ECE
Department of Electrical and Computer Engineering

جامعة بيرزيت
BIRZEIT UNIVERSITY

Security Issues of Wired Equivalent Privacy (WEP)

Dr. Abdalkarim Awad

Ethics

- The goal of this exercise to study the weakness of WEP.
- It is not intended to be used as a tool to steal information or to damage systems.

Objectives

- The main objective of this experiment is to understand how does WEP work
- To understand the weak points of WEP
- To understand ARP and IP packets
- And to write a C program to be able to crack a password of WiFi router that uses WEP

ECE

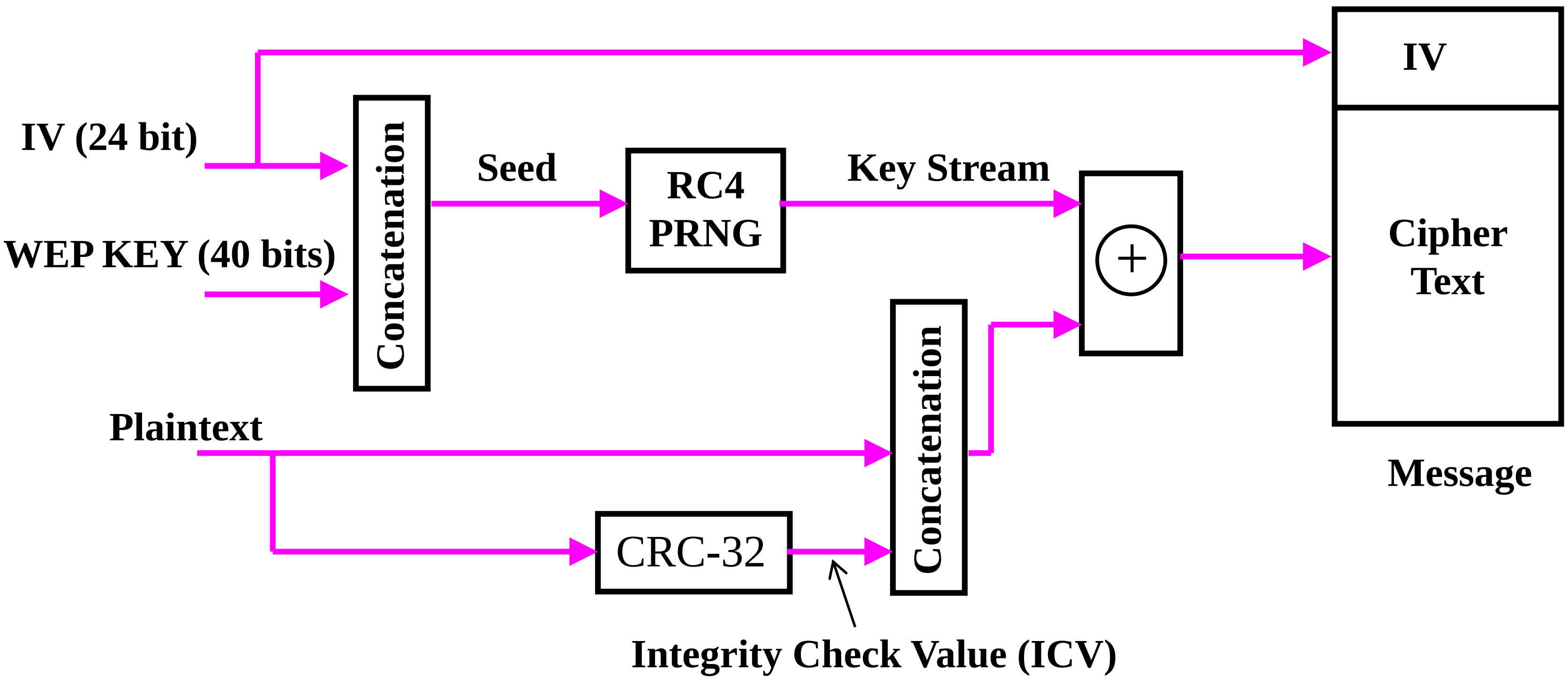
BIRZEIT UNIVERSITY

Requirements

- Wireless AP
- Wireless NIC
- Wireshark
- A C compiler

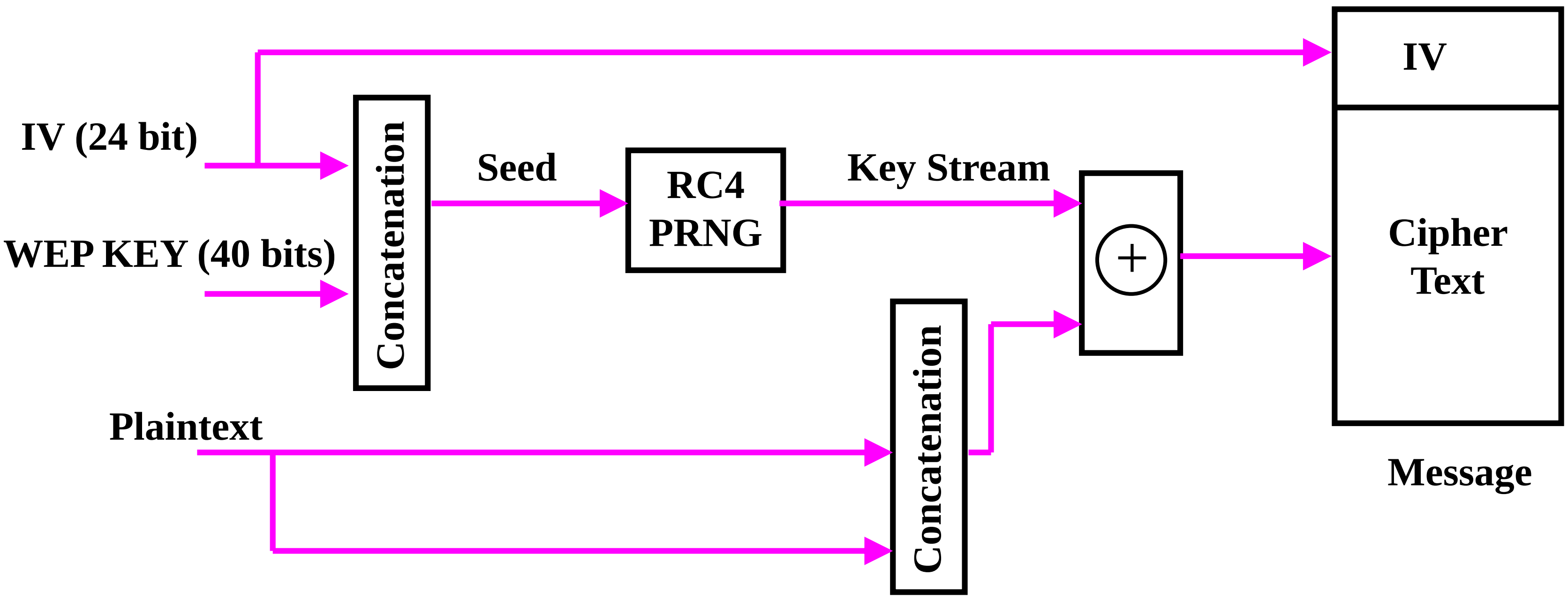
Wired Equivalent Privacy (WEP)

- WEP Encryption uses RC4 stream cipher



Wired Equivalent Privacy (WEP)

- WEP Decryption uses RC4 stream cipher



RC4

- a proprietary cipher owned by RSA DSI
- another Ron Rivest design, simple but effective
- variable key size, byte-oriented stream cipher
- widely used (web SSL/TLS, wireless WEP/WPA)
- key forms random permutation of all 8-bit values
- uses that permutation to scramble input info processed a byte at a time

RC4: Two Steps

- RC4 Key Schedule (Initialization)

```
for i = 0 to 255 do
    S[i] = i;
    T[i] = K[i mod keylen];
j = 0
for i = 0 to 255 do
    j = (j + S[i] + T[i]) (mod 256);
    swap (S[i], S[j]);
```

- RC4 Encryption

```
i = j = 0;
for each message byte M_i
    i = (i + 1) (mod 256);
    j = (j + S[i]) (mod 256);
    swap(S[i], S[j]);
    t = (S[i] + S[j]) (mod 256);
    C_i = M_i XOR S[t];
```

ECE

Example

- The size of S is 8 instead of 256
- K = [1 2 3 6]
- M = [4 2 3 2]
- The step is to generate the stream.
- Initialise the state vector S and temporary vector T. S is initialised so the S[i] = i,
- and T is initialised so it is the key K (repeated as necessary).
- S = [0 1 2 3 4 5 6 7]
- T = [1 2 3 6 1 2 3 6]

Example-Cont

- Now perform the initial permutation on S.

```
j = 0
for i = 0 to 7 do
    j = (j + S[i] + T[i]) (mod 8);
    swap (S[i], S[j]);
```

- at i = 0:
- j = (0 + 0 + 1) mod 8
- = 1
- Swap(S[0],S[1]);
- S = [1 0 2 3 4 5 6 7]

ECE

Example-Cont

- T = [1 2 3 6 1 2 3 6]
- at i = 1:
- `j = (j + S[i] + T[i]) (mod 8);`
- j = 3
- Swap(S[1],S[3])
- S = [1 3 2 0 4 5 6 7];
- At the end of itrations
- S= [2 3 7 4 6 0 1 5];

Example-Cont

- Now we generate 3-bits at a time, k, that we XOR with each 3-bits of plaintext to
- produce the ciphertext. The 3-bits k is generated by:

```
i = j = 0;
for each message byte Mi
    i = (i + 1) (mod 8);
    j = (j + S[i]) (mod 8);
    swap(S[i], S[j]);
    t = (S[i] + S[j]) (mod 8);
    Ci = Mi XOR S[t];
```

Example-Cont

- The first iteration:
- **S** = [2 3 7 4 0 1 6 5]
- i = (0 + 1) mod 8 = 1
- j = (0 + S[1]) mod 8 = 3
- Swap(S[1],S[3])
- **S** = [2 4 7 3 0 1 6 5]
- t = (S[1] + S[3]) mod 8 = 7
- k = S[7] = 5
- Remember, M= [4 2 3 2]
- So our first 3-bits of ciphertext is obtained by: k XOR P
- 5 XOR 4 = 101 XOR 100 = 001= 1

Example-Cont

- The second iteration:
- **S** = [2 4 7 3 0 1 6 5]
- i = (1 + 1) mod 8 = 2
- j = (2 + S[2]) mod 8 = 1
- Swap(S[2],S[1])
- **S** = [2 7 4 3 0 1 6 5]
- t = (S[2] + S[1]) mod 8 = 3
- k = S[3] = 3
- Second 3-bits of ciphertext are:
- 3 XOR 2 = 011 XOR 010 = 001 = 1

Example-Cont

- The third iteration:
- **S** = [2 7 4 3 0 1 6 5]
- i = (2 + 1) mod 8 = 3
- j = (1 + S[3]) mod 8 = 4
- Swap(S[3],S[4])
- **S** = [2 7 4 0 3 1 6 5]
- t = (S[3] + S[4]) mod 8 = 3
- k = S[3] = 0
- Third 3-bits of ciphertext are:
- 0 XOR 2 = 000 XOR 011 = 011 = 3

Example-Cont

- The final iteration:
- **S** = [2 7 4 0 3 1 6 5]
- i = (1 + 3) mod 8 = 4
- j = (4 + S[4]) mod 8 = 7
- Swap(S[4],S[7])
- **S** = [2 7 4 0 5 1 6 3]
- t = (S[4] + S[7]) mod 8 = 0
- k = S[0] = 2
- Last 3-bits of ciphertext are:
- 2 XOR 2 = 010 XOR 010 = 000 = 0

Example-Cont

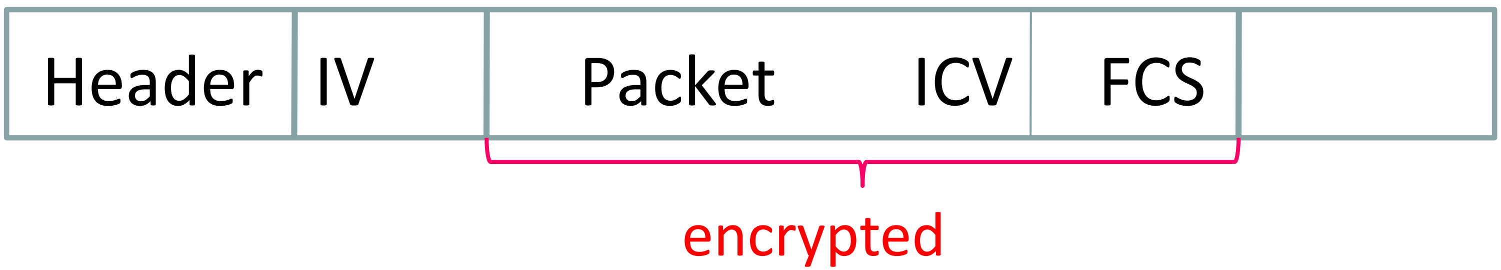
- So to encrypt the plaintext stream **M** = [4 2 3 2] with key **K** = [1 2 3 6] using our simplified RC4 stream cipher we get **C** = [1 1 3 0].

RC4 and WEP

- . WEP requires each packet to be encrypted with a separate RC4 key.
- The RC4 key for each packet is a concatenation of a 24-bit IV (initialization vector) and a 40 or 104-bit long-term key.

RC4 key:	IV (24)	Long-term key (40 or 104 bits)

802.11 frames using WEP



- ICV: integrity check value (for data integrity)
- FCS: frame check sequence (for error detection)
- Both use CRC32

ECE

WEP Vulnerability

- WEP protocol has several flaws
  - **Short IV length**
    - 24 bits IV not sufficient
  - **Clear text IV as part of the key**
    - 24 bits of every key in cleartext
    - Collect and analyze IVs to extract the WEP key

ECE

WEP decryption step-by-step

Step 1: Build the keystream

- Extract the IV from the incoming frame
- Prepend the IV to the key
- Use RC4 to build the keystream

WEP decryption step-by-step

Step 2: Decrypt the plaintext and verify

- XOR the keystream with the ciphertext
- Verify the extracted message with the some known data in the packet

ECE

Initialization vector (IV)

- It’s carried in plaintext in the “encrypted” message!
- It’s only 24 bits!
- There are no restrictions on IV reuse!
- The IV forms a significant portion of the “seed” for the RC4 algorithm!

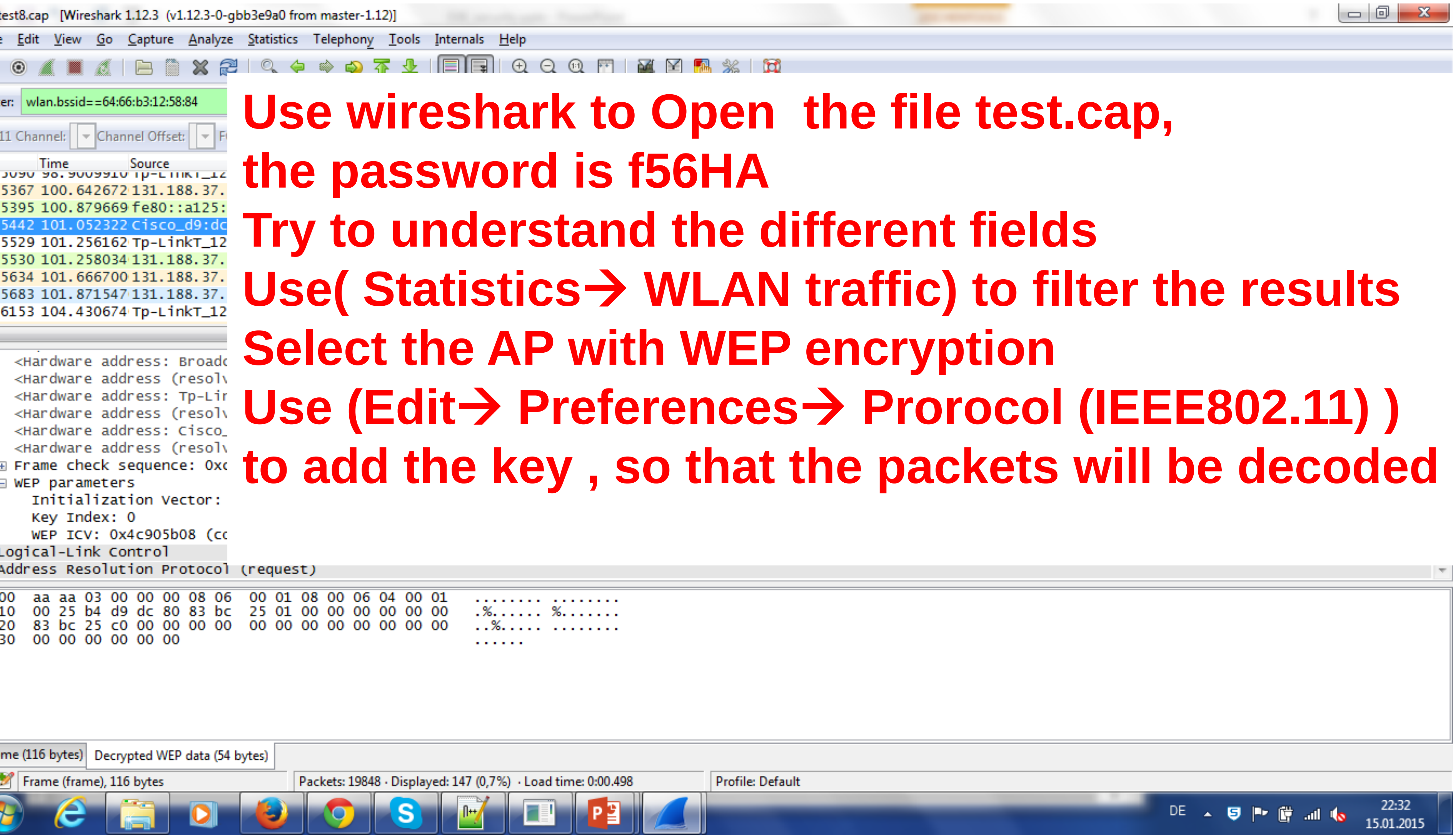
What do we know about the packets?

AA	AA	03	00	00	00	08	??
DSAP	SSAP	CTRL		ORG Code		Ether type	

Can be either IP or ARP

- With 802.11, you know the first eight bytes of a packet
- Many IP services have packets of fixed lengths
- Most WLAN IP addresses follow common conventions.
- Many IP behaviors have predictable responses
- The network part of IP address is known

Example



Use wireshark to Open the file test.cap, the password is f56HA

Try to understand the different fields

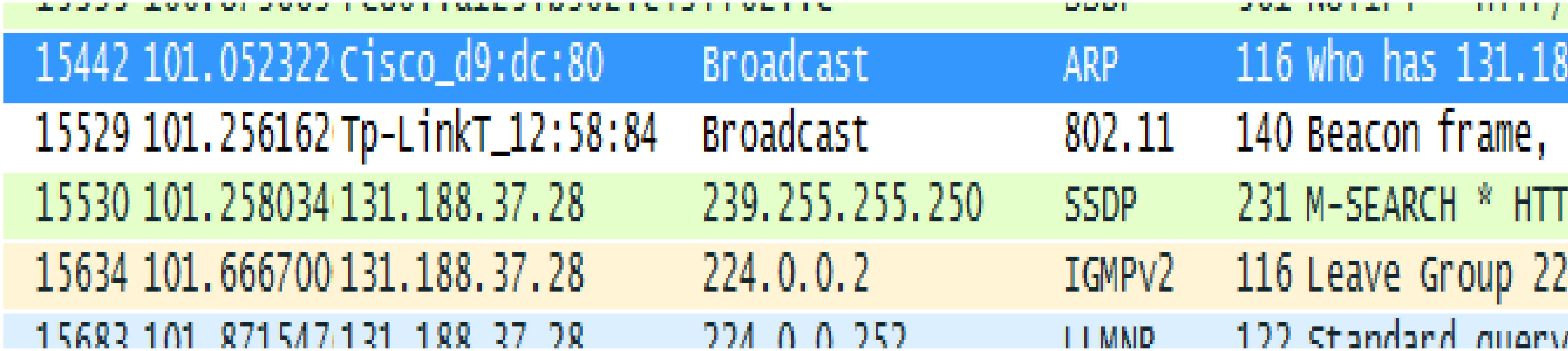
Use(Statistics→ WLAN traffic) to filter the results

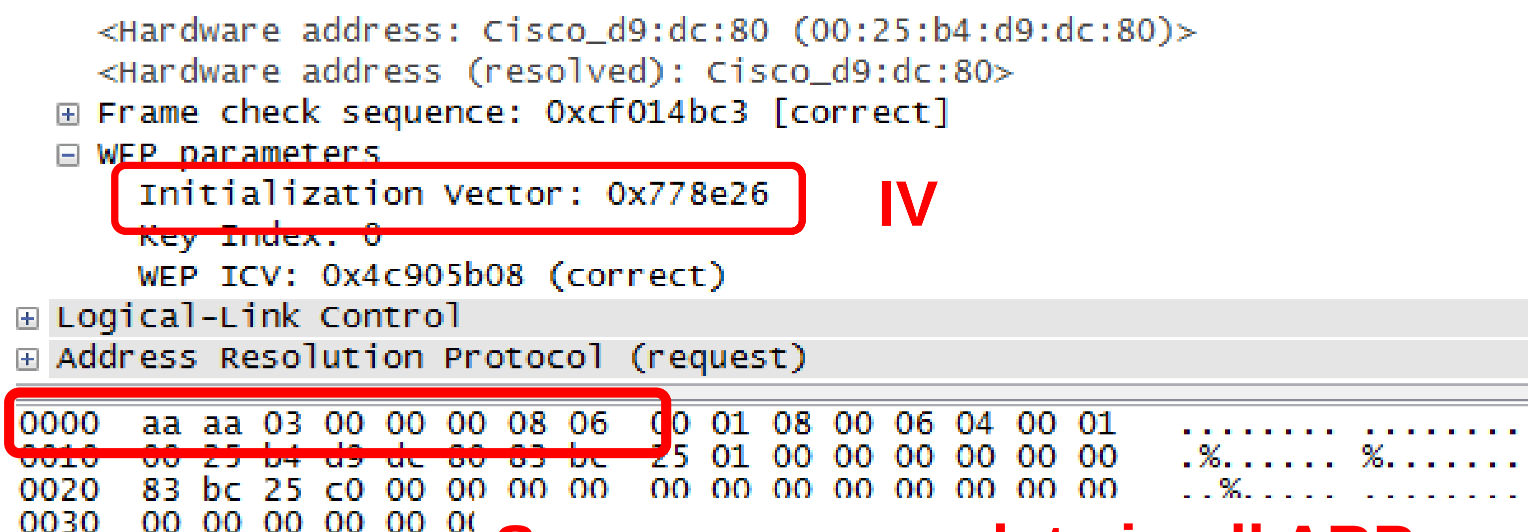
Select the AP with WEP encryption

Use (Edit→ Preferences→ Prorocol (IEEE802.11)) to add the key , so that the packets will be decoded

ECE

ARP packet





IV

Some common data in all ARP packets

Cracking the password

- Brute Force method
- Get the IV from an ARP packet (data packet)
- Get the encrypted data from the Packet as hex
- Assume the password consists from small/capital letters in addition to numbers
- Concatenate a 40 bits (5 chars) key to have the complete Key.
- Key schedule, obtain the vector S based on the key
- Using the encrypted data and S, decode the encrypted message and compare the results in byte 0, 1, 2,3,and 4, with 0xaa, 0xaa, 0x03, 0x00, 0x00, 0x00.
- If the results are true, then the password is cracked

ECE

Bibliography

- Smart Grid: Technology and Applications, 2012, ISBN 1119968682, Wiley, by Janaka Ekanayake, Kithsiri Liyanage, Jianzhong Wu, Akihiko Yokoyama, Nick Jenkins
- Smart Grid : Applications, Communications, and Security by Lars T. Berger and Krzysztof Iniewski
- Computer Networks A Top-Down Approach, James F. Kurose and Keith W. Ross
- Computer Networks A Top-Down Approach (Slides)
- Cryptography and Network Security, William Stallings
- Cryptography and Network Security Lecture slides by Lawrie Brown
- Security and Cryptography, Steven Gordon