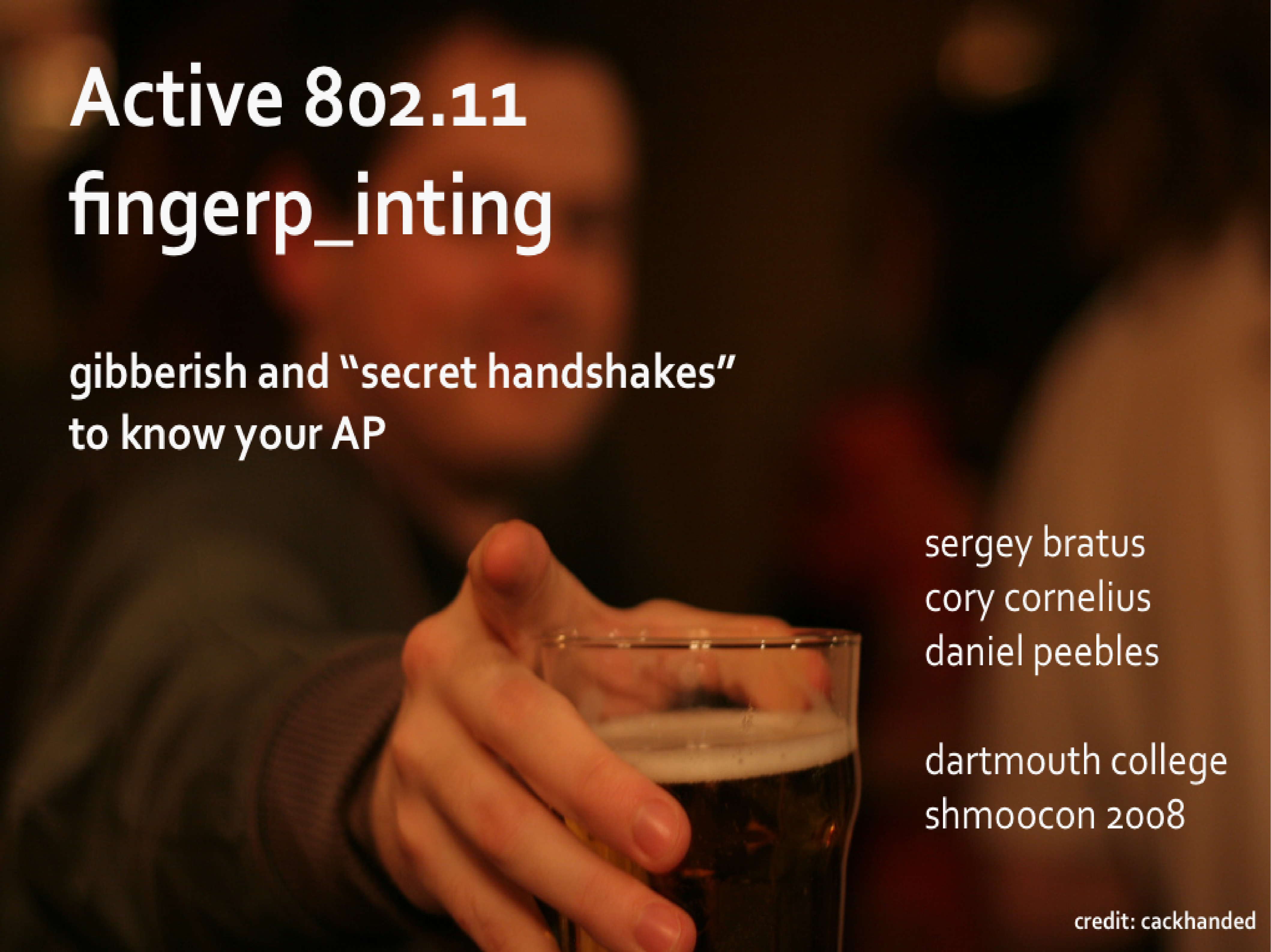


Active 802.11 fingerp_inting

gibberish and “secret handshakes”
to know your AP

sergey bratus
cory cornelius
daniel peebles

dartmouth college

shmoocon 2008

credit: cackhanded

Active 802.11 Fingerprinting: gibberish and “secret handshakes” to know your AP

Sergey Bratus, Cory Cornelius, Daniel Peebles
Dartmouth College

Shmoocon 2008

This talk in 5 minutes (1)

“How it started?”

- TC7, Johnny Cache: different 802.11 clients responded differently to change of BSSID in Auth & Assoc Resp.
  - *Wow, TCP/IP stack fun all over again! (“You are in a maze of twisty implementations, all slightly different”).*

This talk in 5 minutes (2)

"What is this about?"

AP vs clients: is it "*Can the castle fight off barbarians?*"

More like: "Can the peasants find the right castle?"

Famous attacks on clients fake the castle (i.e., the AP):

- Shmoo: "802.11 bait: badass tackle ..." (TC7, '05)
- Dai Zovi, Macaulay: KARMA (CanSecWest '05)
- Simple Nomad: "Hacking the friendly skies"
- Cache & Maynor: "Hijacking a MacBook in 60 sec"
- The Month of Kernel Bugs (Nov 2006), ...

This talk in 5 minutes (3)

“What's in a fingerprint?”

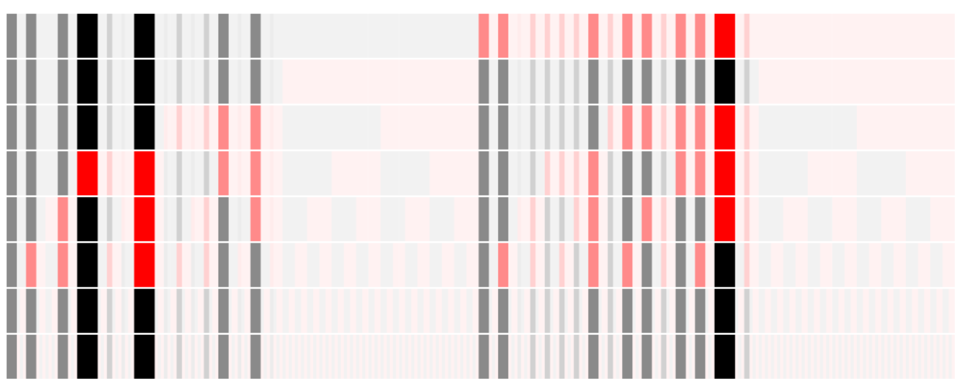
- With enough resources and observations, you can fingerprint almost anything
  - Timings, Electric or RF signal, Fourier analysis, ...
- When cheap and straightforward, it's fun
  - ... like different code logic (*Nmap* & friends)
- Lots of protocol **states** & **fields** => lots of differences
  - ... and some combinations are gibberish
  - 802.11 has lots of these even in L2 headers: (e.g., mismatched ***type*** and ***flags*** in ***Frame Control***)

So test how your AP reacts to gibberish, at a glance.
If the picture is different, it's likely NOT your AP.

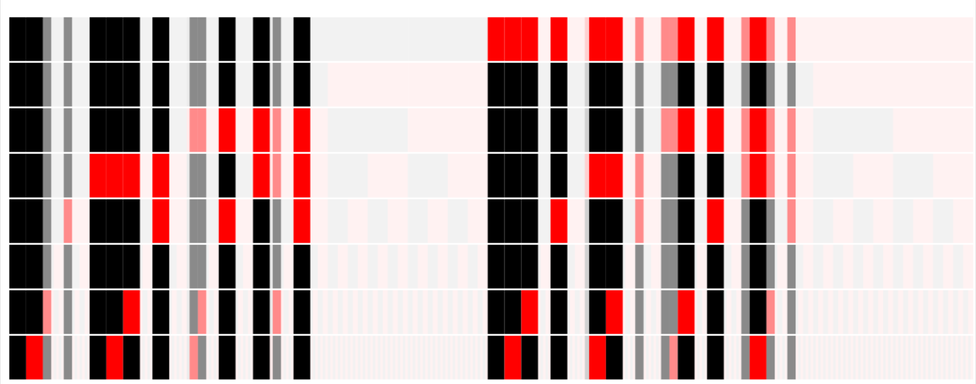
This talk in 5 minutes (4)

“AP responses at a glance”

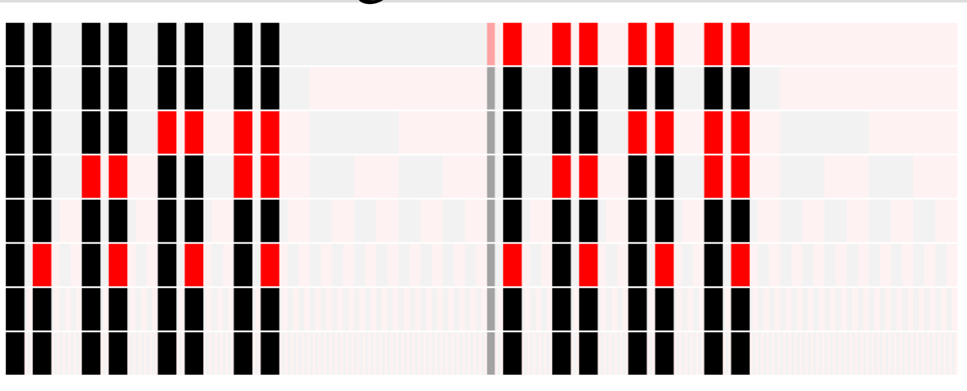
Linksys WRT54g:



Prism II HostAP soft AP:



Madwifi-ng soft AP:

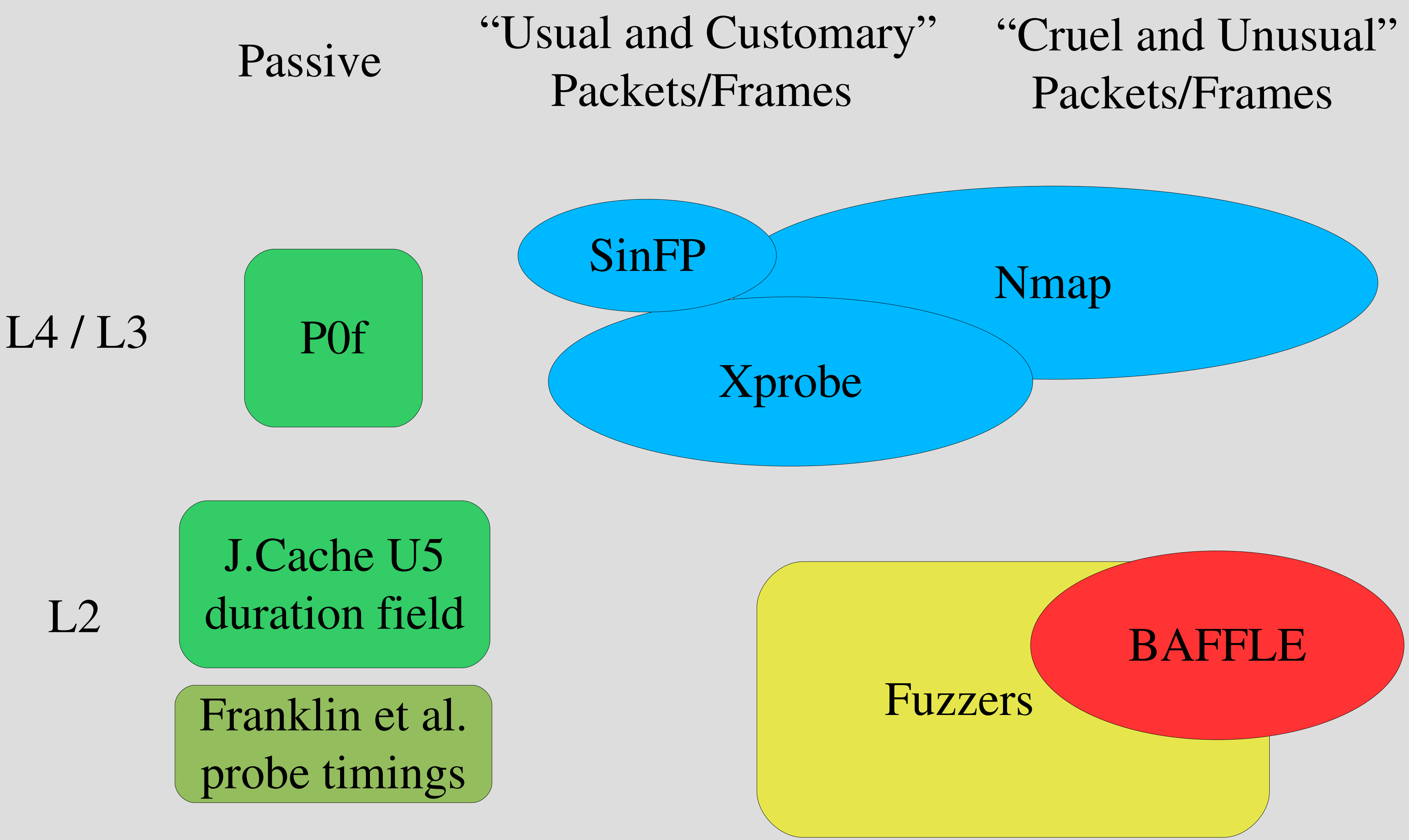


Auth Requests with non-sensical combinations of flags

BAFFLE

- Written in Ruby 1.9
- Uses Ruby LORCON from Metasploit
  - forever indebted to the authors!
- Builds pcap/BPF filters for 802.11 frames from Ruby objects
- A special language for describing tests, stimuli and training

“Where we fit in”



Passive

“Usual and Customary” Packets/Frames

“Cruel and Unusual” Packets/Frames

L4 / L3

P0f

SinFP

Nmap

Xprobe

L2

J.Cache U5 duration field

Franklin et al. probe timings

Fuzzers

BAFFLE

Some history

- L3 TCP/IP stack fingerprints:
  - Classics
  - New developments
- Countermeasures
- L2 802.11 fingerprinting

The Noble Art of L3 Fingerprinting:

“part of a complete TCP/IP VA kit”

- **Nmap** (1998, 2006--)
  - 2nd gen. OS fingerprinting: http://nmap.org/osdetect/
- **Xprobe** (2001, 2002—2005)
    - “fuzzy logic”
- **P0f**, the passive fingerprinter (2000, 2006)
    - preceded by “Siphon”, adopted by Ettercap, many others
- **SinFP** (2005)
    - attempts single-port, 3-packet OS fingerprinting
- …

The Noble Art of L3 Fingerprinting

--Countermeasures--

- Smart, Malan, Jahanian (USENIX, 2000)
  - *“Defeating TCP/IP OS stack fingerprinting”*
  - scrubbers suppress “cruel and unusual” packets, breaking known signatures
- Kathy Wang (DC-12, 2004)
  - *“Frustrating OS fingerprinting with Morph”*
  - don't just mess up signatures, emulate them
- Niels Provos (USENIX, 2004)
  - *“A virtual honeypot framework”,* Honeyd
  - ... emulate them for entire honeynets

The Noble Art of L3 Fingerprinting

--Timing-related--

- Tony Capela (DC-11, 2003): **Ping RTT**
  - “*Fashionably late - what your network's RTT tells...*”
- Kohno, Broido, Claffy (2005): **Clock skew**
  - *“Remote Physical Device Fingerprinting”* paper
- Dan Kaminsky (2005): **IP timers**
  - Fragment reassembly timeouts differs between stacks
- ... many others

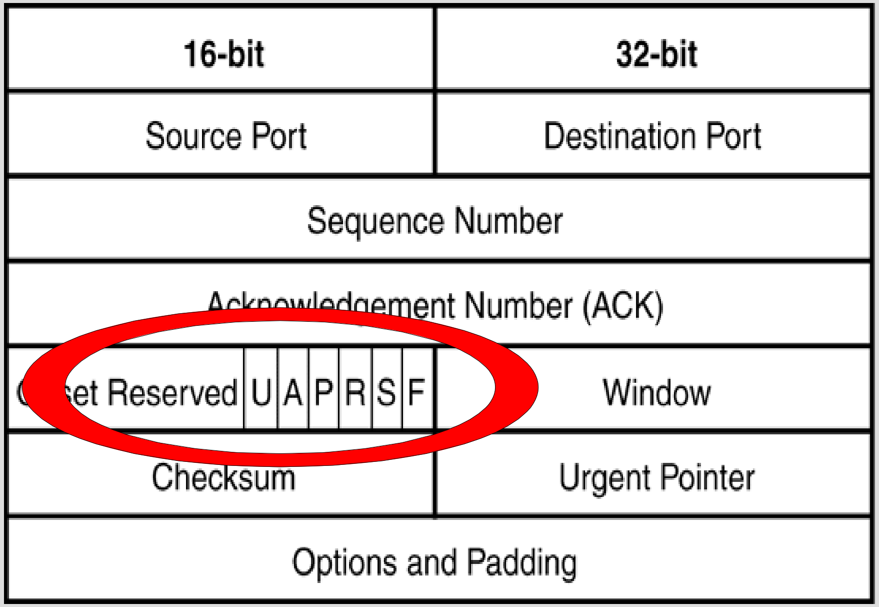
Timeline

- 1998: **Nmap** gets OS fingerprinting
  - 2000: “Scrubbers” suggested to remove anomalies
  - 2001: **Norm** (Handley et al.) normalized TCP at 100,000 pkts/sec (against IDS evasion)
- 2001: **Xprobe** fingerprints less-used but “normal” ICMP, etc.
  - 2004: **Honeyd** fakes responses of different OSes [see nmap.prints, xprobe2.conf]; **Morph**
- 2003, 2005: Timing-related fingerprinting
  - ?

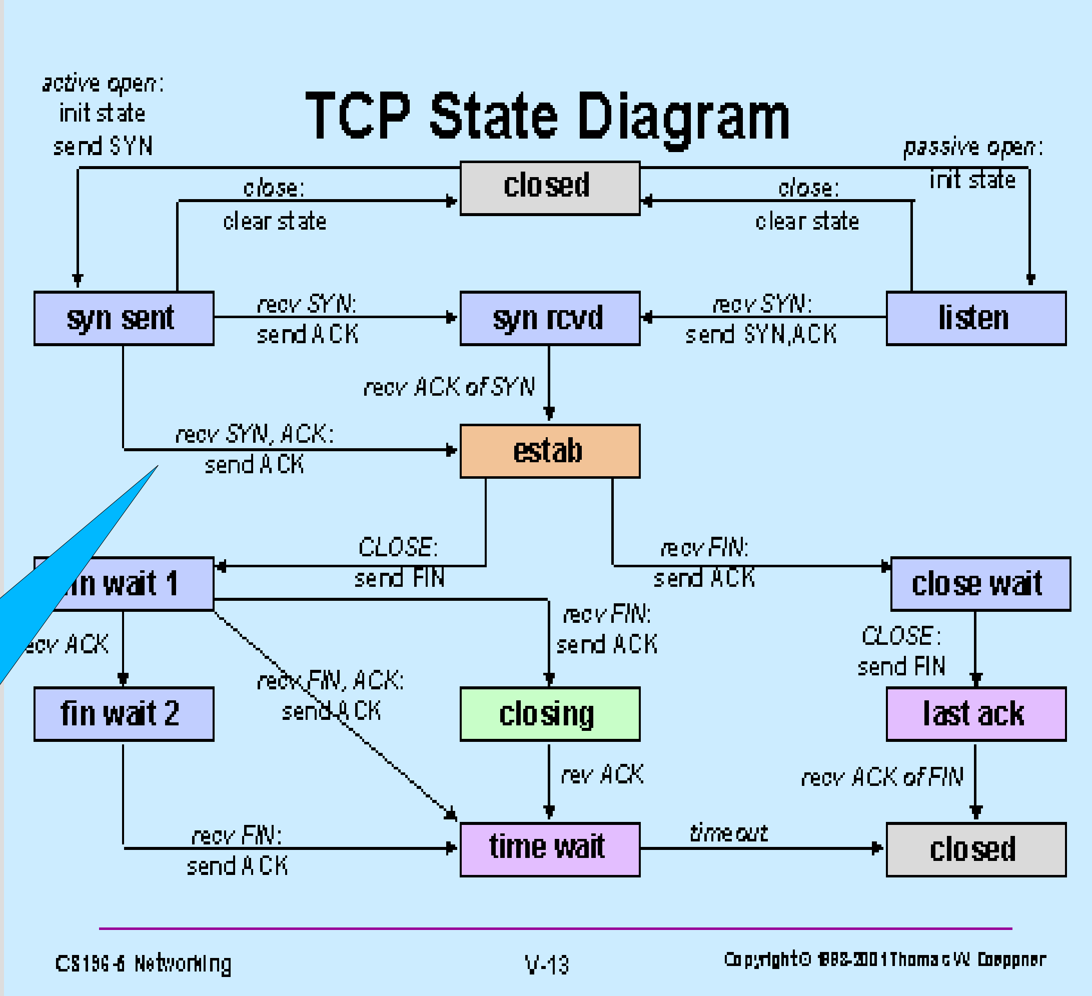
802.11: a whole new L2

- Johnny Cache (Toorcon, 2005)
  - *“802.11 VLANs and Association Redirection”*
  - different client responses to BSSID change in Auth Response and Assoc Response frames from AP
- Johnny Cache (Uninformed 5, 2006)
  - *“Fingerprinting 802.11 implementations via statistical analysis of the duration field”*
  - Passive. “Client associates, gets an IP, loads a few webpages”
- Franklin et al. (USENIX Sec, 2006)
  - *“Passive link layer 802.11wireless device driver fingerprinting”*
  - Client scanning behavior, time intervals between probes
- ...

State machines and “extra bits”: *TCP*



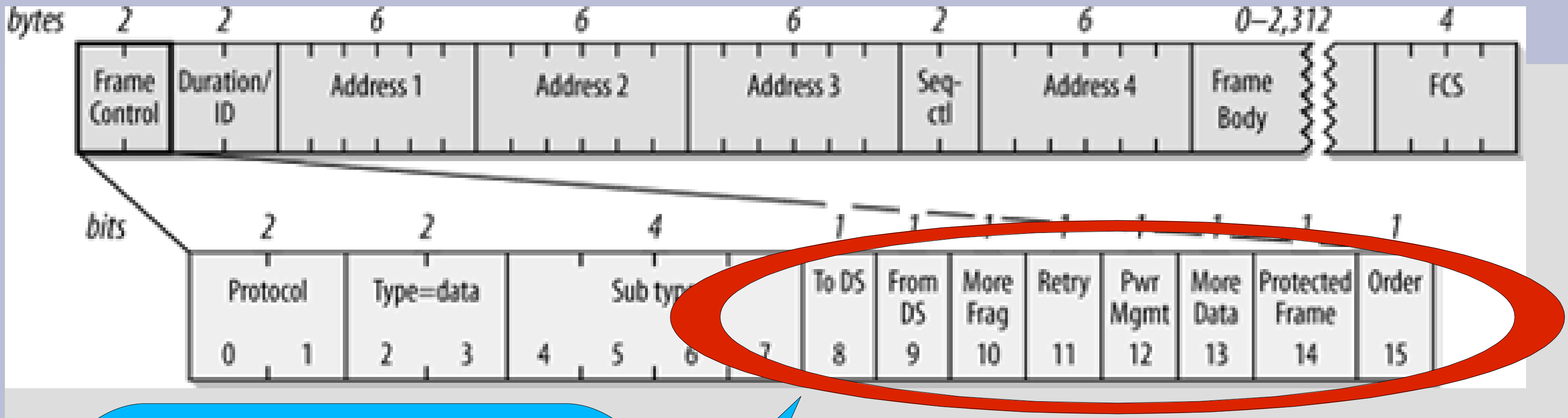
16-bit	32-bit
Source Port	Destination Port
Sequence Number	
Acknowledgement Number (ACK)	
Offset Reserved U A P R S F	Window
Checksum	Urgent Pointer
Options and Padding	



Some fields are meaningless in at least some of the states.

Nmap says hello.

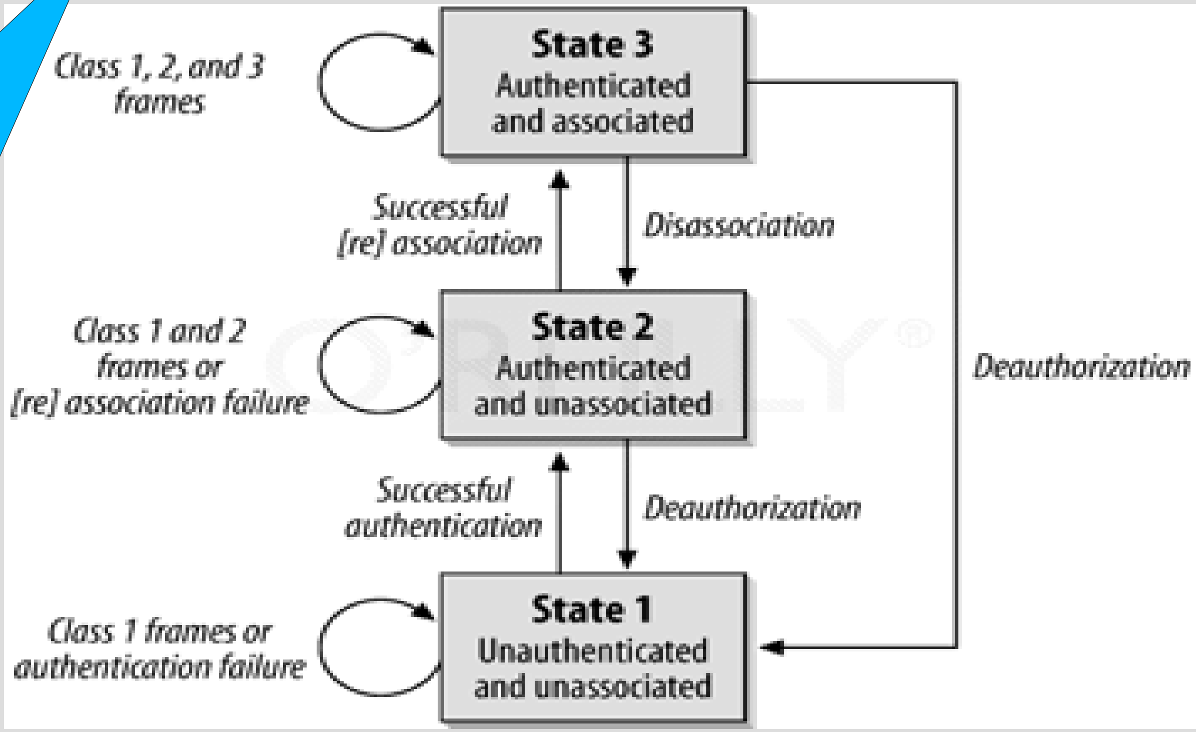
802.11 states and fiddly bits



Not all flags make sense for all types & subtypes.

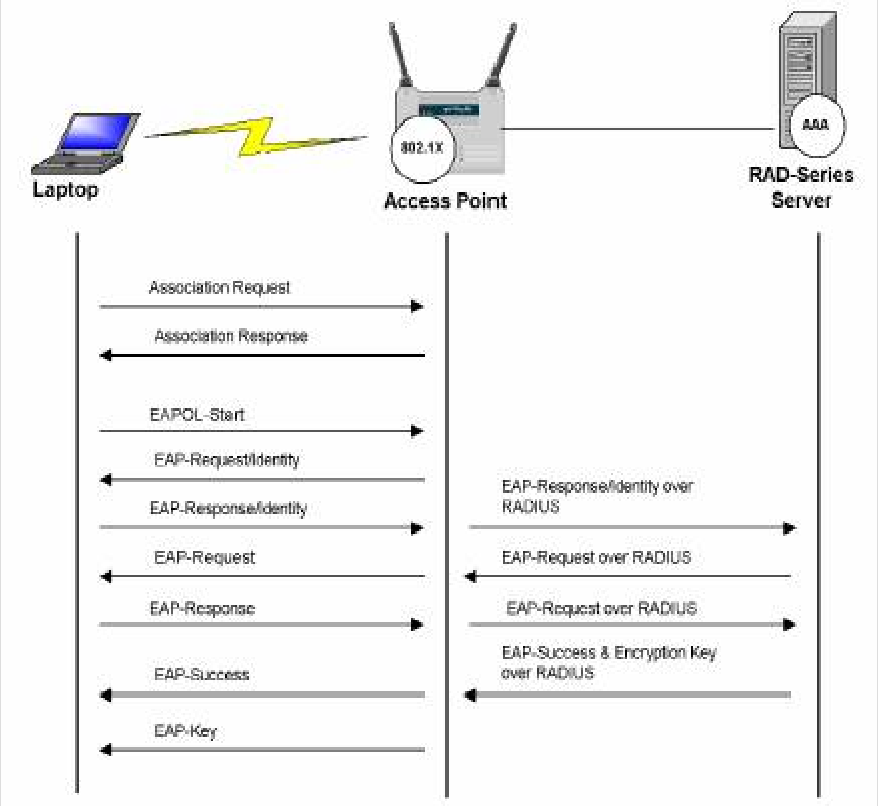
Not all flags make sense for all states.

Hello BAFFLE.


State 3
Authenticated and associated
Class 1, 2, and 3 frames
Successful [re] association
Disassociation
State 2
Authenticated and unassociated
Class 1 and 2 frames or [re] association failure
Deauthorization
Successful authentication
Deauthorization
State 1
Unauthenticated and unassociated
Class 1 frames or authentication failure

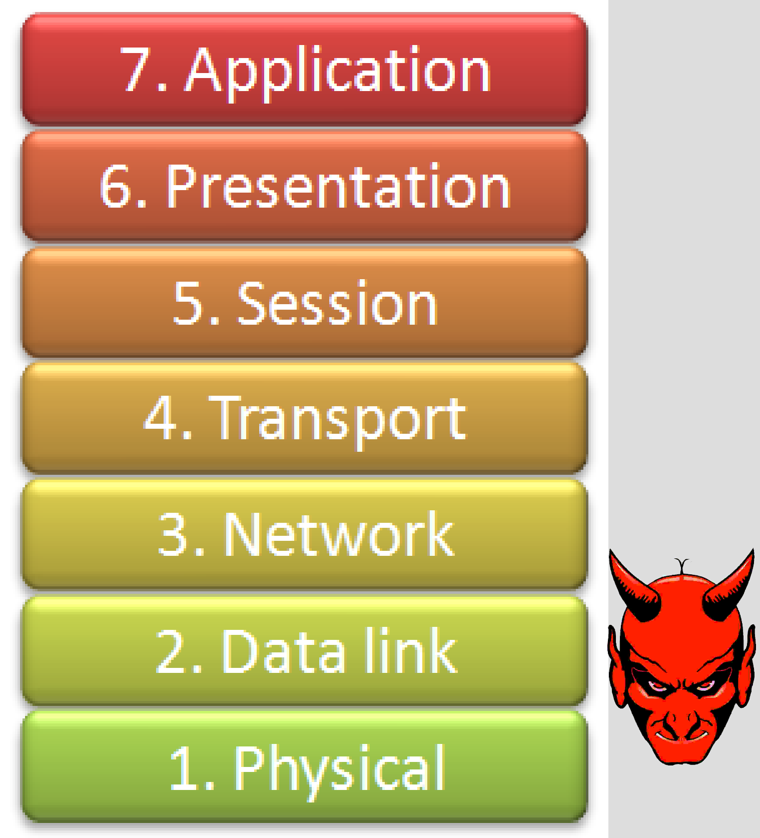
Can a client station trust an AP?

- Is this AP one of a trusted group, or evil faker?
- *Why yes, just exchange some crypto with it, and verify the AP knows the right secrets.*
- Problem solved, right?
- Not exactly: are all these exchanges ***bug-free***?

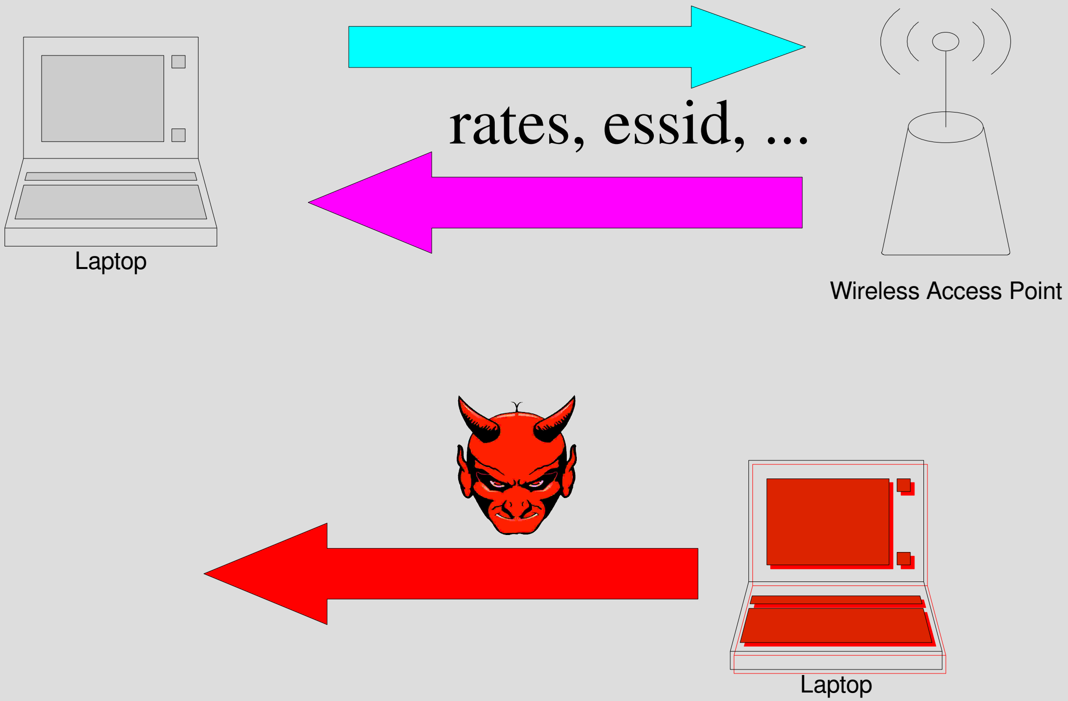


Your L2 is possessed by the devil

- “Hijacking a MacBook in 60 seconds”
- “The month of kernel bugs”, ...

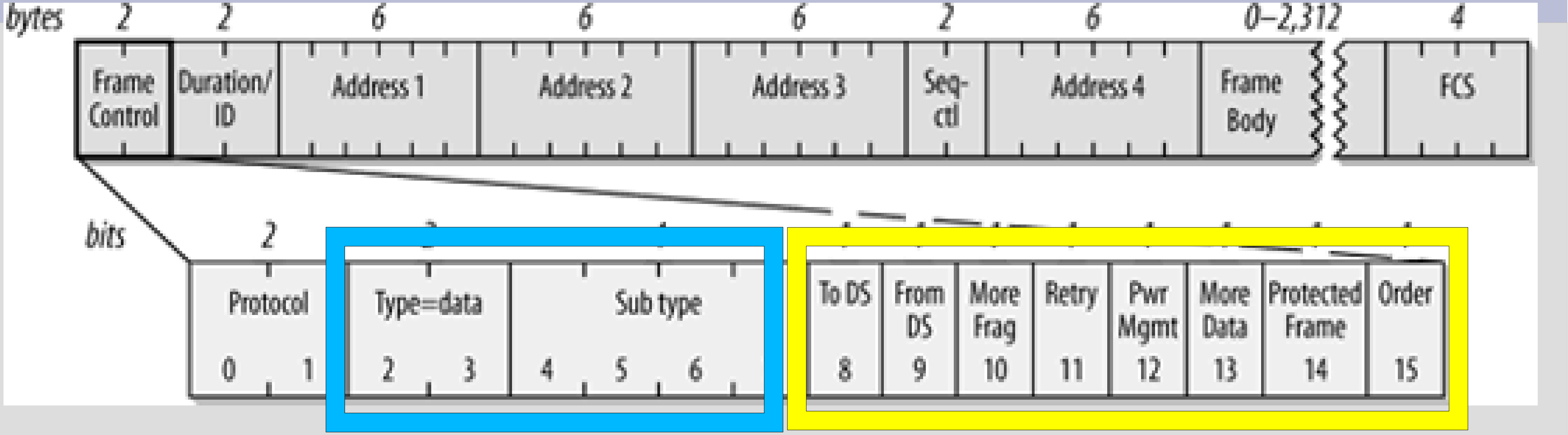


Probe Request -- Probe Response



rates, essid, ...

802.11 fiddly bits



- **Type/Subtype:** Mgmt, Control or Data / various modes
- **ToDS**, **FromDS**: frame from or to distribution system
  - zero on management and control frames
- **MoreFrag**: more L2 fragments to follow
- **PwrMgmt**: station goes into Power Save mode (PS)
- **MoreData**: AP has data buffered for station in PS mode

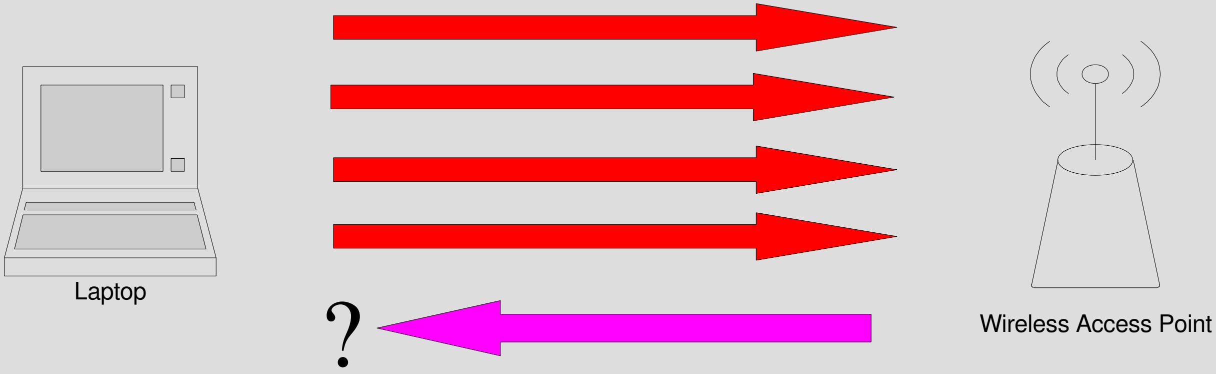
		Association Request	Association Response	Reassociation Request	Reassociation Response	Probe Request	Probe Response	Beacon	ATIM	Disassociation	Authentication	Deauthentication	Action	4 * Reserved	BlockAckReq	BlockAck	Power Save (PS)-Poll	Request To Send (RTS)	Clear To Send (CTS)	Acknowledgment (ACK)	Contention Free (CF)-End	CF-End + CF-ACK	8 * Reserved	Data	*Data + CF-ACK	*Data + CF-Poll	*Data + CF-ACK + CF-Poll	Null Function (no data)	*CF-ACK (no data)	*CF-Poll (no data)	*CF-ACK + CF-Poll (no data)	*QoS Data	*QoS Data + CF-ACK	*QoS Data + CF-Poll	Qos Data + CF-ACK + CF-Poll	*QoS Null (no data)	*QoS CF-Poll (no data)	*QoS CF-ACK + CF-Poll (no data)	Reserved	Reserved
To DS	0	■	■	■	■	🟥■	🟥■	◦	■	■	🟥■	■	■		■	■	■	■	■	■	■	■		■				■												
	1					🟥	🟥				🟥													■				■												
From DS	0	■	■	■	■	🟥■	🟥■	◦	■	■	🟥■	■	■		■	■	■	■	■	■	■	■		■				■												
	1					🟥	🟥				🟥													■				■												
More Fragments	0	■	■	■	■	🟥■	🟥■	◦	■	■	🟥■	■	■		■	■	■	■	■	■	■	■		■				■												
	1	•	•	•	•	🟥•	🟥•	◦	•	•	🟥•	•	•											■				■												
Retry	0	■	■	■	■	🟥■	🟥■	◦	■	■	🟥■	■	■											■				■												
	1	■	■	■	■	🟥■	🟥■	◦	■	■	🟥■	■	■											■				■												
Power Management	0	■	■	■	■	🟥■	🟥■	◦	■	■	🟥■	■	■																											
	1	■	■	■	■	🟥■	🟥■	◦	■	■	🟥■	■	■																											
More Data	0	■	■	■	■	🟥■	🟥■	■	■	■	🟥■	■	■																											
	1	■	■	■	■	🟥■	🟥■	■	■	■	🟥■	■	■																											
Protected Frame	0	■	■	■	■	🟨■	🟥■	■	■	■	🟥■	■	■												■				■												
	1					🟨	🟥				🟥■																													
Order	0					🟥	🟥				🟥													■				■												
	1					🟥	🟥				🟥													■				■												
		Management													Control									Data																Reserved

Legend
- ■ Defined by IEEE 802.11 Specification
- ◦ In IEEE 802.11 Specification but purpose seems undefined
- • In IEEE 802.11 Specification but unlikely
- 🟥 Tested by BAFFLE
- 🟨 Tested by BAFFLE but of limited utility
- □ Not defined in IEEE 802.11 Specification
- * In IEEE 802.11 Specification but mostly unimplemented

Gibberish

- ToDS and FromDS set on Probe & Auth Requests
  - unspecified on Mgmt and Contol frames
- MoreFrags on Probe Reqs and Auth Reqs
  - will the AP wait for more, ignore or respond?
- MoreData from station to AP (say what?)

So: send lots of garbage frames, listed for responses (varying source MACs helps)



“Secret handshake with an AP”

- All you really know about an AP is its BSSID/MAC

- Don't trust your driver?
- Scared of getting too close with an AP before you can learn anything about it through crypto? (and you have to get pretty intimate to use crypto)

- Choose some weird things than your APs do
- Check if the BSSID in question does them

Thanks!

- Johnny Cache for the many inspirations
- Joshua Wright and Mike Kershaw for LORCON
- Uninformed and Toorcon crews
- everyone else who helped us (authors of Ruby, Lapack, Metasploit, ...)