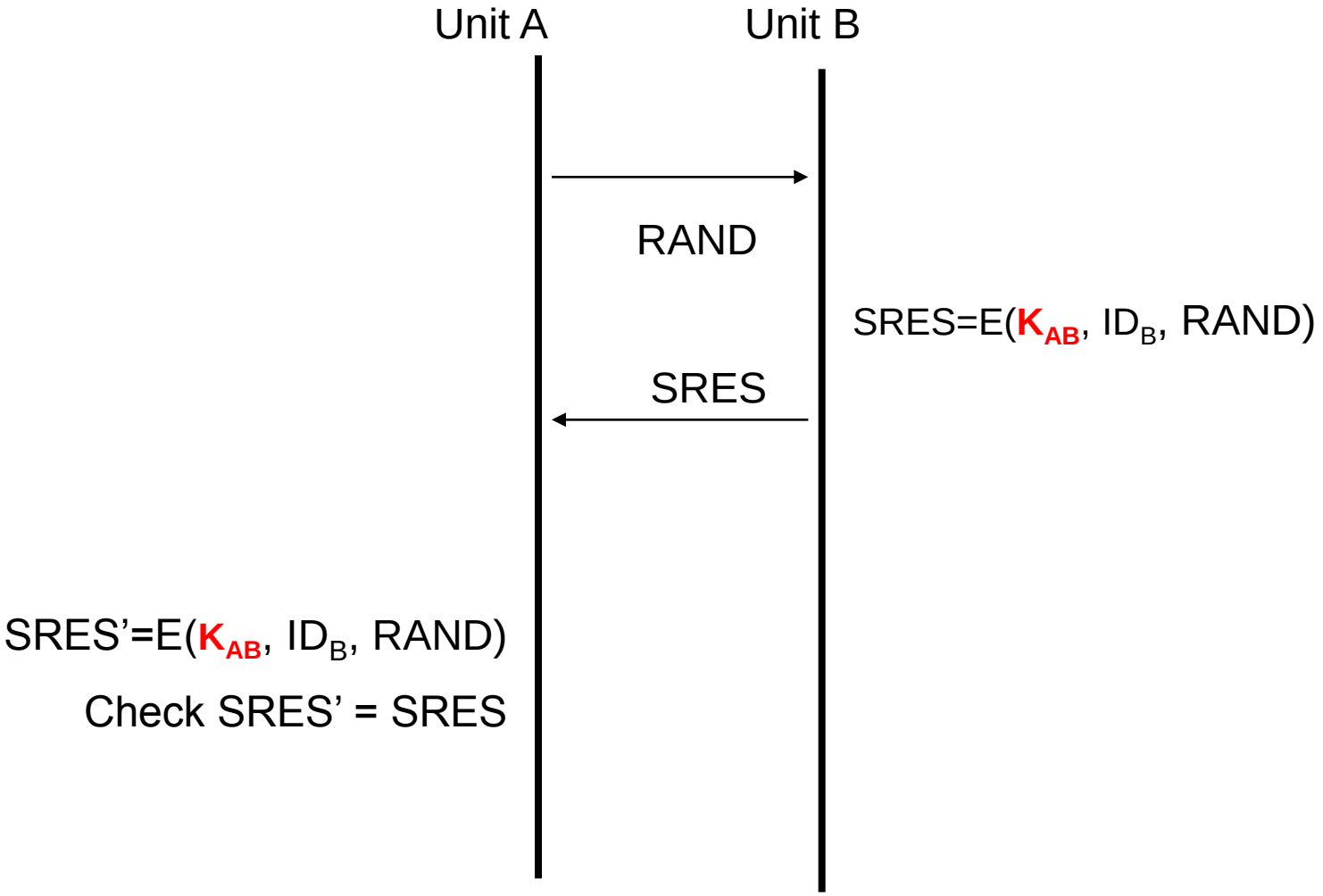
- 802.11 – IEEE Standard, 1997.
- 802 LAN/MAN Standard Committee
  - 802.1d – MAC bridging standard
  - 802.1x – Port-based Network Access Control
  - 802.2 – Logical Link Control
  - 802.3 – Ethernet
    - 802.3z – 100BaseT Fast Ethernet
  - 802.5 – Token Ring
  - 802.11 – Wireless LAN
    - 802.11 – "basic" wireless
    - 802.11a - 5GHz, 54Mb
    - 802.11b – 2.4GHz, 11Mb
    - 802.11e – QoS
    - 802.11f – AP interop
    - 802.11g – faster 802.11b, starting at 20Mbps
    - 802.11h – transmit power control for 802.11a (Europe)
    - 802.11i – better security
    - 802.11j – Japanese 802.11
    - 802.11n – 600Mb MIMO
    - 802.11p – automotive apps
  - 802.15.1 Bluetooth
  - 802.15.4 Low-rate (low power) (ZigBee on top of 802.15.4)
  - 802.16 Wireless Metropolitan Area Network (WMAN)

Security in Wireless PAN (Bluetooth/IEEE 802.15)



Protocol for deriving a shared secret key between two Bluetooth devices: random number exchanged through the initialization key K

Bluetooth Authentication (by the fact that you have the shared key)



A challenge/response protocol for authentication

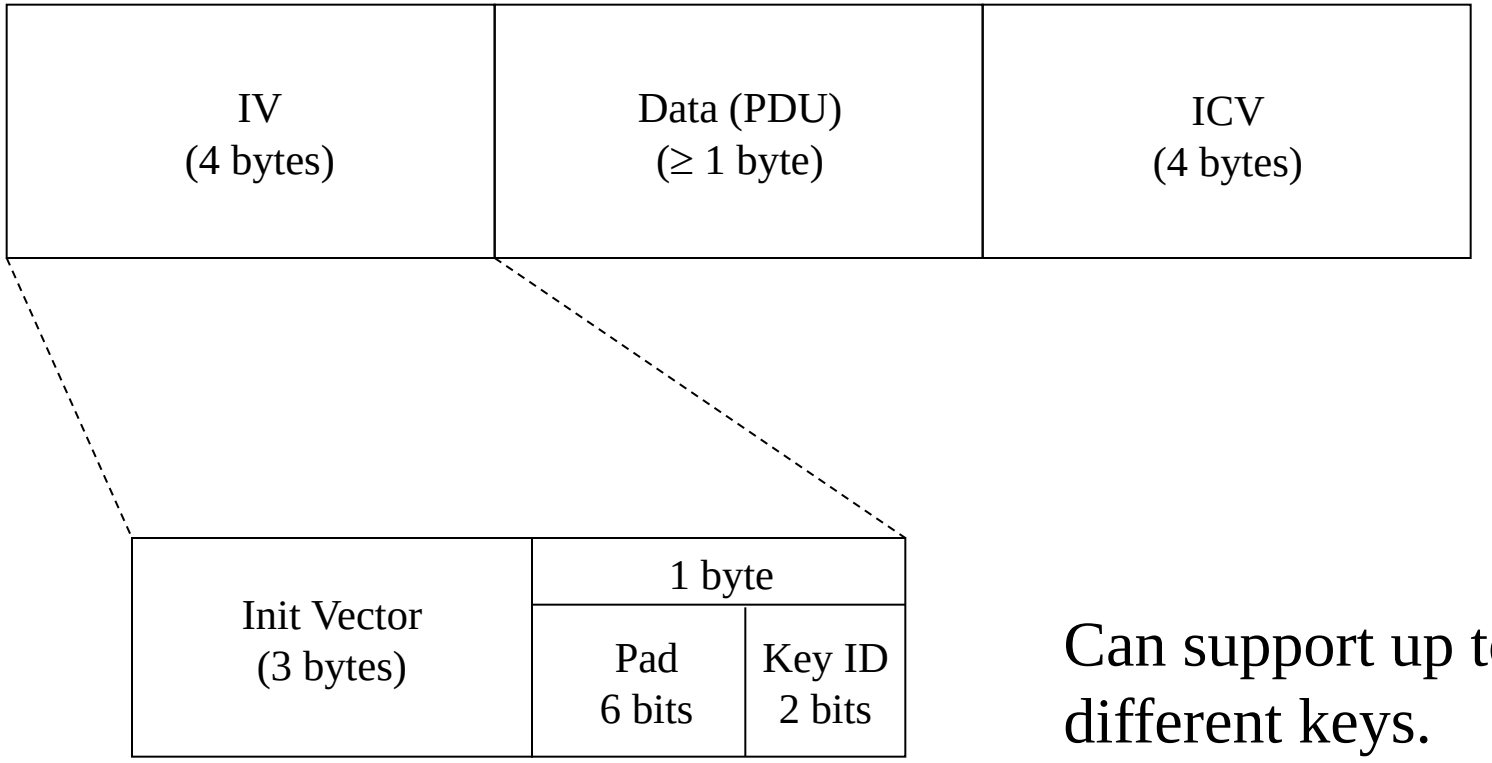
WEP – Protection for 802.11b

- WEP stands for Wired Equivalent Privacy
  - "No worse than what you get with wired systems"
- Criteria:
  - "Reasonably strong"
  - Self-synchronizing – mobile terminals often go in and out of coverage
  - Computationally efficient – in HW or SW since low MIPS CPUs might be used
  - Optional – not required to use it

WEP – How It Works

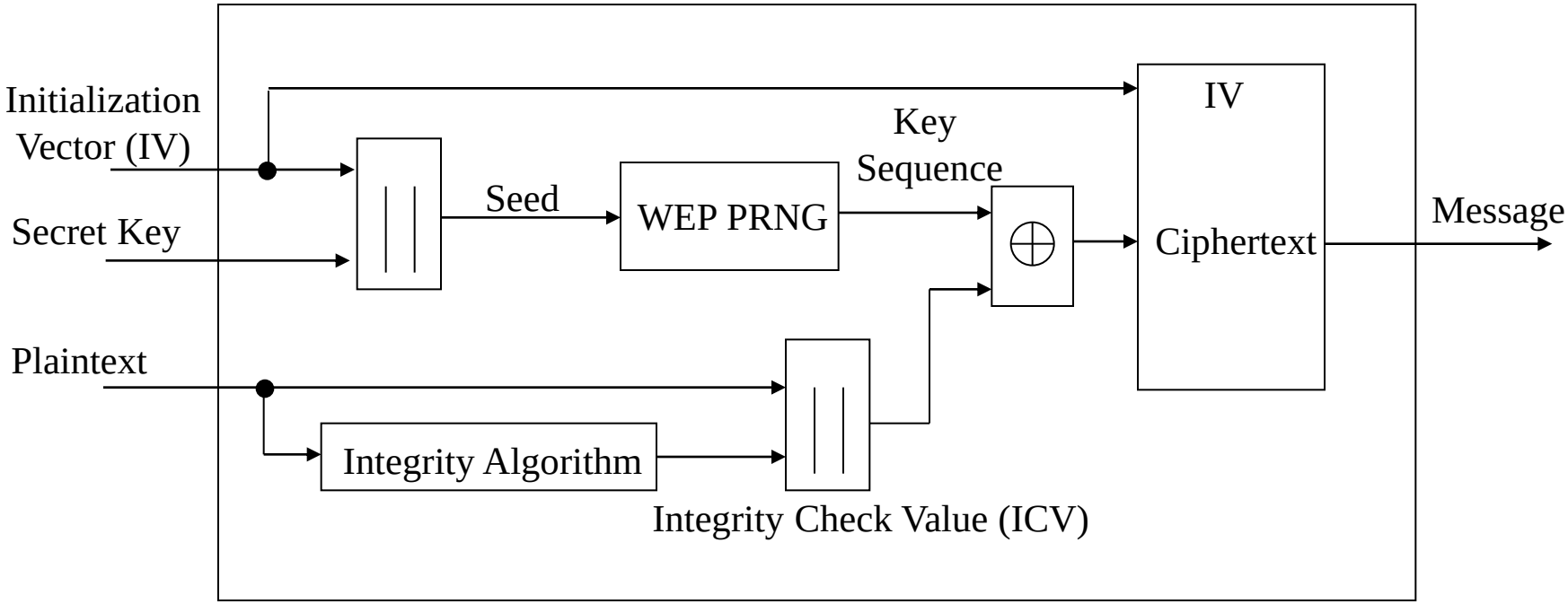
- Secret key (40 bits or 104 bits)
- Initialization vector (IV)
  - (24 bits, by IEEE std.)
- Total of 64 or 128 bits “of protection.”
- RC4-based pseudo random number generator (PRNG)
- Integrity Check Value (ICV): CRC 32

WEP Data Frame



Can support up to 4 different keys.

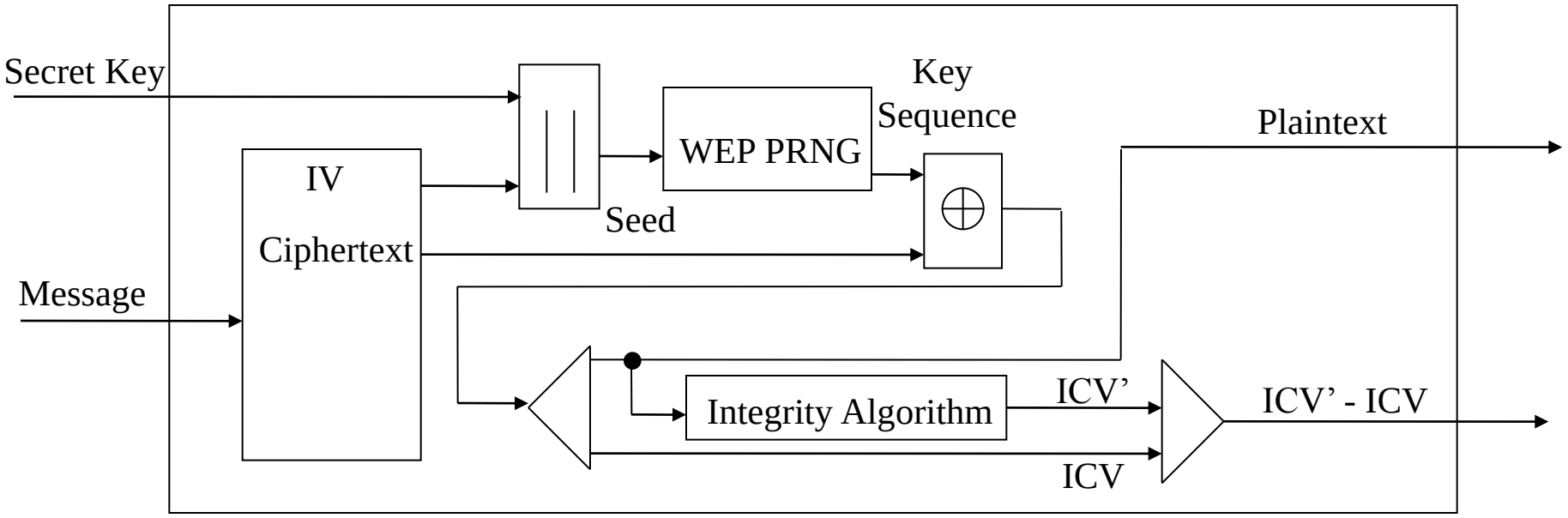
WEP Encryption



WEP Encryption Process

1. Compute ICV using CRC-32 over plaintext msg.
2. Concatenate ICV to plaintext message.
3. Choose random IV, concatenate it to secret key and input it to RC4 to produce pseudo random key sequence.
4. Encrypt plaintext + ICV by doing bitwise XOR with key sequence to produce ciphertext.
5. Put IV (in the clear) in front of cipertext.

WEP Decryption

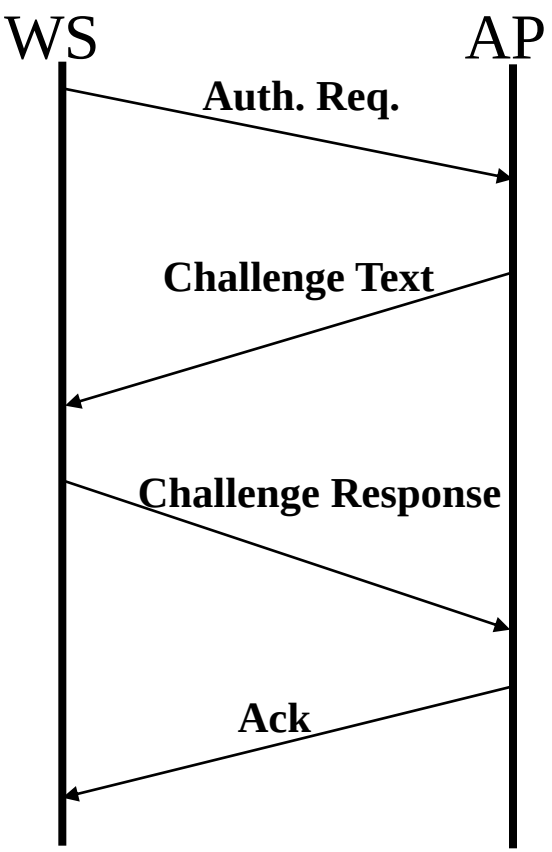


WEP Decryption Process

1. IV of message retrieved + secret key to generate key sequence, k.
2. Ciphertext XOR k ➔ original plaintext + ICV.
3. Verify by computing integrity check on plaintext (ICV') and comparing to recovered ICV.
4. If ICV $\neq$ ICV' then message is in error; send error to MAC management and back to sending station.

WEP Station Authentication

1. Wireless Station (WS) sends Authentication Request to Access Point (AP).
2. AP sends (random) challenge text T.
3. WS sends challenge response (encrypted T).
4. AP sends ACK/NACK.



WEP Weaknesses

- Forgery Attack
  - CRC-32 is weak: Can alter bits in the encrypted message and CRC-32 without knowing plaintext
  - Source and destination in the header are in the clear: can be altered
- Replay Attack
  - Can eavesdrop and record a session and play it back later.
- Collision Attack (keystream reuse: 24 bit IV+40 bit WEP key)
  - 24-bit IV value (sent in the clear) is reused quickly: if sequential: roll-over in < ½ day; if random: after 5000 packets, > 50% of reuse
  - The attacker can determine the pseudo random key sequence for each IV if it can collect a (plaintext, ciphertext) pair for this IV value in its dictionary and can use this key sequence to decrypt future ciphertext with the same IV value
- Weak Key Attack
  - Certain RC4 weak keys tend to generate output strongly correlated with a few bytes of the WEP key. The attacker can collect outputs generated from weak keys and guess the key in a reduced space
  - Weak keys cannot be avoided because of IV reuse. The attacker can wait for packets with certain IV values which create weak keys

Ways to Improve Security with WEP

- All encryption modes of operation should use (secure) MAC, rather than CRC
- Better key management:
  - Change key early and often
- Better: replace with something else
  - WPA (WiFi Protected Access) and 802.11i

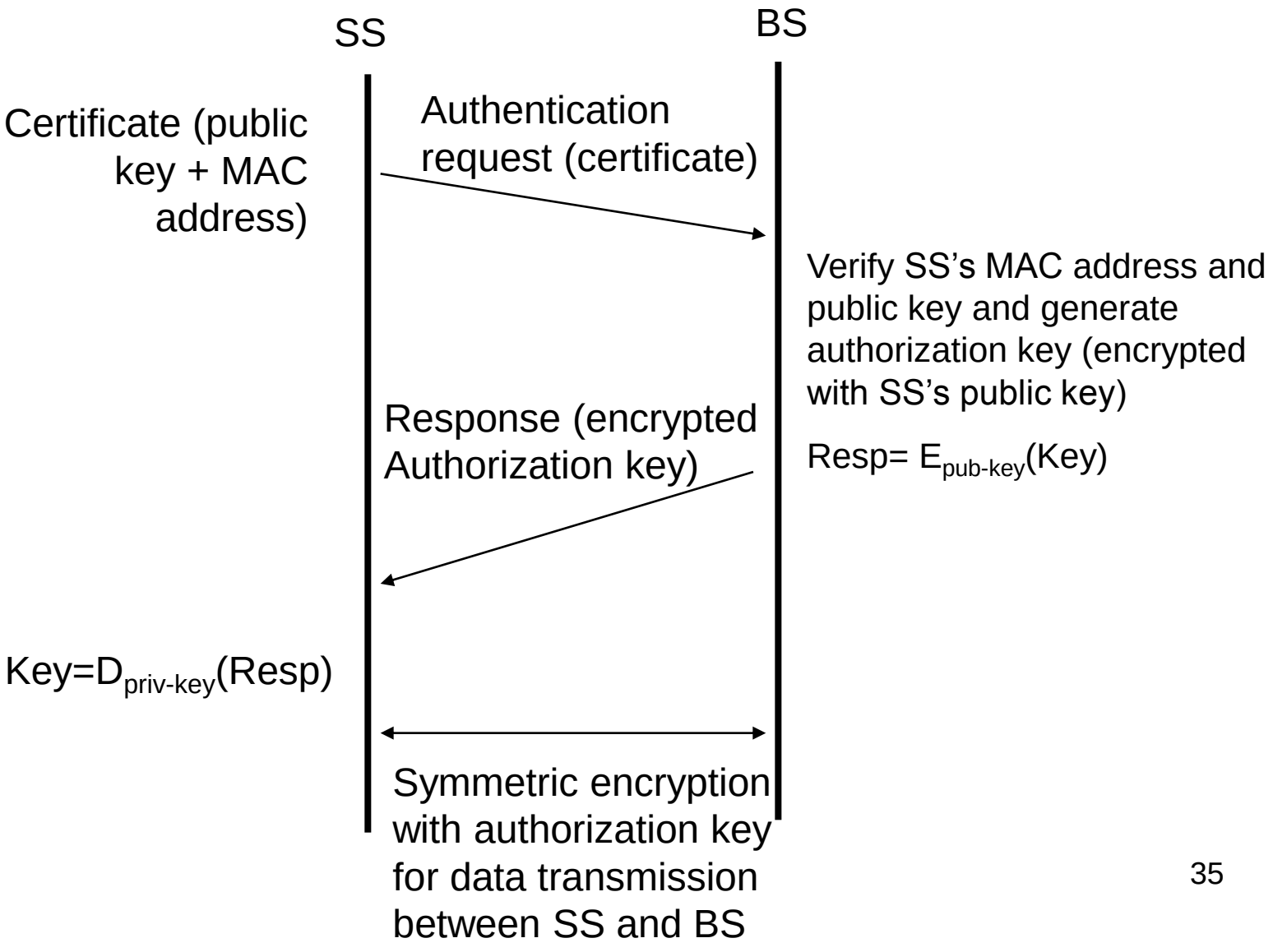
802.11i vs. WEP

- Forgery attack
  - Use MAC (called message integrity code (MIC) or “Michael”) utilizing a cryptographically secure hash to make changing data impossible
  - Hash is applied to packet payload *plus* src and dest addresses
- Replay attack
  - Use 48-bit IV in strictly increasing sequence to prevent replay attacks; the base key is rekeyed when max IV is reached
  - A receiver discards out-of-sequence packets
- Collision (Key-stream) attack
  - Use 48-bit IV, forcing rekey of the base key after 2^{15} packets (early), so every IV value is used just once for a base key
  - Use 802.1X EAPOL (Extensible Authentication Protocol Over LAN) to configure a new key for every association
- Weak Keys of WEP
  - Use per-packet key derived from the base key (along with transmitter address, and IV) which is rekeyed early and often

Security in 802.16 Wireless Metropolitan Area Networks

- Connect base station (BS) at the ISP to the subscriber station (SS) and support speed up to 268 Mbps
- Privacy Key Management (PKM) is the key management protocol allowing BS to control access to the network, and SS and BS to exchange data
  - Use X.509 digital certificates, RSA public key encryption and symmetric encryption based on DES for data exchange

Key Management in 802.16



Security Weakness of 802.16 and Enhancements

- Authentication is only in one direction
  - SS should also be able to authenticate BS
  - Instead of relying on RSA public + private key pair, should have 802.1x style authentication suite
- DES is used for encryption which is not regarded as secure (encryption key is acquired after the authorization key)
  - AES-128 should be used
- No data replay protection
  - IVs should be sequential with base key rekeying to prevent replay attacks

Group Key Management in Mobile Ad Hoc Networks

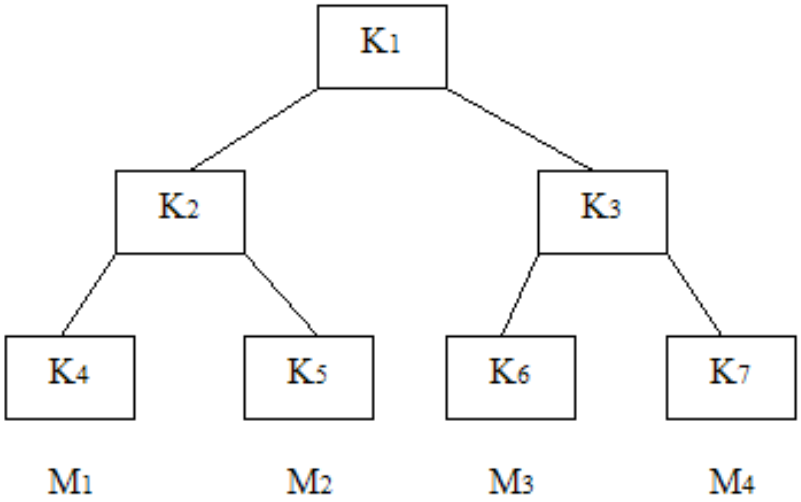
- Centralized: a key server is responsible for generating the group key and distributing it to group members
  - Logical Key Hierarchy (LKH): a key tree is maintained to efficiently update the group key after member join/leave events
- Contributory: all group members contribute to the group key generation
  - Based on Diffie-Hellman key exchange to agree on a secret key

Security Requirements for Group Key Management

- Forward secrecy: this guarantees that an adversary who knows an old group key cannot discover a subsequent group key
  - This ensures a member cannot learn future group keys after it leaves the group
- Backward secrecy: this guarantees that an adversary who knows the current group key cannot discover a previous group key
  - This ensures that a new member who joins the group cannot learn any previous group key

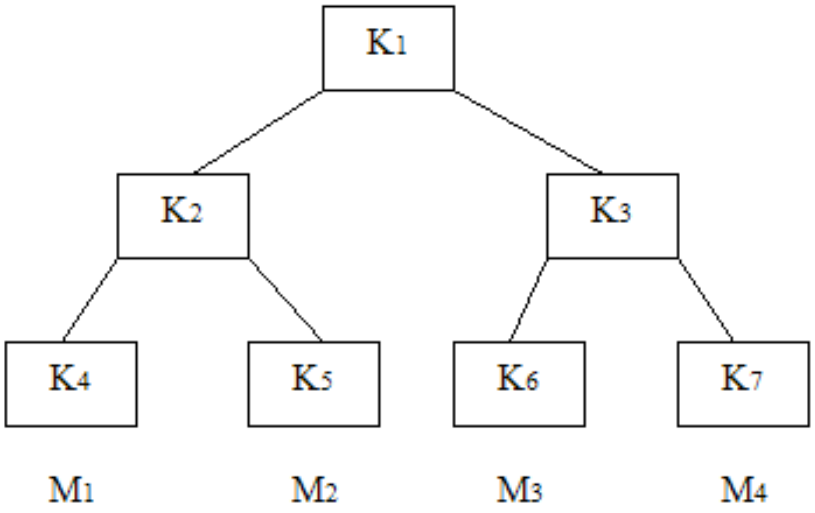
Centralized Group Key Management: Logical Key Hierarchy (LKH)

- A key server maintains a key tree to efficiently update the group key after a member join or leave event.
- Each node in the tree stores a cryptographic symmetric key. The root node stores the group key. A leaf node also represents a member.
- The following invariant is always maintained: each group member knows all the keys from its leaf node up to the root node, but no other key in the key tree. We call the set of keys that a member knows a *key path*.
- For instance, the key path for member M2 consists of K5, K2, and K1.



LKH

- When a new member joins the group, the key server sends to the new member all the keys on the key path over a secure channel.
- When a member leaves the group, the key server needs to update all the keys that the member knows, that is, all the keys on the member's key path.

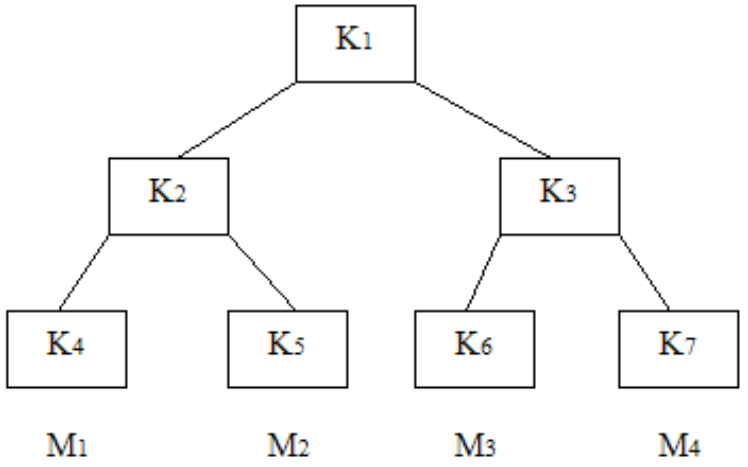


LKH: Member Join Events

- Suppose that a new member M4 just joins the group. Assuming that the last leaf node (corresponding to the key K7) of the tree is empty, the key server places the new member M4 on that leaf, selects new keys on M4's key path, and sends K7', K3', and K1' to M4 over a secure channel.
- In order to update the key paths of existing members, the server broadcasts the following key update message:

$$\{K_3'\}_{K_6}, \{K_1'\}_{K_3'}, \{K_1'\}_{K_2}$$

- Member M3 needs to update K3' and K1' on its key path. Since M3 knows K6, it can decrypt the first part of the key update message and recover K3'; it can then subsequently recover K1' using K3'.

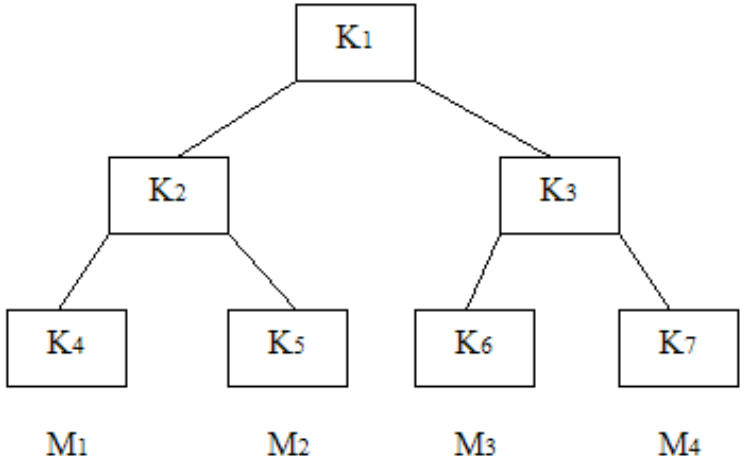


LKH: Member Leave Events

- Assume that member M2 leaves the group. The key server generates new, random keys for all keys on M2's key path: K5', K2' and K1'. The key server broadcasts the following key update message:

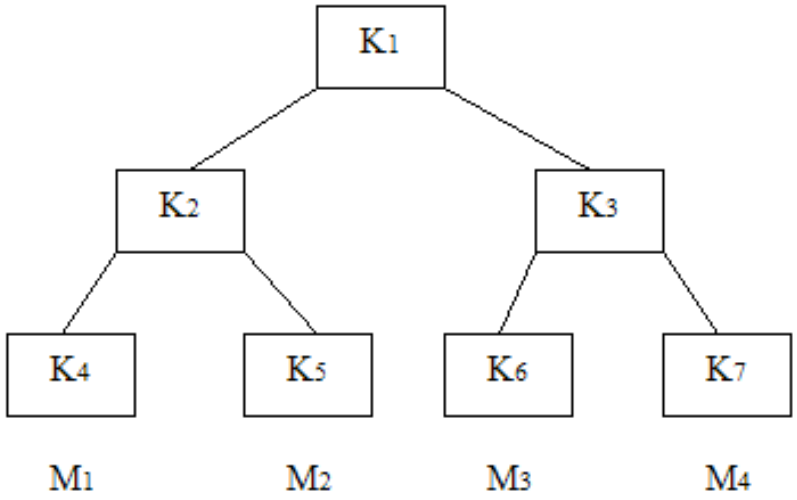
$$\{K_2'\}_{K_4}, \{K_2'\}_{K_5'}, \{K_1'\}_{K_2'}, \{K_1'\}_{K_3}$$

- Member M1 needs the new keys K2' and K1'. Since it knows K4, it can decrypt K2' from the key update message. Since M1 now knows K2', it can decrypt the new group key K1'. Members M3 and M4 only need the new group key. Because they both know K3, they can readily decrypt it from the key update message



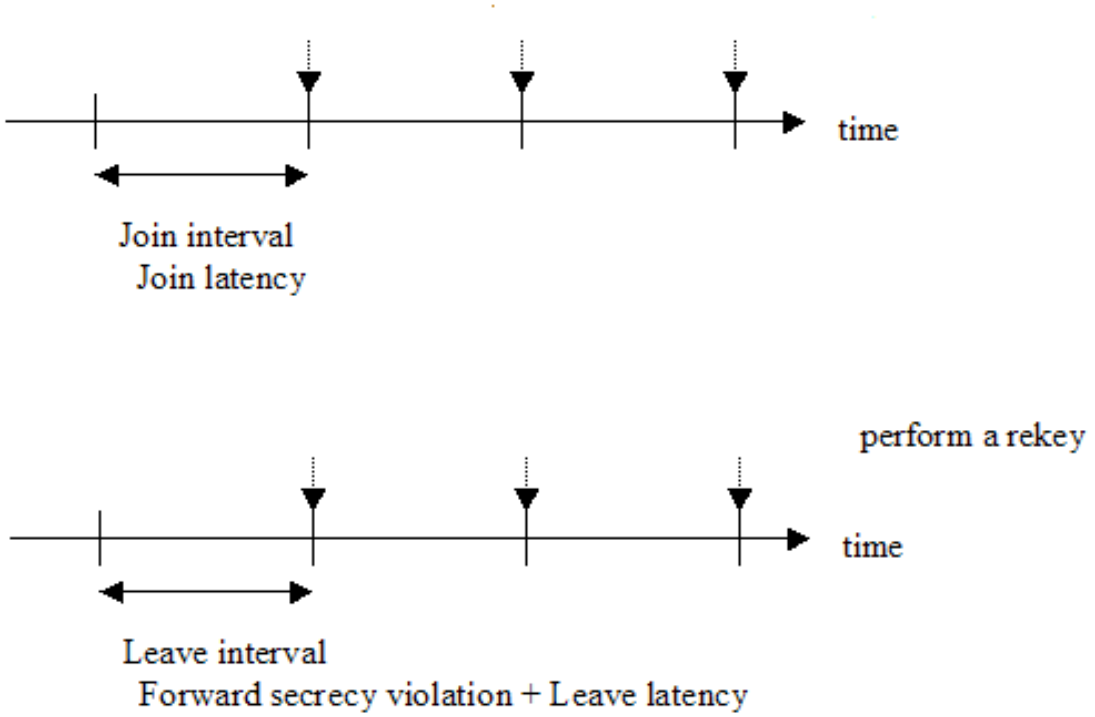
LKH: Message Overhead for Group Key Management

- A new member join event requires a message of length $k^* (2\log_2 (N) – 1)$ bits, where k is the length of a key, and N is the number of members.
- A member leave event requires a message of length $k^*2\log_2 (N)$ bits.
- The main benefit of LKH is that it provides secure key update that only requires a broadcast message size that is logarithmic in the number of group members



LKH Variation: Batch Rekeying

- Instead of immediate rekeying, batch rekeying delays join and leave operations periodically or after collecting a few for computation and energy conservation
- Tradeoff of energy conservation vs. long latency and forward secrecy violation



Contributory Group Key Management [Ref. 17]

- Mostly based on Diffie-Hellman (DH) key exchange algorithm which allows the establishment of a secret key between two entities by means of data exchange through an insecure channel.

Diffie-Hellman Key Exchange

- A and B agree on two randomly selected numbers p and g such that p > g and p is a large prime number.
- A chooses a secret random number α and B chooses a secret random number β
- A computes a public value $T_A = g^{\alpha} \bmod p$ and B computes a public value $T_B = g^{\beta} \bmod p$
- A sends T_A to B, and B sends T_B to A (in the clear)
- A computes $T_B^{\alpha} \bmod p = (g^{\beta})^{\alpha} \bmod p$ and B computes $T_A^{\beta} \bmod p = (g^{\alpha})^{\beta} \bmod p$
- Since $(g^{\beta})^{\alpha} = (g^{\alpha})^{\beta}$ these two entities share a secret key K
- The security is based on the difficulty of calculating K despite of knowing the public values T_A ($g^{\alpha} \bmod p$) and T_B ($g^{\beta} \bmod p$) when the prime number p is sufficiently large
- The weakness is lack of authentication of the two entities

A Contributory Group Key Agreement Protocol: CLIQUES (IKA.1)

- Stage 1: Contributions are collected from group members in rounds. Each group member receives a data set that represents the partial contributions from all group members that have already executed the first stage. The member adds its contribution and sends a new data set to the next group member.
- The set sent by the i^{th} node consists of i intermediate values, each containing (i-1) exponents, and a cardinal value with i exponents.
- For example, M_4 (the 4^{th} node) receives the set $\{g^{S1S2S3}, g^{S1S2}, g^{S1S3}, g^{S2S3}\}$ from M_3 and sends the set $\{g^{S1S2S3S4}, g^{S1S2S3}, g^{S1S2S4}, g^{S1S3S4}, g^{S2S3S4}\}$ to M5.
- The last group member M_n called the group controller will receive a data set whose cardinal value is $g^{S1S2\ldots Sn-1}$ and will compute $K= g^{S1S2\ldots Sn}$

Stage 1 of CLIQUES (IKA.1)

$$M_i \Rightarrow M_{i+1}:$$

$$\{g^{\frac{S_1 S_2 \ldots Si}{S_k}} \mid k \in [1,i]\}, g^{S_1 S_2 \ldots S_i}$$

A Contributory Group Key Establishment Protocol: CLIQUES

- Stage 2: the group controller adds its contribution to each intermediate value and broadcasts this new data set to every other node in the network
- Each intermediate value now consists of the contributions of all group members except one.
- Each group member M_i identifies the appropriate intermediate value (the one that does not contain its contribution) and raises it to its contribution S_i, thus obtaining $K= g^{S1S2...Sn}$

Stage 2 of CLIQUES (IKA.1)

$$M_n \Rightarrow M_i :$$

$$\{ g^{\frac{S_1 S_2 \ldots S_n}{S_i}} \mid i \in [1, n-1] \}$$

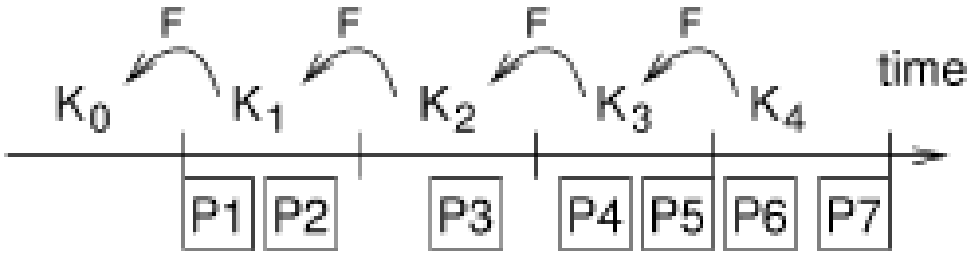
Performance of Contributory Group Key Protocols

- Complexity:
  - Number of messages
    - O(n) for CLIQUES IKA.1
  - Number of exponential operations
    - $O(n^2)$ for CLIQUES IKA.1 because i+1 exponential operations are required by the ith node
- If there are simultaneous broadcast messages, each broadcast is counted as 1 since wireless medium cannot allow simultaneous broadcast
- Able to cope with arbitrary member leave/join and disconnection events in MANETs

Authenticated Broadcast [Ref.19]

- Authenticated broadcast requires asymmetric mechanisms; otherwise, any compromised receiver could forge messages using the shared secret key.
- Asymmetric cryptography mechanisms (e.g., based on PKI) have high computation, communication, and storage overheads and may not be practical for MANETs and sensor networks
- **μTESLA**: an authenticated broadcast protocol which introduces asymmetry through a delayed disclosure of symmetric keys, which results in efficient broadcast authentication

μTESLA for Authenticated Broadcast

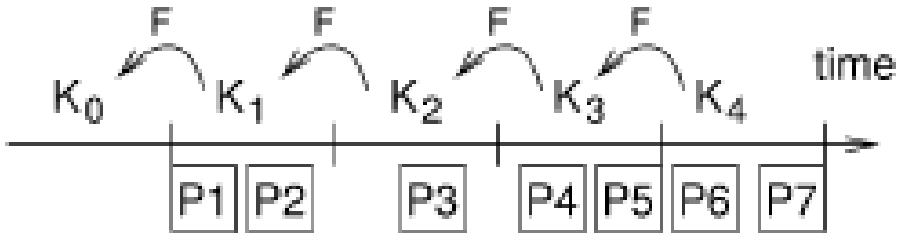


- The sender generates a one-way key chain right-to-left by repeatedly applying the one-way function *F* such that $K_i = F(K_{i+1})$
- The sender uses the keys of the key chain in reverse order. In interval *i*, the MAC key K_i is used to compute the MAC of the packets sent in that time interval.

μTESLA

- Delay disclosure of keys:
  - In time interval *(i + δ)*, the sender reveals key K_i. The key disclosure time delay *δ* is on the order of a few time intervals and must be greater than any reasonable round trip time between the sender and the receivers

μTESLA - Example



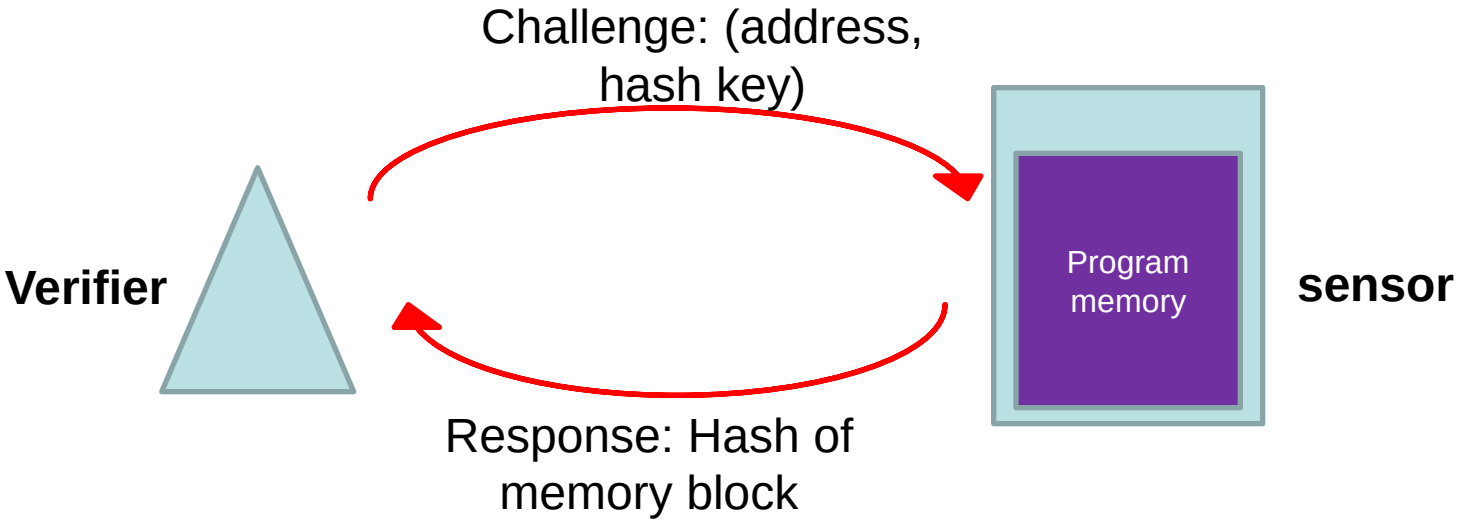
- Assume that $\delta = 2$ and the receiver nodes are loosely time synchronized and know $K0$ (a commitment to the key chain).

- Packets $P1$ and $P2$ sent in interval 1 contain a MAC with key $K1$. Packet $P3$ sent in interval 2 contains a MAC with key $K2$.

- So far, a receiver cannot authenticate any packets yet. Assume that packets $P4$ and $P5$ are all lost, as well as the packet that discloses key $K1$, so the receiver still cannot authenticate $P1$, $P2$, or $P3$. In interval 4 the base station broadcasts key $K2$, which the node authenticates by verifying $K0 = F(F(K2))$. The node derives $K1 = F(K2)$, so it can authenticate packets $P1$, $P2$ with $K1$, and $P3$ with $K2$

Code Attestation [Ref. 18, 20]

- The basic idea of code attestation is to compare the "fingerprint" or "digest" of a memory block at an address specified by the verifier through the challenge-response mechanism.
  - The verifier can be centralized (e.g., a base station) or distributed (e.g., a neighbor).
  - If a node is detected compromise, code can be recovered through code update.



If the response received is the same as the hash of memory block computed by the verifier, then the sensor is clean

Code Attestation

Objective: Improve sensor reliability or prolong sensor lifetime

Sensor failure definition:

- It is compromised but it is not detected before it returns incorrect sensor readings
- Its energy is exhausted

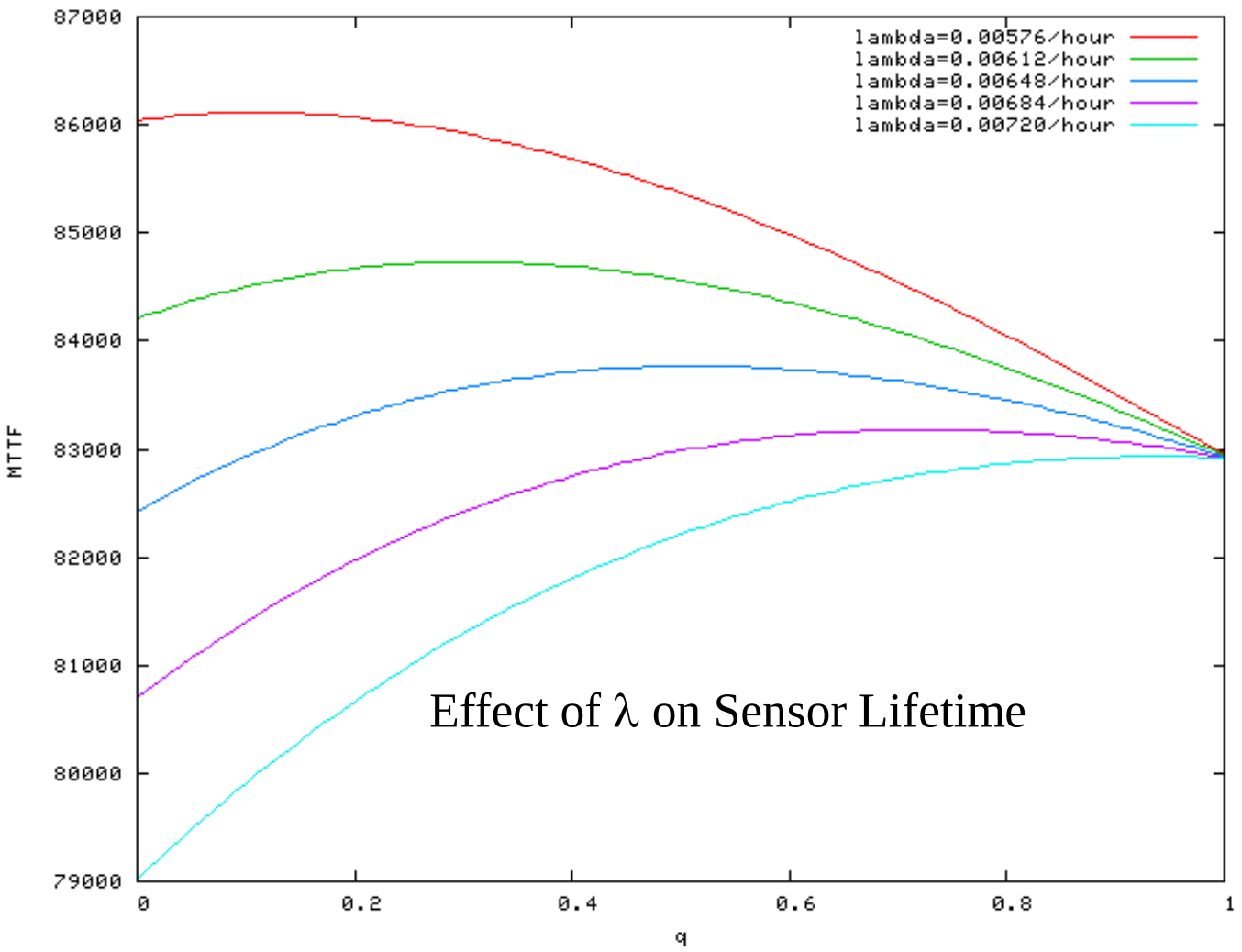
Code Attestation – How often?

q: probability of invoking code attestation at the periodic sensing time

Effect of compromise rate:

When the sensor compromise rate λ is sufficiently low (< 0.005 hour^{-1}), the optimal *q* is 0

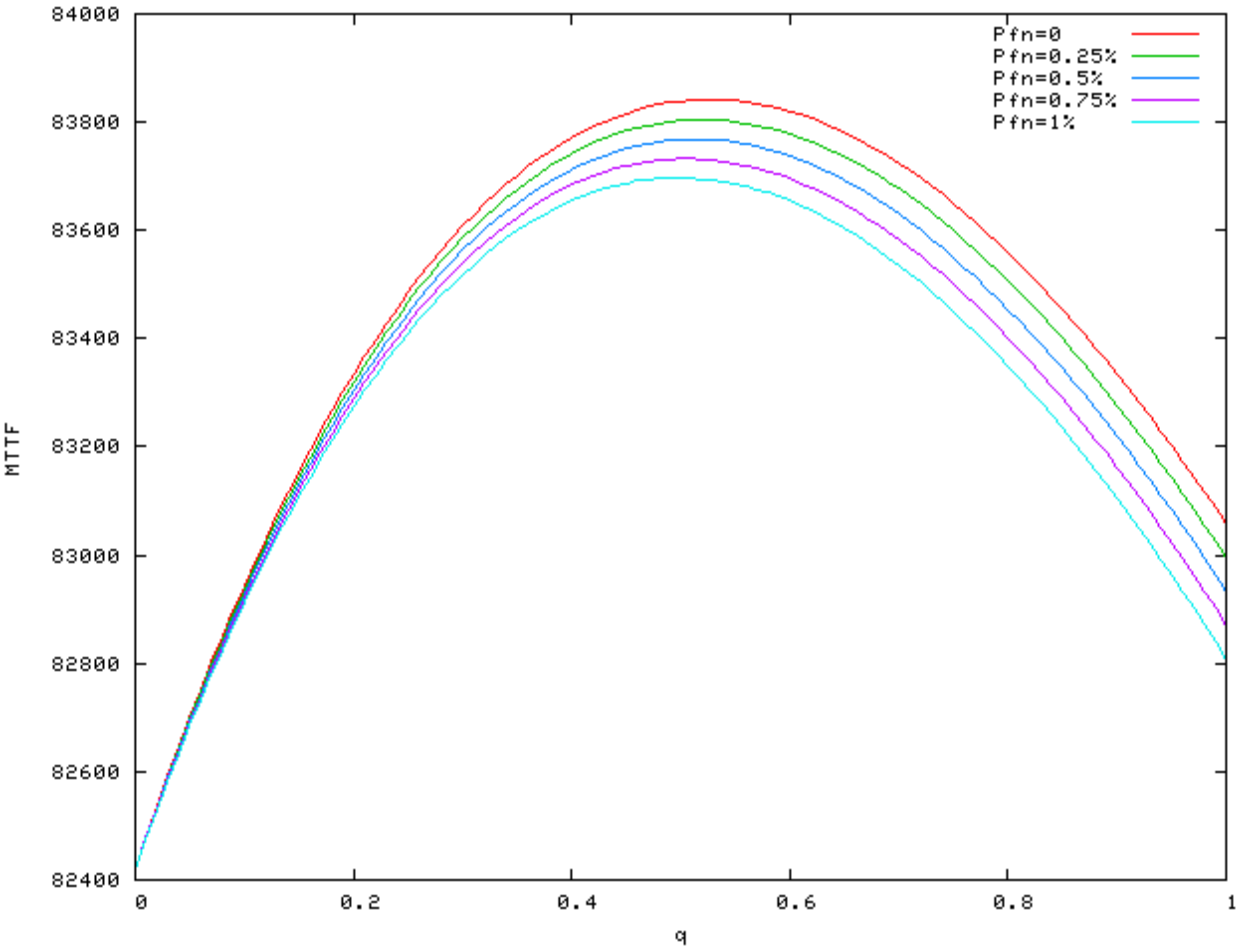
When λ is sufficiently large (>0.007 hour^{-1}), the optimal *q* is 1



Code Attestation – How Often (2)

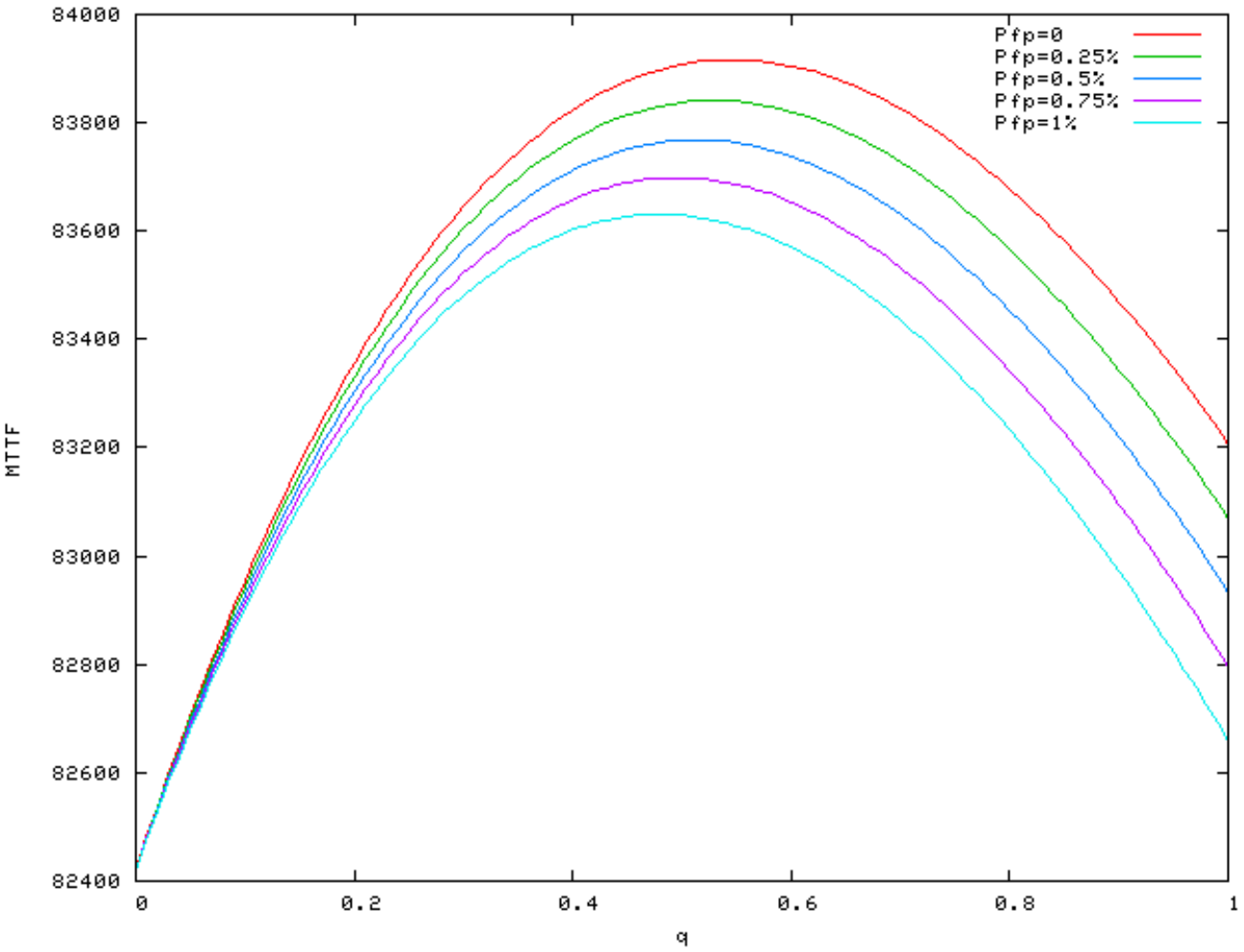
Effect of the IDS false negative probability:

The optimal *q* value for maximizing MTTF decreases as the IDS false negative probability P_{fn} of code attestation increases, because code attestation may misidentify a bad node as a good node and still could cause sensor failure



Effect of P_{fn} on Sensor Lifetime
(E_s=0.02 mjoule, E_c=0.01 mjoule, E_r =0.02 mjoule, λ=0.00648 hour^{-1}, T=4 s, P_{fp}=0.5%).

Code Attestation – How Often (3)



The optimal q also decreases as the IDS false positive probability P_{fp} increases, because code attestation can misidentify a good node as a bad node, thus causing unnecessary failure recovery and extra energy consumption

Effect of P_{fp} on Sensor Lifetime
(E_s=0.02 mjoule, E_c=0.01 mjoule, E_r =0.02 mjoule, λ=0.00648 hour^{-1}, T=4 s, P_{fn}=0.5%).

References

Chapters 12-16, F. Adelstein, S.K.S. Gupta, G.G. Richard III and L. Schwiebert, *Fundamentals of Mobile and Pervasive Computing*, McGraw Hill, 2005.

Other References:

17. **M. Steiner, G. Tsudik and M. Waidner, "Key agreement in dynamic peer groups," *IEEE Trans. Parallel and Distributed Systems*, Vol. 11, No. 8, Aug. 2000, pp. 769-780.**
18. **E. Shi and A. Perrig, "Designing secure sensor networks," *IEEE Wireless Communications,* Vol. 11, No. 6, 2004, pp. 38-43.**
19. **A. Perrig, R. Szewczyk, J.D. Tygar, V. Wen and D.E. Culler, "SPINS: Security protocols for sensor networks," *Wireless Networks*, Vol. 8, 2002, pp. 521-534.**
20. **I.R. Chen, Y. Wang and D.C. Wang, "Reliability of Wireless Sensors with Code Attestation for Intrusion Detection," Information Processing Letters, 2010.**