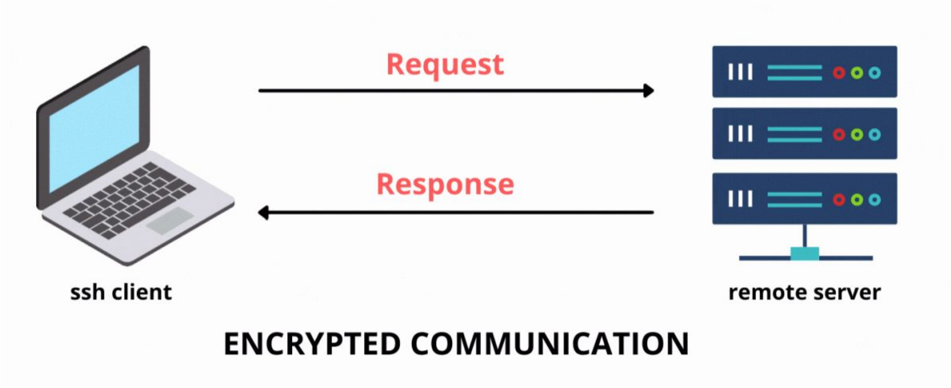
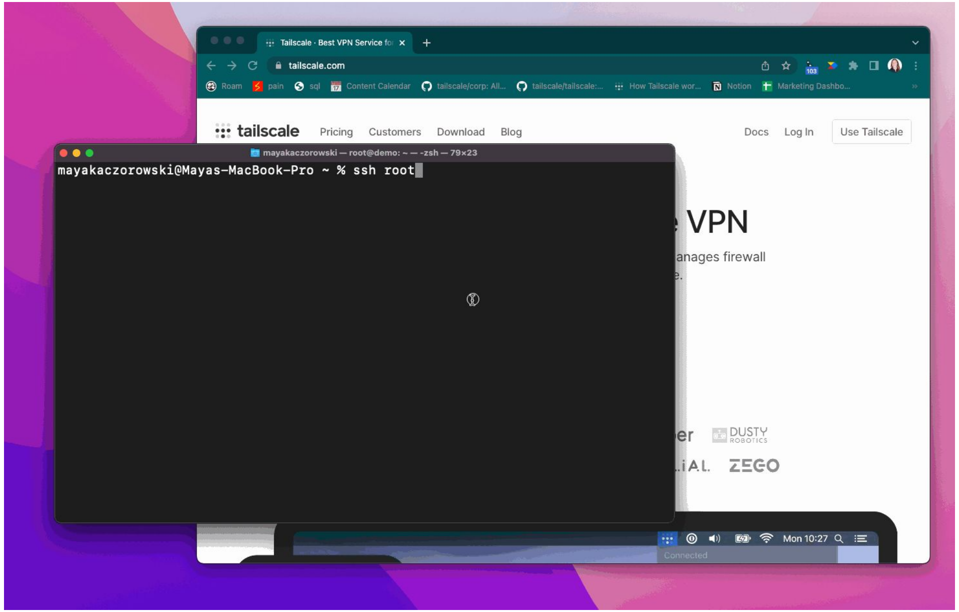
SSH & WEP

Keith Ng & Jacob Hoopes

Secure Shell (SSH)

- Enables encrypted communication between users and servers over the internet
- Replaces unencrypted remote terminal or login programs like Telnet, rlogin, and rsh
- Replaces file transfer programs like FTP and rcp





```
mayakaczorowski@Mayas-MacBook-Pro ~ % ssh root
```

Image source: https://geekflare.com/understanding-ssh/
GIF source: https://tailscale.com/blog/tailscale-ssh/

SSH Versions

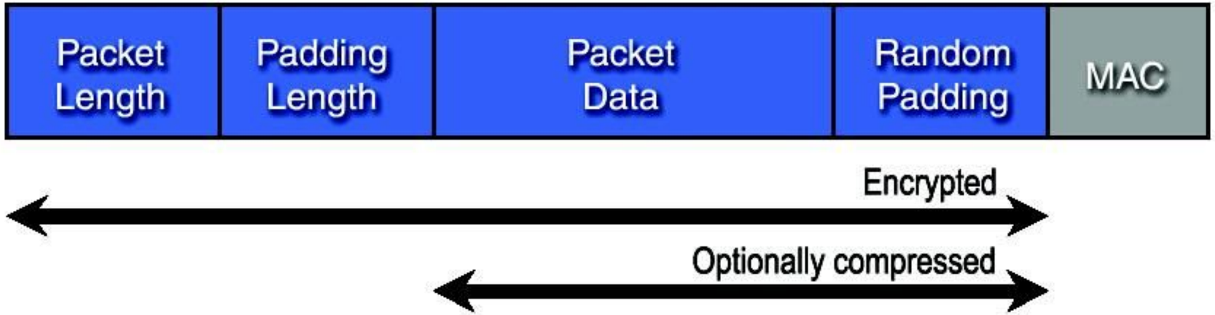
SSHv1

- Introduced in 1995
- Had a number of flaws
  - Encryption: CBC mode that always had an IV of 0
  - Key: Same encryption key used for both directions
  - Data Integrity: Cyclic Redundancy Check on plaintext, checksum was appended to ciphertext

SSHv2

- Introduced in 1996
- Aimed to fix many of the issues in SSHv1, was not backwards compatible
  - Encryption: Supports a range of protocols like AES and Blowfish
  - Key: Creates 2 keys, $k_{u \rightarrow s}$ for user to server and $k_{s \rightarrow u}$ for server to user
  - Data Integrity: MAC

SSHv2 Encryption



1. Plaintext is padded with random bytes so that the length of

   packet len || pad len || message || pad

   is a multiple of the cipher block length (16 bytes for AES)
2. Encrypt using AES in randomized CBC mode
3. Compute MAC

Image source: https://www.researchgate.net/figure/SSH2-Binary-Packet-Protocol_fig4_225129766

SSHv2 Decryption

After an encrypted packet is received, the decryption algorithm:

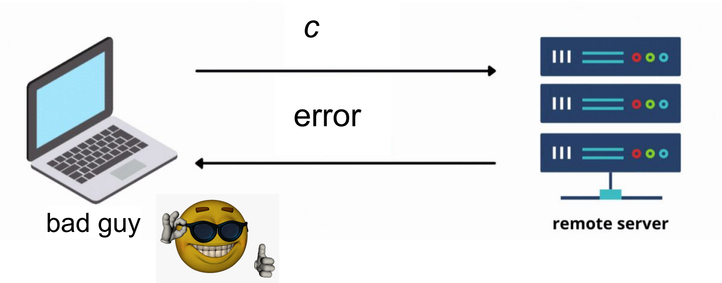
1. Decrypts packet length using the assigned key
2. Reads as much data as specified by packet length
3. Decrypts the rest of the ciphertext
4. Checks the validity with MAC
    a. If valid, removes pad and returns plaintext
    b. Otherwise, sends error message

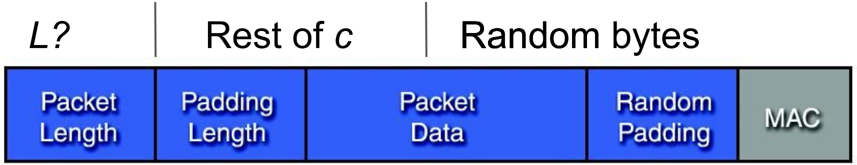
Attacking Non-Atomic Decryption

SSHv2 uses the decrypted packet length *before* data integrity has been verified

Suppose attacker intercepts 16-byte ciphertext *c*

1. Attacker sends *c* to server
2. Server decrypts *c*, assumes first 4 bytes represent packet length *L*
3. Attacker intermittently sends a random byte
4. Once server reads in *L* bytes, it will send an error message
5. Attacker now knows value of *L*, or the first 4 bytes of *c*




L? | Rest of *c* | Random bytes

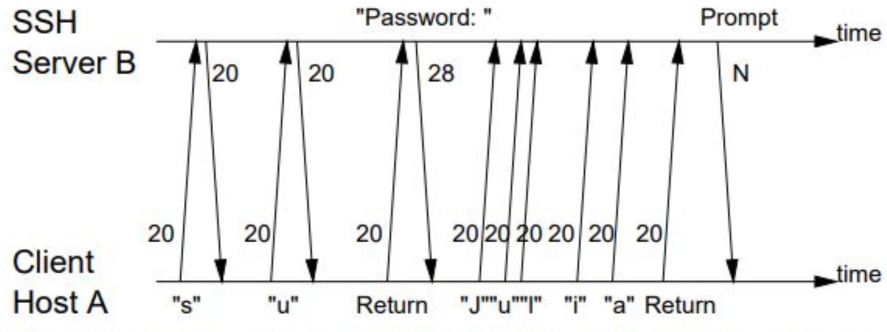
Packet Length | Padding Length | Packet Data | Random Padding | MAC

Attacking With Traffic Analysis

SSHv2 sends a packet for each keystroke

Suppose attacker knows a user is typing a password (i.e. if echo is turned off)

- The attacker can time the differences between consecutive keystrokes
- This could expose information on passwords, such as characters based on distance on a keyboard



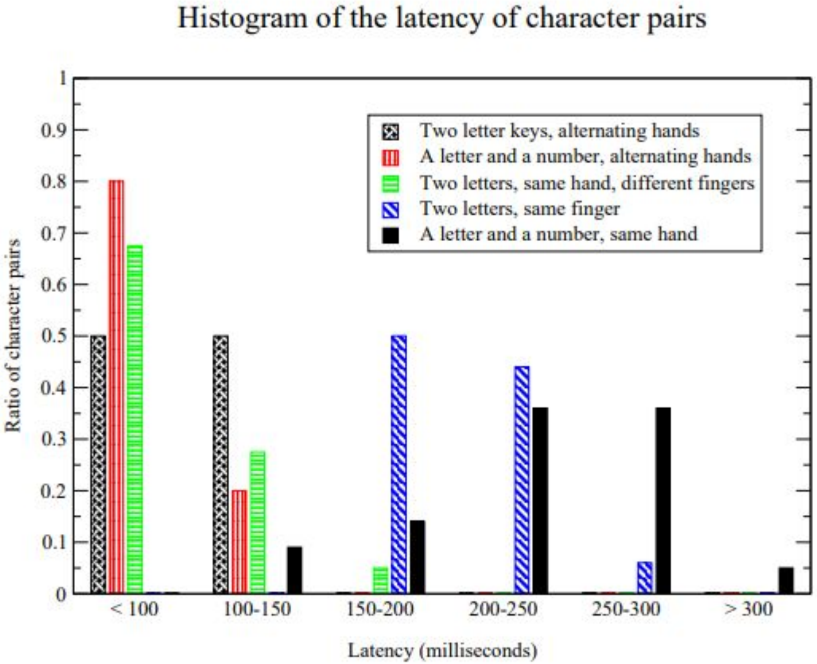


Image source: *Timing analysis of keystrokes and timing attacks on SSH*

WEP - Wired Equivalent Privacy

Aims to ensure a level of security and privacy for wireless connections that is as good as the expected security and privacy of wired connections.

It completely fails.

“The design goal of WEP is to provide data privacy at the level of a wired network. WEP, however, completely fails on this front and gives us an excellent case study illustrating how a weak design can lead to disastrous results.” (BS 389)

How it works

All members of the wireless network share a long term secret key *k* (40 or 128 bits)

Uses the RC4 stream cipher and CRC(*m*), a 32-bit checksum

$c \leftarrow (m \,||\, \mathrm{CRC}(m)) \oplus \mathrm{RC4}(\mathrm{IV} \,||\, k)$

So, transmitted data is (IV, *c*)

Decryption by obtaining a pair (*m*, *s*), where $m = c \oplus \mathrm{RC4}(\mathrm{IV} \,||\, k)$, and $s = \mathrm{IV} \,||\, k$

The receiver accepts if *s* = CRC(*m*) and rejects otherwise

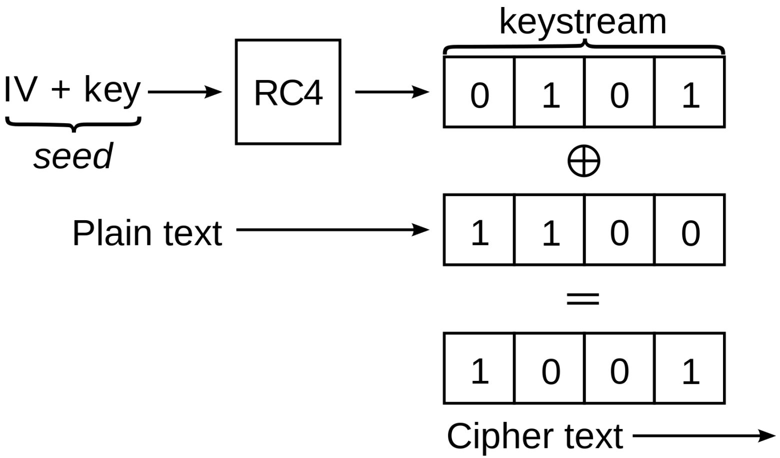


Image source: https://en.wikipedia.org/wiki/Wired_Equivalent_Privacy

Attacks (1) - IV collisions

Stream cipher keys should not be reused! (They did understand this.)

To overcome this, they used the 24-bit IV to make a per-frame key k_f := IV || k .

Many implementations choose IVs poorly, and as IVs are sent in the clear, eavesdroppers can detect IV collisions easily (then can use the two pad attack).

Each frame is at most only 1156 bytes, by the birthday paradox this leads to collisions after transmitting only ~4MB.

IVs generated by a counter will exhaust the IV space after 2^{24} frames, which would only take about a day for a busy wireless access point (also often resets on startup, so low IV values are more common).

Attacks (2) - Related Keys

RC4 is completely insecure in the WEP setting, where key values are chosen as 1 || k, 2 || k, etc.

After about a million frames are sent, an eavesdropper can recover the entire long term secret key k (Fluhrer, Mantin, and Shamir showed this in “Weaknesses in the key scheduling algorithm of RC4”, 2001) (Recent attacks can do this with as few as 40k frames).

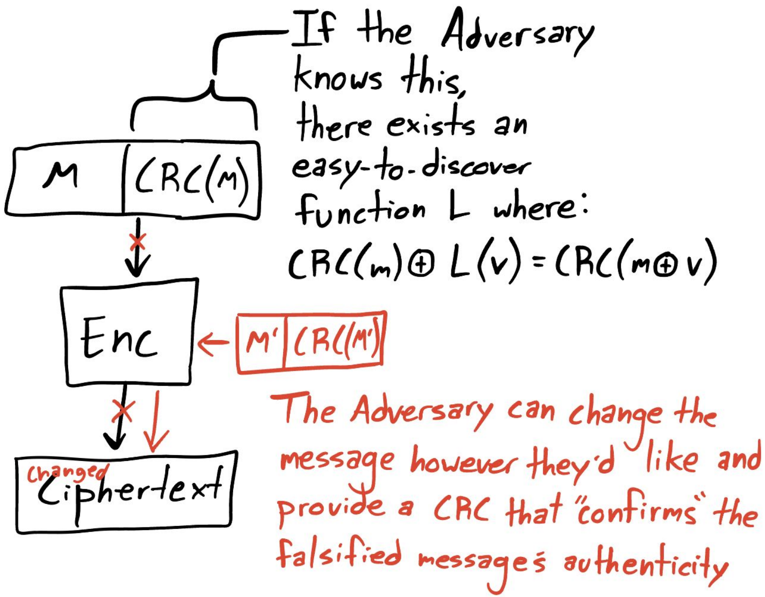
Generating per-frame keys should have been done with a PRF, which could solve this problem, although not some others.

Attacks (3) - Malleability

WEP uses something like MAC then encrypt, though it uses CRC instead of MAC. We've seen that MAC then encrypt does not stand against IND-CCA nor INT-CTXT.

CRC has linearity - given CRC(m) for some m, it is easy to compute CRC($m \oplus \Delta$) for any Δ. So, CRC($m \oplus \Delta$) = CRC(m) $\oplus$ L(Δ). (It satisfies the relation: CRC($x \oplus y$) = CRC(x) $\oplus$ CRC(y) $\oplus$ c, where c is related to $|x|$ and $|y|$.)

The attacker can make *arbitrary modifications* to the ciphertext w/o being detected.



If the Adversary knows this, there exists an easy-to-discover function L where:
$CRC(m) \oplus L(v) = CRC(m \oplus v)$

The Adversary can change the message however they'd like and provide a CRC that "confirms" the falsified message's authenticity

For any $\Delta \in \{0,1\}^\ell$, an attacker can create a new ciphertext $c' \leftarrow c \oplus (\Delta, L(\Delta))$, which satisfies

$$\begin{aligned} c' \;=\; & \mathrm{RC4}(\mathrm{IV} \parallel k) \;\oplus\; (m,\ \mathrm{CRC}(m)) \;\oplus\; (\Delta,\ L(\Delta)) = \\ & \mathrm{RC4}(\mathrm{IV} \parallel k) \;\oplus\; (m \oplus \Delta,\ \ \mathrm{CRC}(m) \oplus L(\Delta)) = \\ & \mathrm{RC4}(\mathrm{IV} \parallel k) \;\oplus\; (m \oplus \Delta,\ \ \mathrm{CRC}(m \oplus \Delta)) \end{aligned}$$

Image source: Boneh & Shoup, A Graduate Course in Applied Cryptography

Attacks (4) - Chosen Ciphertext Attack

Vulnerable to a specific CCA called chopchop which allows the attacker to decrypt an encrypted frame of their choice.

Chopchop takes advantage of a parity bit at the end of the message (it is just the XOR of all the bits in the message).

In a cipher with variable length bit strings, no MAC, no padding, and the parity bit is appended to the end before encryption, chopchop is able to completely decrypt the plaintext.

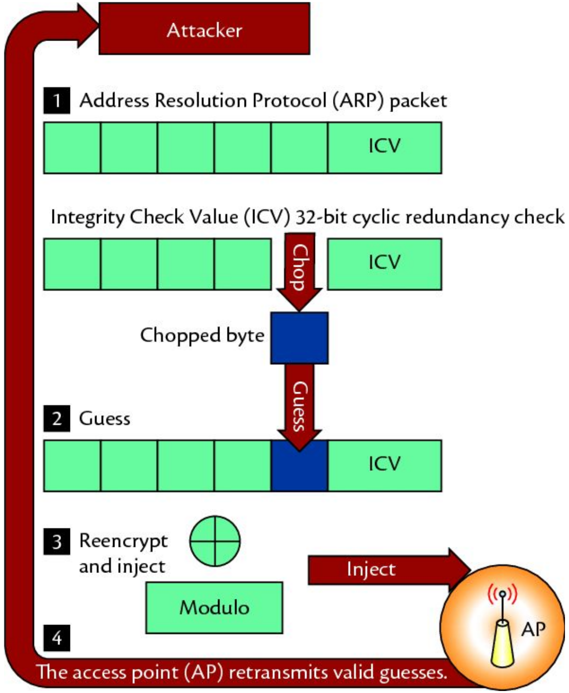

Attacker
1 Address Resolution Protocol (ARP) packet
ICV
Integrity Check Value (ICV) 32-bit cyclic redundancy check
ICV
Chop
Chopped byte
Guess
2 Guess
ICV
3 Reencrypt and inject
Inject
Modulo
4
AP
The access point (AP) retransmits valid guesses.

Image source: https://www.researchgate.net/publication/260635283_The_Insecurity_of_Wireless_Networks

Attacks (5) - Denial of Service

A message to a wireless access point from a client to “dissociate” will cause that client to become disconnected, and require a few seconds to re-establish a connection.

This disassociation message is unauthenticated, meaning that anyone can send it.

A malicious party sending disassociation messages every few seconds will prevent a computer from connecting to the network.

Sources

Boneh & Shoup, *A Graduate Course in Applied Cryptography*

Song, Wagner, and Tian, *Timing analysis of keystrokes and timing attacks on SSH*

https://docstore.mik.ua/orelly/networking_2ndEd/ssh/ch03_05.htm

https://www.techtarget.com/searchsecurity/definition/Wired-Equivalent-Privacy