14. Wireless Network Security

Computer Security Courses @ POLIMI

Prof. Maggi & Prof. Zanero

The Wireless Security Challenge

- For any wireless network (from bluetooth to wifi to satcom)
  - Same security challenges of any network
  - Plus, intrinsic challenges of being radio-transmitted
- Patterns of typical wireless/radio challenges
  - Eavesdropping (radio transmissions not confined)
  - Injection/spoofing (lack of authentication)
  - DoS (radio prone to physical DoS attacks)

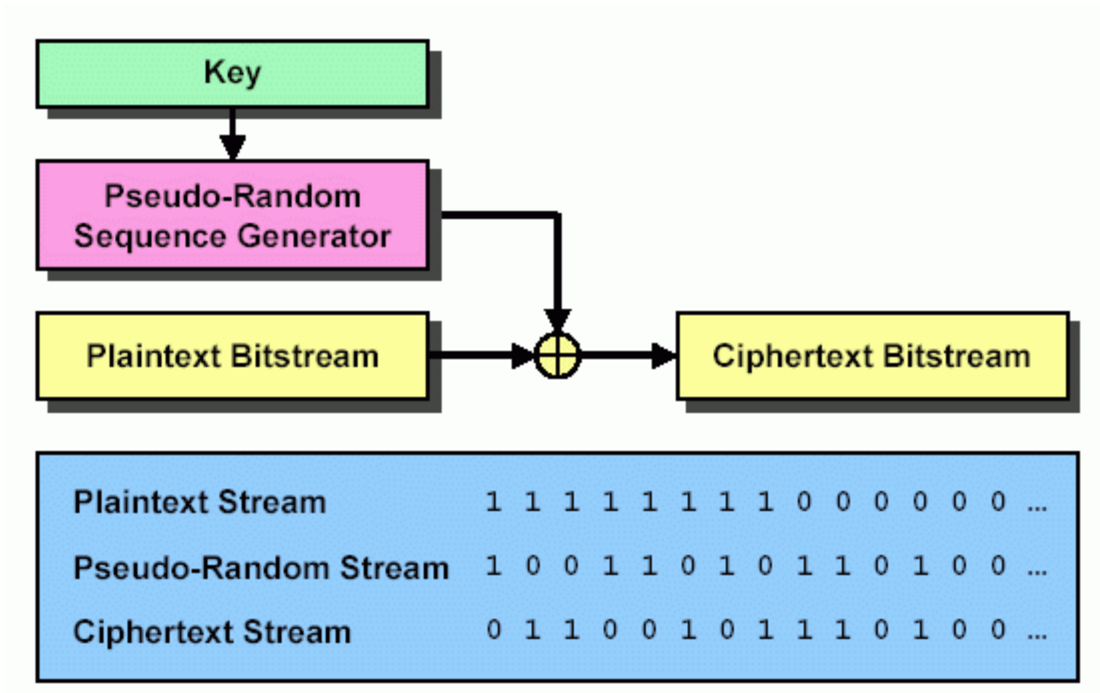
“Wi-fi” Networks

- All starts with IEEE 802.11 in 1997
  - **Objective:** make “Ethernet” (802.3) wireless
  - First release: 2.4 GHz (ISM band), 1–2 Mbps
  - All 802.11 networks use the same protocols for Media Access Control (MAC), namely Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA)
  - Multiple channels provide physical separation, SSID (Service Set ID) provides logical separation
  - Infrastructure vs ad hoc
  - Range of tens of meters
- Subsequent standards (b, g, n, …) improved transmission rates and changed the PHY layer

The First Security Standard: WEP

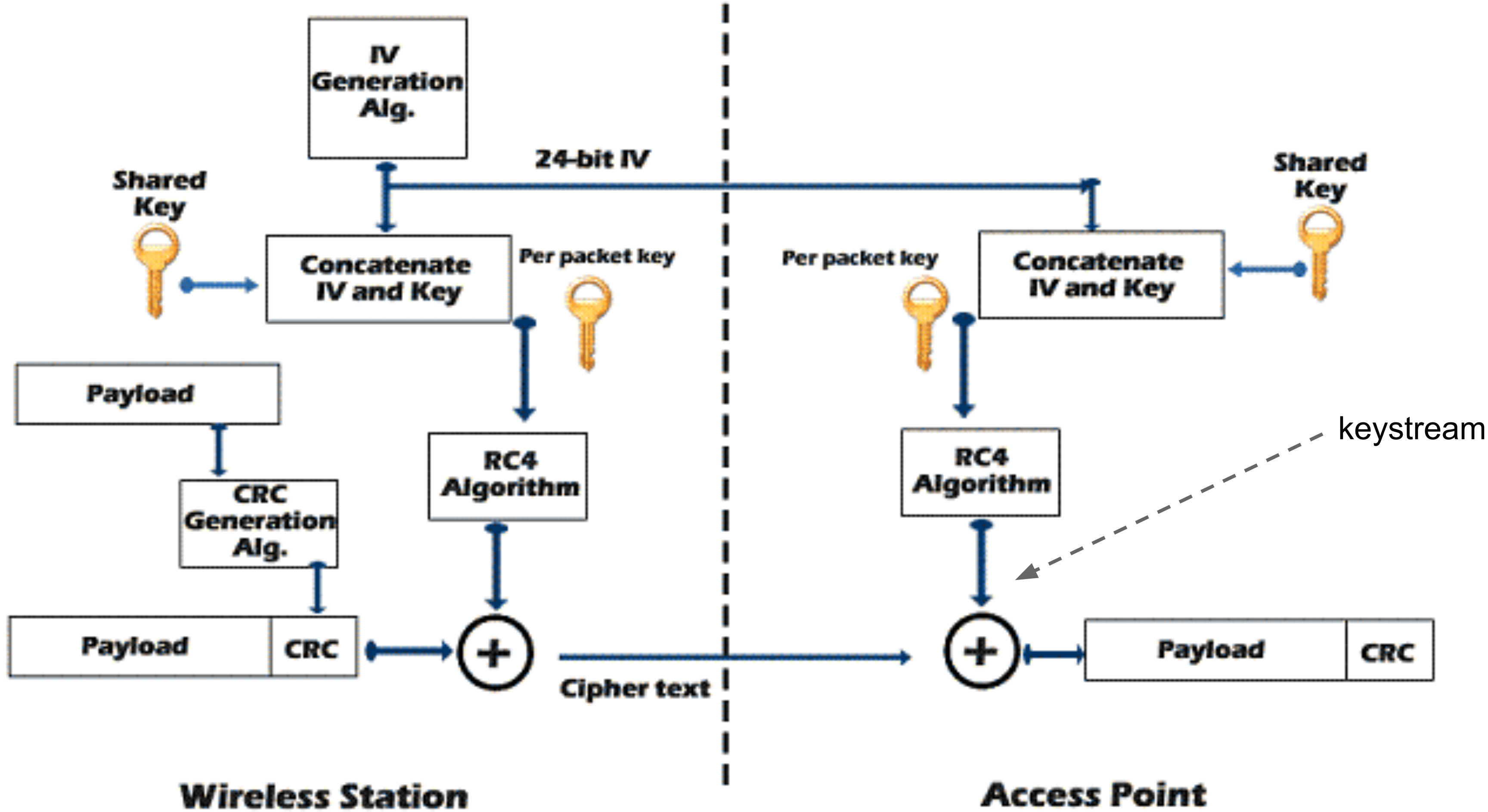
- WEP (Wired Equivalent Privacy) added as of 802.11b
  - **Packet per packet** encryption of payloads
  - Based on a **cipher feedback** (CFB) stream
  - Shared key, manually set on all clients and APs
- The basic stream cipher of WEP is RC4
  - designed by Ron Rivest in 1987, protected as a trade secret by RSA until 1994, when it was leaked on USENET and became public domain
  - IEEE then decided to adopt it in 802.11b's WEP

Stream Cipher in CFB Mode



The **key is the same**, so the **PRS** is going to be the **same** for each packet.

Usage of RC4 in WEP



802.11 header	IV	Payload	CRC

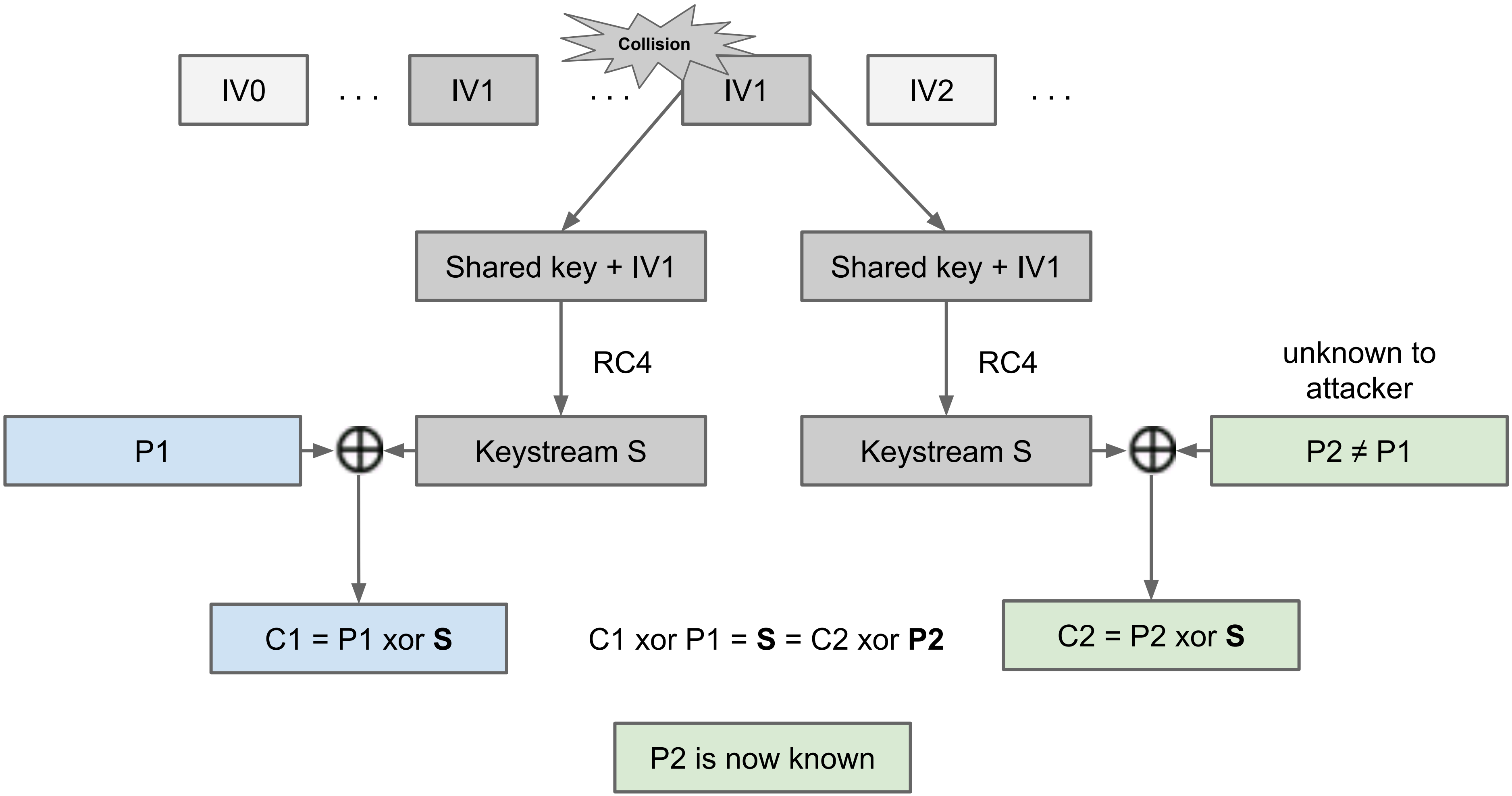
Doomed to Fail?

- Since for a stream cipher in CFB mode:
  - Same plaintext => same ciphertext
  - Key stream is always the same
- To mitigate, a random IV is used:
  - Combines with the key and adds randomness
  - Needs to be transmitted in clear (but this is fine, because it will combine with the key)
- Export of ciphers > 64bits forbidden by ITAR: 128 bit version of RC4 could not be used
  - 64bits per packet key = 24bits IV + 40bits shared key
  - Later on, they couldn't change the format, so:
    - 128bits per packet key = 24 bits IV + 104 bits shared key

Attack 1: Walker (2000)

- http://www.cse.sc.edu/~wyxu/719Spring10/papers/JesseWalkerWEP.doc (difficult to find)
- IV space is small (2^24)
  - Due to birthday paradox, random collision of IVs every 2^12
  - Probability of collision is 99% after 12,430 frames
    - 2–3 seconds of normal 11Mbps traffic
- IV collisions => repeated keystream S
  - Since packets P1 and P2 are both encrypted with S: C1 = P1 xor S, and same for C2
  - If I know P1, I can derive S = C1 xor P1, and thus P2

Attack 1 (visualized)



Attack 2: Borisov, Goldberg, Wagner (2001)

- http://dl.acm.org/citation.cfm?id=381695
- **Flip arbitrary bits** into **encrypted** messages
  - The MIC function is CRC32, which is distributive w.r.t. XOR: CRC(X xor Y) = CRC(X) xor CRC(Y)
  - The transmission is A | CRC(A)
  - I want to flip the bits in mask M, thus A' = A xor M
    - I must send (A xor M) | CRC(A xor M)
    - Easy: CRC(A xor M) = CRC(A) xor CRC(M)
- This depends from the **lack of an authenticated portion** in CRC32

Attack 3: Arbaugh (2001)

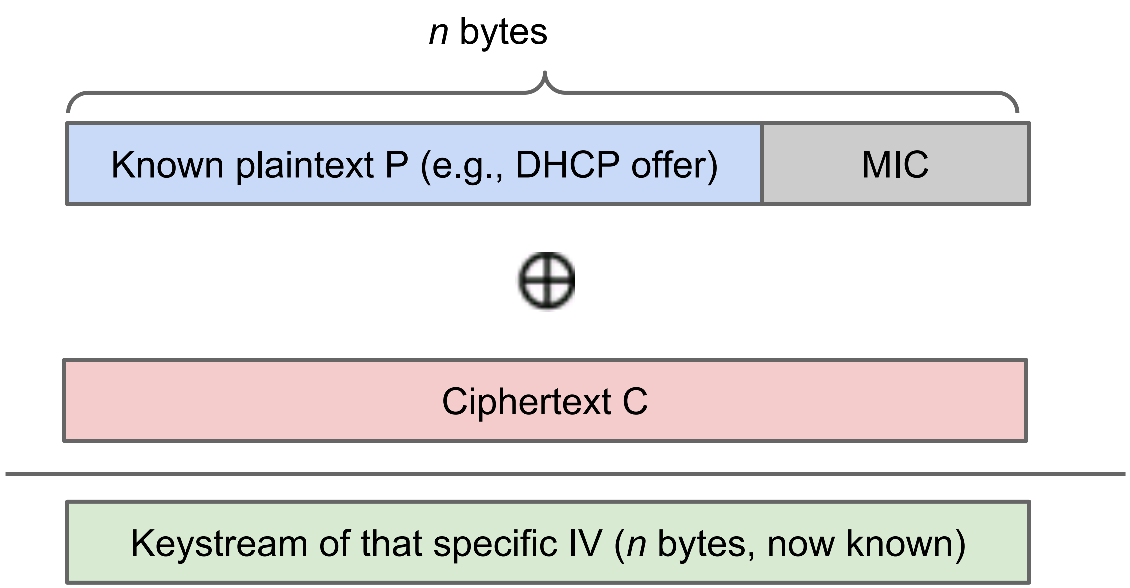
Inductive attack to retrieve the keystream (not the key) starting from a known plaintext

- Suppose we know *n-4* bytes of plaintext (and their 4 bytes of MIC) corresponding to *n* bytes of ciphertext (e.g., a DHCP discover)
- We know, therefore, *n* bytes of the keystream associated to some (unknown) IV

Objective: we want to know *n+1* bytes

http://www.slideserve.com/foy/an-inductive-chosen-plaintext-attack-against-wep

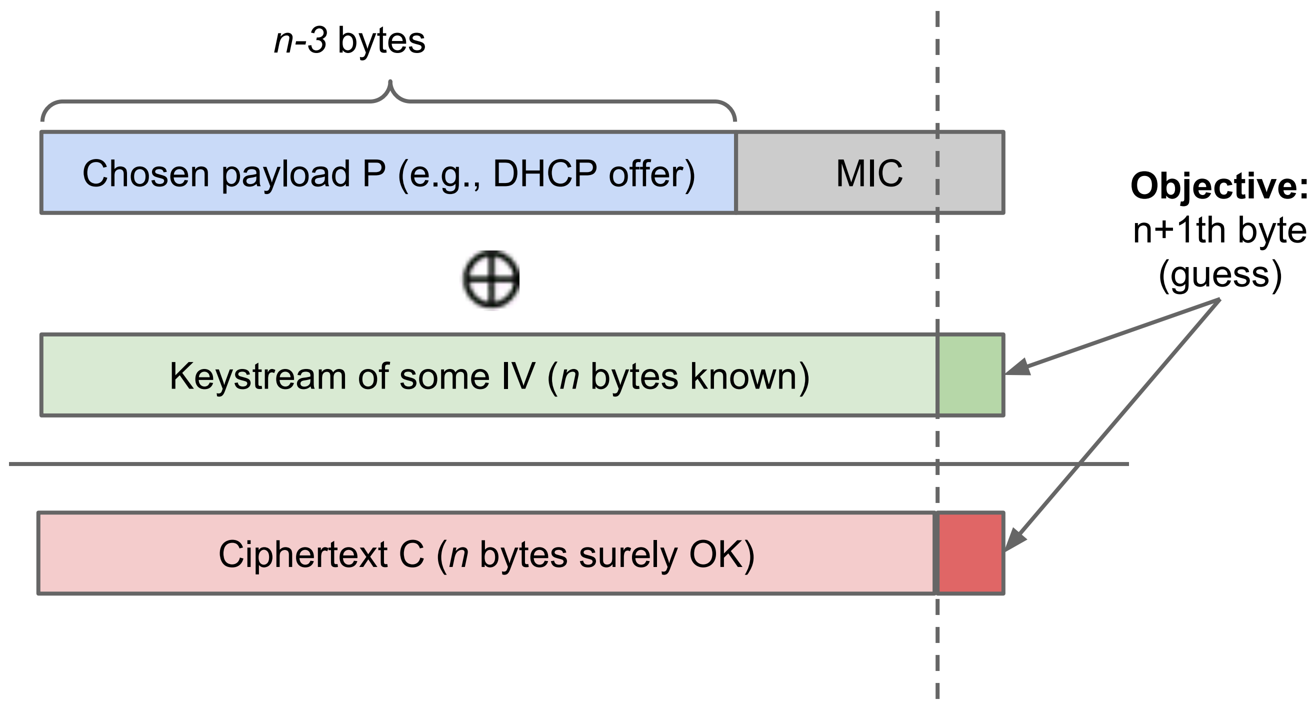
Attack 3 (starting point)



Attack 3 (inductive step)

- We can inject arbitrary messages of length (*n-4*) with that specific IV
- How do we extend it by 1 byte?
  - Pick a message long (*n-3*) which generates an answer if received (e.g., a ping)
  - Encrypt it and try guessing the last byte; if answer received we guessed right.
- Lather, rinse, repeat until we retrieve up to 1500 bytes (MTU) of keystream for that IV
- Repeat for every IV

Attack 3 (inductive step, visualized)



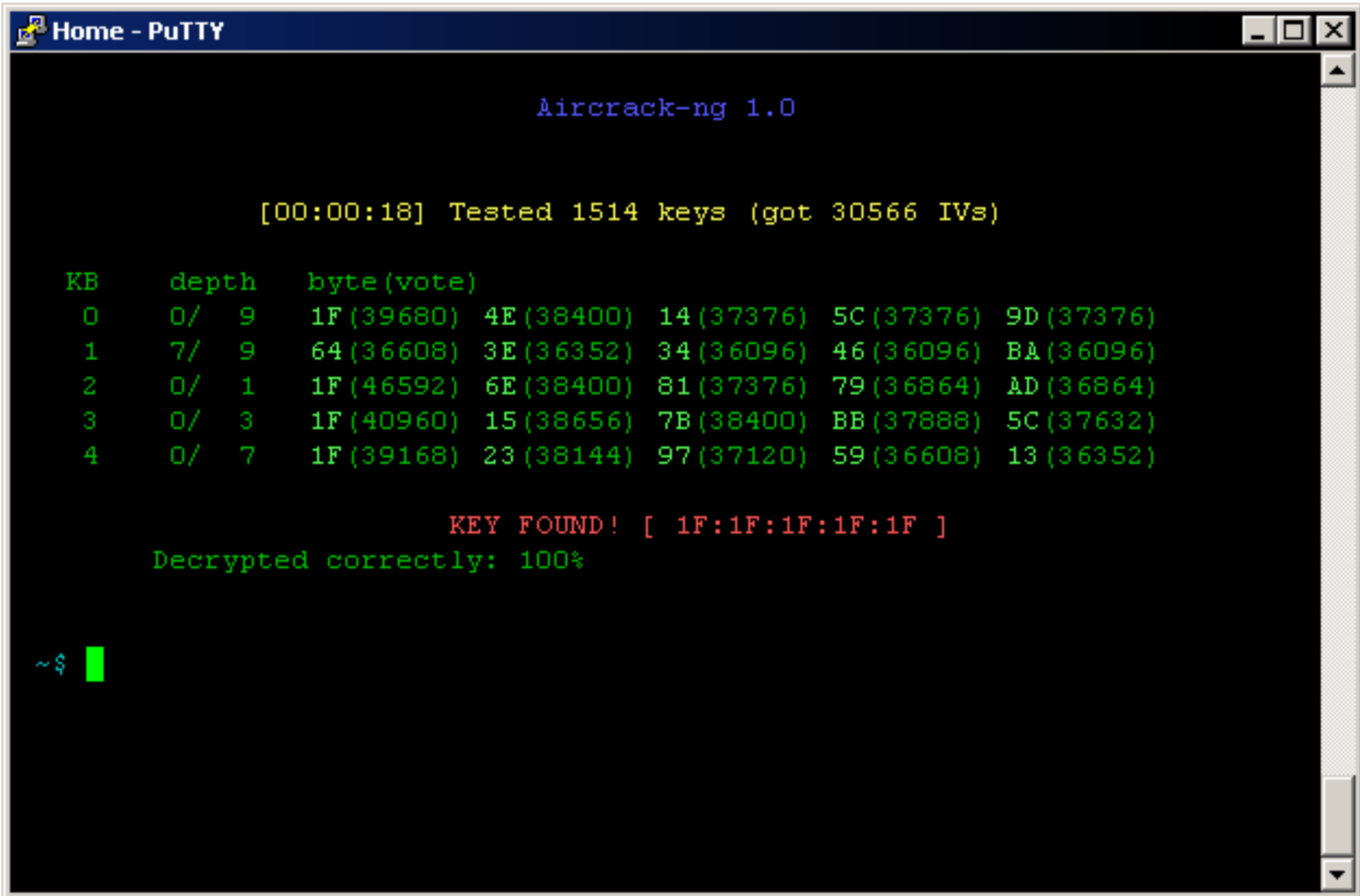
Fatality: Fluhrer, Shamir, Mantin (01)

- Vulnerability in scheduling algorithm of RC4
  - **Passive** statistic attack, which extracts key directly from ciphertext, exploiting a set of weak IV http://wiki-files.aircrack-ng.org/doc/technique_papers/rc4_ksaproc.pdf
  - Needs several million packets (several hours of sniffing) and then breaks the key in a few seconds
  - completely passive
- Implementations against WEP
  - Stubblefield, Ioannidis and Rubin first, but did not release code (and paper published 2002)
  - WEPCrack released first (Aug. 2001)

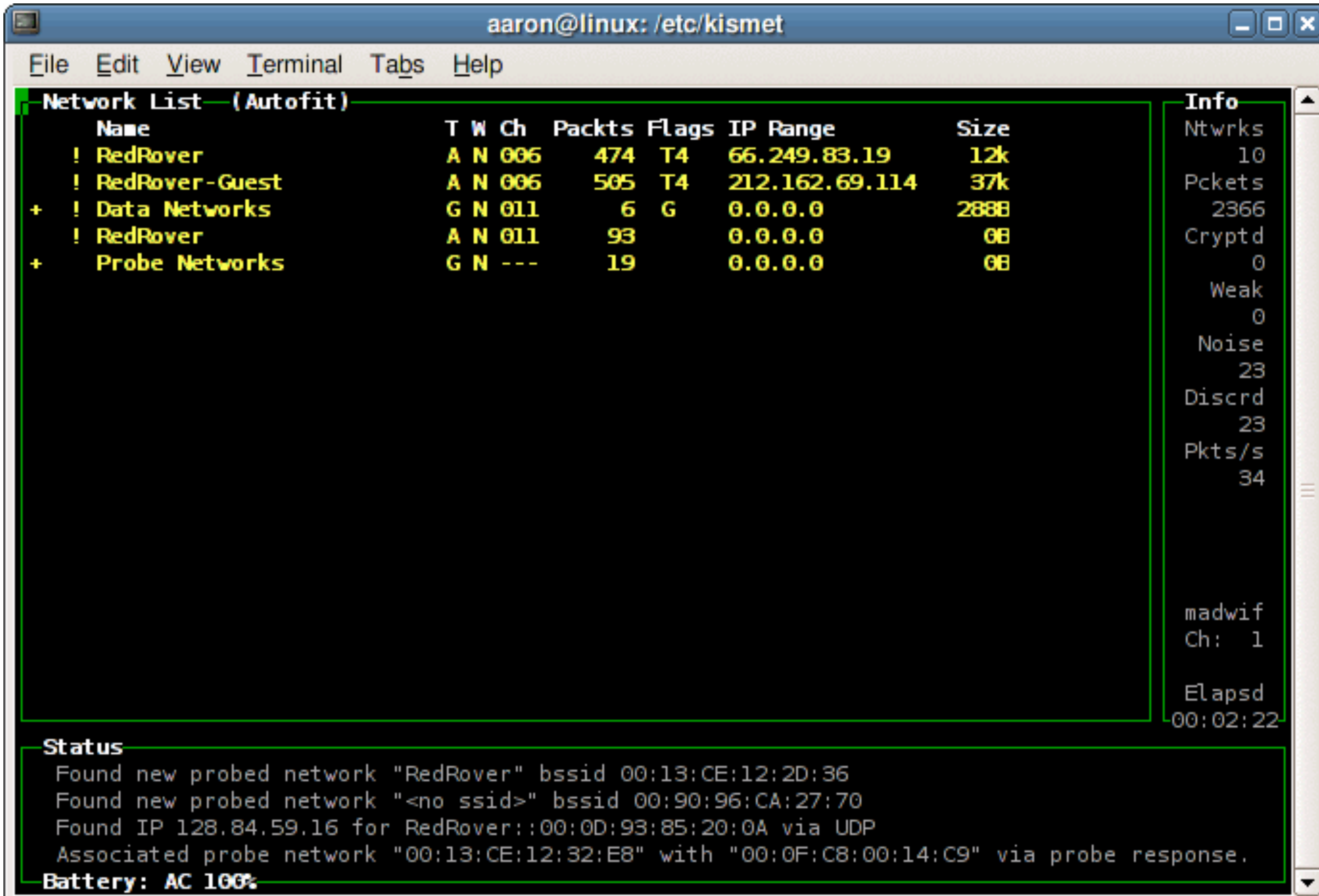
Modern wi-fi tools

- Aircrack-NG: modern implementation of WEP attacks
  - http://www.aircrack-ng.org/
- Wavemon: monitor for signal intensity
  - http://www.erg.abdn.ac.uk/wavemon/
- Kismet: lists networks and their security settings
  - https://www.kismetwireless.net/
  - used for “wardriving” in glorious past

Aircrack-ng

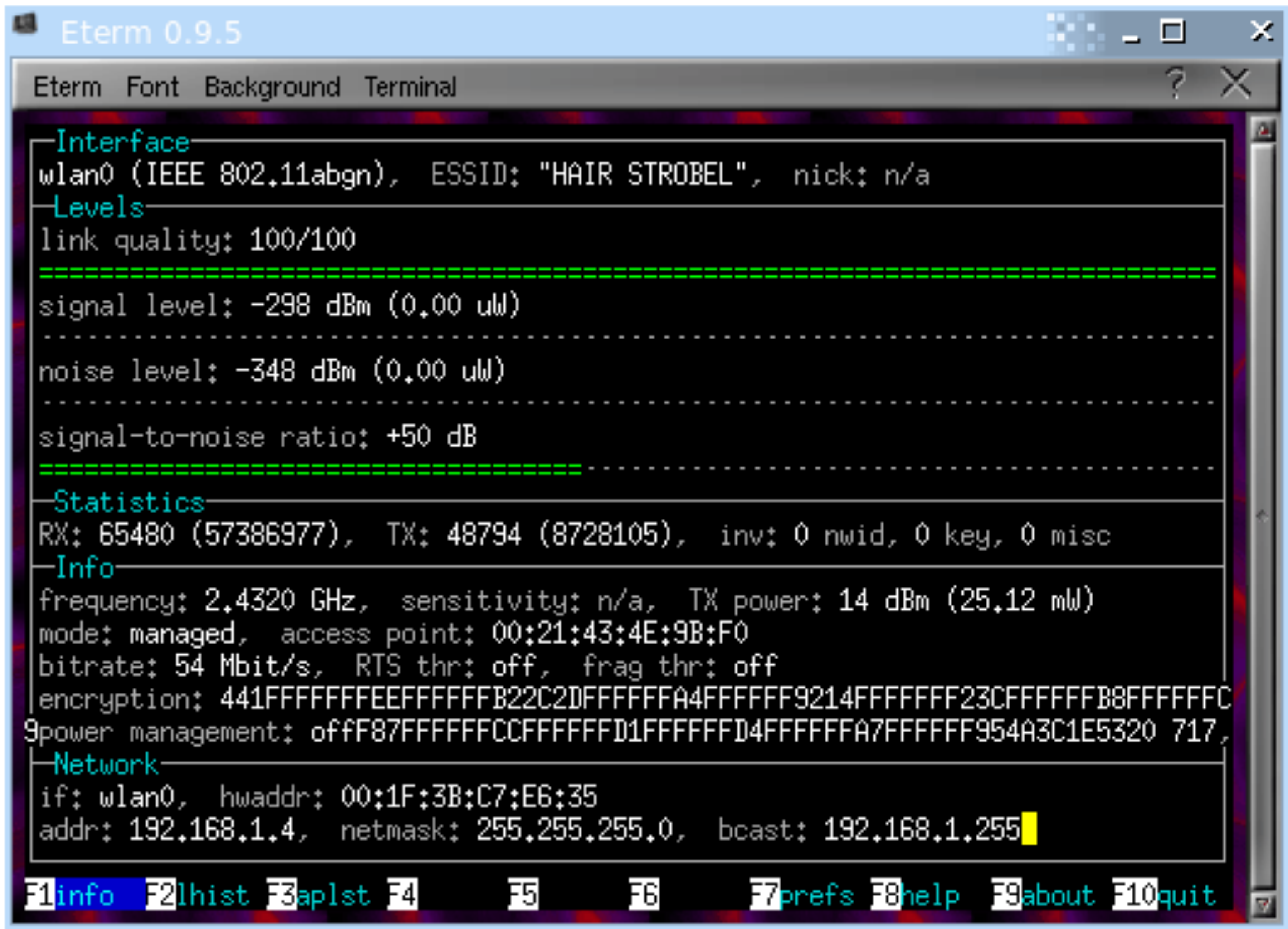

Home - PuTTY
Aircrack-ng 1.0
[00:00:18] Tested 1514 keys (got 30566 IVs)
KB depth byte(vote)
0 0/ 9 1F(39680) 4E(38400) 14(37376) 5C(37376) 9D(37376)
1 7/ 9 64(36608) 3E(36352) 34(36096) 46(36096) BA(36096)
2 0/ 1 1F(46592) 6E(38400) 81(37376) 79(36864) AD(36864)
3 0/ 3 1F(40960) 15(38656) 7B(38400) BB(37888) 5C(37632)
4 0/ 7 1F(39168) 23(38144) 97(37120) 59(36608) 13(36352)
KEY FOUND! [1F:1F:1F:1F:1F]
Decrypted correctly: 100%
~$

Kismet

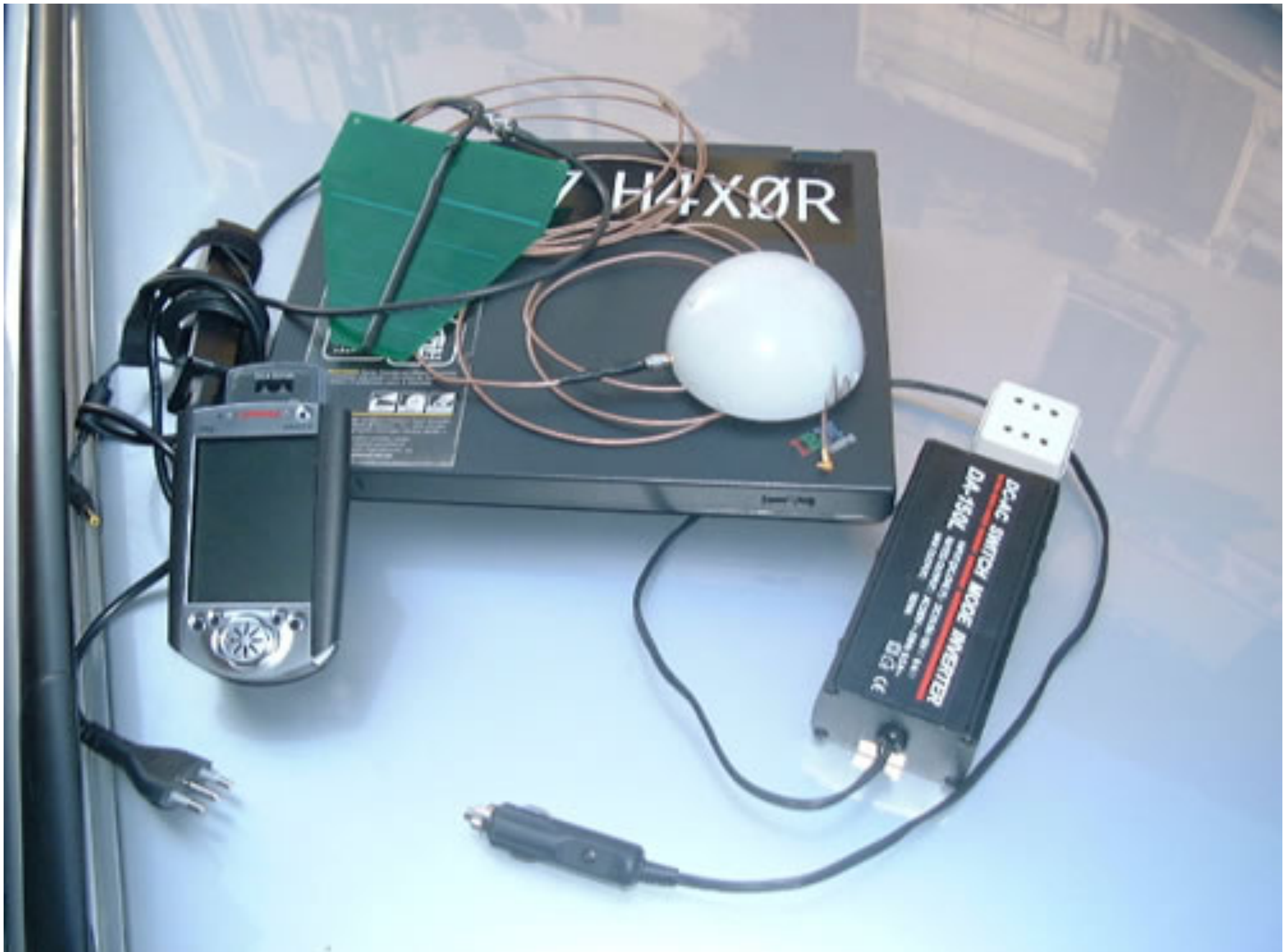


aaron@linux: /etc/kismet
File Edit View Terminal Tabs Help
Network List (Autofit)
Name T W Ch Packts Flags IP Range Size
! RedRover A N 006 474 T4 66.249.83.19 12k
! RedRover-Guest A N 006 505 T4 212.162.69.114 37k
+ ! Data Networks G N 011 6 G 0.0.0.0 288B
! RedRover A N 011 93 0.0.0.0 0B
+ Probe Networks G N --- 19 0.0.0.0 0B
Info
Ntwrks 10
Pckets 2366
Cryptd 0
Weak 0
Noise 23
Discrd 23
Pkts/s 34
madwif
Ch: 1
Elapsd 00:02:22
Status
Found new probed network "RedRover" bssid 00:13:CE:12:2D:36
Found new probed network "<no ssid>" bssid 00:90:96:CA:27:70
Found IP 128.84.59.16 for RedRover::00:0D:93:85:20:0A via UDP
Associated probe network "00:13:CE:12:32:E8" with "00:0F:C8:00:14:C9" via probe response.
Battery: AC 100%

Wavemon


Eterm 0.9.5
Eterm Font Background Terminal
Interface
wlan0 (IEEE 802.11abgn), ESSID: "HAIR STROBEL", nick: n/a
Levels
link quality: 100/100
signal level: -298 dBm (0.00 uW)
noise level: -348 dBm (0.00 uW)
signal-to-noise ratio: +50 dB
Statistics
RX: 65480 (57386977), TX: 48794 (8728105), inv: 0 nwid, 0 key, 0 misc
Info
frequency: 2.4320 GHz, sensitivity: n/a, TX power: 14 dBm (25.12 mW)
mode: managed, access point: 00:21:43:4E:9B:F0
bitrate: 54 Mbit/s, RTS thr: off, frag thr: off
encryption: 441FFFFFFFEEFFFFFFB22C2DFFFFFFA4FFFFFF9214FFFFFFF23CFFFFFFB8FFFFFFC
9power management: offF87FFFFFFCCFFFFFFD1FFFFFFD4FFFFFFA7FFFFFF954A3C1E5320 717,
Network
if: wlan0, hwaddr: 00:1F:3B:C7:E6:35
addr: 192.168.1.4, netmask: 255.255.255.0, bcast: 192.168.1.255
F1info F2lhist F3aplst F4 F5 F6 F7prefs F8help F9about F10quit

Heroic times of wardriving...



Life Beyond WEP: 802.1x and 802.11i

- Two standards developed to patch the WEP fiasco
- Commercial name: WPA, Wi-Fi Protected Access
  - WPA-1
    - 802.1x-LEAP + TKIP (from 802.11i)
  - WPA-2
    - 802.1x-PEAP + AES
- Since August 2003, WPA is a requirement for the “Wi-Fi” logo

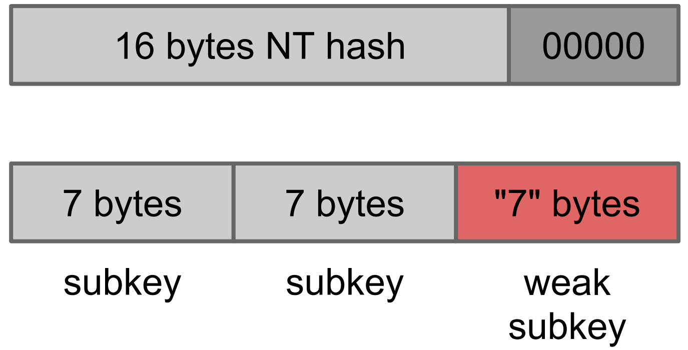
Fixing Authentication With 802.1x

- Standard in the 802.1 family, common to “wired” and “wireless” networks
  - Authentication at layer 2 (at “port” level)
- Specifies use of EAP (RFC 2284)
  - Supports wide range of auth options
  - Variants:
    - EAP-TLS: RFC 2716, uses TLS and a digital certificate for mutual auth and automatic key exchange, but requires a PKI :-(
    - EAP-TTLS: login+password in TLS tunnel, no need of PKI and certificate for clients, less burden :-)
    - LEAP/PEAP deserve a slide on their own.

LEAPing and PEAPing

- LEAP is a proprietary Cisco protocol
  - Uses MS-CHAPv1 for mutual authentication and dynamic key generation
- PEAP: Protected EAP, new standard proposed by Microsoft and Cisco, using CHAPv2
  - Why? Because CHAPv1 is weak, and as a result, LEAP is vulnerable to attack
    - Attack developed by J. Wright and disclosed in August 2003 at DefCon in Las Vegas

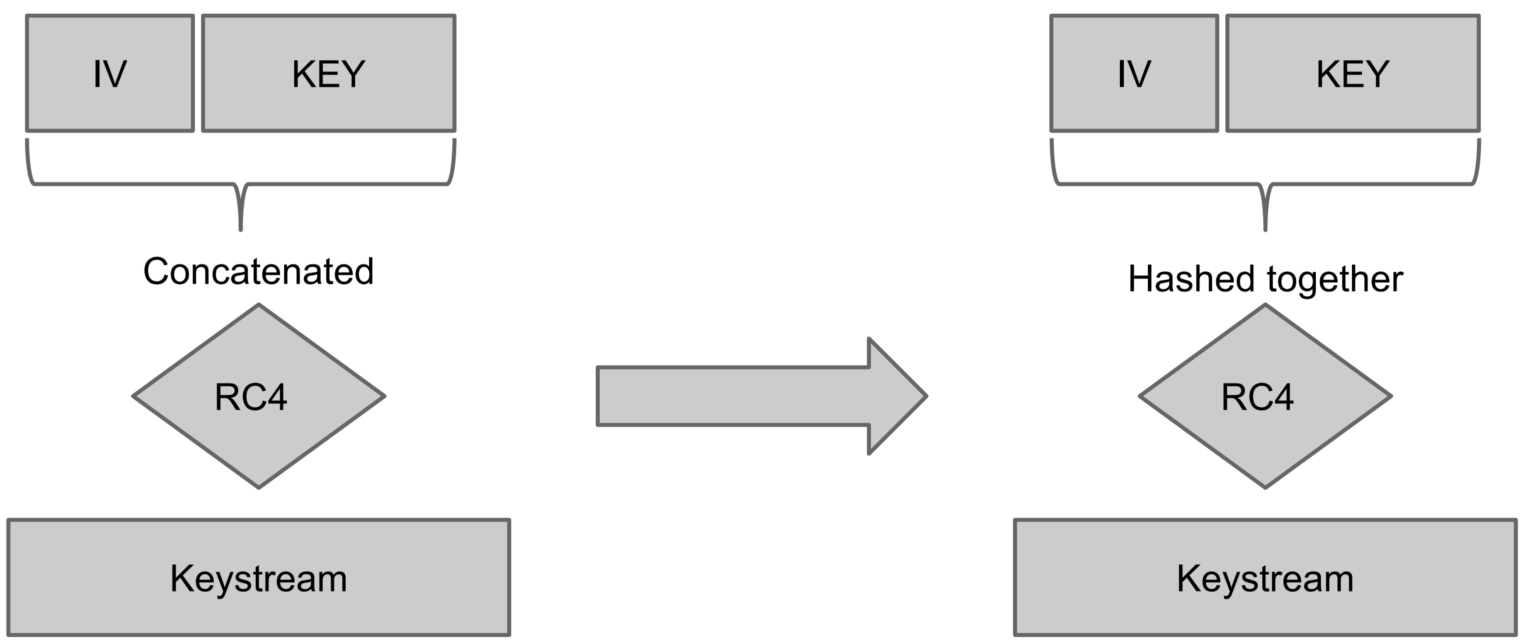
Falling asLEAP



- LEAP uses CHAPv1:
  - Based on NTLM passwords hashes
  - Username in cleartext, ok, fine...
  - “Two bytes vulnerability” on the password
    - NT_HASH: password hashed with MD4 (16 byte hash). No salt(!)
    - 16 byte are brought to 21 adding 5 null (!!)
    - Divided in 3*7 byte (56-bit) subkeys (DES)
    - Each subkey used to encrypt a challenge separately
    - Attacker knows challenge (sent in clear), and that third subkey has just 2 non-zero bytes
    - 2 byte = 65k combinations: can guess
    - Hash not salted: can reduce space of keys (on a dictionary of 3 million, I need to test just 3M/65k = 45 of them)
    - http://asleap.sourceforge.net
- Cisco suggested “a strong password policy”...

802.11i - Fixing Encryption

- Replacing RC4 with AES is the primary recommendation
  - But it requires hardware change!
  - Interim patch: TKIP (Temporal Key Integrity Protocol)



802.11i - Fixing Message Integrity

- Interim fix also CRC-32 until AES can be used
- Algorithm “Michael”
  - Before, with WEP
    - MIC = *f*(payload)
  - Now, with WPA
    - MIC = *f*(<u>random seed, src MAC, dst MAC</u>, payload)
  - Both <u>seed</u> and MIC are <u>in the encrypted payload</u>
    - <u>no more blind bit flipping</u>
  - Details
    - http://www.uow.edu.au/~jennie/WEB/WEB05/Michael.pdf
  - A practical attack against WPA-PSK with TKIP and Michael http://dl.aircrack-ng.org/breakingwepandwpa.pdf

Summary

- WEP: fiasco, insecure at any key size
- WPA-1 insecure, at least in PSK mode
- WPA-1 in EAP mode: currently secure
- WPA-2 currently considered secure