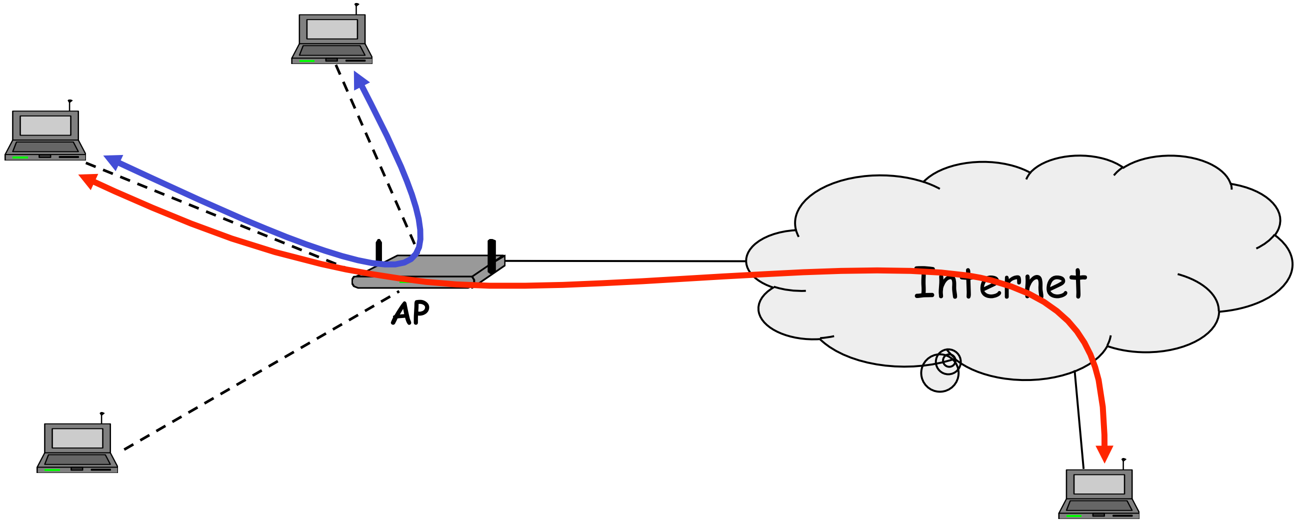
Securing 802.11 (WiFi) networks

Claude Castelluccia- INRIA

IEEE 802.11 Protocol Primer

802.11 protocol

- 802.11 is a wireless LAN standard
  - 2 frequency bands: 2.4GHz and 5GHz.
- This is the most widely deployed wireless LAN technology.
- It is composed of a Access point(s) and mobile devices



802.11 protocol

- IEEE802.11 has two modes of operations:
  - Infrastructure mode (ESS)
    - All communications (even between two mobile devices of same network) are through the access point (AP)
    - The AP is coordinating the communications.
  - Infrastructure-less mode (IBSS) or ad-hoc mode
    - Mobile devices (STA) can communicate directly
    - No AP is necessary!
  - Most networks operate in infrastructure mode…

Basics of Operation in Infrastructure Mode

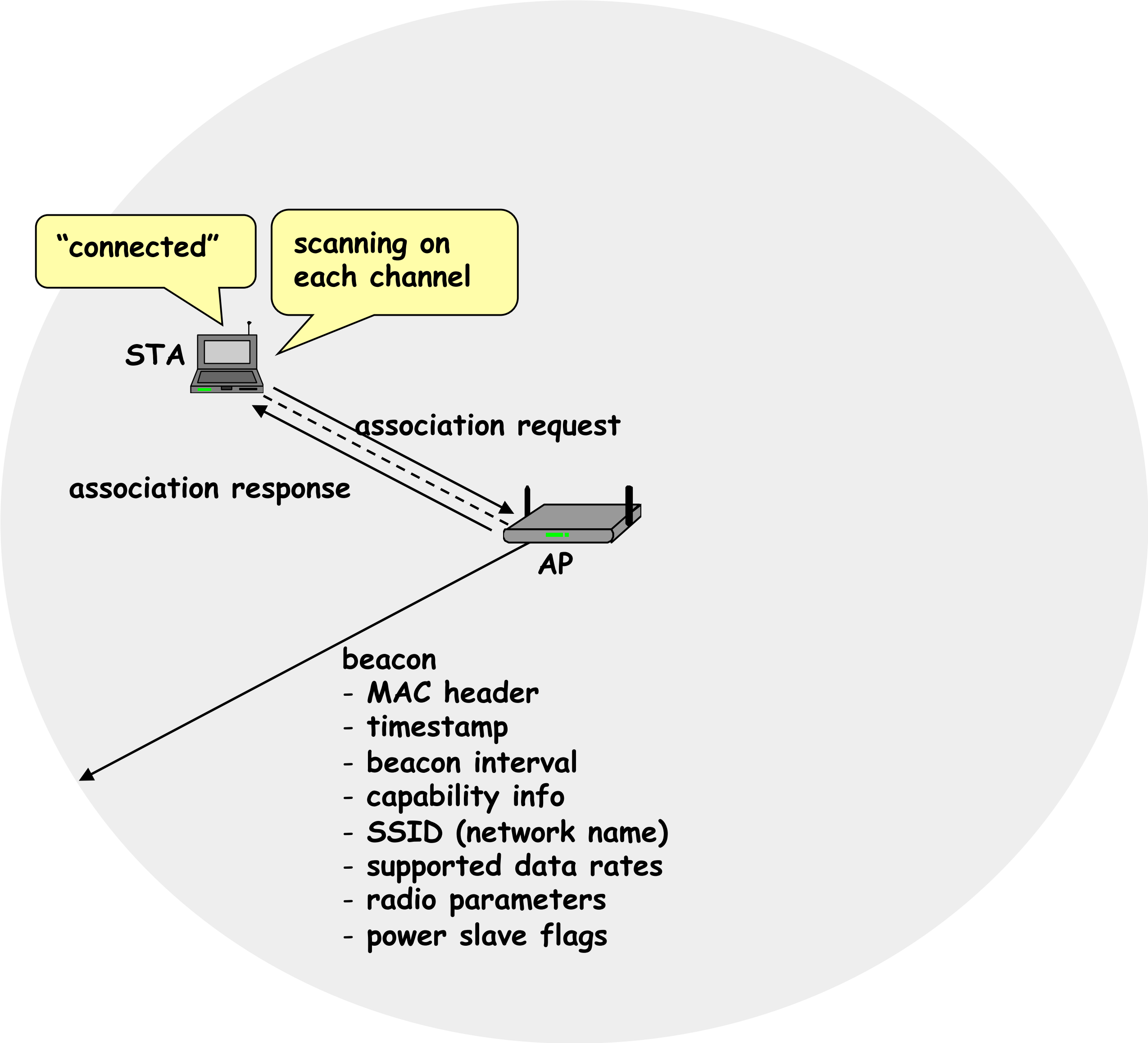
- Let's consider an access point (AP) and a mobile station (STA)
- The AP uses several radio frequencies (called ***channels***) to communicate with the STAs
- The AP advertises its presence, on each channel, by transmitting short wireless messages at regular intervals (10 times a second)
  - These messages are called ***beacons***
- The STA must then tune into each channel and listen for beacon messages
  - This process is called ***scanning***
  - This process can be accelerated by probing (i.e. the STA sends a request)
- The STA may discover severals APs in a large network and must decide to which it intends to connect (based on signal strength, security policy, roaming agreement, SSID,…)

Basics of Operation in Infrastructure Mode (2)

- When the STA is ready to connect to the AP, it first sends an (***authenticate) request message*** to the AP.
- The AP immediately responds by sending an ***authenticate response message*** indicating acceptance *(no security is used in this example)*
- The STA sends an ***association request*** message
- The AP responds with an ***association reply*** message…
- The **STA is then connected** (associated in the term) and can send data through the AP!

- There are 3 types of messages:
  - **Control**: short msgs that tell devices when to start or stop transmission
  - **Management**: messages use to negotiate and control the association.
  - **Data**: messages that contain the data…

Introduction to WiFi



"connected"

scanning on each channel

STA

association request

association response

AP

beacon
- MAC header
- timestamp
- beacon interval
- capability info
- SSID (network name)
- supported data rates
- radio parameters
- power slave flags

IEEE 802.11 Security Solutions

Access mechanisms

open network (no protection)

- assumption: there are no unauthorized users in the range of the network
- problems: range is hard to determine (unpredictable propagation of the signals, directional antennas, ...)

closed network

- using SSIDs for authentication (Service Set Identifier)
- MAC filtering
- shared keys
- authentication servers

MAC filtering

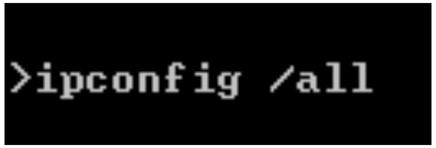
- MAC address filtering
  - only devices with certain MAC addresses are allowed to associate
  - **needs pre-registration of all device at the AP**

- MAC can be sniffed and forged
  - **sent in clear text in each packet (can be sniffed)**
  - can be forged

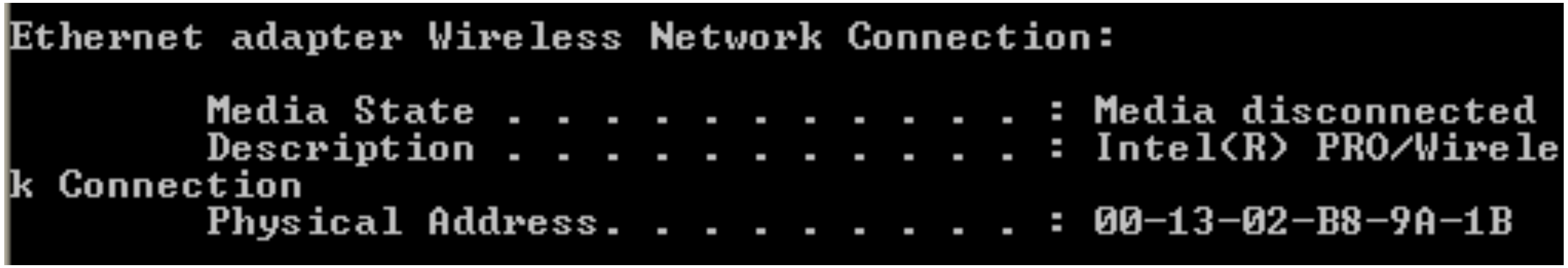
Device identification – MAC addresses

- “Hardcoded” addresses in WiFi cards (“unique device identifiers”)
  - *all* devices have different addresses

- Concept taken over from ethernet addresses



```
>ipconfig /all
```



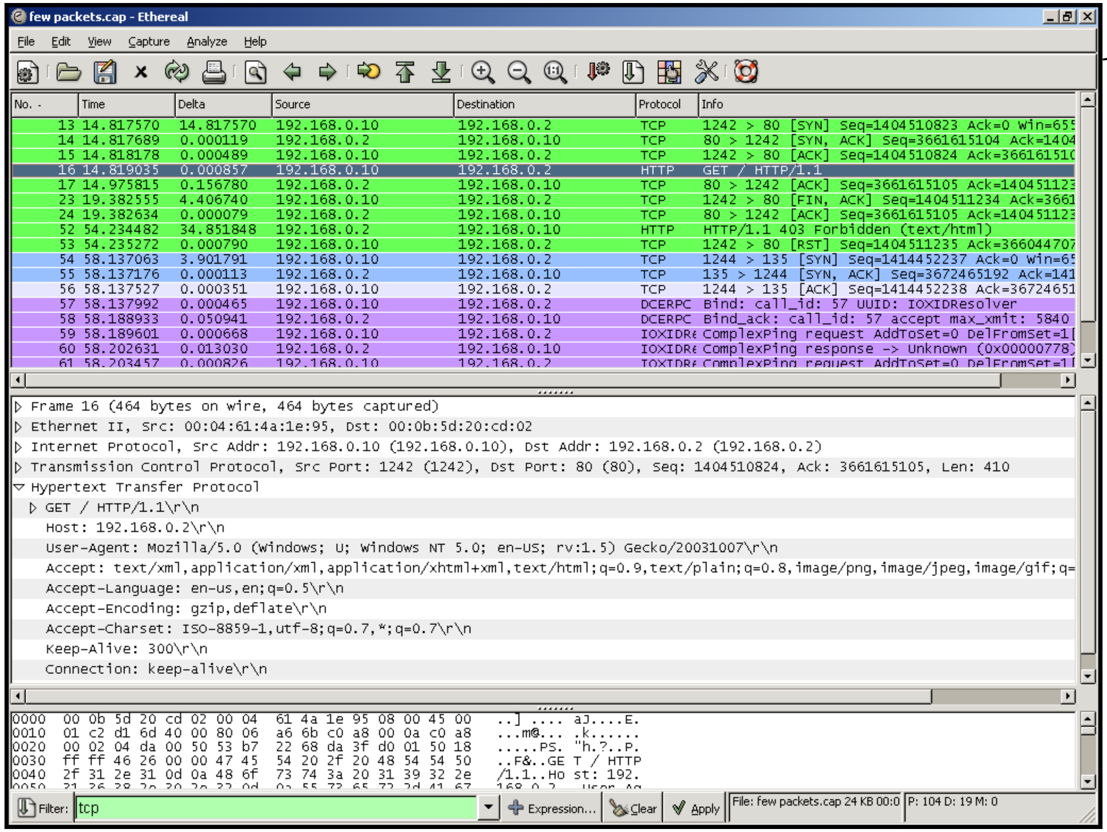
```
Ethernet adapter Wireless Network Connection:

        Media State . . . . . . . . . . . : Media disconnected
        Description . . . . . . . . . . . : Intel(R) PRO/Wirele
k Connection
        Physical Address. . . . . . . . . : 00-13-02-B8-9A-1B
```

3 simple steps for overcoming MAC filtering

1. Put your card in promiscuous mode (accepts all packets).

2. Sniff the traffic and find out which MAC addresses are accepted by the AP



few packets.cap - Ethereal
File Edit View Capture Analyze Help

```
No. -  Time       Delta      Source         Destination    Protocol Info
    13 14.817570  14.817570  192.168.0.10   192.168.0.2    TCP      1242 > 80 [SYN] Seq=1404510823 Ack=0 Win=655
    14 14.817689  0.000119   192.168.0.2    192.168.0.10   TCP      80 > 1242 [SYN, ACK] Seq=3661615104 Ack=1404
    15 14.818178  0.000489   192.168.0.10   192.168.0.2    TCP      1242 > 80 [ACK] Seq=1404510824 Ack=36616151C
    16 14.819035  0.000857   192.168.0.10   192.168.0.2    HTTP     GET / HTTP/1.1
    17 14.975815  0.156780   192.168.0.2    192.168.0.10   TCP      80 > 1242 [ACK] Seq=3661615105 Ack=140451123
    23 19.382555  4.406740   192.168.0.10   192.168.0.2    TCP      1242 > 80 [FIN, ACK] Seq=1404511234 Ack=3661
    24 19.382634  0.000079   192.168.0.2    192.168.0.10   TCP      80 > 1242 [ACK] Seq=3661615105 Ack=140451123
    52 54.234482  34.851848  192.168.0.2    192.168.0.10   HTTP     HTTP/1.1 403 Forbidden (text/html)
    53 54.235272  0.000790   192.168.0.10   192.168.0.2    TCP      1242 > 80 [RST] Seq=1404511235 Ack=366044707
    54 58.137063  3.901791   192.168.0.10   192.168.0.2    TCP      1244 > 135 [SYN] Seq=1414452237 Ack=0 Win=65
    55 58.137176  0.000113   192.168.0.2    192.168.0.10   TCP      135 > 1244 [SYN, ACK] Seq=3672465192 Ack=141
    56 58.137527  0.000351   192.168.0.10   192.168.0.2    TCP      1244 > 135 [ACK] Seq=1414452238 Ack=36724651
    57 58.137992  0.000465   192.168.0.10   192.168.0.2    DCERPC   Bind: call_id: 57 UUID: IOXIDResolver
    58 58.188933  0.050941   192.168.0.2    192.168.0.10   DCERPC   Bind_ack: call_id: 57 accept max_xmit: 5840
    59 58.189601  0.000668   192.168.0.10   192.168.0.2    IOXIDRε  ComplexPing request AddToSet=0 DelFromSet=1[
    60 58.202631  0.013030   192.168.0.2    192.168.0.10   IOXIDRε  ComplexPing response -> Unknown (0x00000778)
    61 58.203457  0.000826   192.168.0.10   192.168.0.2    IOXIDRε  ComplexPing request AddToSet=0 DelFromSet=1[
```

```
▷ Frame 16 (464 bytes on wire, 464 bytes captured)
▷ Ethernet II, Src: 00:04:61:4a:1e:95, Dst: 00:0b:5d:20:cd:02
▷ Internet Protocol, Src Addr: 192.168.0.10 (192.168.0.10), Dst Addr: 192.168.0.2 (192.168.0.2)
▷ Transmission Control Protocol, Src Port: 1242 (1242), Dst Port: 80 (80), Seq: 1404510824, Ack: 3661615105, Len: 410
▽ Hypertext Transfer Protocol
  ▷ GET / HTTP/1.1\r\n
    Host: 192.168.0.2\r\n
    User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.0; en-US; rv:1.5) Gecko/20031007\r\n
    Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,image/jpeg,image/gif;q=
    Accept-Language: en-us,en;q=0.5\r\n
    Accept-Encoding: gzip,deflate\r\n
    Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7\r\n
    Keep-Alive: 300\r\n
    Connection: keep-alive\r\n
```

```
0000  00 0b 5d 20 cd 02 00 04  61 4a 1e 95 08 00 45 00   ..] .... aJ....E.
0010  01 c2 d1 6d 40 00 80 06  a6 6b c0 a8 00 0a c0 a8   ...m@... .k......
0020  00 02 04 da 00 50 53 b7  22 68 da 3f d0 01 50 18   .....PS. "h.?..P.
0030  ff ff 46 26 00 00 47 45  54 20 2f 20 48 54 54 50   ..F&..GE T / HTTP
0040  2f 31 2e 31 0d 0a 48 6f  73 74 3a 20 31 39 32 2e   /1.1..Ho st: 192.
```

Ethereal/wireshark

3. Change your MAC address (need a card that can do that)

```
# ifconfig ath0 hw ether <mac address of C>
```

SSID-based access control

- SSID = Service Set IDentifier (network name)
- a 32-character unique identifier
- attached to the header of packets
- acts as a password when a mobile device tries to connect to the WLAN
- SSID differentiates one WLAN from another
- all devices attempting to connect to a specific WLAN must use the same SSID

SSID-based access control

- SSIDs can be sniffed *(e.g. http://www.ethereal.com)*
  - advertised by the APs
  - contained in SSID response frames

- Overcoming SSID-based access control
  - Sniff SSID (either sent by the clients or advertised by the AP)
  - Set your SSID to the same value ...

- MAC/SSID access control: not a bad protection from unskilled neighbors (much better than no authentication/protection)

IEEE 802.11 WEP

Protected access using WEP

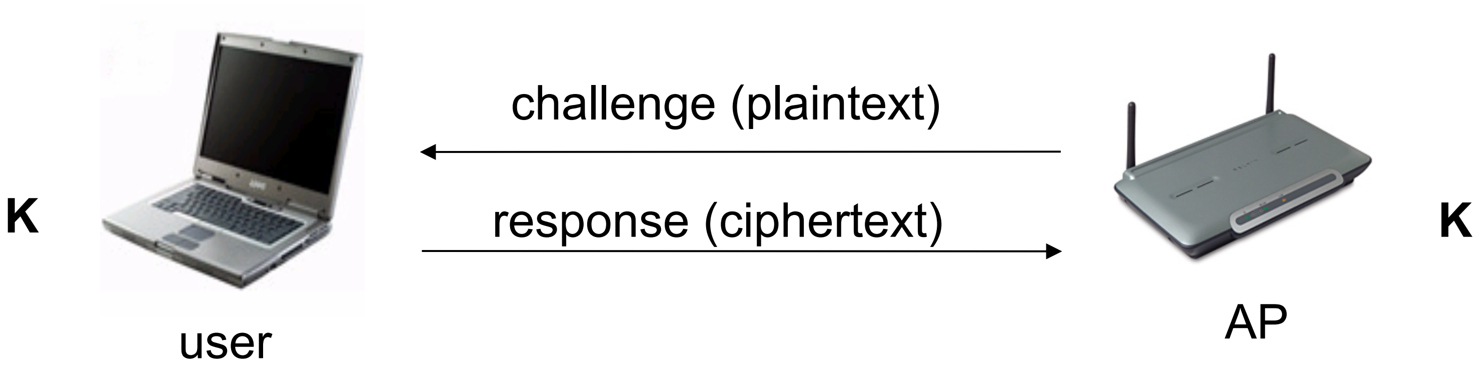
- WEP = Wired Equivalent Privacy
  - part of the IEEE 802.11 specification

- Goal
  - make the WiFi network *at least as secure as a wired LAN* (that has no particular protection mechanisms)
  - WEP has never intended to achieve strong security
  - (at the end, it hasn't achieved even weak security)

- Services
  - access control to the network
  - message confidentiality
  - message integrity

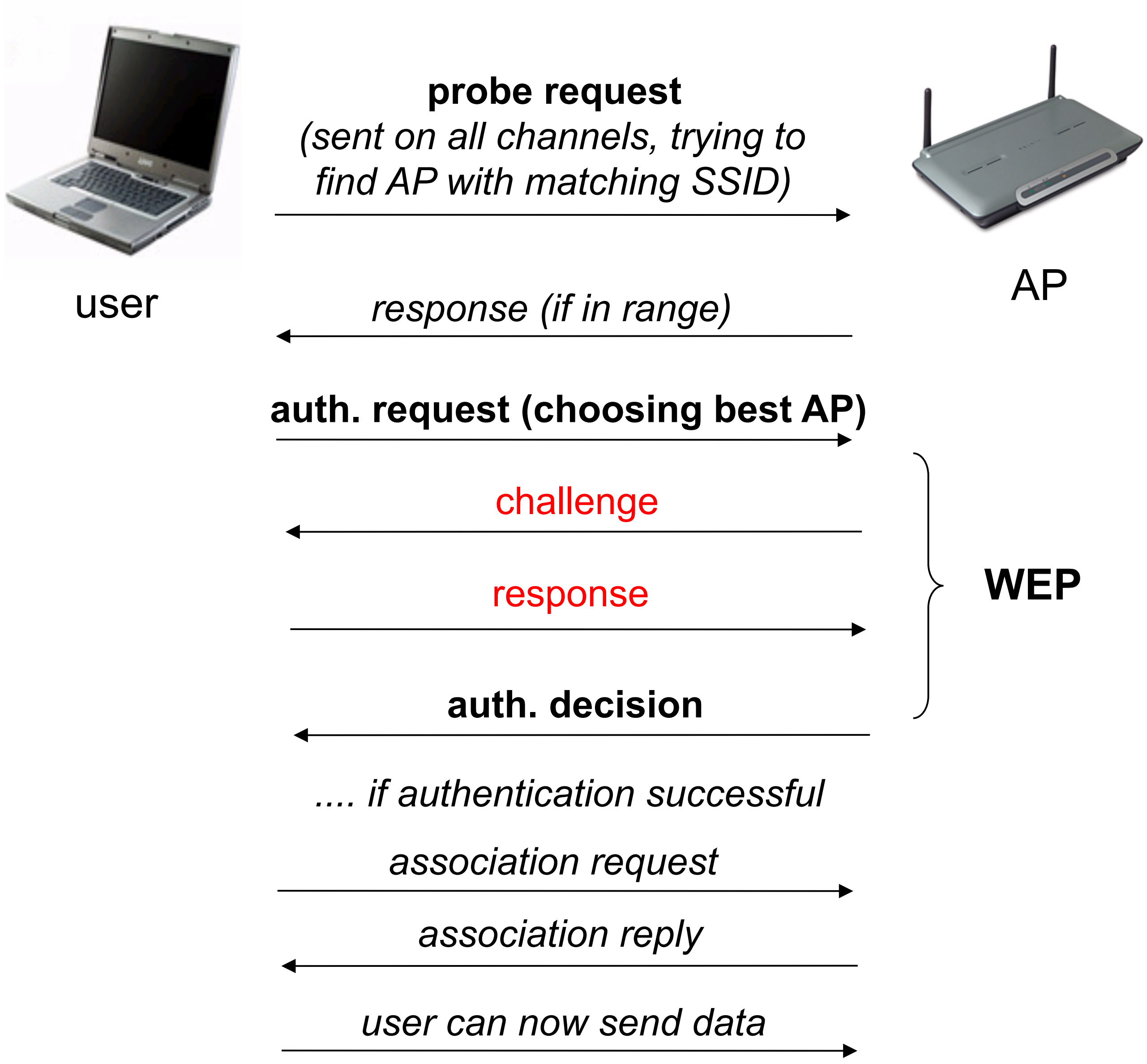
WEP Overview

- Based on a shared key between the station and the AP (40 bit or 104 bit)
- Based on the RC4 symmetric stream cipher
- 24-bit Initialization Vector (IV)



- The payload of every packet is encrypted (confidentiality) with its CRC value (integrity)
- Authentication through ‘standard’ challenge-response authentication protocol ... using the shared key ...

WEP-authentication



user

AP

probe request
(sent on all channels, trying to find AP with matching SSID)

response (if in range)

auth. request (choosing best AP)

challenge

response

WEP

auth. decision

.... if authentication successful

association request

association reply

user can now send data

Authentication Protocol

- Goal: the base station verifies that a client joining the network really knows the shared secret key k.

- The base station sends a challenge string to the client
  B->A: CHAL
- The client sends encrypted challenge:
  A -> B: v, <CHAL,c(CHAL)> $\oplus$ RC4(v,k)

- The base station checks if the challenge is correctly encrypted, and if so, accepts the client.
  - i.e. An 802.11 receiver will accept a packet if, after decryption, it contains a correct checksum of the plaintext.
-

The WEP Protocol: Confidentiality and Integrity

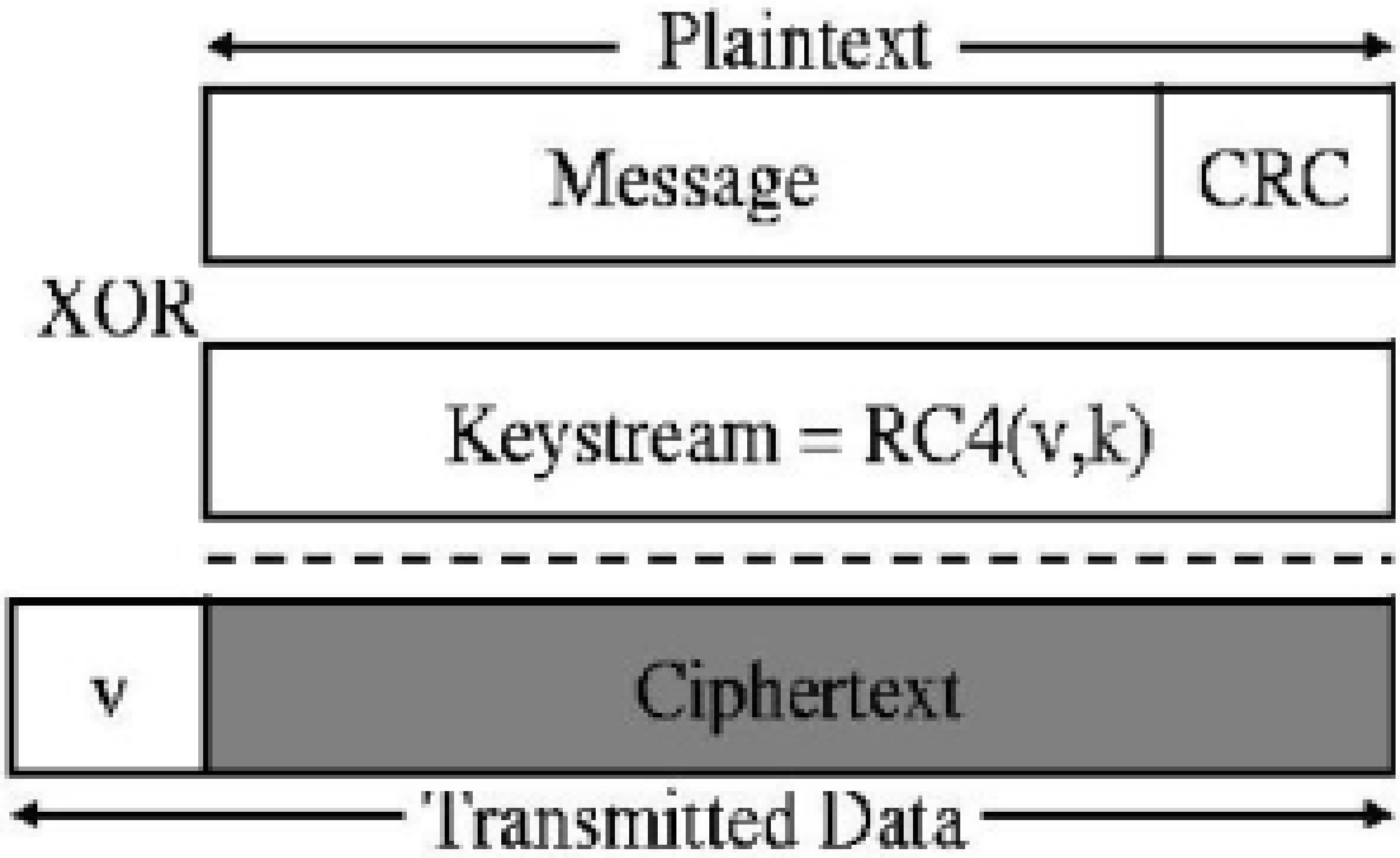
- Sender and receiver share a secret key k.
- Two classes of WEP implementation:
  - classic WEP as documented in standard (40-bit key)
  - extended version developed by some vendors (128-bit key)
- In order to transmit a message M:
  P = <M, c(M)>

  pick IV v and generate RC4(v,k)

  $C = P \oplus RC4(v,k)$
  A -> B: v, $(P \oplus RC4(v,k))$

IMPORTANT (for later): v is selected by A!

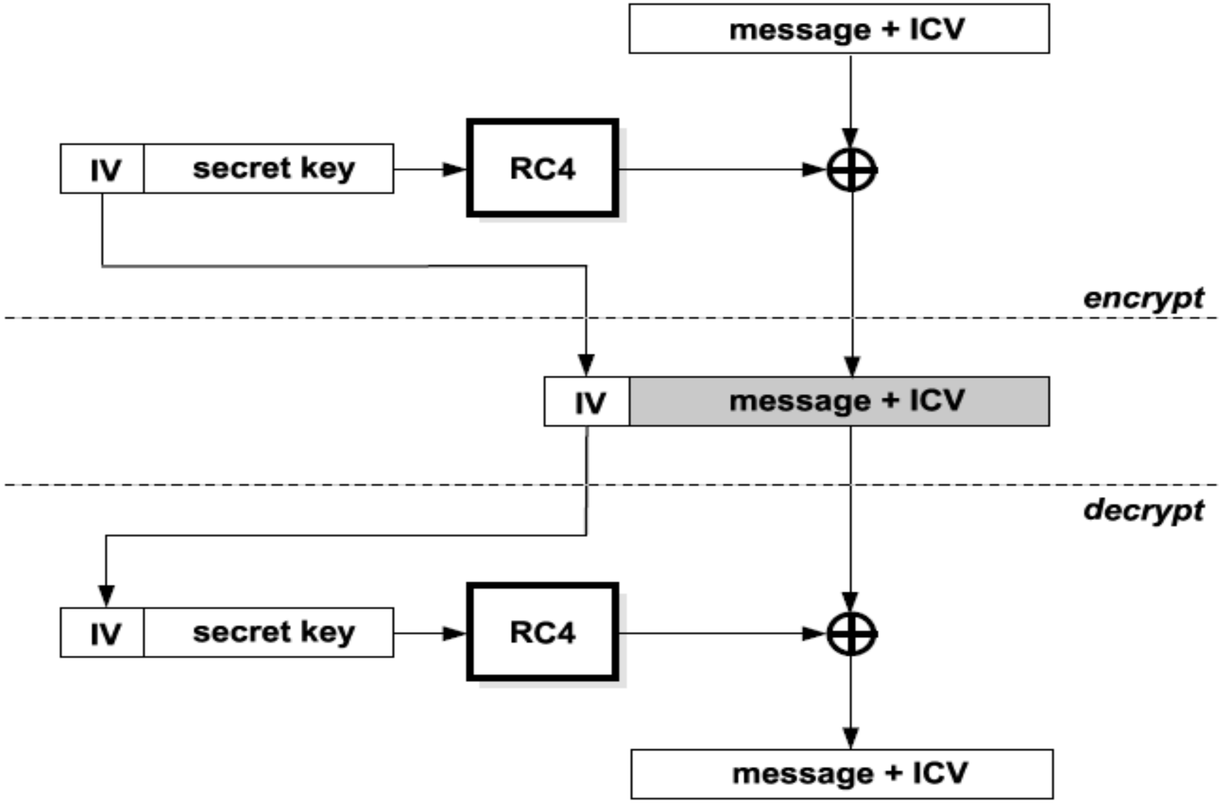
WEP, Pictorially



WEP (cont.)

- Upon receipt:
  generate RC4(v,k)
  $P = C \oplus RC4(v,k) =$
  $P \oplus RC4(v,k) \oplus RC4(v,k)$
  check if c=c(M)
- If so, accept the message M as being the one transmitted

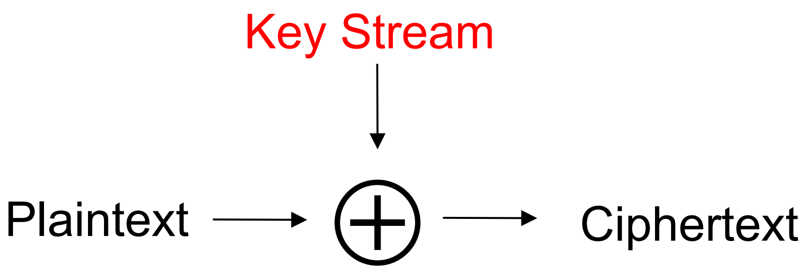
WEP confidentiality/integrity protection (1)



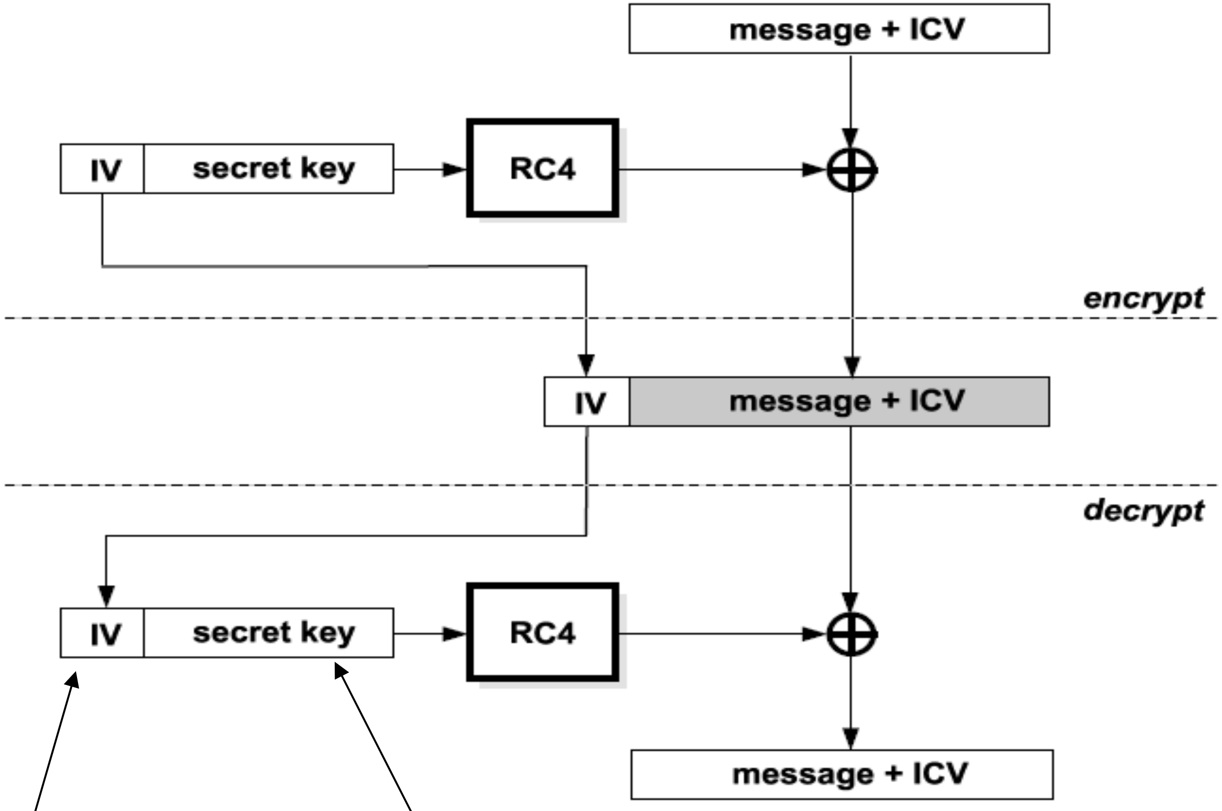
- RC4 Generates a key stream of a desired length from the key
- The key stream is XORed with plaintext data
- The result is ciphertext data

WEP confidentiality/integrity protection (2)

- RC4 is a stream cipher
  - given a short input key, it produces a pseudorandom sequence (key stream)
  - the key stream is always the same for the same key
  - A different IV (initialization vector) is therefore needed for each message!
    - The key stream is initialized for each message…
    - …so it is tolerant to packet loss…
- The output of the key stream is XORed with the plaintext to obtain a ciphertext:



WEP confidentiality/integrity protection (bis)



Different for each packet
(24 bits)

Fixed
(40 or 104 bits)

WEP Flaws: Confidentiality

WEP Flaws- Confidentiality

- The keystream for WEP is RC4(IV,k).
- k is a fixed shared secret, that changes rarely, if ever (in many setups, every user shares the same k).
- If two packets ever get transmitted with the same value of IV, you reuse the keystream.
- Two-time pad attack…

Steam Cipher’s "Two-Time Pad" Problem

- You must **never** encrypt two messages with the same keystream S.
- Suppose P1 and P2 are both encrypted with the same S.
  - Then C1 = P1 $\oplus$ K and C2 = P2 $\oplus$ K
  - C1 $\oplus$ C2 = P1 $\oplus$ K $\oplus$ P2 $\oplus$ K = P1 $\oplus$ P2
- So the adversary learns the XOR of two plaintexts!
- Usually, just knowing the XOR of two plaintexts is enough to recover them.
  - For example, if the adversary knows P1, he can get P2…

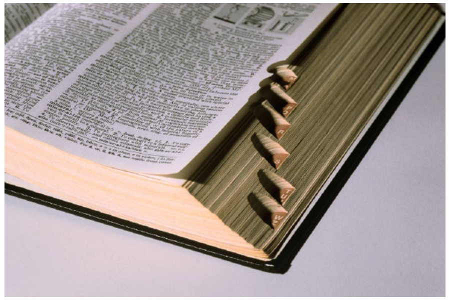
The "Two-Time Pad" Problem

- Note that if an adversary knows a (P1,C1) pair
  - then S = C1 xor P1
  - the adversary can sent C2 = M2 xor S
    - C2 will be successful decrypted by Bob!
- As a result in RC4,
  - The key must never be used twice!
- If not, many attacks are possible

- In WEP, the key between the AP and STA is fixed
  - Therefore WEP uses a different IV (initialization vector) per message

WEP Flaws- Confidentiality (2)

- How many possible values of IV are there?
- IV only occupies 24 bits of the header = at most there are 2^24 (about 16 million of IV).
- After 16 million packets, you have to repeat one!
- It is even worse than that!
  - All the 802.11 cards reset their IV counter to 0 every time they were activated, and incremented by 1 for each packet transmitted.
- This means that low IV values get reused at the beginning of every wireless session.
- This makes collisions much more common.



WEP Flaws- Confidentiality (3)
Decryption Dictionaries

- Adversary knows both the C and the P for some packets encrypted with a given IV v.
- Easy if he knows the P (pings, ARP request/reply, or spam email!).
- RC4(k,v) = P $\oplus$ C
  - Note: no need to know the value of the shared secret k.
- Store keystream in a table, indexed by v.

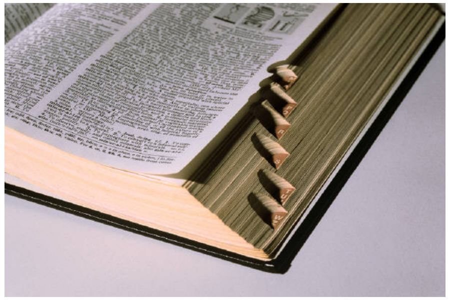
v1: RC4(k, v1)

v2: RC4(k, v2)

v3: RC4(k, v3)

....

- Table is at most 1500 * 2^24 bytes = 24 GB
  - 1500 is the max. frame size.



WEP Flaws- Confidentiality (3)
Decryption Dictionaries

- Next time a packet with an IV stored in the table passes by, look up the keystream, XOR it against the packet, and read the data!

- If the cards that are being used have the IV-reset-to-0 property, then most IV's will be small, and the dictionary will be even smaller!

WEP Confidentiality?
-broken!
-easy to decrypt messages!!

WEP Flaws: Authentication and Access Control

Authentication Protocol

- Goal: the base station verifies that a client joining the network really knows the shared secret key k.

- The base station sends a challenge string to the client
  AP ->Client: CHAL
- The client sends encrypted challenge:
  Client -> AP : v, <CHAL,c(CHAL)> $\oplus$ RC4(v,k)

- The base station AP checks if the challenge is correctly encrypted, and if so, accepts the client.
  - i.e. An 802.11 receiver will accept a packet if, after decryption, it contains a correct checksum of the plaintext.
-

WEP flaws – Authentication and access control

- Flaw1: authentication is one-way only
  - AP is not authenticated to STA
  - STA may associate to a rogue AP
- Flaw2: the same shared secret key is used for authentication and encryption
  - weaknesses in any of the two protocols can be used to break the key
  - different keys for different functions are desirable
- Flaw3: no session key is established during authentication
  - access control is not continuous
  - once a STA has authenticated and associated to the AP, an attacker send messages using the MAC address of STA
  - correctly encrypted messages cannot be produced by the attacker, but replay of STA messages is still possible

WEP flaws–Authentication and access control (2)

- Flaw4: The checksum algorithm used is CRC-32.
  - CRC's are used to detect random errors; they are useless against malicious errors ☹
  - There is already a CRC at a lower layer of the protocol to detect random bit errors in transmission.

- 3 attacks
  - Message modification
  - Message injection
  - Authentication spoofing

Message Modification Attack

- GOAL:The adversary wants to modify legitimate messages

- Make use of CRC-32 properties
  - It is independent of the shared secret and the IV.
  - It is linear: $c(M \oplus D) = c(M) \oplus c(D)$
  - We can make a controlled modification and get unnoticed.
    - Assume a message M was transmitted, and the ciphertext was C and the IV was v (i.e. C and v are known to the adversary).

Message Modification Attack (2)

- $C = RC4(v,k) \oplus \langle M, crc32(M) \rangle$
- A -> B: $\langle v, C \rangle$
- Possible to find C′ s.t. it decrypts to M′ and $M' = M \oplus \Delta$
  Δ = arbitrarily chosen by the attacker
- A -> B: $\langle v, C' \rangle$
  - $C' = C \oplus \langle \Delta, crc32(\Delta) \rangle$
    $= RC4(v,k) \oplus \langle M, crc32(M) \rangle \oplus \langle \Delta, crc32(\Delta) \rangle$
    $= RC4(v,k) \oplus \langle M \oplus \Delta, crc32(M) \oplus crc32(\Delta) \rangle$
    $= RC4(v,k) \oplus \langle M \oplus \Delta, crc32(M \oplus \Delta) \rangle$
    $= RC4(v,k) \oplus \langle M', crc32(M') \rangle$
- Receiver checks that $c' = c(M')$
- Accept message M′ as the one transmitted!

Message Injection Attack

- GOAL: The adversary wants to inject its own messages!
- The adversary just needs to know a single plaintext, and its corresponding encrypted packet.
- A -> B: <v,C>
- $P \oplus C = P \oplus RC4(v,k) \oplus P = RC4(v,k)$

- Construct M' and P' = <M',c(M')>
- $C' = RC4(v,k) \oplus P'$
- "A" -> B: <v,C'>

- The IV is selected by A (i.e. the attacker in this case)!
  - So it can be replayed!
  - How to prevent this attack?

Authentication Spoofing

- GOAL: the Adversary want to connect to the network without the credential:
- The adversary just needs to know a single plaintext, and its corresponding encrypted packet.
- A -> B: <v,C>
- $P \oplus C = P \oplus RC4(v,k) \oplus P = RC4(v,k)$
- The base station sends a challenge string CHAL to the adversary.
- The adversary replies with
  $v, \langle CHAL, c(CHAL) \rangle \oplus RC4(v,k)$
- This is the correct response, so the base station accepts the adversary.
- Success even though he never did learn the value of k!

Some WEP cracking tools…

- Airedump to collect data…
- Aircrack-ng to crack the key…

WEP: what went wrong?

- IV is too short: 56-64 bits should be used…
- IV should be set to random value, not zero when reset!
- AP must keep a list of used value IV
  - Require some memory
  - => Key must be changed periodically…
- A MAC should be used for message integrity instead of a checksum
- Although RC4 is believed to be secure
  - It is easy to make mistakes
  - And build insecure systems
- This is often the case
  - Attacks are on the protocols not on the crypto. Algorithms…

– Note: there are also some attacks to recover the key k

IEEE 802.11i (WPA and WPA2)

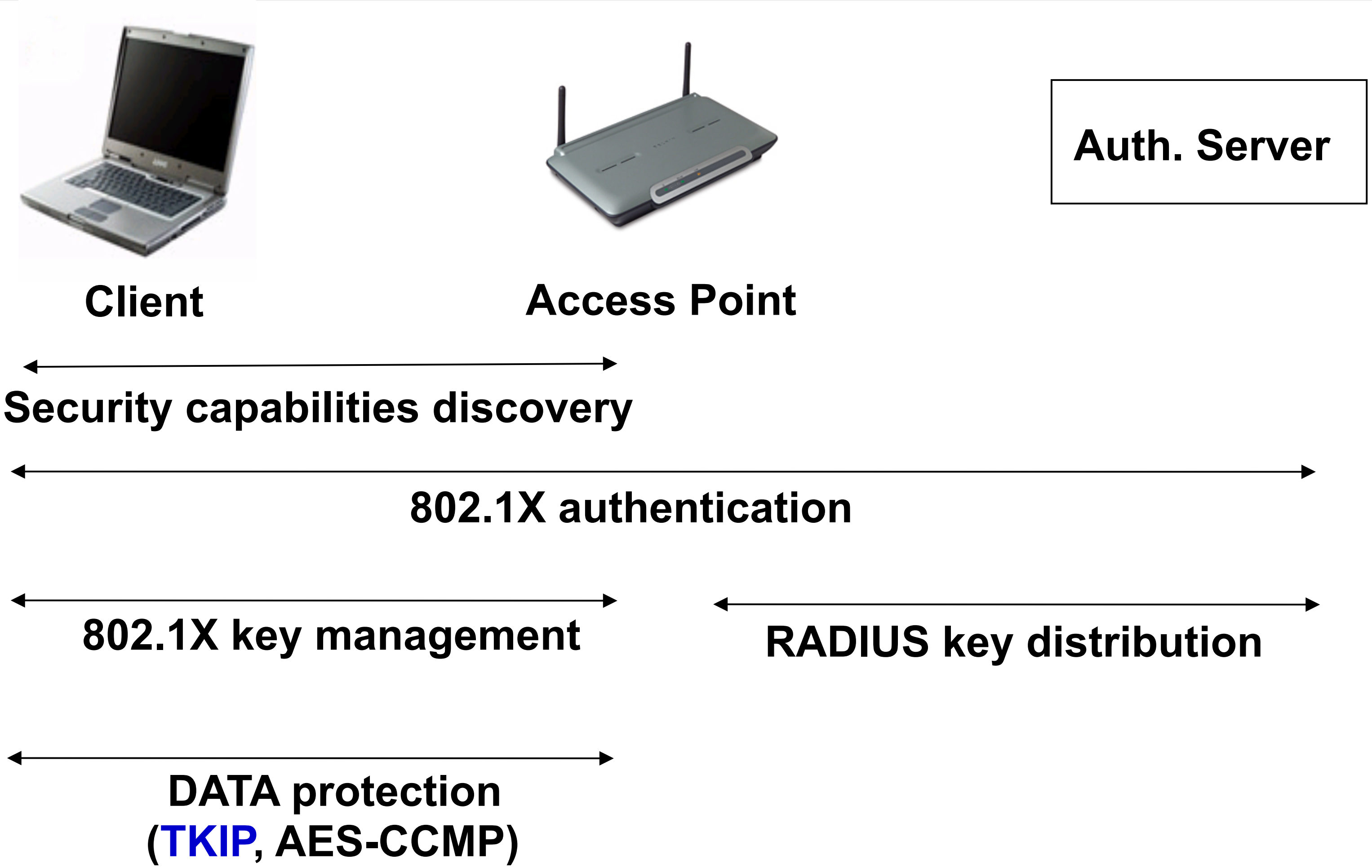
Overview of 802.11i

- after the collapse of WEP, IEEE started to develop a new security architecture → 802.11i
- main novelties in 802.11i wrt to WEP
  - access control model is based on 802.1X
  - flexible authentication framework (based on EAP)
    - authentication can be based on strong protocols (e.g., TLS)
  - authentication process results in a shared session key (which prevents session hijacking)
  - different functions (encryption, integrity) use different keys derived from the session key using a one-way function
  - integrity protection is improved
  - encryption function is improved
  - Pairwise key is enforced (in WEP, mobile uses same key)

Overview of 802.11i

- 802.11i defines the concept of RSN (Robust Security Network)
  - integrity protection and encryption is based on AES (in CCMP mode)
  - nice solution, but needs new hardware → cannot be adopted immediately
- 802.11i also defines an optional protocol called TKIP
  - integrity protection is based on **Michael**
  - encryption is based on RC4, but WEP’s problems have been avoided
  - ugly solution, but runs on old hardware (after software upgrade)
- **industrial names**
  - **TKIP → WPA (WiFi Protected Access)**
  - **RSN/AES-CCMP → WPA2**

802.11i overview



Client

Access Point

Auth. Server

Security capabilities discovery

802.1X authentication

802.1X key management

RADIUS key distribution

DATA protection
(TKIP, AES-CCMP)

802.11i overview

802.1x authentication

- Mutually authenticate Client and AS
- Generate Master Key as a side effect of authentication
- Generate Pairwise MK as an access authorization token
- Generate 4 keys for encryption/integrity

We will detail this phase later….

Data protection

- Provides data confidentiality and integrity
- 2 possibles schemes
    - TKIP (optional)
    - AES-CCMP

Let’s consider for now that the STA and AS share a master key (we’ll see later on how to do that)…and let’s look at TKIP...

TKIP/WPA1

Temporary Key Integrity Protocol (WPA)

- TKIP is a secure and available as an upgrade to WEP systems.
- The implementation of WEP almost depends on the hardware assist functions.
  - **RC4 is implemented on hard inside the card chip**
- The hardware assist functions in these earlier systems cannot support AES-CCMP.
  - Implementing AES-CCMP means changing the cards!
- **TKIP uses existing RC4 and upgrades the firmware.**
- Provides confidentiality and integrity.
- Ugly, but works with existing hardware.
- Usually used with manually configured master key (although 802.1x could be used)

Changes from WEP to TKIP/WPA

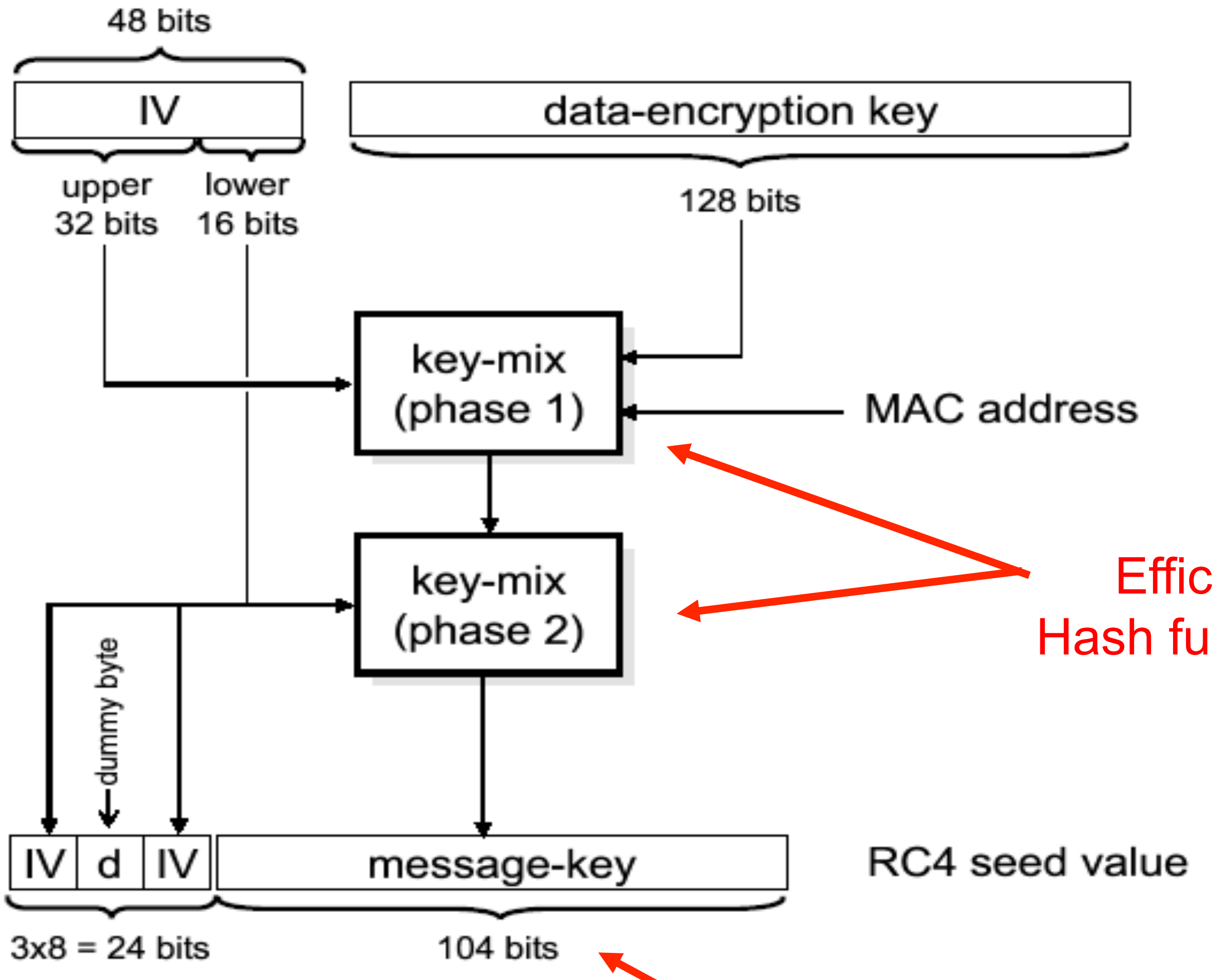
- IV selection and use: as counter (sequence no)
  - This reduces collisions (birthday attack paradox)
  - Prevent replay attack (AP just has to store latest value).
- Increase the size of IV: from 24 to 48, to avoid ever reusing the same IV.
- Per-packet key Mixing: change the key for every frame
- Message integrity: add a message integrity protocol.
- Key management: add a mechanism to distribute and change the broadcast keys.

The same encryption scheme than WEP is used: RC4

Per-Packet Key Mixing

- In WEP, there was a single key used for everything.
  - If compromised…everything is compromised!
- TKIP uses multiple keys
  - The session keys (renewed periodically)
  - derived from a single master key
- The per-packet key mixing derives a key for each packet
  - The session keys and master keys do not change at **every packet!**

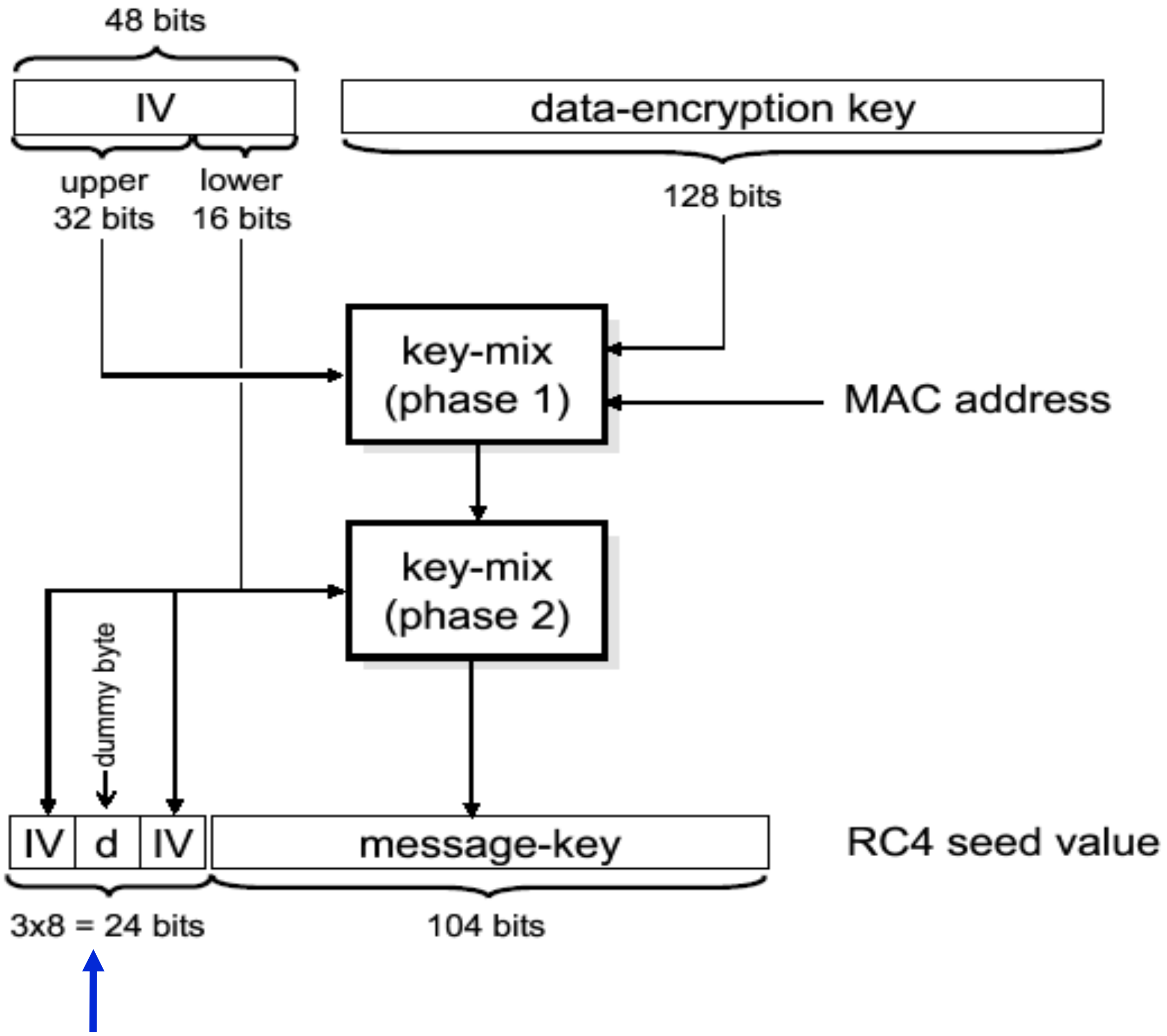
Per-Packet Key Mixing (2)



Efficient Hash functions

Different for each frame!

Per-Packet Key Mixing against weak key (3)



To avoid weak key

TKIP – Integrity

- Replaces ICV (Integrity Check Value) with MIC
  - MIC – Message Integrity Code or Message Authentication Code (MAC)
    - Computed using a non-reversible function and a secret key
  - Protects against bit-flip attacks by adding tamper-proof hash to messages
- TKIP uses MICHAEL a newly designed scheme
    - Can run on low power processor and without hardware support
    - Compute on 8-byte check value
    - Used with a secret key!
  - Part of firmware

Summary

- TKIP
  - uses RC4 → runs on old hardware
  - corrects WEP’s flaws
  - mandatory in WPA, optional in RSN (WPA2)
  - Temporary solution… but will probably be around for awhile ;-)

WPA2

802.11i/WPA1:
a comprehensive redesign of WiFi security

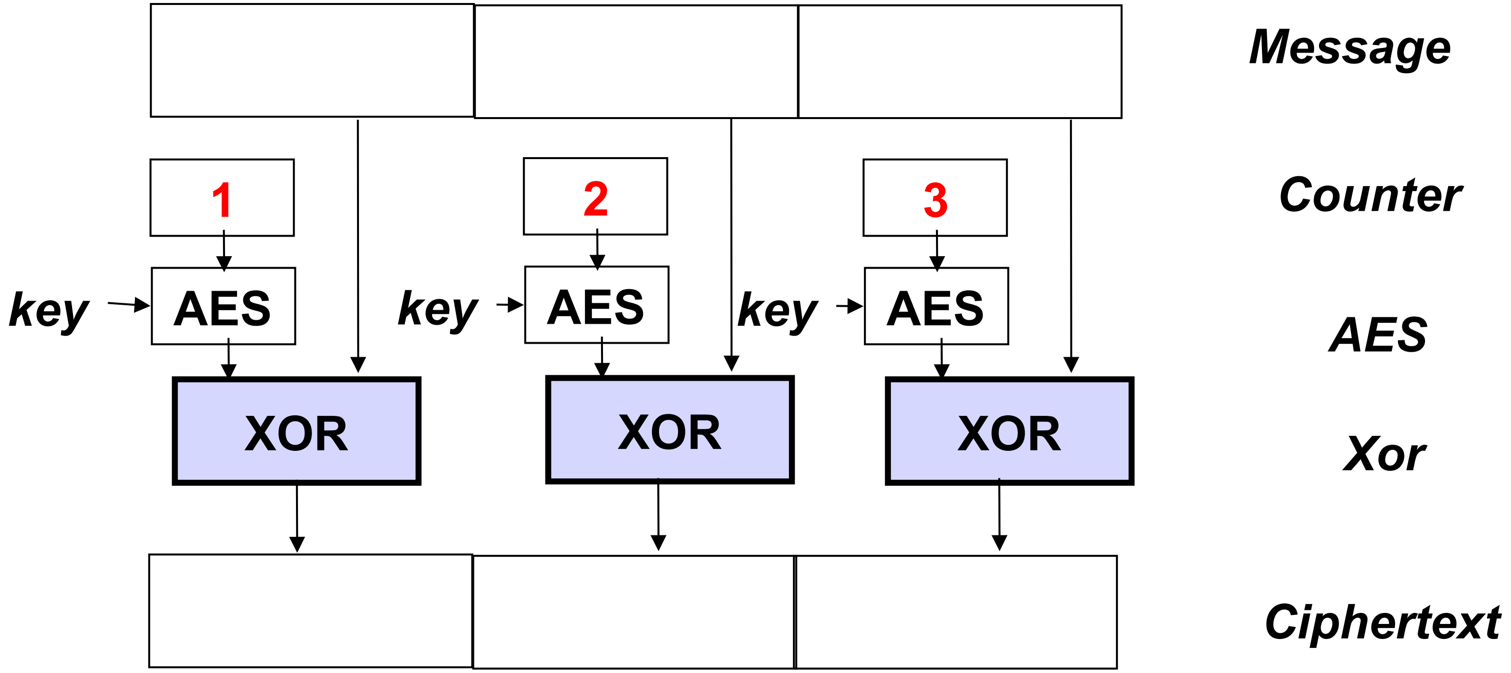
- Robust Security Network (RSN) for establishing secure communications
  - Uses 802.1x for authentication
  - Replaces TKIP
- AES replaces RC4 w/TKIP,
  - Counter Mode with Cipher Block Chaining (CCM) (CCM=counter mode + CBC MAC)
    - Counter mode for encryption
    - CBC-MAC provides data integrity/authentication
  - 128-bit keys, 48-bit IV
  - CCMP mandatory with RSN
  - Ensures data confidentiality and integrity
- Dubbed “WPA2” by WiFi Alliance

AES: Advance Encryption Standard

- Block Cipher:
  - message is decomposed into blocks
  - Each block is encrypted independently
  - Used the Rijndael Algorithm
  - Allows different block sizes and key sizes
    - 128, 192 and 256 bits.

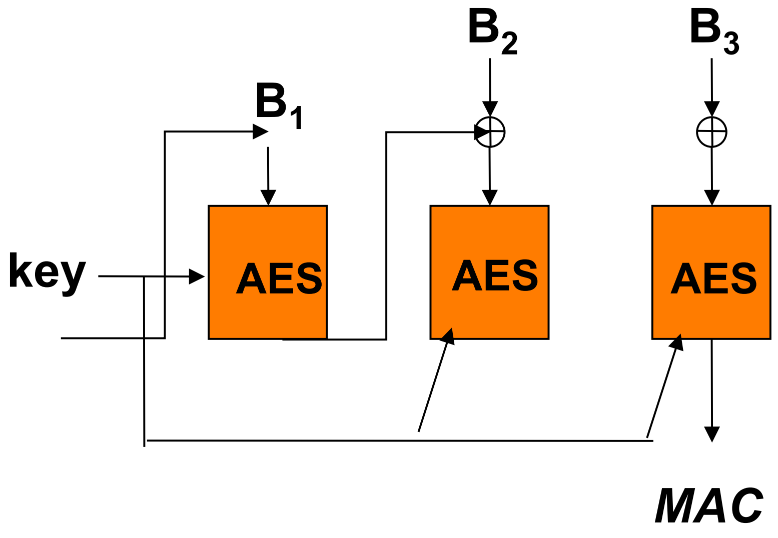
AES – Encryption: Counter Mode of Operation

- AES can be used differently: mode of operation
- Counter Mode of Operation
  - Message divided into blocks
  - A counter i is encrypted
  - E(i) ⊕ E(Bi) produces the encrypted message block

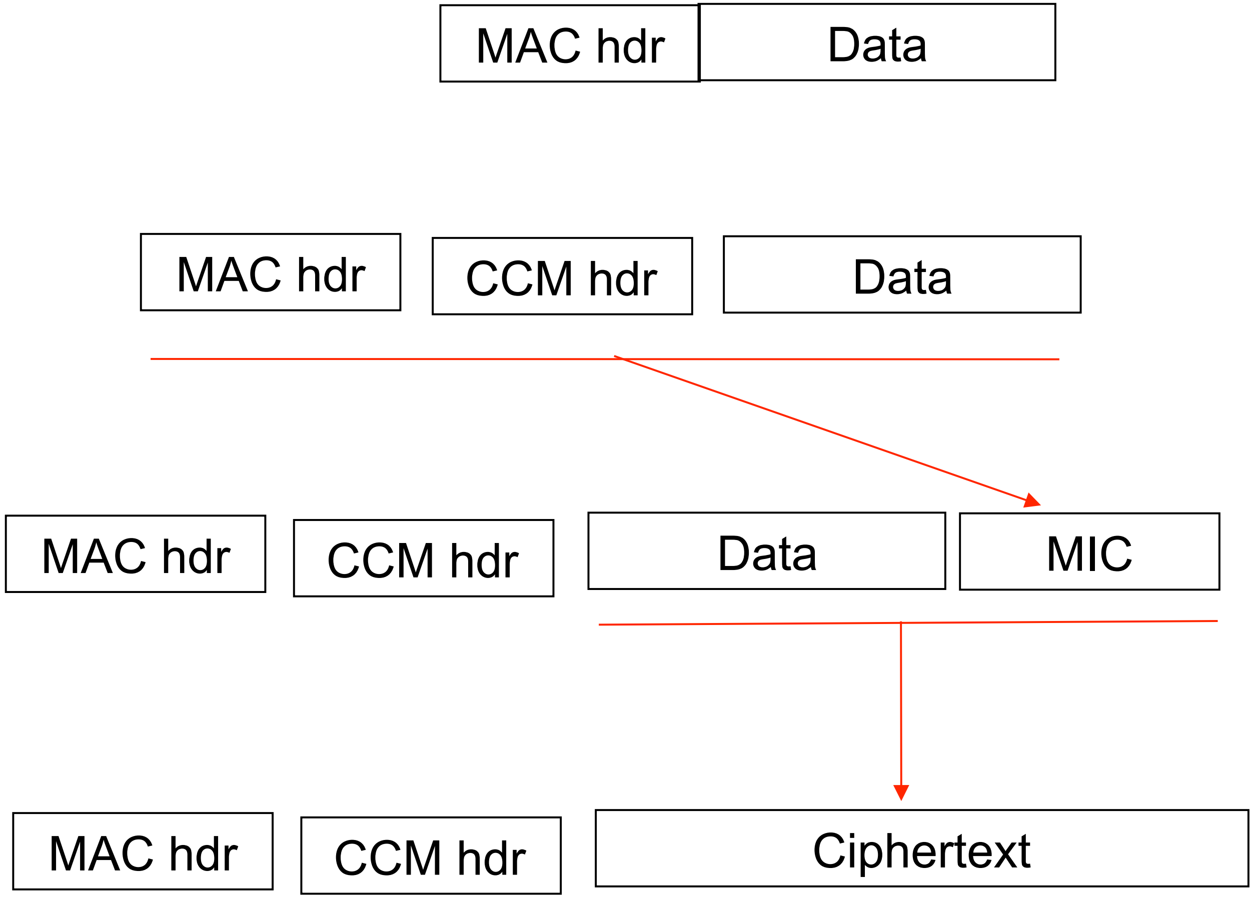


AES-MAC: CBC-MAC mode of operation

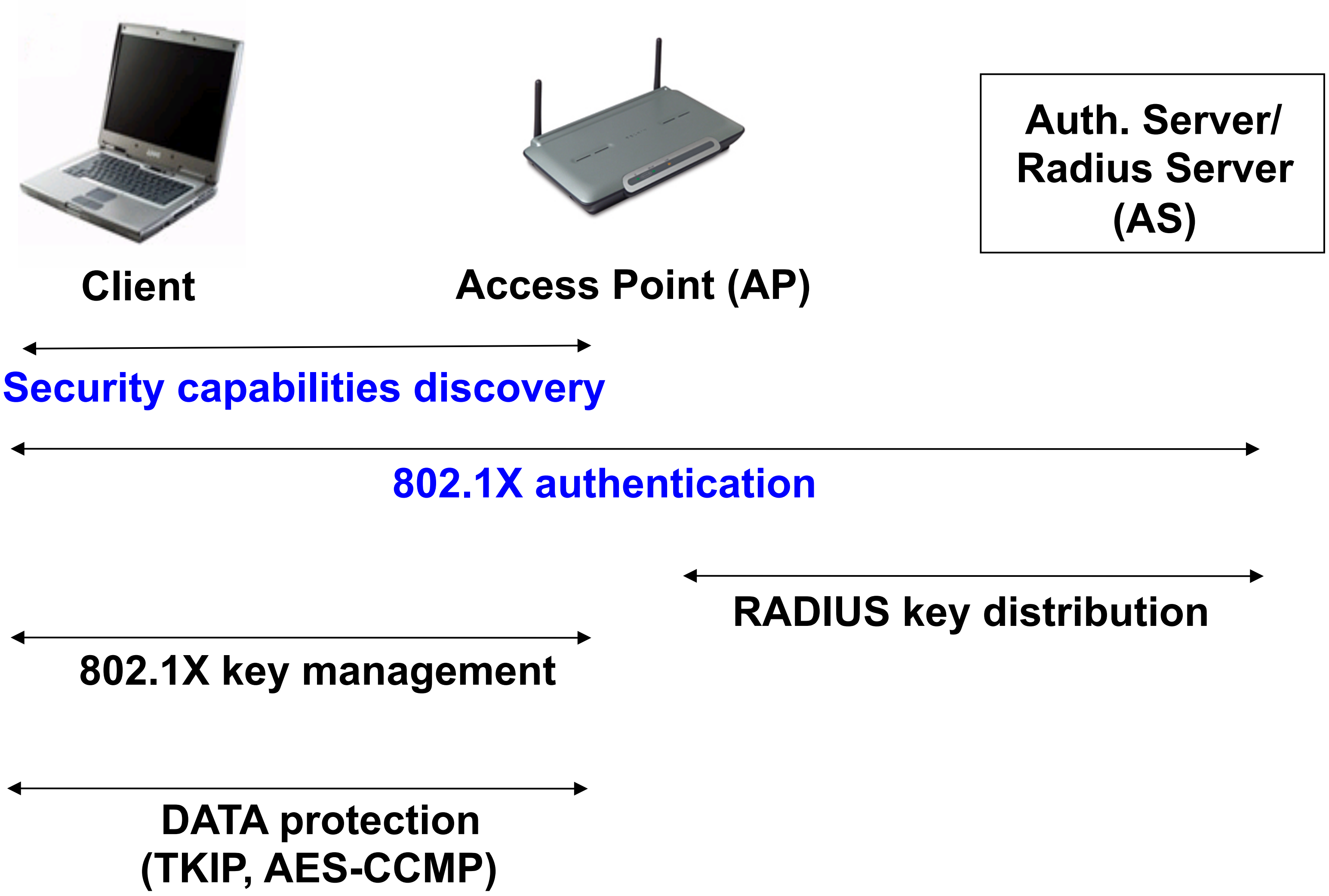
- AES is also used to compute MAC
  - One algorithm for encryption and MAC!
- CBC is used to compute a MIC (Message Integrity Code)
  - 1. Take the first block and encrypt it using AES
  - 2. XOR the result with the second block and then encrypt the result
  - 3. XOR the result with next block and encrypt that…and so on!



How is CCM Used in RSN….



802.11i overview
Authentication-Authorization Phases



Client

Access Point (AP)

Auth. Server/
Radius Server
(AS)

Security capabilities discovery

802.1X authentication

RADIUS key distribution

802.1X key management

DATA protection
(TKIP, AES-CCMP)

802.11i protocol steps

Discovery

- AP advertises network security capabilities to Clients

802.1x authentication

- Implement access control
- It was not originally designed for wireless networks
- It was designed to allow an unauthorized device to be physically attached to a LAN infrastructure

802.1x

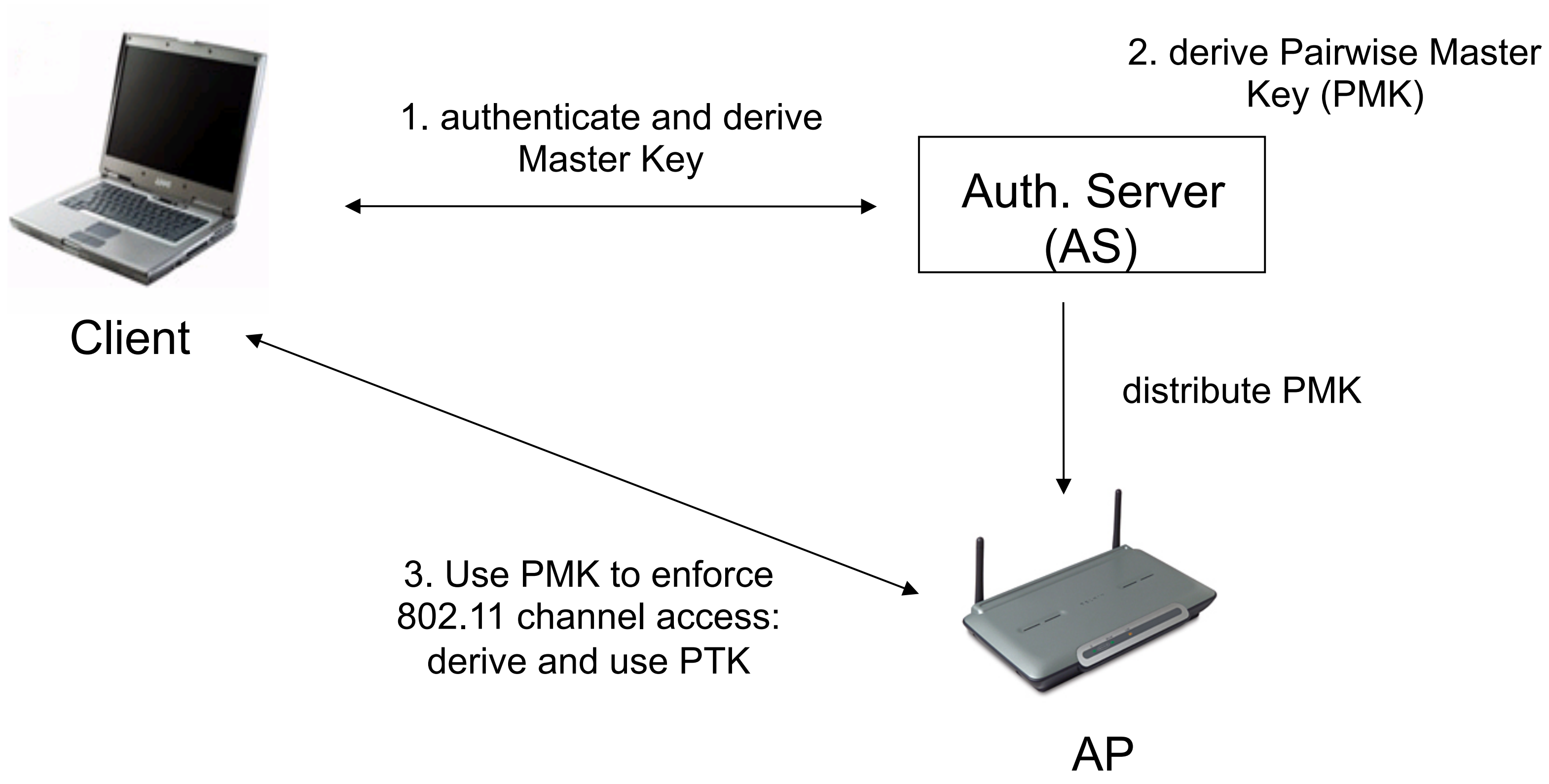
802.1x authentication

- Mutually authenticate Client and AS (Auth. Server)
  - Using password, Challenge/response, TLS,…
  - Generate Master Key as a side effect of authentication
  - Generate Pairwise MK (PMK) as an access authorization token
  - AS send PMK to AP (via Radius)
- Client and AP Generate 4 keys for encryption/ integrity from PMK
- Client authenticates AP!

Key derivation

- Client and AS derive a PMK (Pairwise Master Key)
  - As a result of EAP protocols (TLS for example)
  - From a manually configured master key (MK)…
- AS then sends PMK to the AP
  - Via Radius protocol
- Four separate keys for two layers' protection (the PTK: Pairwise Transient Keys) are then derived by AP and Client
  - Data Encryption key
  - Data Integrity key
  - EAPOL-Key Encryption key
  - EAPOL-Key Integrity key

802.11i/RSN key Hierarchy



2. derive Pairwise Master Key (PMK)

1. authenticate and derive Master Key

Auth. Server (AS)

Client

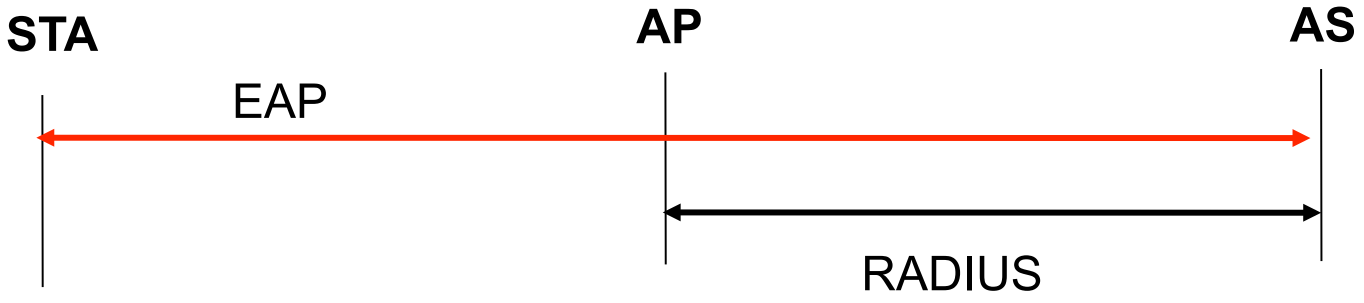
distribute PMK

3. Use PMK to enforce 802.11 channel access: derive and use PTK

AP

- MK ≠ PMK or AP could make access control decision instead of AS
- MK is fresh and bound to the session between Client and *AS*
- Easy revocation of AP

802.1X: the protocols (EAP)

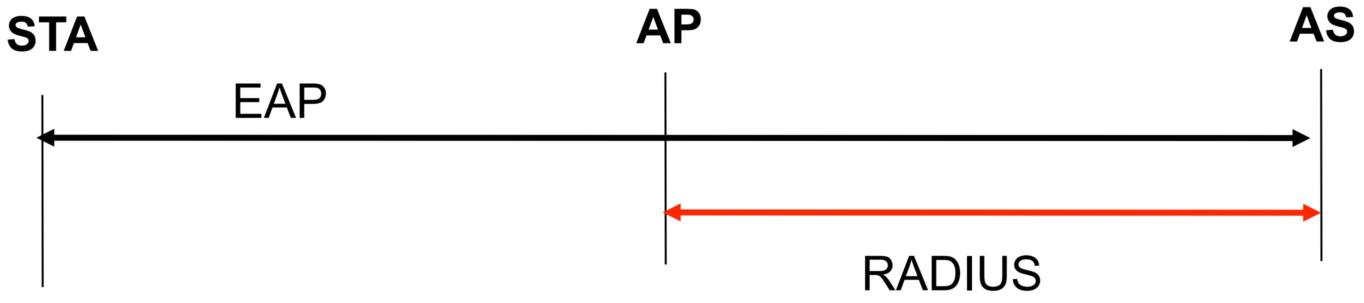


- STA communicates with AS using EAP (Extensible Authentication Protocol)
  - EAP (Extensible Authentication Protocol) [RFC 3748]
    - carrier protocol designed to transport the messages of "real" authentication protocols (e.g., TLS)
    - very simple, four types of messages:
      - EAP request – carries messages from the supplicant to the authentication server
      - EAP response – carries messages from the authentication server to the supplicant
      - EAP success – signals successful authentication
      - EAP failure – signals authentication failure
    - Authenticator doesn't understand what is inside the EAP messages, it recognizes only EAP success and failure

Protocols – LEAP, EAP-TLS, PEAP, EAP-SIM

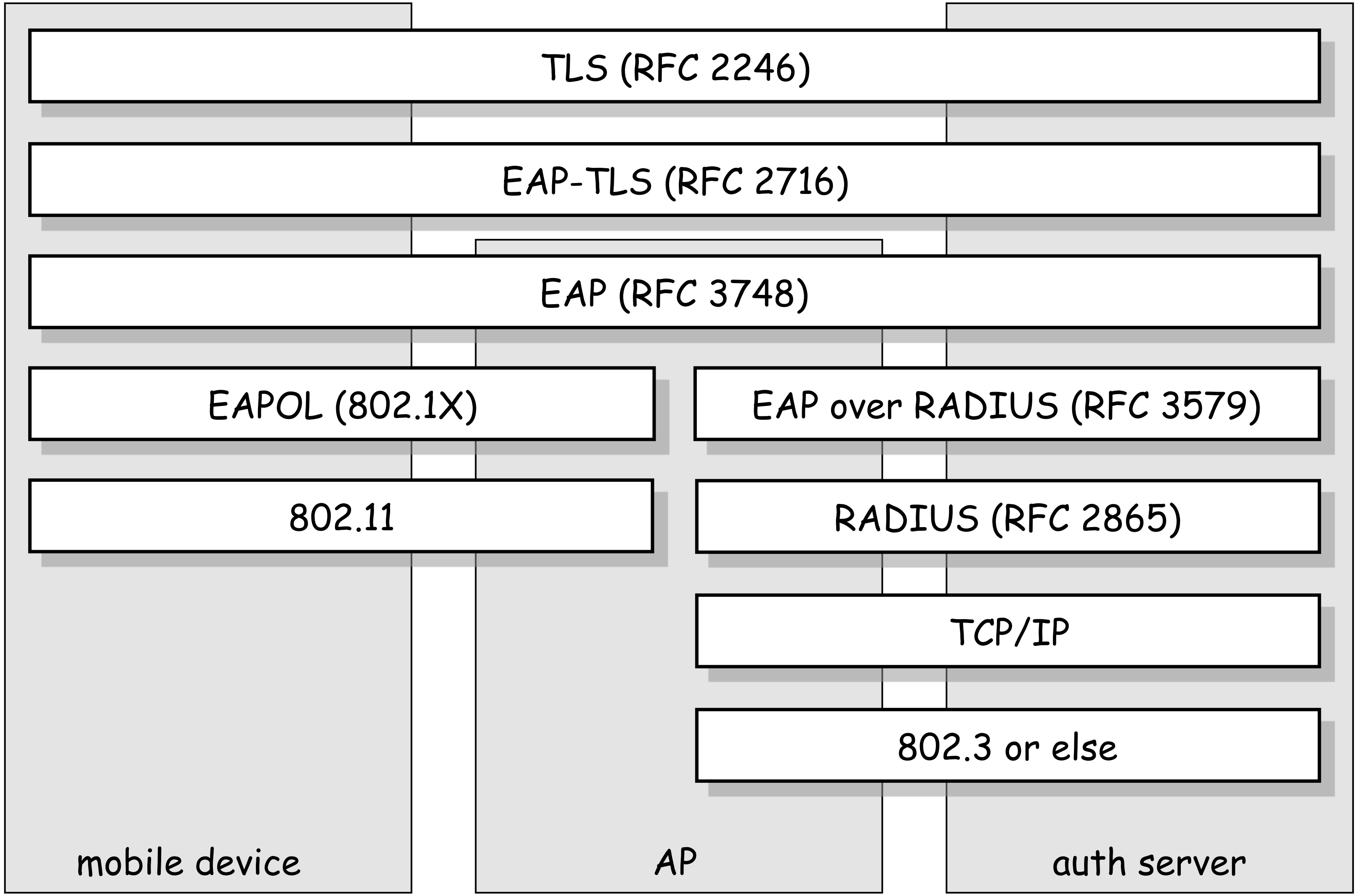
- LEAP (Light EAP)
  - developed by Cisco
  - similar to MS-CHAP extended with session key transport
- EAP-TLS (TLS over EAP)
  - only the TLS Handshake Protocol is used
  - server and client authentication, generation of master secret
  - TLS master secret becomes the session key
  - mandated by WPA, optional in RSN
- PEAP (Protected EAP)
  - phase 1: TLS Handshake without client authentication
  - phase 2: client authentication protected by the secure channel established in phase 1
- EAP-SIM
  - extended GSM authentication in WiFi context

802.1X: the protocols (RADIUS)



- AP and AS communicates using RADIUS protocol
  - RADIUS (Remote Access Dial-In User Service) [RFC 2865-2869, RFC 2548]
    - used to carry EAP messages between the AP and the AS
    - MS-MPPE-Recv-Key attribute is used to transport the session key from the auth server to the AP
    - RADIUS is mandated by WPA and optional for RSN

Summary of the protocol architecture



TLS (RFC 2246)

EAP-TLS (RFC 2716)

EAP (RFC 3748)

EAPOL (802.1X)

EAP over RADIUS (RFC 3579)

802.11

RADIUS (RFC 2865)

TCP/IP

802.3 or else

mobile device

AP

auth server

In summary: WEP vs. WPA vs. WPA2

	WEP	WPA	WPA2
Encryption	RC4	RC4	AES
Key rotation	None	Dynamic session keys	Dynamic session keys
Key distribution	Manually typed into each device	Automatic distribution available	Automatic distribution available
Authentication	Uses WEP key as AuthC	Can use 802.1x & EAP	Can use 802.1x & EAP

Summary

- the new security standard for WiFi is 802.11i
  - access control model is based on 802.1X
  - flexible authentication based on EAP and upper layer authentication protocols (e.g., TLS, GSM authentication)
  - improved key management
  - TKIP
    - uses RC4 → runs on old hardware
    - corrects WEP’s flaws
    - mandatory in WPA, optional in RSN (WPA2)
  - AES-CCMP
    - uses AES in CCMP mode (CTR mode and CBC-MAC)
    - needs new hardware that supports AES