2550 Intro to cybersecurity

L25: net + wifi security & Review

abhi shelat

NMAP

Nmap ("Network Mapper") is an open source tool for network exploration and security auditing. It was designed to rapidly scan large networks, although it works fine against single hosts. Nmap uses raw IP packets in novel ways to determine what hosts are available on the network, what services (application name and version) those hosts are offering, what operating systems (and OS versions) they are running, what type of packet filters/firewalls are in use, and dozens of other characteristics. While Nmap is commonly used for security audits, many systems and network administrators find it useful for routine tasks such as network inventory, managing service upgrade schedules, and monitoring host or service uptime.

```
abhi@ry1:~$ sudo nmap -O shelat.khoury.northeastern.edu
Starting Nmap 7.95 ( https://nmap.org ) at 2024-12-02 21:29 UTC
Nmap scan report for shelat.khoury.northeastern.edu (129.10.111.166)
Host is up (0.0015s latency).
rDNS record for 129.10.111.166: vista.khoury.northeastern.edu
Not shown: 995 filtered tcp ports (no-response)
PORT     STATE  SERVICE
80/tcp   open   http
113/tcp  closed ident
443/tcp  open   https
2000/tcp open   cisco-sccp
5060/tcp open   sip
Device type: general purpose|firewall
Running (JUST GUESSING): Linux 4.X|3.X|2.6.X (94%), IPFire 2.X (88%)
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:2.6.32 cpe:/
o:ipfire:ipfire:2.25
Aggressive OS guesses: Linux 4.0 - 4.4 (94%), Linux 3.11 - 4.9 (89%), Linux 2.6.32 (89%), Linux 2.6.32 or 3.10
(89%), Linux 3.10 (89%), Linux 3.10 - 3.16 (89%), Linux 4.15 (89%), Linux 4.19 - 5.15 (89%), IPFire 2.25 firewall
(Linux 4.14) (88%), Linux 3.10 - 3.12 (87%)
No exact OS matches for host (test conditions non-ideal).

OS detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 9.63 seconds
```

OS fingerprinting

subtle differences in implementations allows an attacker to determine OS and version numbers.

```
MacBook-Pro:p8 abhi$ sudo nmap -O localhost
Password:
Starting Nmap 7.91 ( https://nmap.org ) at 2021-04-20 05:23 EDT
Nmap scan report for localhost (127.0.0.1)
Host is up (0.00014s latency).
Other addresses for localhost (not scanned): ::1
Not shown: 993 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
1025/tcp  open  NFS-or-IIS
1080/tcp  open  socks
1110/tcp  open  nfsd-status
3000/tcp  open  ppp
8086/tcp  open  d-s-n
49161/tcp open  unknown
Device type: general purpose
Running: Apple macOS 10.14.X
OS CPE: cpe:/o:apple:mac_os_x:10.14
OS details: Apple macOS 10.14 (Mojave) (Darwin 18.2.0 - 18.6.0)
Network Distance: 0 hops
```

lsof - list open files

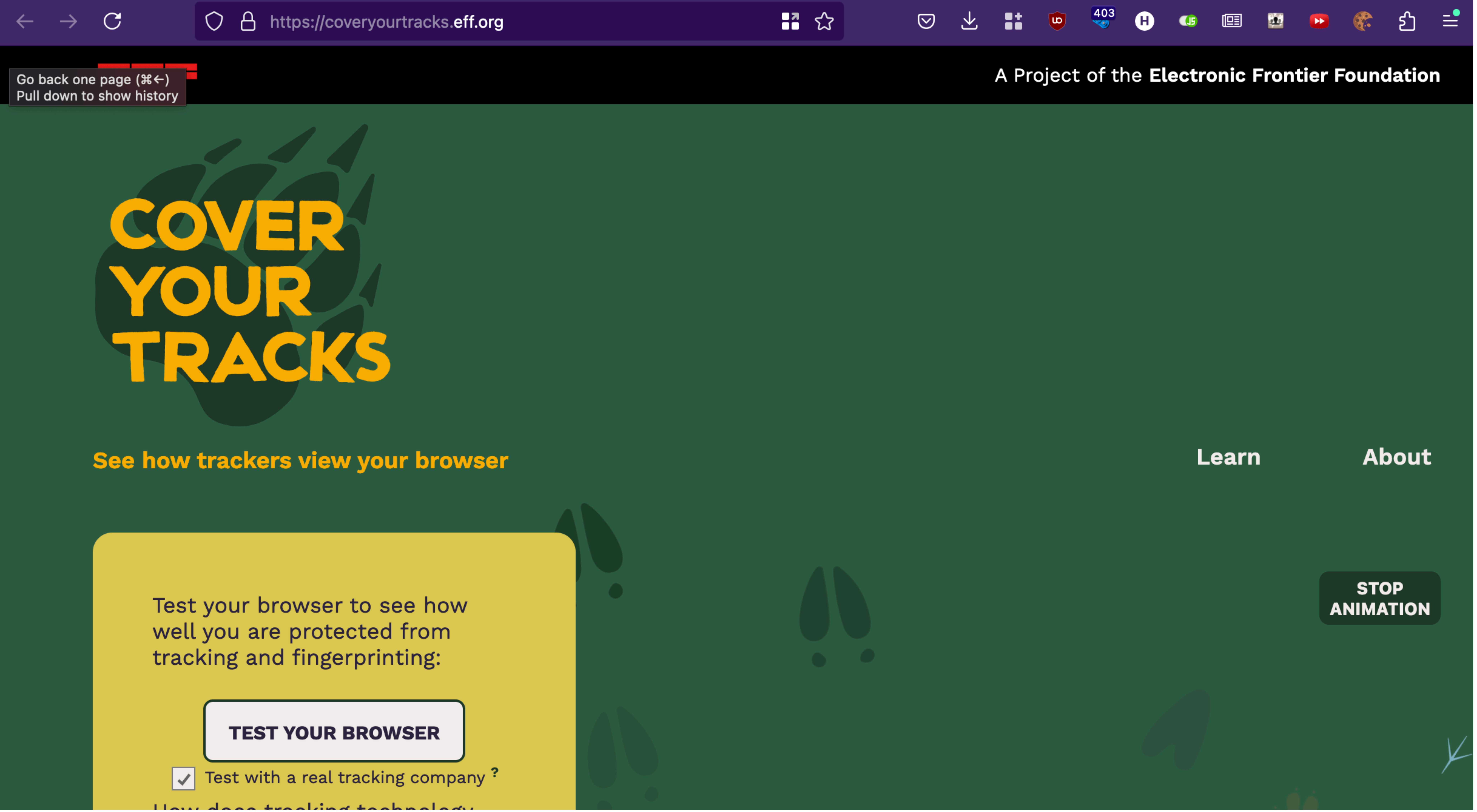
Lsof revision 4.93.2 lists on its standard output file information about files opened by processes for the ... UNIX dialects...

An open file may be a regular file, a directory, a block special file, a character special file, an executing text reference, a library, **a stream or a network file** (Internet socket, NFS file or UNIX domain socket.)

```
abhi@ry1:~$ sudo lsof -i
COMMAND      PID            USER   FD   TYPE  DEVICE SIZE/OFF NODE NAME
systemd-n   1886 systemd-network   15u  IPv4 1276970      0t0  UDP ry1:bootpc
systemd-r   1889 systemd-resolve   13u  IPv4   23769      0t0  UDP localhost:domain
systemd-r   1889 systemd-resolve   14u  IPv4   23770      0t0  TCP localhost:domain (LISTEN)
tailscale   2093            root   16u  IPv6   31905      0t0  UDP *:41641
tailscale   2093            root   17u  IPv4   31906      0t0  UDP *:41641
tailscale   2093            root   18u  IPv4   39850      0t0  TCP ry1:35048->ec2-3-124-108-117.eu-central-1.compute.amazonaws.com:https (ESTABLISHED)
tailscale   2093            root   19u  IPv4   28766      0t0  TCP ry1.wolf-arctic.ts.net:44994 (LISTEN)
tailscale   2093            root   20u  IPv6   28767      0t0  TCP ry1.wolf-arctic.ts.net:44994 (LISTEN)
tailscale   2093            root   21u  IPv4 1275046      0t0  TCP ry1:44416->ec2-54-161-152-147.compute-1.amazonaws.com:https (ESTABLISHED)
tailscale   2093            root   28u  IPv4  774732      0t0  TCP ry1:45764->derp1f.tailscale.com:https (ESTABLISHED)
rsync       2156            root    5u  IPv4   16996      0t0  TCP *:rsync (LISTEN)
rsync       2156            root    6u  IPv6   16997      0t0  TCP *:rsync (LISTEN)
sshd        2208            root    3u  IPv4   40020      0t0  TCP *:ssh (LISTEN)
sshd        2208            root    4u  IPv6   40022      0t0  TCP *:ssh (LISTEN)
coder       2851           coder    3u  IPv4   44134      0t0  TCP ry1.wolf-arctic.ts.net:3000 (LISTEN)
coder       2851           coder    8u  IPv4   56336      0t0  UDP *:56264
coder       2851           coder    9u  IPv4   44138      0t0  TCP ry1:54576->74.118.138.247:https (ESTABLISHED)
coder       2851           coder   10u  IPv6   56337      0t0  UDP *:56264
coder       2851           coder   15u  IPv4   45781      0t0  TCP localhost:36334->localhost:42441 (ESTABLISHED)
coder       2851           coder   19u  IPv4 1249989      0t0  TCP localhost:59988->localhost:42441 (ESTABLISHED)
coder       2851           coder   20u  IPv4 1271353      0t0  TCP localhost:41842->localhost:42441 (ESTABLISHED)
coder       2851           coder   21u  IPv6   31496      0t0  UDP *:33124
coder       2851           coder   22u  IPv4   31497      0t0  UDP *:53141
coder       2851           coder   25u  IPv4 1255207      0t0  TCP localhost:58202->localhost:42441 (ESTABLISHED)
postgres    2872           coder    7u  IPv4   55553      0t0  TCP localhost:42441 (LISTEN)
postgres    2872           coder    9u  IPv4   55555      0t0  UDP localhost:47759->localhost:47759
postgres    2874           coder    9u  IPv4   55555      0t0  UDP localhost:47759->localhost:47759
postgres    2875           coder    9u  IPv4   55555      0t0  UDP localhost:47759->localhost:47759
postgres    2876           coder    9u  IPv4   55555      0t0  UDP localhost:47759->localhost:47759
postgres    2877           coder    9u  IPv4   55555      0t0  UDP localhost:47759->localhost:47759
postgres    2878           coder    9u  IPv4   55555      0t0  UDP localhost:47759->localhost:47759
postgres    2879           coder    9u  IPv4   55555      0t0  UDP localhost:47759->localhost:47759
postgres    2899           coder    9u  IPv4   55555      0t0  UDP localhost:47759->localhost:47759
postgres    2899           coder   10u  IPv4   43557      0t0  TCP localhost:42441->localhost:36334 (ESTABLISHED)
postgres  102309           coder    9u  IPv4   55555      0t0  UDP localhost:47759->localhost:47759
postgres  102309           coder   10u  IPv4 1241905      0t0  TCP localhost:42441->localhost:58202 (ESTABLISHED)
postgres  104478           coder    9u  IPv4   55555      0t0  UDP localhost:47759->localhost:47759
postgres  104478           coder   10u  IPv4 1271290      0t0  TCP localhost:42441->localhost:59988 (ESTABLISHED)
postgres  104809           coder    9u  IPv4   55555      0t0  UDP localhost:47759->localhost:47759
postgres  104809           coder   10u  IPv4 1250011      0t0  TCP localhost:42441->localhost:41842 (ESTABLISHED)
sshd      105250            root    4u  IPv4 1278337      0t0  TCP ry1.wolf-arctic.ts.net:ssh->mbp.wolf-arctic.ts.net:63349 (ESTABLISHED)
sshd      105308            abhi    4u  IPv4 1278337      0t0  TCP ry1.wolf-arctic.ts.net:ssh->mbp.wolf-arctic.ts.net:63349 (ESTABLISHED)
abhi@ry1:~
```

Network Anonymity

My browser essentially determines my identity.



Browser data exposed through APIs provides a pseudo-identifier.

Your Results

Your browser fingerprint **appears to be unique** among the 184,780 tested in the past 45 days.

Currently, we estimate that your browser has a fingerprint that conveys **at least 17.5 bits of identifying information.**

The measurements we used to obtain this result are listed below. You can **read more about our methodology, statistical results, and some defenses against fingerprinting here**.

User Agent

Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:133.0) Gecko/20100101 Firefox/133.0

Bits of identifying information: 8.93
One in x browsers have this value: 487.55

HTTP_ACCEPT Headers

text/html, */*; q=0.01 gzip, deflate, br, zstd en-US,en;q=0.5
Bits of identifying information: 2.07
One in x browsers have this value: 4.21

Screen Size and Color Depth

1440x900x30

Bits of identifying information: 7.12
One in x browsers have this value: 138.83

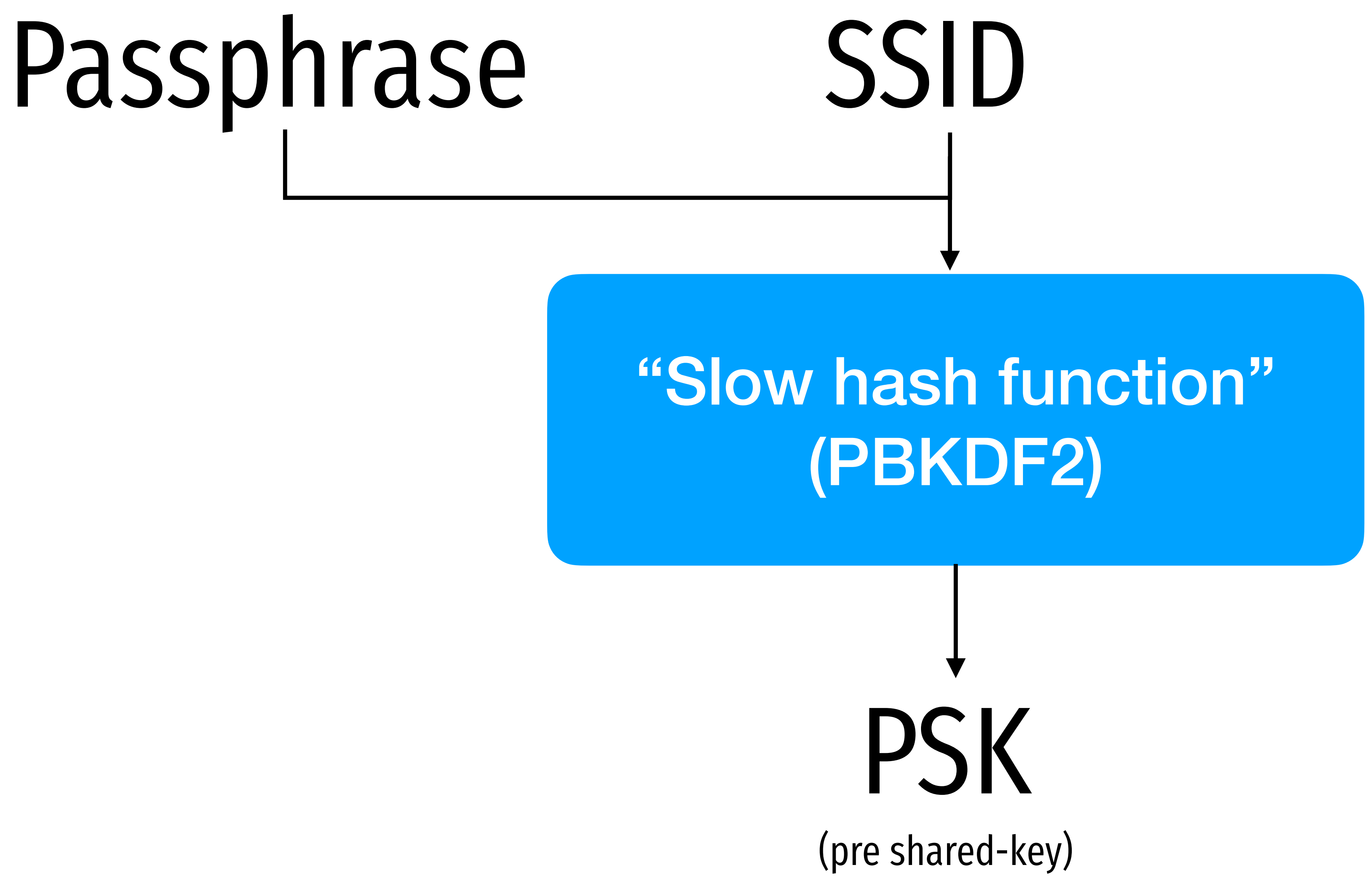
System Fonts

Andale Mono, Arial, Arial Black, Arial Hebrew, Arial Narrow, Arial Rounded MT Bold, Arial Unicode MS, Comic Sans MS, Courier, Courier New, Geneva, Georgia, Helvetica, Helvetica Neue, Impact, LUCIDA GRANDE, Microsoft Sans Serif, Monaco, MYRIAD, Palatino, Tahoma, Times, Times New Roman, Trebuchet MS, Verdana, Wingdings, Wingdings 2, Wingdings 3 (via javascript)

Bits of identifying information: 17.5
One in x browsers have this value: 184780.0

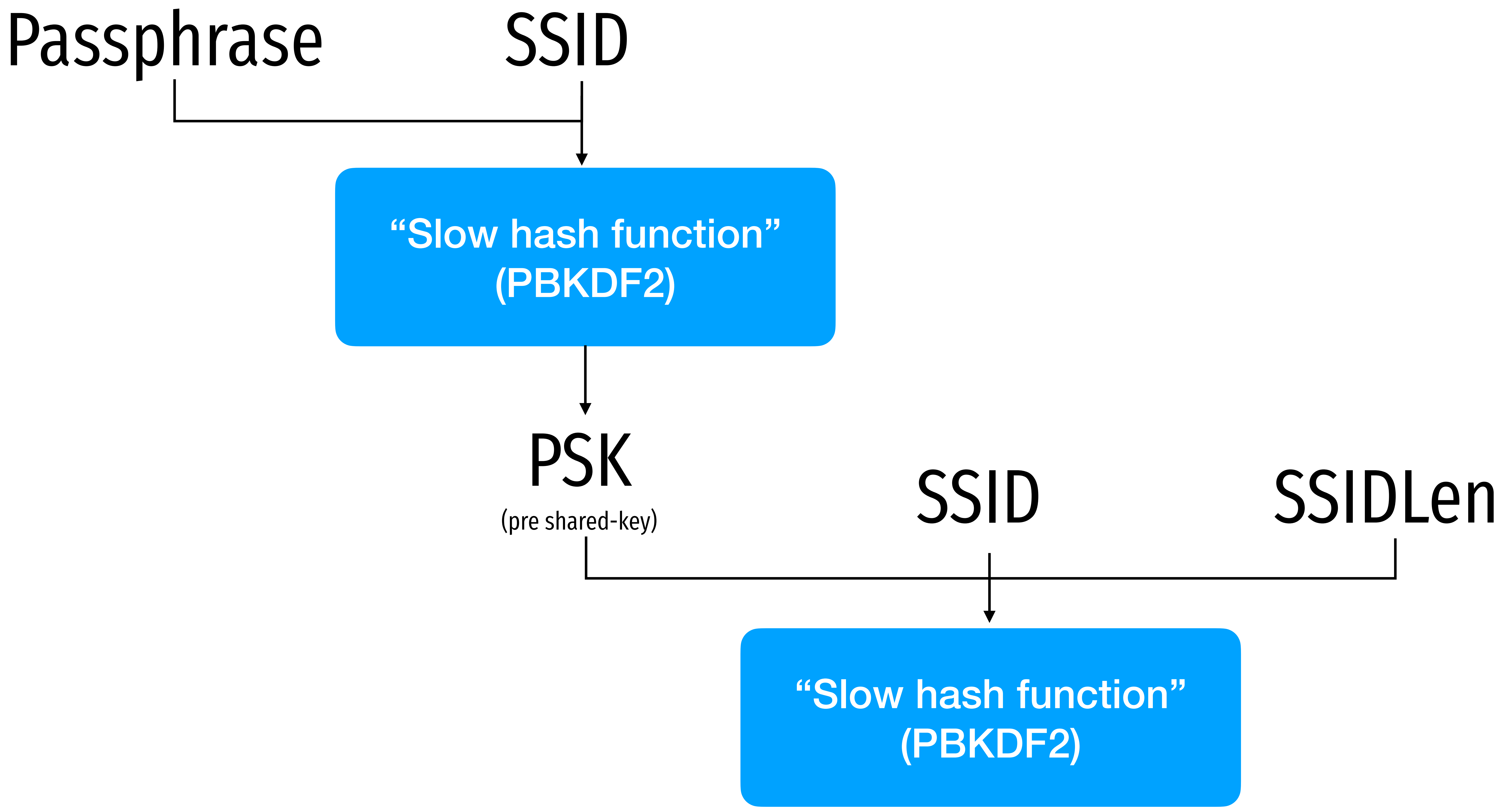
WPA2

WPA2 uses a passphrase to generate a PSK, which is used to generate a PMK and PTK.



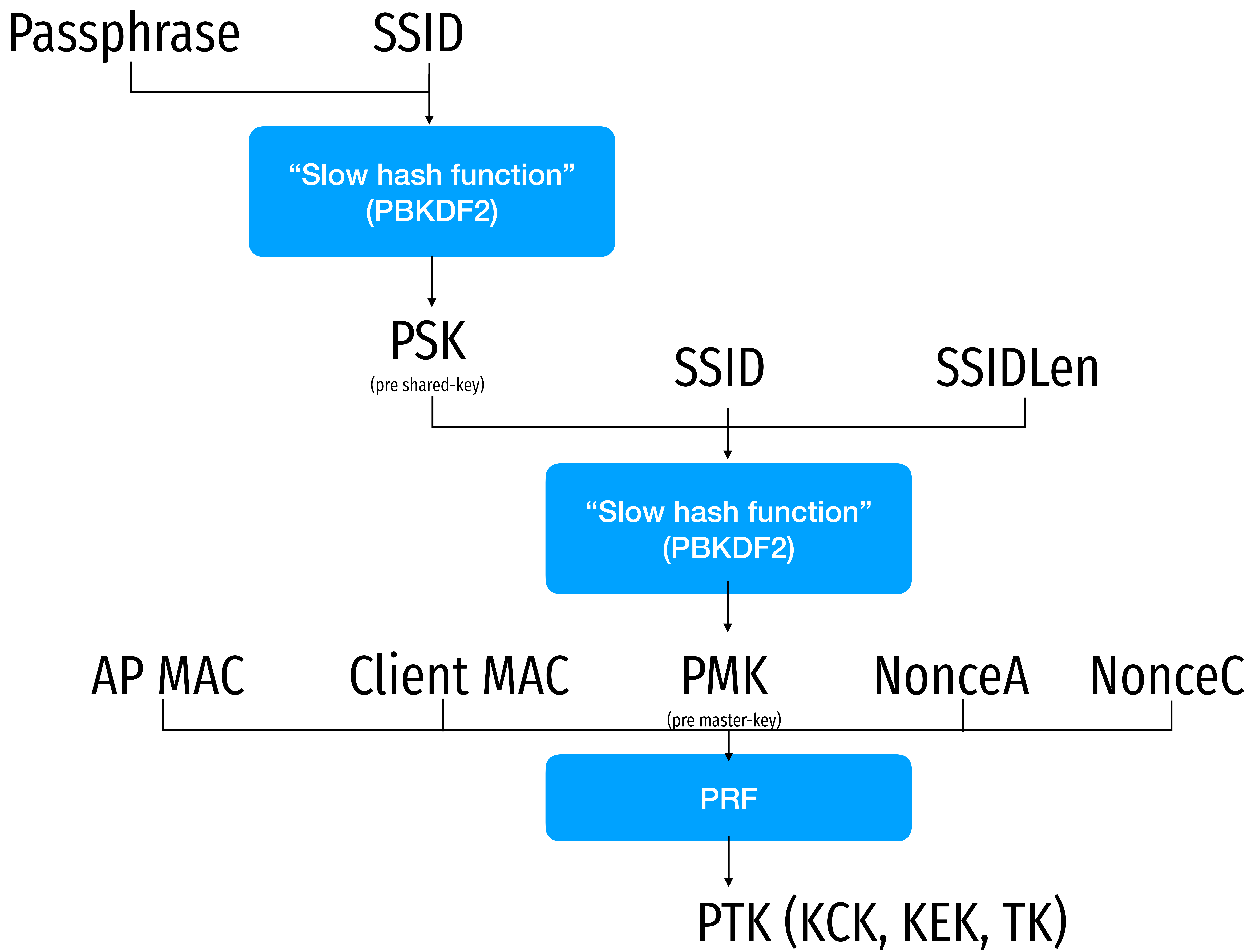
WPA2

WPA2 uses a passphrase to generate a PSK, which is used to generate a PMK and PTK.



WPA2

WPA2 uses a passphrase to generate a PSK, which is used to generate a PMK and PTK.



Wifi 4-way handshake

passphrase

[calculate PMK]

passphrase

[calculate PMK]

Wifi 4-way handshake

passphrase

passphrase

[calculate PMK]

NonceA

[calculate PMK]

[compute PTK]

Wifi 4-way handshake

passphrase

passphrase

[calculate PMK]

[compute PTK]

NonceA

NonceC + MsgIntCode

The KCK is used to compute MIC.

[calculate PMK]

[compute PTK,
Verify MIC]

Wifi 4-way handshake

passphrase

passphrase

[calculate PMK]

NonceA

[calculate PMK]

[compute PTK]

NonceC + MsgIntCode

The KCK is used to compute MIC.

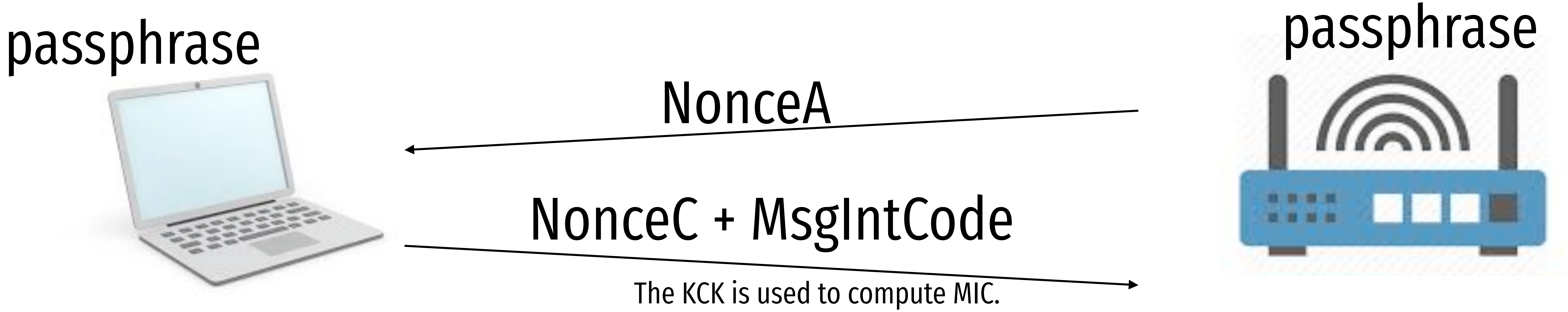
[compute PTK,
Verify MIC]

[verify MIC]

KeyInstall + MsgIntCode

KeyInstalled + MsgIntCode

How to attack a WPA2 session

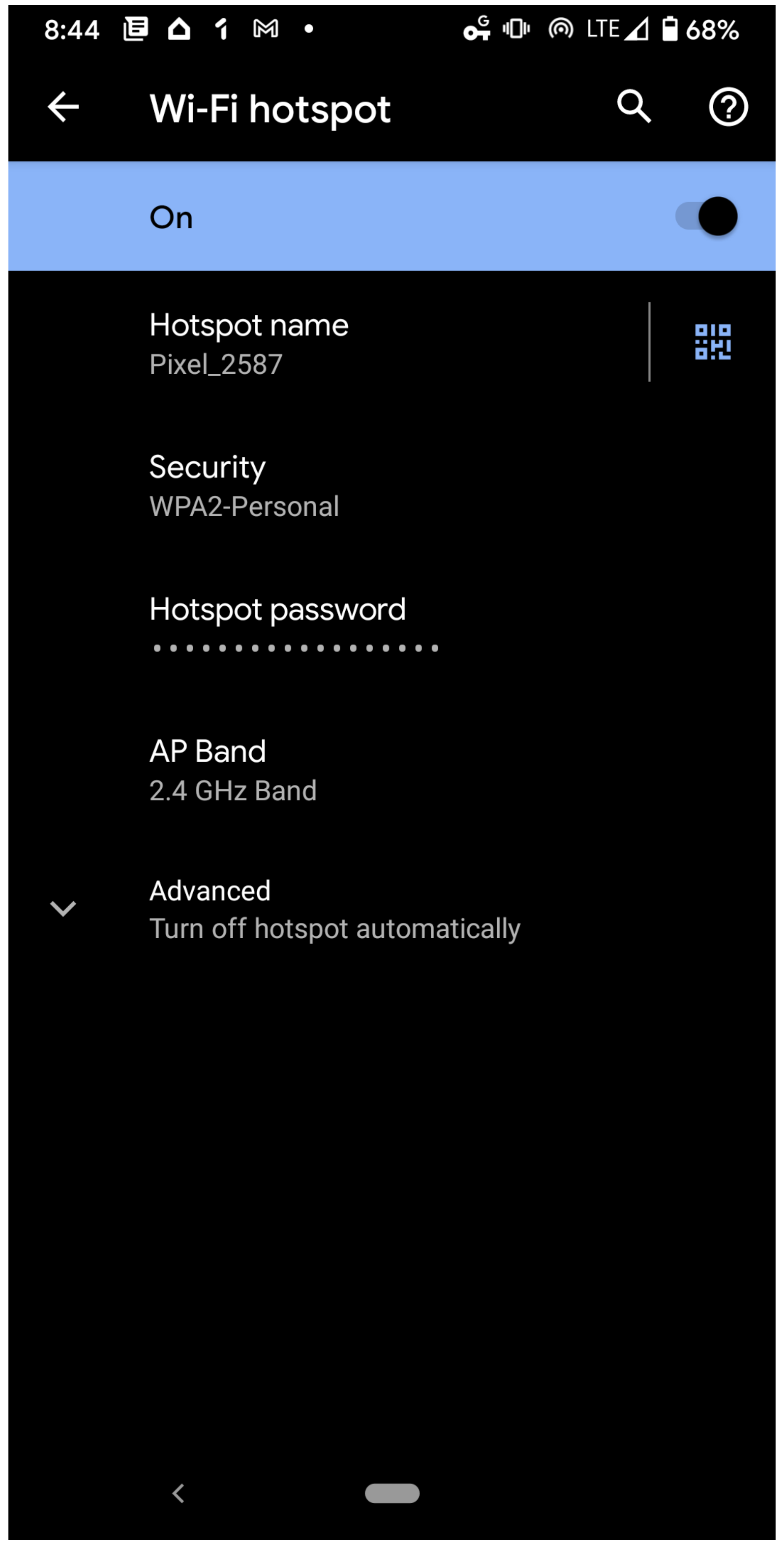


Listen for the NonceA, NonceC, MIC value, AP MAC, client MAC, SSID.
For each passphrase in the dictionary:

    Use passphrase to compute PMK, PTK.
    Use PTK to compute a MIC and test whether it is equal to captured one.

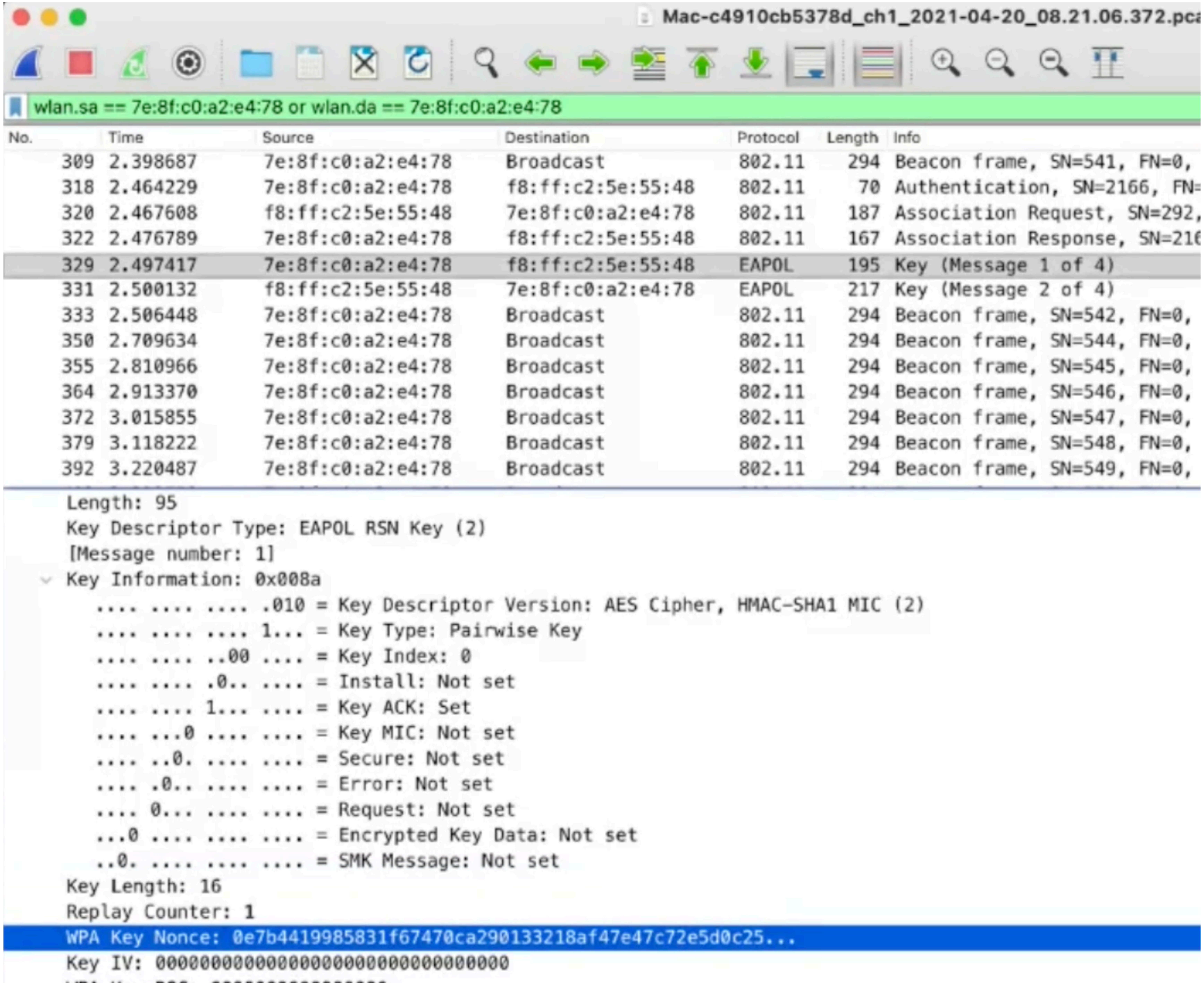
Demo

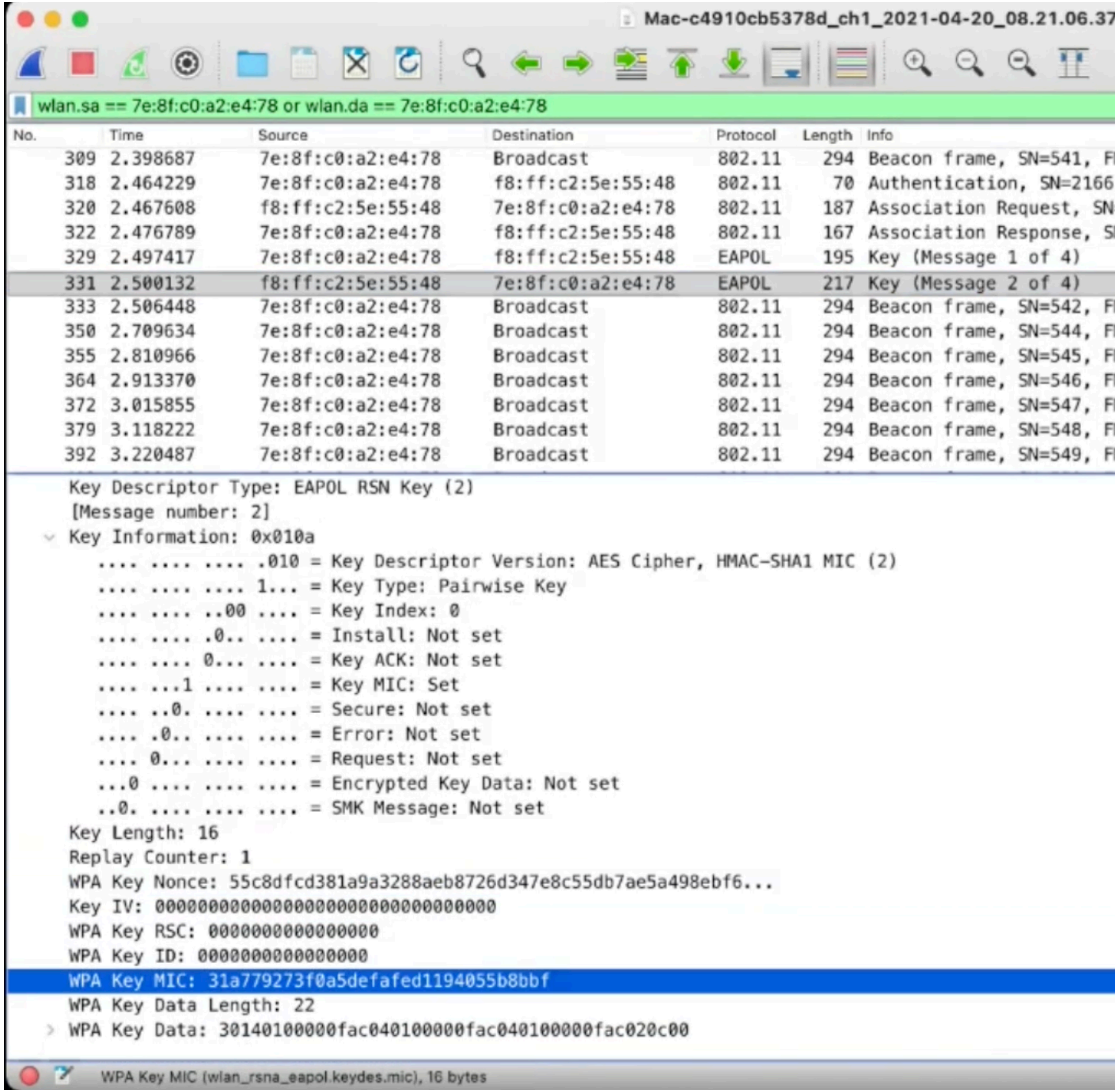
Honest user
trying to connect

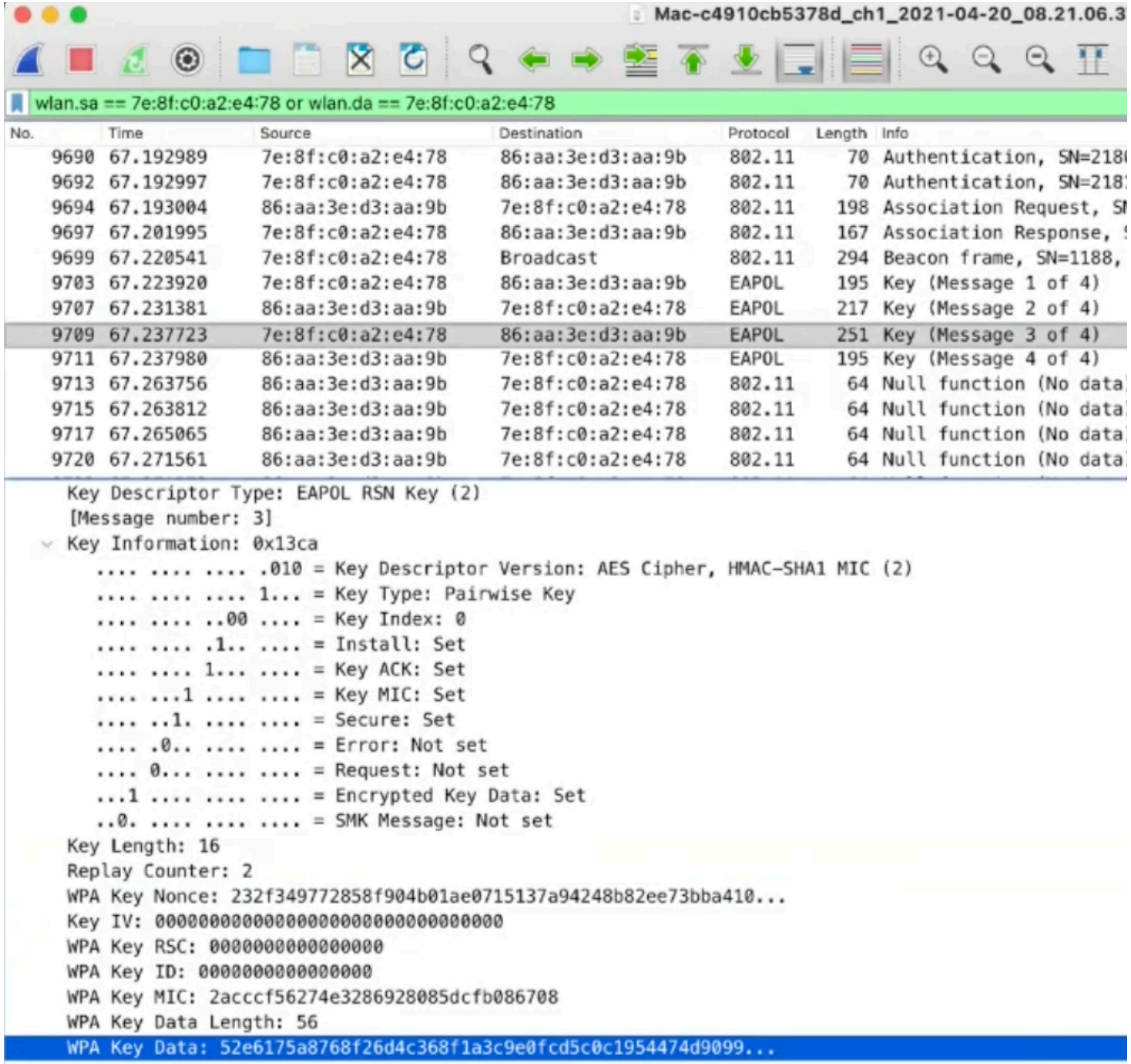


Pixel phone sharing wifi

Attacker using packet capture over wifi


Mac-c4910cb5378d_ch1_2021-04-20_08.21.06.372.pca
wlan.sa == 7e:8f:c0:a2:e4:78 or wlan.da == 7e:8f:c0:a2:e4:78
No. Time Source Destination Protocol Length Info
309 2.398687 7e:8f:c0:a2:e4:78 Broadcast 802.11 294 Beacon frame, SN=541, FN=0,
318 2.464229 7e:8f:c0:a2:e4:78 f8:ff:c2:5e:55:48 802.11 70 Authentication, SN=2166, FN=
320 2.467608 f8:ff:c2:5e:55:48 7e:8f:c0:a2:e4:78 802.11 187 Association Request, SN=292,
322 2.476789 7e:8f:c0:a2:e4:78 f8:ff:c2:5e:55:48 802.11 167 Association Response, SN=216
329 2.497417 7e:8f:c0:a2:e4:78 f8:ff:c2:5e:55:48 EAPOL 195 Key (Message 1 of 4)
331 2.500132 f8:ff:c2:5e:55:48 7e:8f:c0:a2:e4:78 EAPOL 217 Key (Message 2 of 4)
333 2.506448 7e:8f:c0:a2:e4:78 Broadcast 802.11 294 Beacon frame, SN=542, FN=0,
350 2.709634 7e:8f:c0:a2:e4:78 Broadcast 802.11 294 Beacon frame, SN=544, FN=0,
355 2.810966 7e:8f:c0:a2:e4:78 Broadcast 802.11 294 Beacon frame, SN=545, FN=0,
364 2.913370 7e:8f:c0:a2:e4:78 Broadcast 802.11 294 Beacon frame, SN=546, FN=0,
372 3.015855 7e:8f:c0:a2:e4:78 Broadcast 802.11 294 Beacon frame, SN=547, FN=0,
379 3.118222 7e:8f:c0:a2:e4:78 Broadcast 802.11 294 Beacon frame, SN=548, FN=0,
392 3.220487 7e:8f:c0:a2:e4:78 Broadcast 802.11 294 Beacon frame, SN=549, FN=0,
Length: 95
Key Descriptor Type: EAPOL RSN Key (2)
[Message number: 1]
Key Information: 0x008a
.... .010 = Key Descriptor Version: AES Cipher, HMAC-SHA1 MIC (2)
.... 1... = Key Type: Pairwise Key
....00 .... = Key Index: 0
.... .0.. .... = Install: Not set
.... 1... .... = Key ACK: Set
....0 .... = Key MIC: Not set
.... ..0. .... = Secure: Not set
.... .0.. .... = Error: Not set
.... 0... .... = Request: Not set
...0 .... = Encrypted Key Data: Not set
..0. .... = SMK Message: Not set
Key Length: 16
Replay Counter: 1
WPA Key Nonce: 0e7b4419985831f67470ca290133218af47e47c72e5d0c25...
Key IV: 00000000000000000000000000000000


Mac-c4910cb5378d_ch1_2021-04-20_08.21.06.37
wlan.sa == 7e:8f:c0:a2:e4:78 or wlan.da == 7e:8f:c0:a2:e4:78
No. Time Source Destination Protocol Length Info
309 2.398687 7e:8f:c0:a2:e4:78 Broadcast 802.11 294 Beacon frame, SN=541, F
318 2.464229 7e:8f:c0:a2:e4:78 f8:ff:c2:5e:55:48 802.11 70 Authentication, SN=2166
320 2.467608 f8:ff:c2:5e:55:48 7e:8f:c0:a2:e4:78 802.11 187 Association Request, SN
322 2.476789 7e:8f:c0:a2:e4:78 f8:ff:c2:5e:55:48 802.11 167 Association Response, S
329 2.497417 7e:8f:c0:a2:e4:78 f8:ff:c2:5e:55:48 EAPOL 195 Key (Message 1 of 4)
331 2.500132 f8:ff:c2:5e:55:48 7e:8f:c0:a2:e4:78 EAPOL 217 Key (Message 2 of 4)
333 2.506448 7e:8f:c0:a2:e4:78 Broadcast 802.11 294 Beacon frame, SN=542, F
350 2.709634 7e:8f:c0:a2:e4:78 Broadcast 802.11 294 Beacon frame, SN=544, F
355 2.810966 7e:8f:c0:a2:e4:78 Broadcast 802.11 294 Beacon frame, SN=545, F
364 2.913370 7e:8f:c0:a2:e4:78 Broadcast 802.11 294 Beacon frame, SN=546, F
372 3.015855 7e:8f:c0:a2:e4:78 Broadcast 802.11 294 Beacon frame, SN=547, F
379 3.118222 7e:8f:c0:a2:e4:78 Broadcast 802.11 294 Beacon frame, SN=548, F
392 3.220487 7e:8f:c0:a2:e4:78 Broadcast 802.11 294 Beacon frame, SN=549, F
Key Descriptor Type: EAPOL RSN Key (2)
[Message number: 2]
Key Information: 0x010a
.... .010 = Key Descriptor Version: AES Cipher, HMAC-SHA1 MIC (2)
.... 1... = Key Type: Pairwise Key
....00 .... = Key Index: 0
.... .0.. .... = Install: Not set
.... 0... .... = Key ACK: Not set
....1 .... = Key MIC: Set
.... ..0. .... = Secure: Not set
.... .0.. .... = Error: Not set
.... 0... .... = Request: Not set
...0 .... = Encrypted Key Data: Not set
..0. .... = SMK Message: Not set
Key Length: 16
Replay Counter: 1
WPA Key Nonce: 55c8dfcd381a9a3288aeb8726d347e8c55db7ae5a498ebf6...
Key IV: 00000000000000000000000000000000
WPA Key RSC: 0000000000000000
WPA Key ID: 0000000000000000
WPA Key MIC: 31a779273f0a5defafed1194055b8bbf
WPA Key Data Length: 22
WPA Key Data: 30140100000fac040100000fac040100000fac020c00
WPA Key MIC (wlan_rsna_eapol.keydes.mic), 16 bytes


Mac-c4910cb5378d_ch1_2021-04-20_08.21.06.3
wlan.sa == 7e:8f:c0:a2:e4:78 or wlan.da == 7e:8f:c0:a2:e4:78
No. Time Source Destination Protocol Length Info
9690 67.192989 7e:8f:c0:a2:e4:78 86:aa:3e:d3:aa:9b 802.11 70 Authentication, SN=218
9692 67.192997 7e:8f:c0:a2:e4:78 86:aa:3e:d3:aa:9b 802.11 70 Authentication, SN=218
9694 67.193004 86:aa:3e:d3:aa:9b 7e:8f:c0:a2:e4:78 802.11 198 Association Request, S
9697 67.201995 7e:8f:c0:a2:e4:78 86:aa:3e:d3:aa:9b 802.11 167 Association Response,
9699 67.220541 7e:8f:c0:a2:e4:78 Broadcast 802.11 294 Beacon frame, SN=1188,
9703 67.223920 7e:8f:c0:a2:e4:78 86:aa:3e:d3:aa:9b EAPOL 195 Key (Message 1 of 4)
9707 67.231381 86:aa:3e:d3:aa:9b 7e:8f:c0:a2:e4:78 EAPOL 217 Key (Message 2 of 4)
9709 67.237723 7e:8f:c0:a2:e4:78 86:aa:3e:d3:aa:9b EAPOL 251 Key (Message 3 of 4)
9711 67.237980 86:aa:3e:d3:aa:9b 7e:8f:c0:a2:e4:78 EAPOL 195 Key (Message 4 of 4)
9713 67.263756 86:aa:3e:d3:aa:9b 7e:8f:c0:a2:e4:78 802.11 64 Null function (No data
9715 67.263812 86:aa:3e:d3:aa:9b 7e:8f:c0:a2:e4:78 802.11 64 Null function (No data
9717 67.265065 86:aa:3e:d3:aa:9b 7e:8f:c0:a2:e4:78 802.11 64 Null function (No data
9720 67.271561 86:aa:3e:d3:aa:9b 7e:8f:c0:a2:e4:78 802.11 64 Null function (No data
Key Descriptor Type: EAPOL RSN Key (2)
[Message number: 3]
Key Information: 0x13ca
.... .010 = Key Descriptor Version: AES Cipher, HMAC-SHA1 MIC (2)
.... 1... = Key Type: Pairwise Key
....00 .... = Key Index: 0
.... .1.. .... = Install: Set
.... 1... .... = Key ACK: Set
....1 .... = Key MIC: Set
.... ..1. .... = Secure: Set
.... .0.. .... = Error: Not set
.... 0... .... = Request: Not set
...1 .... = Encrypted Key Data: Set
..0. .... = SMK Message: Not set
Key Length: 16
Replay Counter: 2
WPA Key Nonce: 232f349772858f904b01ae0715137a94248b82ee73bba410...
Key IV: 00000000000000000000000000000000
WPA Key RSC: 0000000000000000
WPA Key ID: 0000000000000000
WPA Key MIC: 2acccf56274e3286928085dcfb086708
WPA Key Data Length: 56
WPA Key Data: 52e6175a8768f26d4c368f1a3c9e0fcd5c0c1954474d9099...

Why does this attack succeed?

Most people use bad passwords for wifi.

How to mitigate network attacks?


FIREWALL

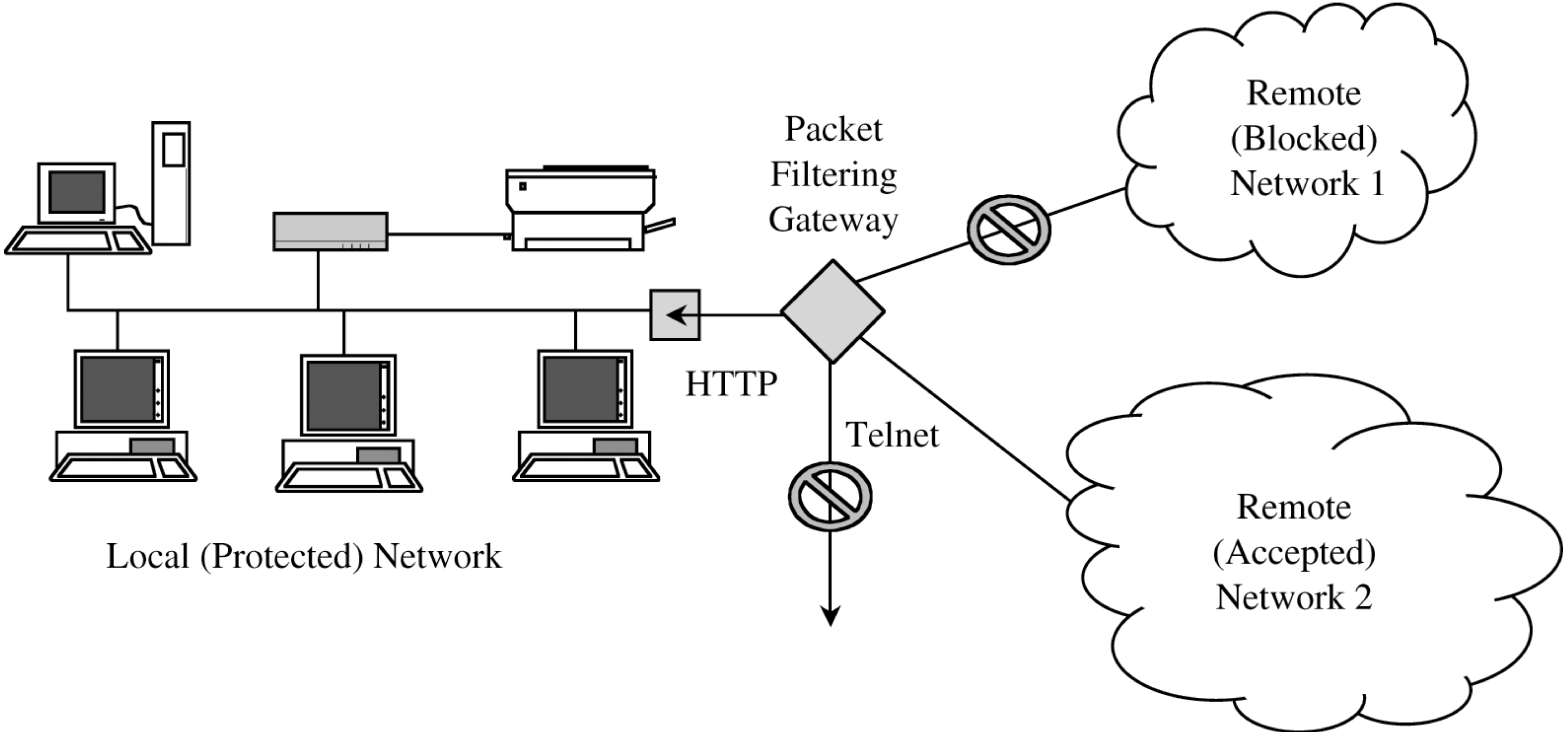
Firewalls

Stateless Packet Filter	Rules based on addr/port + header info
Statefull Packet Filter	above + state between each packet
Statefull Packet Inspection	above + can inspect the data of the package

StateLESS Packet Filter

Rules based on addr/port + header info

Look at the packet and decide immediately whether to drop or forward.



- Local subnet has all traffic from remote network 1 blocks (say, network with IP address 253.128.x.x)

- Allow some traffic from Remote Network 2 (say, 253.127.x.x), but only if it is destined for port 80 (web-traffic), Drop all other ports

Review

Our main topics

Authentication, passwords

Authorization

Cryptography

Systems security

Web
Network

Passwords and Authentication

What is authentication?

Classes of secrets?

Methods and attacks against passwords?

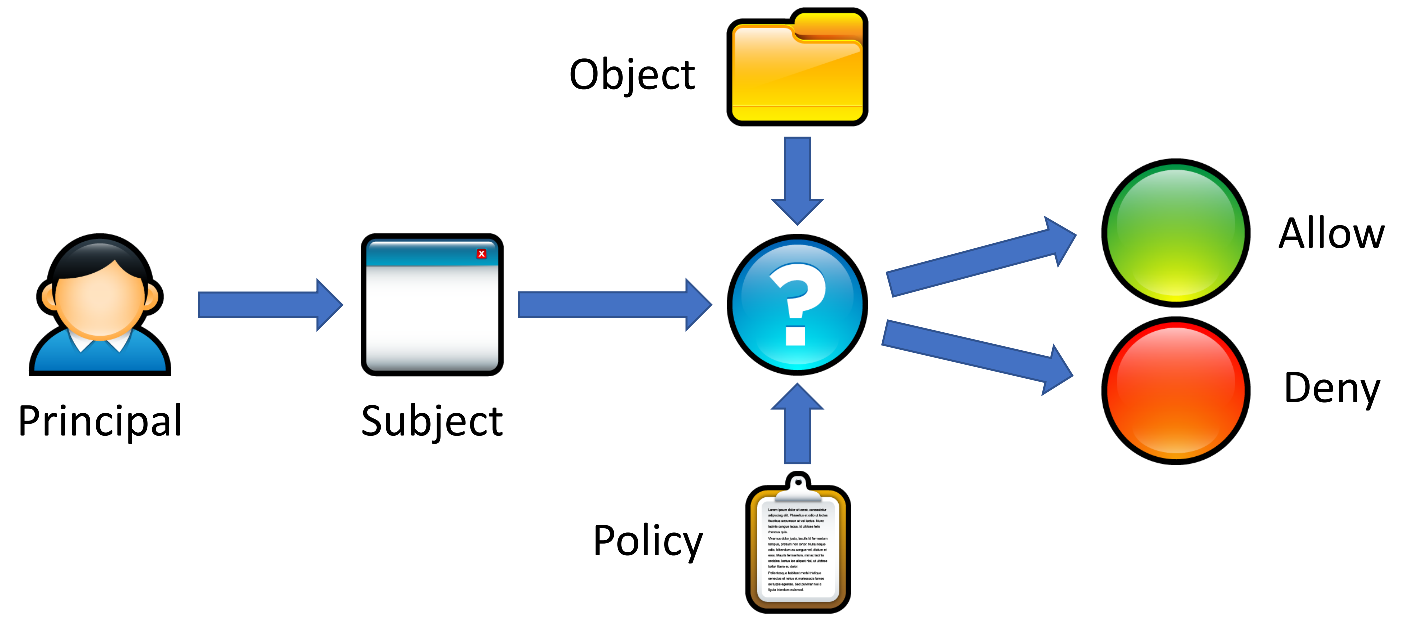
Passwords in the real (distributed) world, Oauth, 2fa.

Authorization

Basics of an access control check

Access Control Check

- Given an access request from a subject, on behalf of a principal, for an object, return an access control decision based on the policy

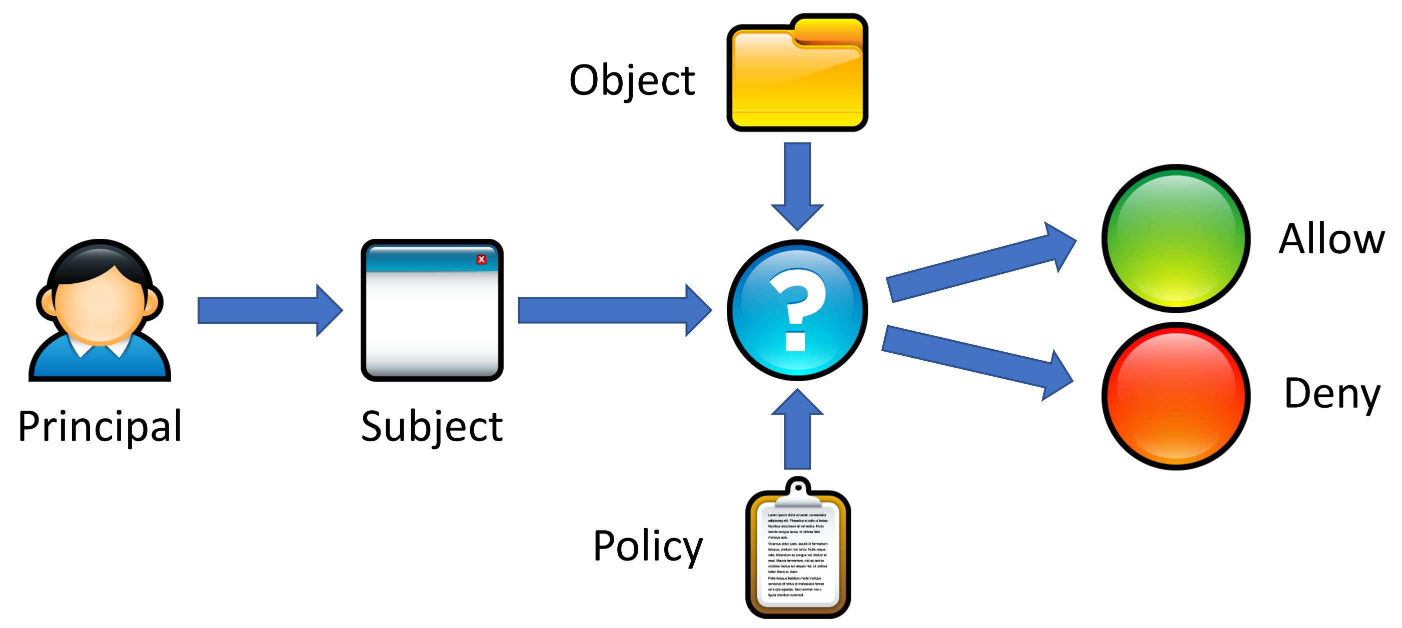


Authorization

Basics of an access control check

Access Control Check

- Given an access request from a subject, on behalf of a principal, for an object, return an access control decision based on the policy



Two types

Access Control Models

- Discretionary Access Control (DAC)
  - The kind of access control you are familiar with
  - Access rights propagate and may be changed at subject’s discretion
  - Implemented in Windows and Linux
  - Main issues:
    - Ambient authority (subjects inherit all permissions of principals)
    - Confused deputies (subject doesn’t know which principal it serves); setuid
- Mandatory Access Control (MAC)
  - Access of subjects to objects is based on a system-wide policy managed by admin ∂
  - Denies users full control over resources they create
  - Bell-LaPadula: MAC for confidentiality (uses Multi Level Security)
  - Biba: MAC for integrity
  - Main issues:
    - Inflexible and complicated to manage
    - Do not prevent side channel attacks

Cryptography

Privacy:

Authenticity:

Hashing:

Cryptography

Privacy: Encryption

Authenticity:

Hashing:

Cryptography

Privacy:	Encryption
Authenticity:	Signature & MACs
Hashing:	

Cryptography

Privacy:	Encryption
Authenticity:	Signature & MACs
Hashing:	SHA-256, collision resistance

System Security: Attack Surfaces

- Steal the device and use it
- Social Engineering
  - Trick the user into installing malicious software
  - Spear phishing
- OS-level attacks
  - Backdoor the OS
  - Direct connection via USB
  - Exploit vulnerabilities in the OS or apps (e.g. email clients, web browsers)
- Network-level attacks
  - Passive eavesdropping on the network
  - Active network attacks (e.g. man-in-the-middle)

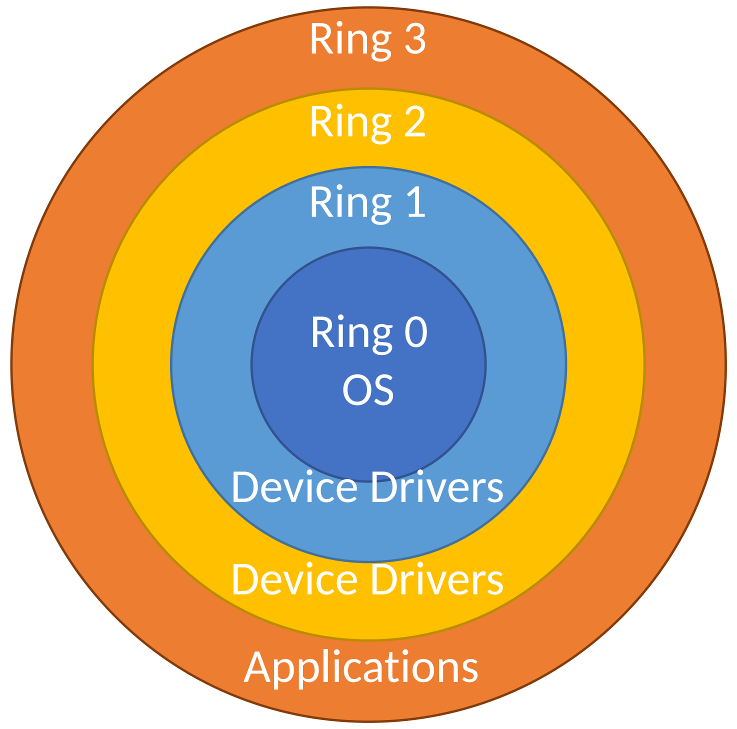
Modern defense: Isolation

Rings:

Most modern CPUs support protected mode

x86 CPUs support three rings with different privileges

- Ring 0: Operating System
  - Code in this ring may directly access any device
- Ring 1, 2: device drivers
  - Code in these rings may directly access some devices
  - May not change the protection level of the CPU
- Ring 3: userland
  - Code in this ring may not directly access devices
  - All device access must be via OS APIs
  - May not change the protection level of the CPU



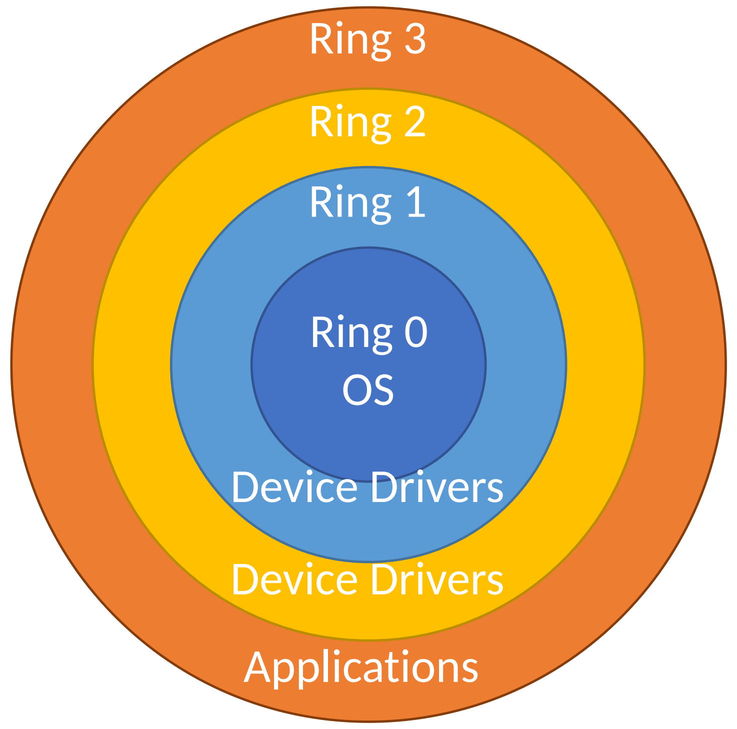
Modern defense: Isolation

Rings:

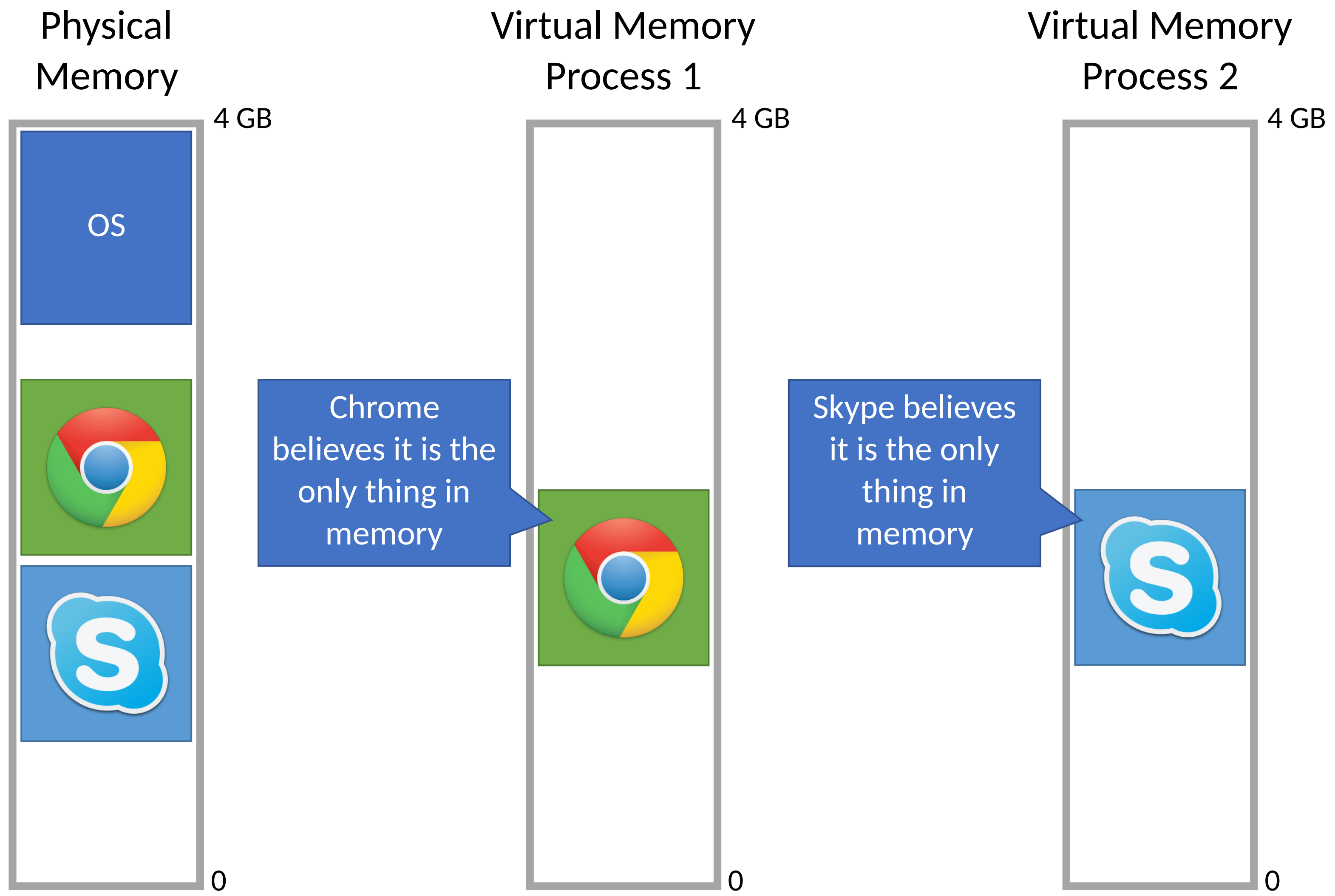
Most modern CPUs support protected mode

x86 CPUs support three rings with different privileges

- Ring 0: Operating System
  - Code in this ring may directly access any device
- Ring 1, 2: device drivers
  - Code in these rings may directly access some devices
  - May not change the protection level of the CPU
- Ring 3: userland
  - Code in this ring may not directly access devices
  - All device access must be via OS APIs
  - May not change the protection level of the CPU



Virtual Memory:



Basis for tools

Security Technologies

Authentication

- Physical and remote access is restricted

Access control

- Processes cannot read/write any file
- Users may not read/write each other's files arbitrarily
- Modifying the OS and installing software requires elevated privileges

Firewall

- Unsolicited communications from the internet are blocked
- Only authorized processes may send/receive messages from the internet

Anti-virus

- All files are scanned to identify and quarantine known malicious code

Logging

- All changes to the system are recorded
- Sensitive applications may also log their activity in the secure system log

Exploits

Anatomy of an exploit

Program Crash

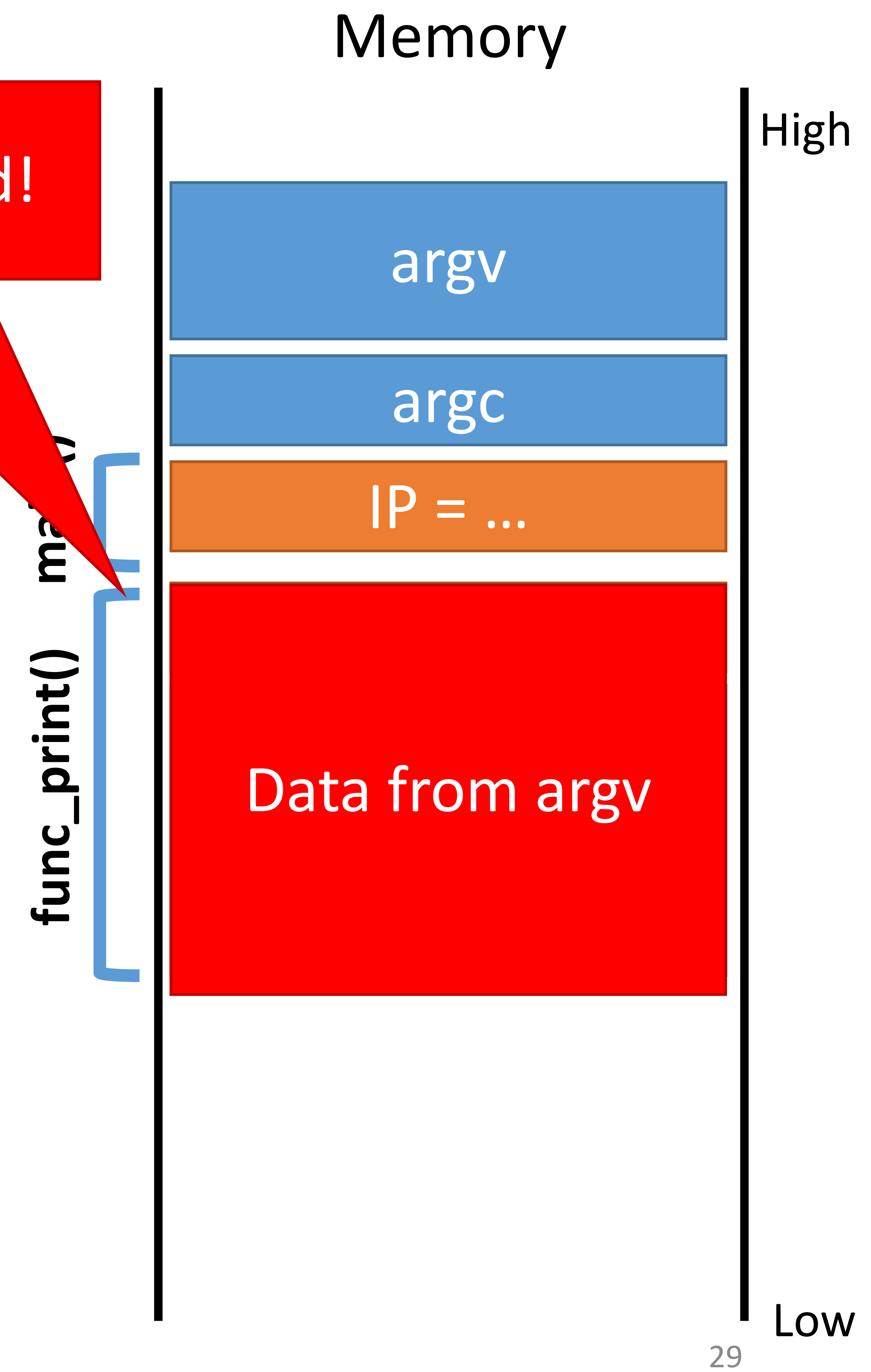
```
0:  void func_print(char s[]) {
      // only holds 32 characters, max
      char buffer[32];
1:    strcpy(buffer, s);
2:    printf(“%s\n”,buffer);
3:  }

4:  void main(i
5:    for (int i=1; i < argc; i++) {
6:      func_print(argv[i]);
7:    }
8:  }
```

IP

Program crashes :(

Saved IP is destroyed!



Mitigations

- Stack canaries
  - Compiler adds special sentinel values onto the stack before each saved IP
  - Canary is set to a random value in each frame
  - At function exit, canary is checked
  - If expected number isn’t found, program closes with an error
- Non-executable stacks
  - Modern CPUs set stack memory as read/write, but no eXecute
  - Prevents shellcode from being placed on the stack
- Address space layout randomization
  - Operating system feature
  - Randomizes the location of program and data memory each time a program executes

SQL Injection

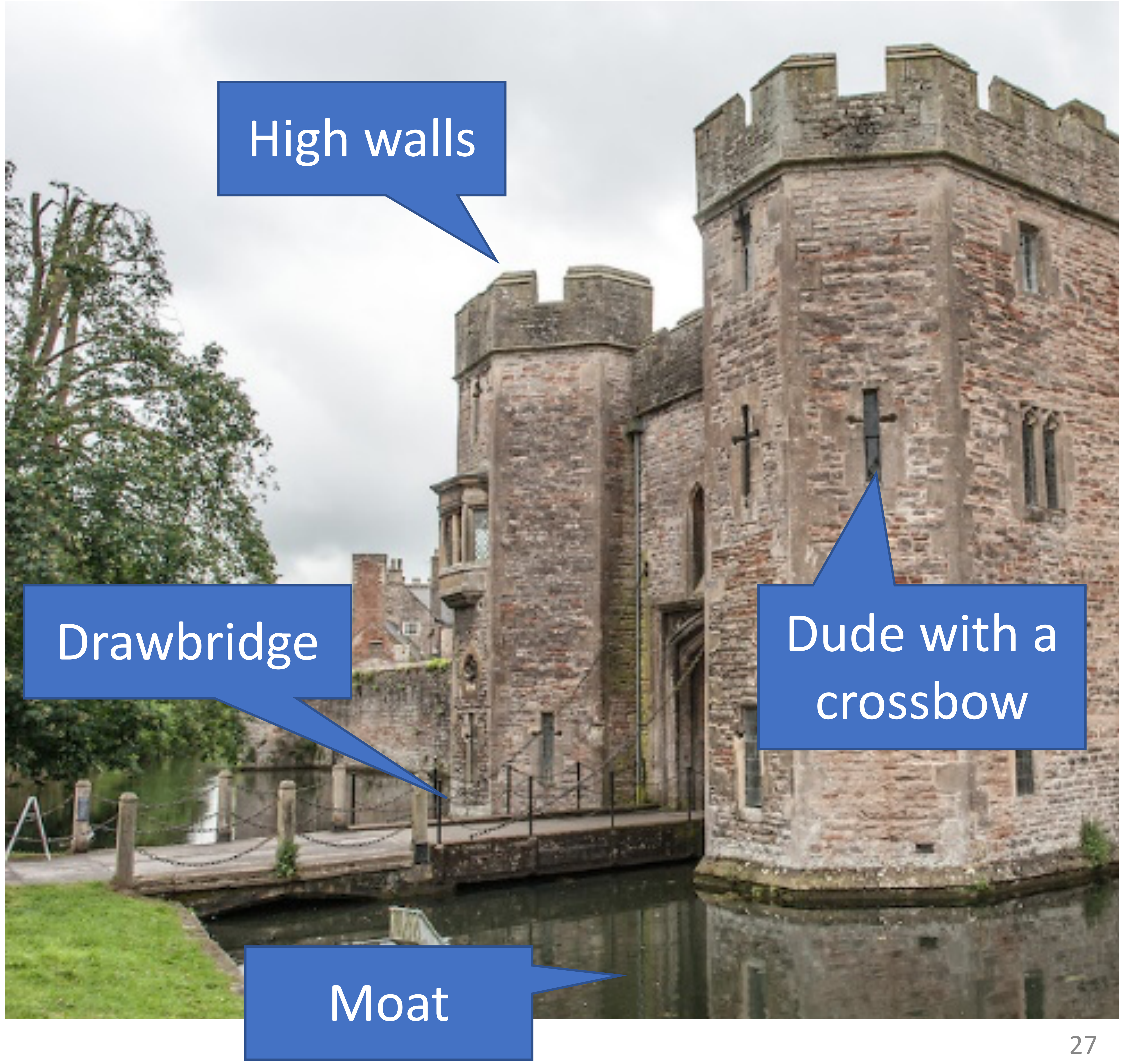
‘SELECT * FROM user_tbl WHERE user="%s" AND pw="%s";'

form[‘username’]	form[‘password’]	Resulting query
alice	123456	‘… WHERE user="alice" AND pw="123456";’
bob	qwerty1#	‘… WHERE user="bob" AND pw="qwery1#";’
goofy	a"bc	‘… WHERE user="goofy" AND pw="a"bc";’
weird	abc" or pw="123	‘… WHERE user=”weird" AND pw="abc" or pw="123";’
eve	" or 1=1; --	‘… WHERE user=”eve" AND pw="" or 1=1; --";’
mallory"; --		‘… WHERE user="mallory"; --" AND pw="";’

Systems Security Principles

Defense in Depth

1. Fail-safe Defaults
2. Separation of Privilege
3. Least Privilege
4. Open Design
5. Economy of Mechanism
6. Complete Mediation
7. Compromise Recording
8. Work Factor



5 Lessons

verify assumption about input, reject bad/unforseen inputs

Lesson 1:

Never trust input from the user

Lesson 2:

Never mix code and data

"write a page or execute a page"

Lesson 3:

Use the best tools at your disposal

Lesson 4:

(get this in 2550)

Awareness and Vigilance

Lesson 5:

Patch!

Topics we did not cover

- Crimeware Botnets
- Post-quantum cryptography
- Crypto currencies and smart contracts
- Protocol Security (TLS, wireless, SDN)
- Side channel attacks
- Secure Hardware Technologies (TPM, TXT)
- Distributed System Security and Resilience
- Privacy and regulations
- Fuzzing and software testing
- Formal verification
- Mobile and IoT security
- Machine Learning for Security
- Adversarial Machine Learning

Failures

O

I

D

A

Failures

O peration

I

D

A

Failures

O peration

I mplementation

D

A

Failures

O peration

I mplementation

D esign

A

Failures

O peration

I mplementation

D esign

A bstraction