

Nomadic Communications Labs



Alessandro Villani
avillani@science.unitn.it



WEP Cracking

WEP: Wired Equivalent Privacy

- ❑ The aim declared of the **WEP** (Wired Equivalent privacy) key is (*Nomina sunt consequentia rerum*) providing “a security level on the wireless channel equivalent to what one can expect in the case of wired networks”
- ❑ Some have thought WEP as the sole mechanism for the access control
- ❑ Other as the solution of all the security problems

WEP: Wired Equivalent Privacy

- ❑ The shared key must be installed on the Access Point and on the client
- ❑ On the devices up to 4 keys are configurable but the standard does not specify how manage these keys: in the practice just one is used
- ❑ PROBLEM:
 - It is not possible install/update it
 - It is the same for all the users of the same AP

WEP: How it works

- ❑ WEP is based on the RC4 algorithm of the RSA
- ❑ It is a system of encryption based on a shared key
- ❑ The shared key is 40 bits (or 104 bits) long
- ❑ It is joined with an ***initialization vector*** (IV) 24 bits long
- ❑ In this way a seed of 64 bits (or 128 bits) is obtained for the RC4

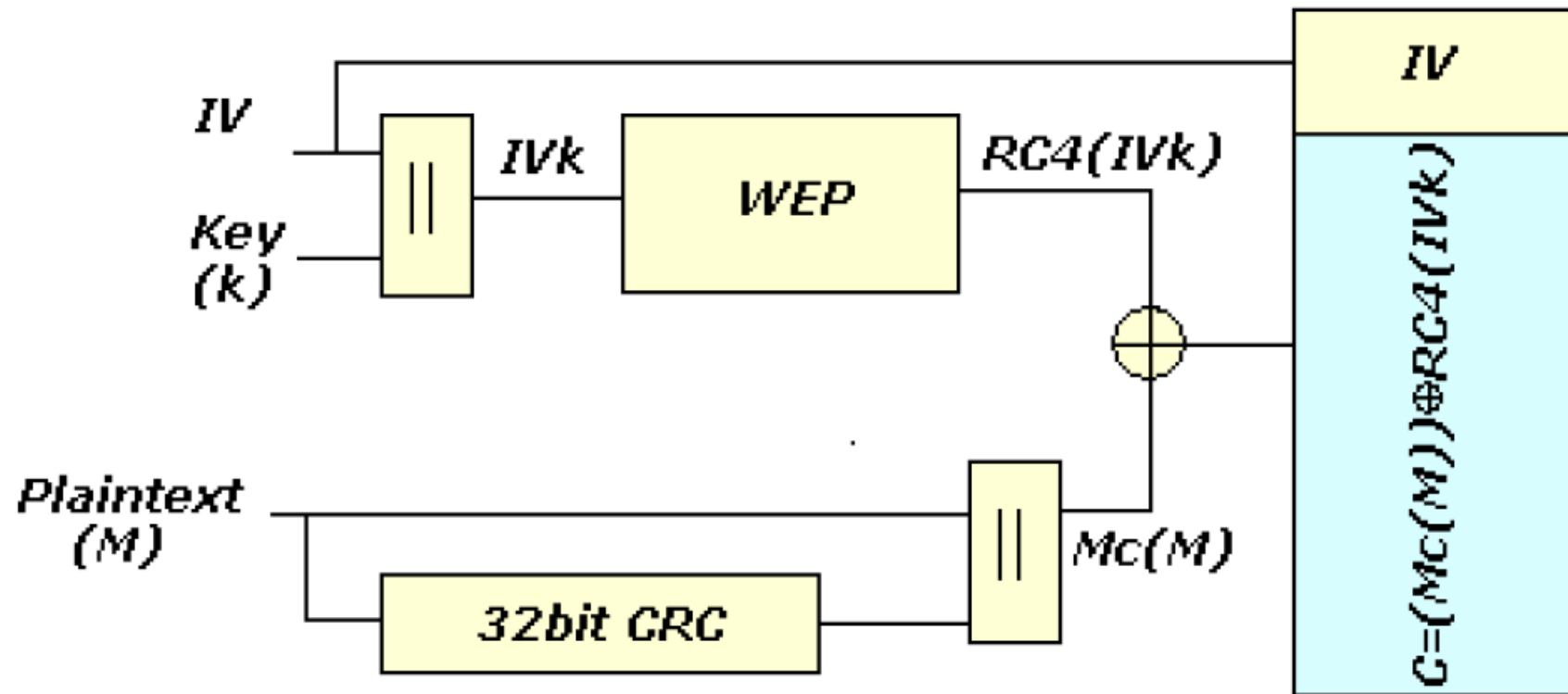
WEP: How it works

- To send a data packet:
 - Given the payload M , the 32 bits CRC $c(M)$ is calculated and concatenated to $M \rightarrow M \cdot c(M)$
 - The k key is concatenated to the IV defined for the packet $\rightarrow IV \cdot K$
 - The RC4 algorithm is initialized using this packet and a sequence of bytes is produced $\rightarrow RC4(IV \cdot k)$
 - Now $M \cdot c(M)$ is xor-ed with $RC4(IV \cdot k) \rightarrow C = (M \cdot c(M)) \oplus RC4(IV \cdot k)$
 - The 3 bytes of the IV are transmitted as clear text (together with the index of the WEP key)

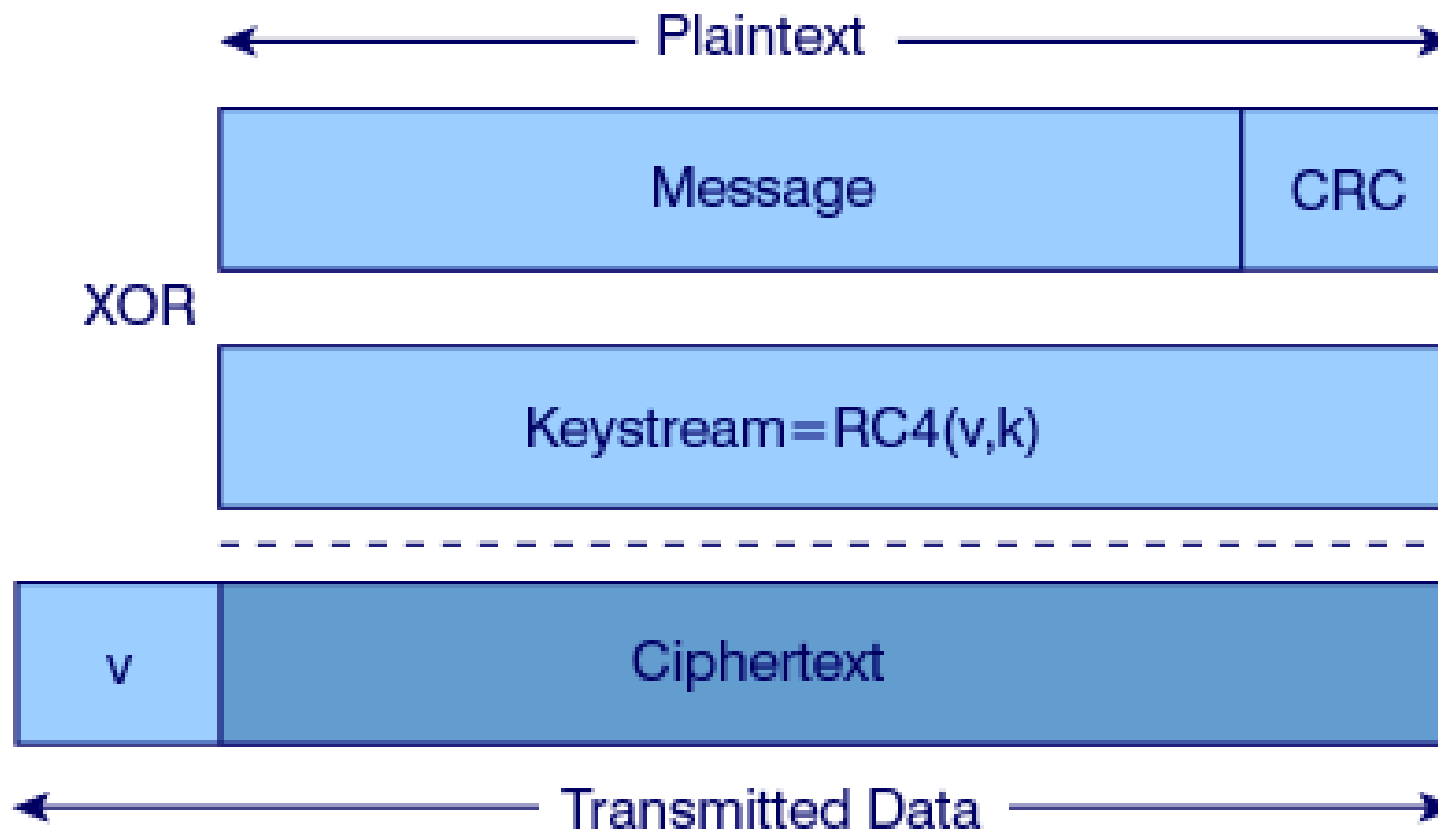
WEP: How it works

- ❑ The receiving side concatenates the IV received with the shared WEP key so that it can rebuild $RC4(IV \cdot k)$ → This is the reason for which the IV must be transmitted in clear text
- ❑ The receiving side decrypts the payload and if the CRC is equal then the packet is valid otherwise the packet is discarded

WEP: How it works



WEP: How it works




WEP: RC4

- Key Scheduling Algorithm
- RC4 uses a vector of status 256 octets long $S[256]$ and two counters i, j
- Initialization of the status:
 - $S[n] = n, i = 0, j = 0$
 - In the temporary vector T of 256 octets inserts the $IV \cdot K$ key, repeating it if short
 - S is run exchanging the elements of the vector:
for $i = 0$ to 255
 $j = (j + S[i] + T[i \bmod 8]) \bmod 256$
 swap ($S[i], S[j]$)

WEP: RC4

- Pseudo Random Generation Algorithm.
Generation of the keystream:
 - To generate an octet z of the keystream starting from the actual status (S, i, j) :
 - $i = (i + 1) \bmod 256$
 - $j = (j + S[i]) \bmod 256$
 - swap $(S[i], S[j])$
 - $t = (S[i] + S[j]) \bmod 256$
 - $z = S[t]$
 - At the beginning $i=0, j=0$ and discard T
 - The generation process continues until there is no more data



Weakness and Vulnerability of Wep

WEP: Reuse of the coding

- If we use the same IV, the same byte sequence (keystream) is generated from RC4
- Encrypting two messages p_1 and p_2 we have:
 - $C_1 = P_1 \oplus \text{RC4}(\text{IV} \cdot k)$
 - $C_2 = P_2 \oplus \text{RC4}(\text{IV} \cdot k)$
 - $C_1 \oplus C_2 = P_1 \oplus \text{RC4}(\text{IV} \cdot k) \oplus P_2 \oplus \text{RC4}(\text{IV} \cdot k)$
 $= P_1 \oplus P_2$
- So with the xor of two ciphered messages we get the xor of the two messages as clear text

WEP: Reuse of the coding

- ❑ If one of the two messages is known, the other is obtained
- ❑ If we have many messages codified with the same keystream it is easy to go back to the original messages
- ❑ The protocols impose many similarities to the packets!
- ❑ So: do not reuse the keystream
- ❑ But: 24 bits of IV means **16.777.216** different keystream: TOO FEW!

WEP: Reuse of the coding

- ❑ The standard recommends (but it is not mandatory) that the IV should change in a random way after every transmitted packets
- ❑ Some cards generate the 24 bits of the IV using a counter set at zero every time they are initialized and then they increase the counter of 1
 - this increases the probability that the key is reused (the low IV values are more frequent and always transmitted at the beginning of a session)

WEP: Brute Force Attacks

- It can use a list of “easy” keys
- Analyzing the whole research space given
 - Requires up to 45 days with 40 bits
 - Not feasible for 104 bits keys
- Two packets are enough in general (to be sure that the CRC does not coincide by chance also with a wrong WEP key)

WEP: Attacks Based on Weak IV

- S. Fluhrer, I. Mantin, A. Shamir have shown that some weaknesses exist in the algorithm of generation of the keys in RC4 → *“Weakness in the Key Scheduling Algorithm of RC4”*
- The attack described in their article, besides being extremely fast, requires a time which **increases linearly** with the length of the WEP key!

WEP: Attacks Based on Weak IV

- The fact that a large part of the key (3 bytes) is transmitted in clear and make the cracking easier:
 - The first three iterations of the KSA are easily deducible for the fact that the first three digits of the key are well known (remembered: the IV is transmitted in clear)!
- It is possible to see that there is a probability of 5% than the values in $S[0]$ - $S[3]$ do not change after the first 3 iterations of the KSA

WEP: Attacks Based on Weak IV

- It has been demonstrated that the IV of a certain type are subject to be cracked:

$$(B+3:255:x)$$

where B is the byte of the secret key (the WEP key) that we are cracking

- Then for every byte of the key there are 256 Weak IV

WEP: Attacks Based on Weak IV

- ❑ The first values of the encrypted data is the SNAP (*Sub Network Attachment Point*) header. It is a standard (of layer 2) for the transmission of IP datagram on IEEE 802 network
- ❑ The not encrypted header is AA in hexadecimal
- ❑ The xor of the first encrypted data with AA, will provide the first byte of the PRGA
- ❑ This information allows rebuilding the first digit of the WEP key if we have a Weak IV of the type (3:255:x)

AV1

WEP: Attacks Based on Weak IV

- We analyze the first step of the algorithm to produce the first byte of the keystream:

$$i = (i + 1) \bmod 256 \rightarrow i = 1$$

$$j = (j + S[i]) \bmod 256 \rightarrow j = S[1]$$

$$\text{swap}(S[i], S[j]) \rightarrow \text{swap}(S[1], S[S[1]])$$

$$t = (S[i] + S[j]) \bmod 256 \rightarrow t = S[1] + S[S[1]]$$

$$z = S[t] \rightarrow z = S[S[1] + S[S[1]]]$$

- So the first byte is function of:

$$S[1], S[S[1]] \text{ e } S[S[1] + S[S[1]]]$$

WEP: Attacks Based on Weak IV

- The first two steps of the generation of the vector S with IV (3:255:x) are the following:

- $i = 0, j = 0$

$S \rightarrow$	0	1	2	3	4
$T \rightarrow$	3	255	x	W_1	

- $i = 0, j = (j + S[i] + T[i \bmod 8]) = (0 + S[0] + T[0]) = 0 + 0 + 3 = 3 \rightarrow$
swap($S[0], S[3]$)

$S \rightarrow$	3	1	2	0	4
$T \rightarrow$	3	255	x	W_1	

WEP: Attacks Based on Weak IV

□ $i = 1, j = 3$

S →	3	1	2	0	4
T →	3	255	x	W_1	

□ $i = 1, j = (j + S[i] + T[i \bmod 8]) = (3 + S[1] + T[1]) = 3 + 1 + 255 = 3 \rightarrow \text{swap}(S[1], S[3])$

S →	3	0	2	1	4
T →	3	255	x	W_1	

WEP: Attacks Based on Weak IV

- At the next step $j = (3 + S[2] + T[2]) = (3 + (2 + x))$ that is j move forwards of $x + 2$ with x known
- Every IV behaves in different way depending on x , but we are able to rebuild the configuration of the vector S
- From here on the evolution of S depends on the key, and with a probability of 5% (as we said previously) the first 3 values of S do not change

WEP: Attacks Based on Weak IV

- Beyond the first byte of the key the operation gets complicated because it requires to go through the PRGA for several steps and so we could not be able to infer with a reasonable probability the exchanges of S
- Also other Weak IV families exist

WEP: Attacks Based on Weak IV

- ❑ Some producers of wireless cards have started building cards which avoid IV weak
- ❑ The space of IV available is further reduced (some thousands less)
- ❑ Observe that, to complete the attack, it is enough that only one client does not avoid the weak IVs



WPA Cracking

WPA

- ❑ Given the problem of WEP, IEEE started to work to the new standard: 802.11i
- ❑ WPA (released at the beginning of 2003) was intended as an intermediate solution till the final approval of 802.11i (2004)
- ❑ WPA supports many of the features of 802.11i (the design of the WPA protocol is based on the Draft 3 of 802.11i standard), but WPA is not an IEEE standard

WPA

- There are two variants of this protocol: enterprise and personal:
 - Enterprise: require an IEEE 802.1X authentication server, which will distributes different keys to different users
 - Personal: WPA utilizes the "pre-shared key" (PSK) mode, where all the computer will use the same passphrase

WPA

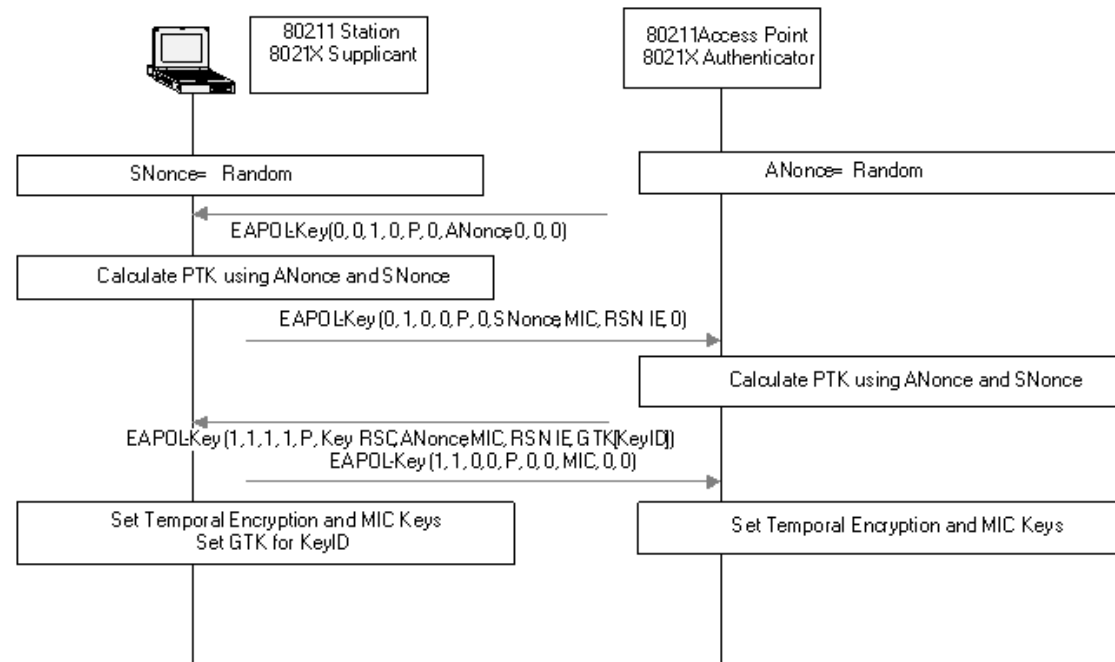
- ❑ The passphrase may be from 8 to 63 printable ASCII characters or 64 hexadecimal digits (256 bits)
- ❑ In PSK mode, security depends on the strength and privacy of the passphrase: the weak passphrases users typically select are vulnerable to password cracking attacks (off line attack based on dictionary)

WPA

- ❑ In WPA, every station is permitted to associate with the AP
- ❑ The AP has the option to start 802.1X authentication, exchanging Extensible Authentication Protocol (EAP) messages to verify user/server identities
- ❑ After authentication (if any), the AP start a four-way handshake to derive the keys for the session

WPA: 4 way handshaking

- The scheme of the 4 way handshaking is the following:



WPA: 4 way handshaking

- The AP sends a nonce-value to the STA (ANonce)
- The client now has all the attributes to construct the PTK (*Pairwise Transient Key* - 64 bytes)
- The STA sends its own nonce-value (SNonce) to the AP together with a MIC (*Message Integrity Code*)

WPA: 4 way handshaking

- ❑ The AP sends the GTK (*Groupwise Transient Key* – 32 bytes) with another MIC
- ❑ The STA sends a confirmation to the AP
- ❑ The GTK is used to decrypt multicast and broadcast traffic. This key has to be updated:
 - For the expiry of a preset timer
 - When a device leaves the network: this is to prevent the device from receiving any more multicast or broadcast messages from the AP

WPA: The Crack


- ❑ The four-way handshake occurs whenever you connect to a WLAN using WPA or WPA2. It also occurs periodically thereafter, whenever the AP decides to refresh transient keys
- ❑ If we use WPA-Enterprise, every session starts from a different PMK, delivered during 802.1X authentication
- ❑ In WLANs using WPA-Personal the station and AP must use the PSK!
- ❑ WPA-PSK crackers like aircrack, kisMAC, coWPAtty try to guess the PSK by capturing and analyzing the four-way handshake messages

WPA: The Crack

- A minimum PSK (8 lowercase letters) has 26^8 possible combinations (208.827.064.576), so trying all of those possible passphrases would take far too long
- Working from a dictionary file cuts that effort by just trying all 8-character words! Huge password dictionaries are readily available for use with conventional password crackers like “John the Ripper”, and they can be fed into PSK crackers

WPA: The Crack

- ❑ Multi-pass hashing for every word in those files still takes time (depending on SSID and PSK length)
- ❑ But conventional password crackers to speed up the cracking use rainbow tables (long lists of pre-computed password hashes). Use default SSID and words in dictionary files
- ❑ For example coWPAtty use a variation on rainbow tables to speed PSK cracking by three orders of magnitude



Airsnort & AirCrack-ng:
software for the cracking of
WEP/WPA keys

Airsnort

- ❑ Several tools exist which allow to determine in an automatic way a WEP key
- ❑ One of these is Airsnort, downloadable to the address:
<http://airsnort.shmoo.com/>
- ❑ It is a linux program now also for windows
- ❑ It requires the wireless card in monitor mode
- ❑ It works for instance with the cards Prism2, Orinoco and Cisco

Airsnort

- ❑ Once activated, the program captures the packets and simultaneously tries to crack the WEP key:
 - All the non data packets (except the beacon) are dropped
 - The packets not encrypted are dropped
 - The encrypted packets are selected and the ones considered not interesting are dropped
- ❑ The packets considered interesting are the Weak IV identified by Fluhrer, Mantin and Shamir (plus several *Weak IV* identified afterwards)

Airsnort

- ❑ Every 10 weak IV acquired, airsnort uses a probabilistic attack
- ❑ It is possible to define how deep the analysis of the tree of the various possibilities must be
- ❑ A value n of the parameter “breadth” indicates that the algorithm will try the n more probable values for each position of the key
- ❑ About 1000 weak IV for a key to 64 bits and about 2000 for a key to 128 bits are required

Airsnort

- Test of attack completed using:
 - An Access Point Avaya AP3
 - Two laptop to produce traffic
 - A laptop with a Netgear wireless card and Airsnort
- Set up a 64 bits WEP to, that is 40 key bits, that is 5 characters → WNLAB
- After about 15 minutes of acquisition with about 550.000 packets (540.000 encrypted) and 919 Weak IV, the key has been determined!

AirSnort

The screenshot shows the AirSnort application window. At the top, there is a menu bar with 'File', 'Edit', 'Settings', and 'Help'. Below the menu bar, there are several controls: a radio button for 'scan' (unselected) and a radio button for 'channel' (selected), with a dropdown menu showing '6'. To the right, there are two dropdown menus for 'Network device' (set to 'eth1') and 'Driver type' (set to 'Host AP/Orinoco'), with a 'Refresh' button next to the first. Further right, there are two spinners for '40 bit crack breadth' (set to 3) and '128 bit crack breadth' (set to 2).

C	BSSID	Name	WEP	Last Seen	Last IV	Chan	Packets	Encrypted	Interesting	Unique	PW: Hex	PW: ASCII
A	00:20:A6:50:DA:C1	WNLAB	Y	Thu May 19 17:07:05 2005	06:1F:AE	6	546271	538842	919	501752	57:4E:4C:41:42	WNLAB
	FF:FF:FF:FF:FF:FF			Thu May 19 17:14:22 2005	00:00:00		1003	0	0	0		

At the bottom of the window, there are three buttons: 'Start', 'Stop', and 'Clear'.

Airsnort

- In the following table some runs of Airsnort:

Key length	No. of packets	Encrypted packets	weak IV
40	283618	278860	120
40	546271	538842	919
40	283895	280098	100
40	283876	280057	102
40	AV5 702466	676083	252
104	285328	281596	104
104	285798	282076	850
104	575137	567385	933

Slide 44

AV5

25/4/2006: AirSnort su auditor live CD

Alessandro Villani, 4/26/2006

WPA: AirCrack-ng

- ❑ We need to configure the CISCO AP to support WPA-PSK:
 - SSID Manager:
 - ❑ Client Authentication Key Management
 - Key Management → Mandatory
 - WPA → Check the box
 - WPA Pre-shared Key → Choose a simple string (not less than 8 chars)
 - Encryption Manager:
 - ❑ Encryption Modes
 - Cipher → TKIP

WPA: AirCrack-ng

The screenshot displays the Cisco IOS Series Bridge - Security - SSID Manager web interface. The browser window title is "Cisco IOS Series Bridge - Security - SSID Manager - Mozilla Firefox". The address bar shows "http://192.168.10.5/ap_sec_ap-client-security.shtml".

SECURITY

- Admin Access
- Encryption Manager
- SSID Manager**
- Server Manager
- AP Authentication
- Intrusion Detection
- Local RADIUS Server
- Advanced Security
- SERVICES +
- WIRELESS SERVICES +
- SYSTEM SOFTWARE +
- EVENT LOG +

Configuration Fields:

- SSID: NCG
- VLAN: < NONE > (with a [Define VLANs](#) link)
- Backup 1: []
- Backup 2: []
- Backup 3: []
- Interface: Radio0-802.11G
- Network ID: [] (0-4096)

Client Authentication Settings

Methods Accepted:

- Open Authentication: < NO ADDITION >
- Shared Authentication: < NO ADDITION >
- Network EAP: < NO ADDITION >

Server Priorities:

EAP Authentication Servers

- Use Defaults [Define Defaults](#)
- Customize
- Priority 1: < NONE >
- Priority 2: < NONE >
- Priority 3: < NONE >

MAC Authentication Servers

- Use Defaults [Define Defaults](#)
- Customize
- Priority 1: < NONE >
- Priority 2: < NONE >
- Priority 3: < NONE >

Client Authenticated Key Management

Key Management: Mandatory CCKM WPA

WPA Pre-shared Key: [] ASCII Hexadecimal

WPA: AirCrack-ng

Cisco IOS Series Bridge - Security - Encryption Manager - Mozilla Firefox

File Edit View History Bookmarks Tools Help

http://192.168.10.5/ap_sec_ap-key-security.shtml

Customize Links Work

CISCO SYSTEMS

Cisco Aironet 1300 Series Wireless Bridge

Hostname ap ap uptime is 2 hours, 18 minutes

Security: Encryption Manager

Encryption Modes

None

WEP Encryption

Cisco Compliant TKIP Features: Enable Message Integrity Check (MIC)
 Enable Per Packet Keying (PPK)

Cipher

Encryption Keys

	Transmit Key	Encryption Key (Hexadecimal)	Key Size
Encryption Key 1:	<input type="radio"/>	<input type="text"/>	128 bit
Encryption Key 2:	<input checked="" type="radio"/>	<input type="text"/>	128 bit
Encryption Key 3:	<input type="radio"/>	<input type="text"/>	128 bit
Encryption Key 4:	<input type="radio"/>	<input type="text"/>	128 bit

Global Properties

Broadcast Key Rotation Interval: Disable Rotation
 Enable Rotation with Interval: (10-10000000 sec)

WPA Group Key Update: Enable Group Key Update On Membership Termination
 Enable Group Key Update On Member's Capability Change

Apply Cancel

WPA: AirCrack-ng

- ❑ To acquire data:
 - `airdump-ng -channel X eth0 -w /root/dump`
- ❑ Connect a client to the AP (we need to acquire the WPA handshake)
- ❑ Run the cracking tool:
 - `aircrack-ng -a 2 -w DICT.TXT dump-01.cap`

WPA: AirCrack-ng

□ An example:

```
bt ~ # airodump-ng
```

```
Airodump-ng 1.0 beta1 r857 - (C) 2006,2007 Thomas d'Otreppe  
Original work: Christophe Devine  
http://www.aircrack-ng.org
```

```
usage: airodump-ng <options> <interface>[,<interface>,...]
```

```
CH 2 ][ Elapsed: 1 min ][ 2008-04-26 00:01 ][ WPA handshake: 00:11:92:90:BD:00
```

BSSID	PWR	RXQ	Beacons	#Data, #/s	CH	MB	ENC	CIPHER	AUTH	ESSID
00:80:C8:B8:39:EB	0	0	3	0 0	1	22	WEP	WEP	<length: 0>	
00:11:92:90:BD:00	0	100	770	1009 2	2	1.	WPA	TKIP	PSK	NCG

BSSID	STATION	PWR	Rate	Lost	Packets	Probes
00:11:92:90:BD:00	00:15:00:3C:3B:1A	0	1- 1	0	1146	NCG
(not associated)	00:1E:4C:00:88:B2	0	0- 1	0	3	
(not associated)	00:90:4B:64:4D:E7	0	0- 1	0	4	

WPA: AirCrack-ng

□ An example:

```
Opening dump.pcap-01.cap
Reading packets, please wait...
Read 2591 packets.
```

#	BSSID	ESSID	Encryption
1	00:11:92:90:BD:00	NCG	WPA (1 handshake)
2	00:80:C8:B8:39:EB		No data - WEP or WPA

```
Index number of target network ? 1
```

```
Opening dump.pcap-01.cap
Reading packets, please wait...
      Aircrack-ng 1.0 beta1
[00:00:00] 0 keys tested (0.00 k/s)
```

```
      KEY FOUND! [ OVERVIEW ]
```

```
Master Key      : 50 F0 E1 87 B6 9E BA 03 CA 93 C3 FB 1F 97 A2 92
                  C8 04 D5 E7 C2 EF 11 28 A2 08 D5 CC A1 11 22 F6
```

```
Transcient Key : 1E 98 C0 6F BD A7 6F 2E 8E 49 B1 CF 1F CC 91 89
                  59 52 82 A7 BB C0 5E 09 44 07 26 4C 6D BA D3 7D
                  5B 3B 2A 77 7E 41 8C DC 37 3B 23 CA 17 2B E0 E3
                  36 B7 FB 72 73 B3 4D FD 11 B2 CD E5 C8 4A DA D9
```

```
HEAPOL HMAC     : 0E 36 DA 0D A9 88 C2 29 94 10 08 E6 AE 1C A0 F8
```



The Report

WEP and IV: the task

- Many wireless cards generate IV in a predictable way
- Today we will analyze the IV sequences for some wireless cards
- You have to work with two laptop:
 - One where you install the wireless card (verify also the integrated wireless network card)
 - The other in monitor mode to acquire and analyze the traffic

WEP and IV: the task

□ You have to:

- Acquire some data generated from the laptop you intend to analyze (ping the AP)
- Take a contiguous sequence of two or more packets
- Look at the first 3 bytes that are the IV!
- Restart the network card, to verify if the IV sequence start each time from the same value
- Do a deep analysis of the IV trying to demonstrate if they are random number of sequence

Airsnort & WEP Cracking: the task

- ❑ Change a WEP Key and verify the number of packets required to break it:
 - Set a simple (an ascii word) 64 bit WEP key
 - Set a complex (random hex) 64 bit WEP key
 - Set a simple (an ascii word) 128 bit WEP Key
 - Set a complex (random hex) 128 bit WEP key
- ❑ Use iperf to generate a lot of traffic
- ❑ Use a laptop with airsnort to break the key

AirCrack-ng & WPA Cracking: the task

- ❑ Change a WPA Key and verify the time required to break it:
 - Set a simple (ASCII dictionary word) 8 chars WPA key
 - Set a complex (random hex) 64 bit WPA key
 - Set a simple (ASCII dictionary word) WPA Key of length x with $8 < x < 64$
 - Set a complex (random hex) WPA key of the same number of bits of before
 - Use a laptop with aircrack-ng to break the key
- ❑ N.B.: consider as unbreakable the key if you don't find the key after one hour