Lecture 11

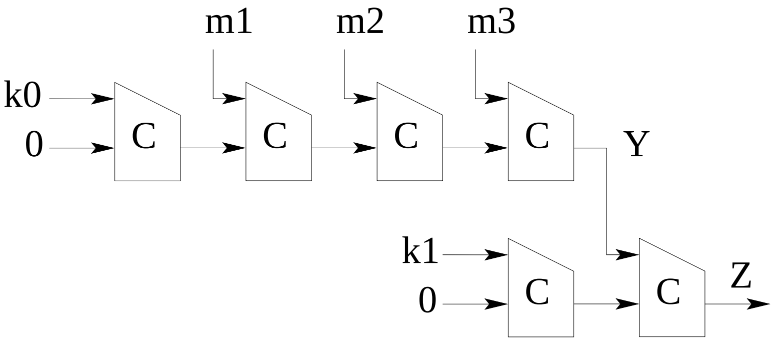
In this lecture we look at the construction of NMAC, HMAC and CBC-MAC, as well as attacks on broken systems like SSLv2, SSH1 and 802.11 WEP.

1 MAC Constructions

1.1 NMAC

The NMAC is constructed using compression functions and two keys k_0 and k_1 as follows

$$NMAC_{k_0,k_1}(m) = H(k_1||H(k_0||m))$$



To show that NMAC is a good MAC we need to show that is acts as a good PRF. In two steps we need to show that the top part producing $Y \leftarrow H(k_0||m)$ is a 2-universal hash and that the bottom part producing $Z \leftarrow H(k_1||Y)$ is a good PRF.

1. We don't need the top part to be a strongly 2-universal hash function, we just need the probability of a collision to be low. The key k_0 is chosen at random after the message is provided and the output Y of the top compression function is a random value that's not accessible. This makes the probability of messages colliding low.

2. The bottom part will act as a PRF if the inputs to the final compression function are random. Thus, the output $Z \leftarrow H(k_1||Y)$ on a random input key k_1 and random looking input Y will behave as an output of a PRF.

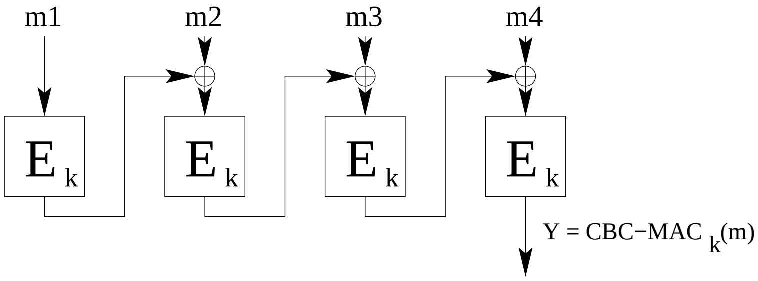
1.2 HMAC

NMAC was a thought experiment by the makers of HMAC. HMAC is a tweak to NMAC to have one key but gives up some elements of NMAC. Single key functionality could be obtained by using NMAC with a single key of double the length, but HMAC allows for a shorter key. HMAC is constructed with constants C_0 and C_1 as

$$HMAC_k(m) = H(k \oplus C_1||H(k \oplus C_0||m))$$

1.3 CBC-MAC

A MAC that's used widely in practice is the CBC-MAC. It is constructed by cipher-block chaining and is nice block ciphers are readily available. The construction is illustrated below.



$CBC\text{-}MAC_k(m)$ can use AES as the block cipher with a single key k. It is secure as a MAC if used with fixed length messages, but not if messages can be of varying lengths. If messages can be of varying lengths, it's easy to construct two messages m and m' such that their CBC-MACs will be the same. Let m be any one-block sized message. Then $CBC\text{-}MAC_k(m) = AES_k(m)$. We can construct the second message $m' = m||AES_k(m) \oplus m$ and it's easy to see that $CBC\text{-}MAC_k(m') = CBC\text{-}MAC_k(m)$. Thus for variable length messages with common prefixes, CBC-MAC is clearly not a good MAC.

CBC-MAC is however a 2-universal hash. It is not strongly 2-universal, but almost 2-universal up to ϵ. If collision probability is low, then it is ϵ-weak 2-universal. This is also true for variable length messages. When used with an n-bit block cipher, the chances that outputs of CBC-MAC for two messages are the same are approximately $1/2^n + \epsilon$ where ϵ is the insecurity of the block cipher used.

2 Broken Systems

2.1 SSL

SSL is a application layer protocol created for managing the security of message transmissions in a network. SSL version 1 was totally broken and never saw the light of day. The flaws described here are for SSL version 2. In use today are SSL versions 3 and 3.1 as well as TLS.

Every message m is transmitted in the form

$$len(m)||E_k(m||pad)$$

The message is padded and encrypted with a shared secret key. The length of the message is prepended in the clear to the ciphertext. The same key is used in both directions of a bidirectional channel between a browser and a web server.

2.1.1 Attacks

1. Since the same key is used for communication in both directions, the protocol is vulnerable to a reflection attack. A message can be recorded going in one direction

and replayed in the other. This gives third parties access to add messages.

2. The length of the message is not encrypted and not protected in any way. An attacker can change the length field and can truncate messages easily. There is also no mechanism that protects the integrity of the message.

2.1.2 Lessons

1. A good protocol needs key separation to avoid reuse of keys in multiple contexts.
2. Any value affecting the processing of a message needs to be MACed.

2.2 SSH

SSH was intended as a secure way of client-server message communication. Client and server exchange session keys in a secure way and use them to communicate throughout the session. The server maintains and changes an hourly key hk, and uses it to generate two session keys $(k, k') = PRG(hk)$. The protocol uses the two different keys in different directions.

Message format is as follows

$$\begin{aligned} Client \rightarrow Server &: RC4(k) \oplus (m||CRC(m)) \\ Server \rightarrow Client &: RC4(k' \oplus SID) \oplus (m||CRC(m)) \end{aligned}$$

As the server temporarily uses same hk to get k, k', it uses the session id SID with RC4 to produce the ciphertext. The protocol forgets to XOR k with the session id for client to server communication, thus creating a two time pad. For confidentiality, the protocol wants RC4 to be like a PRG, producing new output to be used as a one time pad.

2.2.1 Integrity

CRC is linear and can be modified if its XORed with δ as $CRC(m) \oplus CRC(\delta) = CRC(m \oplus \delta)$. By letting C be the original ciphertext $RC4(k) \oplus (m||CRC(m))$ a man in the middle can perform bit-flipping to transmit an acceptable message by forwarding $C \oplus (\delta||CRC(\delta))$.

Fixing this by replacing CRC with SHA1 is not enough as this modification is still vulnerable to known plain text attacks. The attack can be carried out if the attacker knows the ciphertext $C = RC4(k) \oplus (m||SHA1(m))$ and the message m. The attacker can replace ciphertext with $C \oplus (m \oplus m'||SHA1(m) \oplus SHA1(m'))$ to replace the message m with m'. To ensure message integrity the it would be best to use a MAC.

2.3 WEP

WEP is used for wireless encryption of IP packets and has every flaw in the world. The message format is

$$IV, RC4(k, IV) \oplus (m||CRC(m))$$

where k is the network key for the entire network and IV is a 24-bit random value, which in reality is an incremented counter starting at 0. All members of the network have and use the same key. WEP is a stateless protocol, which creates the need for an IV that is prepended to the message and sent in the clear.

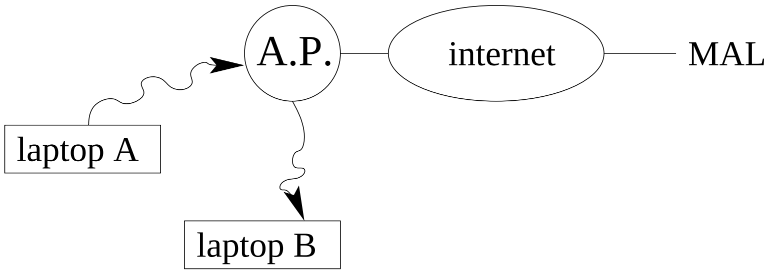
2.3.1 Attacks

1. Bit flipping due to the linearity of CRC. Same attack as in SSH.

2. An attacker can gain access control with the knowledge of one plaintext message m and the corresponding ciphertext. With m and the ciphertext, the attacker can recover the value of $RC4(k, IV)$ and use it to inject packets with the same IV. The IV is prepended in the clear and the it's reuse is possible as the protocol is stateless. In practice it is very easy to get known plaintext and eavesdrop of the ciphertext of ARP or DHCP packets transmitted at startup.

3. Key stream reuse is possible due to the short IV. With a 24-bit IV an attacker needs 2^{24} packets to wrap around and get a 2-time pad. Since the key k is shared by the entire network, an attacker needs 2^{24} pads from the whole network, not just one user. An IV of 0 counter is common as with some implementations, after sleep a machine starts up again with the IV counter set at 0.

4. Dictionary attack of 2^{24} known texts is also possible. An attacker can send 2^{24} packets to someone on the network and eavesdrop on the network Access Point encrypting the messages coming from the outside as it forwards it to a destination inside the network. This attack does require a long time to complete.

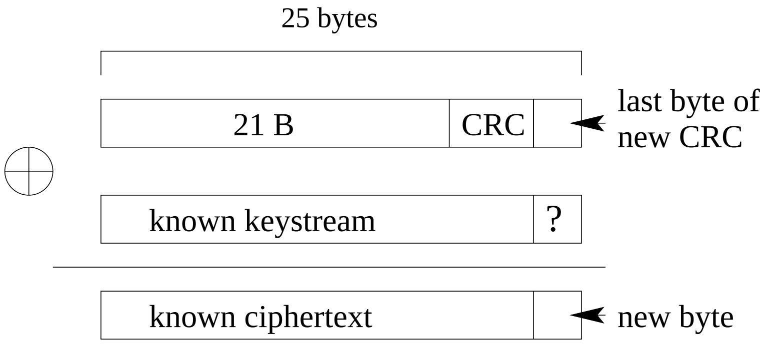


5. A packet redirection attack takes advantage of the fact that the Access Point is also a router. The attacker MAL somewhere on the Internet needs to intercept packets from A to B on the same wireless network. He needs to flip the bits in the IP header to change the IP dest to MAL's address on the outside. The Access Point will forward the packet to MAL in the clear without re-encrypting it. All the attacker needs to know is the original IP address of B to flip the bits in the header correctly. This is clear example of why INT-PTXT is a useful notion.

6. Fluhrer, Mantin, Shamir key recovery attack. The attack exploits funnyness of RC4 and needs to know second byte of key stream for RC4. This is if the first bytes of packet were constant the key can be recovered with some work. The WEP protocol is fine if different keys are used with RC4, but not so with $k||IV$ as then RC4 is ran on closely related seeds. With the usage of RC4 as a PRG, nothing is guaranteed

if seeds are related as a PRG promises nothing in this situation. Some seeds are weak and reveal info about other bytes of the seed. This attack was devastating and convinced people that WEP was broken. Proprietary solutions to WEP change keys pretty frequently.

7. An inductive attack assumes a short known segment of the key stream recovered from known plain text such that for APR packets. Given an IV and the first 24 bytes of the key stream, the attack extends the key stream one byte at a time. This attack completely breaks WEP as given an IV, the attacker can figure out what the value of RC4(k, IV) is and (a) can decrypt and read any packet and (b) can inject new packets.



Given 24 bytes of a ciphertext, the key stream and the concatenation of a 21 byte message with its 3 byte CRC, the attacker guesses the next byte, send the new packet and sees if there is an ack for the packet. There is an ack if the CRC for the new message checks out. If the CRC checks out, then the attacker can reconstruct the next byte in the key stream, as he knows what the CRC of the new 22 byte message should be. If not, the attacker tries the next 255 possibilities.

The attack uses the ability to take a ciphertext and use the recipient as an oracle to get 1 bit of information, instrumenting a reaction attack. This shows the need for chosen ciphertext confidentiality with IND-CCA2.