Week 5 - “Don’t Roll Your Own”

“Don’t roll your own” = never try to make your own cryptographic function. Will always break

Wired Equivalent Privacy (WEP)

- Stream cipher made by a group of people with good intentions, but is a very insecure at what it does.
- Wired Equivalent Privacy is a **security protocol** that is designed to provide a wireless local area network (WLAN) with a level of security and privacy comparable to what is usually expected of a wired LAN
- This is secure, the key must **never** be used twice.
- WEP uses a 24 bit initialisation vector (IV) to produce 2^24 keystreams, which means that with enough trace, a key will be guaranteed to be used again. Using XOR, they can figure out the plaintext
- **Process**:
    - A stream of bits (A) is sent from one point to another
    - Another random stream of bits (B) is generated using an algorithm, such as RC4 (very insecure)
    - A (xor) B -> C (Encrypted Stream of bits)
    - However, C (xor) B -> Gives back A again (XOR can be reversed)
        - The XOR function turns A into a stream of bits with the statistical properties of randomness. (Since something random XOR something else will also result in something random)
- Danger when someone transmits the same data under the same key - same data in the same frame
- Fragmentation Attack
    - intercept packet, obtain keystream•construct IP header in 4 byte fragments•concatenate with encrypted payload•IP header points to Internet host controlled by attacker•send packets to AP (Access Point) which forwards along to Internet host
    - The big problem: data & control mixed

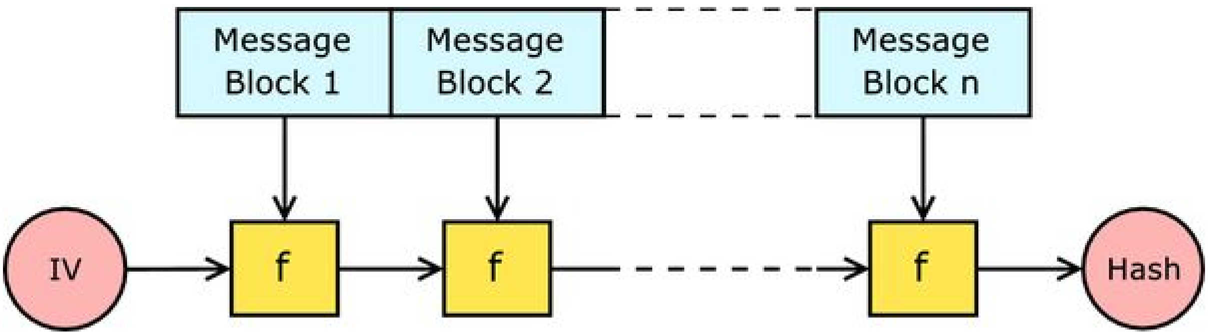
Hashes

- MD5, SHA0, SHA1 (161 bit digest; high collision) - broken
- SHA2, SHA3 - not broken
- SHA3 only one **not** vulnerable to length extension attacks - a type of attack where an attacker can use Hash(message1) and the length of message1 to calculate Hash(message1 ‖ message2) for an attacker-controlled message2, without needing to know the content of message1
- What does it mean to be "broken"? A: Property can be violated faster than brute force (birthday attack, if you can find 2 of the same hash more frequently than root(number of possible hashes)); reveals an underlying structural flaw in the hash

- What is a **birthday attack**?
  - The birthday attack is used as an idea around how good hash functions should be designed by taking the example of birthday paradox. It says that in a room full of people (N), the amount of guesses it takes to find two people with the same birthday on average is sqrt(N). The space here is 365 > N.
  - This is because if you have N people, the probability that two people have the same birthday is 1 - Prob(Everyone has unique birthdays).
    - If you'd want this probability to be more than 0.5 (highly likely) then it turns out you need close to sqrt(n) tries at most (mathematically derived). Lower number of people => lesser guesses needed, but also lesser chances of finding collision.
  - Applying the same analogy on a hash function, it should take sqrt(n) tries to find two inputs in a group of inputs (size n) that hash to the same value. Higher the value of sqrt(n), the stronger the hash function (It takes longer to find hash collision), so that it becomes computationally infeasible to find collisons.
  - A birthday attack won't work if the passwords are salted

Merkle Damgard Construction

Block cipher which builds **collision-resistant cryptographic hash functions** from collision-resistant one-way compression functions (used in the design of MD5, SHA1, SHA2).



1. break message into a series of blocks (512); may require padding if not divisible by 512
2. start with IV (initialisation vector)
3. Given hash function f, it takes in IV and block 1, output goes into f again with block 2 etc.
4. After doing this with all blocks, final hash is produced

- **Bank message problem:**
  - Insert shared secret at the front of the hash, as the first block
  - MAC = h(key | data) - his hash function, key|data is the password concatenated with the message

- **Flaw - length extension attack**:
  - If someone has sent a message, and you have the hash and the message length, you can **extend the message** to say whatever you want.
  - You can take the hash you have, pass it into f again with a new message, thereby extending the original message.

- What happens if you put the password on the end of the message, instead of the beginning?

Digital Signatures + RSA

- Digital Signatures **RSA** (Ron Rivest, Adi Shamir and Leonard Adleman), DSA(Digital Signature Algorithm)
- If you ever need to sign a large file, **hash the file** then **sign the encrypted hash**.
- Digital signature is a mathematical scheme for verifying the authenticity of digital messages or documents. A valid digital signature, where the prerequisites are satisfied, gives a recipient very strong reason to believe that the message was created by a known sender (authentication), and that the message was not altered in transit (integrity)
- RSA relies on people not being able to factorise the large number that is the mod, if someone is able to efficiently find the factors of a number, RSA will break.
- Signatures can be moved from one document to the other if they have the same hash
  - Based on collision attack described below

Hash Collisions

- Summary: **you can make it look like someone signed something they didn't**
- Half the number of bits in the hash size needed to do a collision attack. Why are collisions bad? (ans. above)
- **Collision resistance: It should be hard to find any two things that have the same hash**
- Example:
  - In a PDF that says "I promise I will give you $100", you receive a signature with the original document. In interest of attacker to change the amount to something higher.
  - If they find another pdf with the same hash, the signature can apply to both.
    - **How**? Change 1 bit in each document that doesn't change anything visible in the document, and keep hashing them, until you find 2 identical hashes.
  - After finding the collided hashes, there is a hash for the document that says $100, which now also maps to a document that says you'll give $1,000,000.
  - The attacker can now move the signature to the new document which says you'll give $1 mil

Key Stretching + Salting

- Types of password attacks:
  - Online: you type the password into the browser (manual)
    - can be locked out by site
  - Offline: you steal the file containing the passwords (likely hashed) and attempt to decrypt them locally
- **Salting** a hash is a random string that gets **appended** to the password before it gets hashed. Salt can be a single string, or a unique string for each user. If your salt is unique for each user, **each user's password is almost guaranteed to be unique before hashing**. The goal is to make the amount of work to decode a hash as great as possible, and to prevent attackers from using precomputed rainbow tables. (which are a collection of hashes for common passwords).
- **Key stretching** is designing a hash to be **slow**, such that it adds even more work for a hacker to decrypt a table of passwords. The added time for an individual user would be negligible by comparison. Examples are Crypt, bcrypt, script.

Data & Control

- Separate data and control - Users have access to data, but they shouldn't have access to control (eg. the captain crunch case)
- Separating data and control completely seems to be 'an impossible problem'.
- Famous example: Captain Crunch whistle. Something about phone lines providing free calls when a 2600hz tone is heard (user-provided audio (data) affecting control). Can be recorded and played back, or reproduced, eg with a whistle that was given away with Captain Crunch cereal that happened to be 2600hz.