802.11 Security – WEP

Shambhu Upadhyaya
Wireless Network Security
CSE 566 (Lecture 9)

Requirements of 802.11 WLAN Security

- **Encryption and Data Privacy**
  Requires mechanism to provide data privacy and integrity
  - The security mechanism should enforce the integrity of data under any circumstances
- **Authentication and Access Control**
  Authentication should be mutual
  - A framework to facilitate the transmission of authentication messages between clients, access points and authentication servers

WEP – Wired Equivalent Privacy

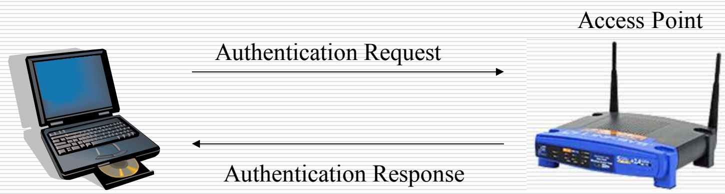
- WEP
  - Used to protect link-layer communications from eavesdropping and other attacks
  - Determines what encryption and authentication method is used to secure wireless data
- WEP has two types of Protection:
  - Secret Key
  - Encryption

WEP Description

- According to 802.11 WEP is:
  - Reasonably strong: Changing the Key (K) and Initialization Vector (IV)
  - Self synchronizing- critical for data link level encryption
  - Efficient- Both hardware and software implementations
  - It is Optional
- Two levels of security:
  - Open security – really means no security
  - Shared security – known secret key
- Authentication: Two parts to WEP
  - Authentication
  - Encryption

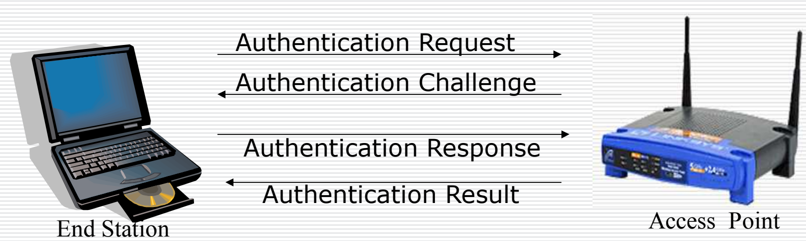
Authentication Sequence in 802.11

- Open Authentication



Authentication Sequence in 802.11

- Shared Key Authentication

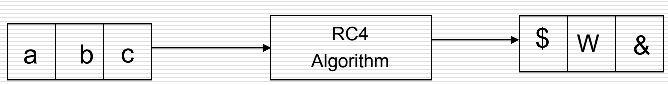


WEP Limitations

- Shared key authentication in WEP doesn't provide mutual authentication
- With 802.11 WEP, the AP and client stations on a particular WLAN must use the same encryption key
- A major problem with the 802.11 standard is that the keys are cumbersome to change
- There is no key management provision in the WEP protocol
- So, there is no security if many users sharing the identical key continue to use for long periods of time
- If one station is lost or stolen, it will threaten the security of all stations using this key

WEP

- WEP uses stream ciphers
- Stream ciphers - Sequence of ordinary data to sequence of encrypted data
- WEP uses RC4 Algorithm to encrypt data
- Initialization Vector (IV) + Secret key → Combined RC4 Key



RC4

- Encryption algorithm used to encrypt the data sent over the airwaves
- Scrambles each and every byte of data sent in a packet
- RC4 consists of two parts:
  - The Key Scheduling Algorithm (KSA)
  - The Psuedo Random Generation Algorithm (PRGA)

Key Scheduling Algorithm

- First part of the encryption process
- Algorithm
  - 1. Assume N = 256
  - 2. K[ ] = Secret Key array
  - 3. Initialization:
  - 4. For i = 0 to N – 1
  - 5. S[i] = i
  - 6. j = 0
  - 7. Scrambling:
  - 8. For i = 0 ... N – 1
  - 9. j = j + S[i] + K[i]
  - 10. Swap(S[i], S[j])

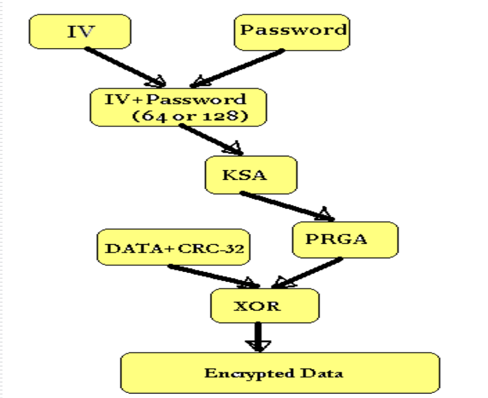
Pseudo Random Generation Algorithm

- Pseudo Random Generation Algorithm outputs a streaming key based on the KSA's pseudo random state array
- Streaming key + plaintext data ---> stream of encrypted data
- Algorithm
  - 1. Initialization:
  - 2. i = 0
  - 3. j = 0
  - 4. Generation Loop:
  - 5. i = i + 1
  - 6. j = j + S[i]
  - 7. Swap(S[i], S[j])
  - 8. Output z = S[S[i] + S[j]]
  - 9. Output XORed with data

Cyclic Redundancy Checksum

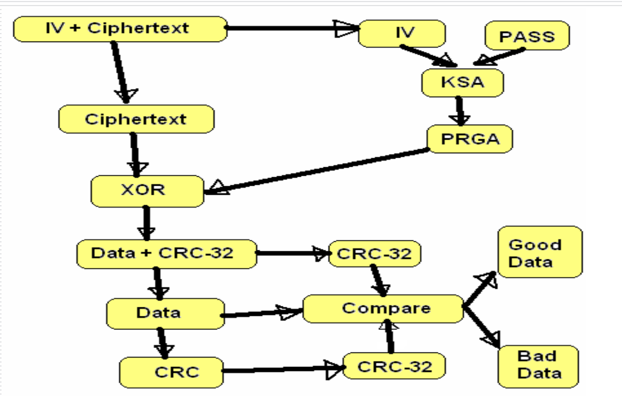
- Final part of the data-transmission process
- Used to check the integrity of the transmitted data
- CRC – comparing the CRC before packaging the data and after transmission
- NEW CRC matches ORIGINAL CRC – Packet is complete else corrupted

Encryption Algorithm



RC4 Encryption Process

Decrypting algorithm



RC4 Decryption Process

Cracking WEP

- Vulnerabilities in WEP
  - IV is sent as plaintext with the encrypted packet. Hence by sniffing it is easy to find the first 3 characters of the secret key
  - The KSA and PRGA leak information during the first few iterations of their algorithm
    - The i will always be 1, and j will always equal S[1] for the first iteration of the PRGA
    - KSA is easily duplicable for the first three iterations as the first 3 characters of the secret key are passed as plaintext
  - XOR is a simple process that can be easily used to deduce any unknown value if the other two values are known

Cracking WEP – FMS Attack

- Fluhrer Mantin Shamir (FMS) Attack
  - Identified certain IVs that leak information about the secret key
  - Reduces the key space so brute force is practically possible
  - This assumes that the attacker has knowledge of the first few bytes of plain text
  - Because of RFC 1042 (SNAP headers), all IP and ARP packets always start with 0xAA
  - Therefore, the first few bytes of plaintext are always known
  - Requires collection of ~ 500,000-2,000,000 packets and < 1 minute cracking time
  - Works with both 40-bit and 104-bit independent of how the key is generated

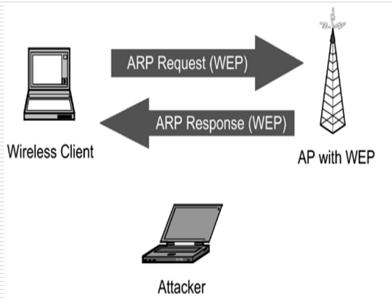
Cracking WEP – FMS Attack

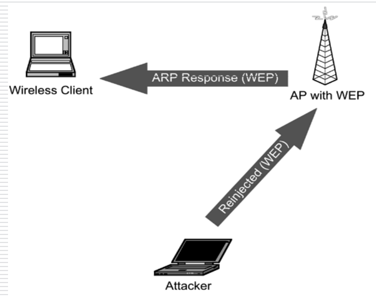
- This attack requires huge amount of data collection
- In a high traffic network, this can be accomplished in a matter of hours
- However, in a low traffic environment, this process can take days or weeks
- To expedite this process some attackers artificially generate network traffic in order to capture cipher text to crack the key

Cracking WEP – FMS Attack

- One possible packet injection attack works like this:
  - The attacker captures an encrypted text looking for a known protocol negotiation based on the size of the packet. E.g., an ARP request has a predictable size (28 bytes)
  - Once captured, the attacker re-injects the encrypted packet (ARP request) over and over again
  - The ARP response will generate new traffic, which the attacker can then capture
  - If the attacker repeats this process over and over again, it is possible to generate enough traffic for a successful FMS attack in about an hour

Cracking WEP – FMS Attack





The attacker captures a legitimate, encrypted packet and guesses that it is an ARP request based on a known size (28 bytes)

The attacker floods the network with the re-injected ARP request. This results in a flood of ARP responses, which the attacker captures as part of an FMS attack

Methods to crack WEP

Vulnerabilities of WEP:

- WEP key recovery
- Unauthorized decryption and the violation of data integrity
- Poor key management
- No access point authentication

- Tools used to crack WEP keys
  - WEP crack (https://sourceforge.net/projects/wepcrack/)
  - Air snort (https://airsnort.soft112.com/)
  - Many more…..

Improving WEP's Security

- Recommended Practice includes
  - Per-link keys
    - Unique key per STA
  - IV Sequencing
    - Check for monotonically increasing IVs
    - Weak IV avoidance
  - 104-bit keys
    - IV + Key = 128-bits
  - Rapid Rekey
    - Derive WEP keys from master key
    - Change encryption key frequently

Suggested Improvements to WEP

- IV Sequence check protects from both intentional and unintentional IV reuse. Protection from IV reuse makes it harder to mount attacks
- Longer Key requires adversary to acquire more packets for key recovery
- Authenticated Key Refreshing provides a secure and synchronized mechanism for re-keying
- Frequent rekeying makes it harder to recover (derived) encryption key. Even if key is cracked, it's only the temporal encryption key
- MAC-Layer Rekeying allows for faster refresh
- Implementation is backward compatible. All improvements are additions on top of current WEP implementations

References

- Jon Edney and William Arbaugh, Real 802.11 Security, Addison-Wesley, 2004 (Chapter 6)