The Final Nail in WEP's Coffin

Andrea Bittau[1] Mark Handley[1] Joshua Lackey[2]

May 24, 2006

[1]University College London.
[2]Microsoft.

Wired Equivalent Privacy

WEP is the 802.11 standard for encryption.

- Pre-shared key for whole network.
- Protects data privacy since data is encrypted.
- Access control: need key to transmit.

In practice, only half of the networks are encrypted.

- In the subset of encrypted networks, WEP is most adopted.

Popularity (%) of WEP and its alternatives based on our survey

Region	WEP	WPA	802.11i
London	76	20	4
Seattle region	85	14	1

Goals when attacking WEP

- Decrypt data in packets.
- Obtain access to the network by being able to transmit data.
- Recover the WEP key.

Contribution

Today, millions of packets are required to break a WEP key.

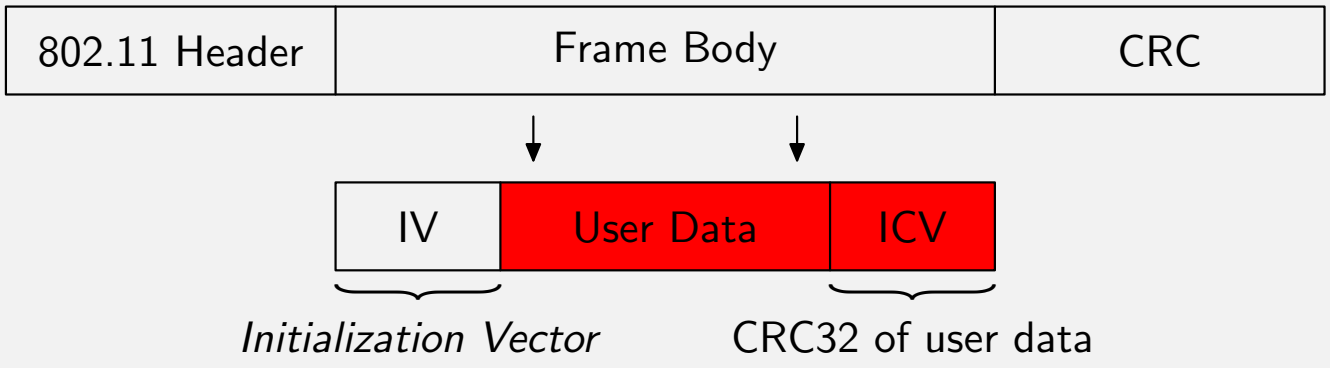
Our fragmentation attack allows:

1. Transmitting arbitrary data after eavesdropping a single data packet on an 802.11 WEP protected network.
2. Real-time decryption given that network is connected to Internet.

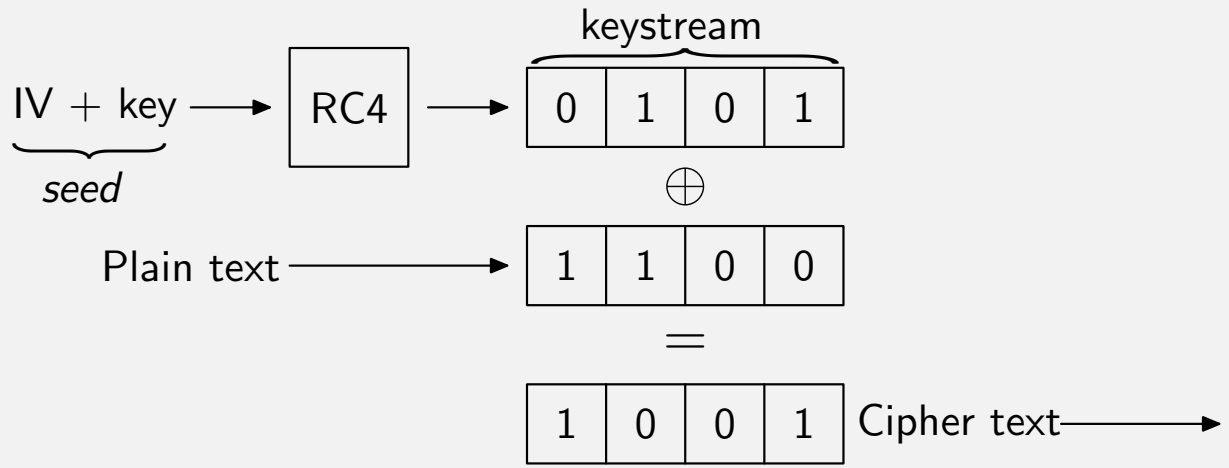
Outline

WEP operation

Data frame format



Encryption



History of WEP attacks

. . . and how the *real* problem was ignored

Year 2000. Keystream attacks (independent of WEP key):

- Design flaw: WEP allows keystream reuse.
- Attacks were thought impractical:
  - Need plain-text to recover keystream.
  - Need 2^{24} keystreams to decrypt all possible packets.

Year 2001. Weak IV attacks (recover WEP key):

- Need millions of packets. Could take hours, and usually, days.
- Use EAP-based solutions to re-key, say, every ten minutes.

Year 2006. Our contribution: fragmentation attack.

- Keystream attack which may be performed within minutes.

History of WEP attacks

. . . and how the *real* problem was ignored

Year 2000. Keystream attacks (independent of WEP key):

- Design flaw: WEP allows keystream reuse.
- Attacks were thought impractical:
  - Need plain-text to recover keystream.
  - Need 2^{24} keystreams to decrypt all possible packets.

Year 2001. Weak IV attacks (recover WEP key):

- Need millions of packets. Could take hours, and usually, days.
- Use EAP-based solutions to re-key, say, every ten minutes.

Year 2006. Our contribution: fragmentation attack.

- Keystream attack which may be performed within minutes.

History of WEP attacks

. . . and how the *real* problem was ignored

Year 2000. Keystream attacks (independent of WEP key):

- Design flaw: WEP allows keystream reuse.
- Attacks were thought impractical:
  - Need plain-text to recover keystream.
  - Need 2^{24} keystreams to decrypt all possible packets.

Year 2001. Weak IV attacks (recover WEP key):

- Need millions of packets. Could take hours, and usually, days.
- Use EAP-based solutions to re-key, say, every ten minutes.

Year 2006. Our contribution: fragmentation attack.

- Keystream attack which may be performed within minutes.

Fragmentation attack
Outline

Transmission

1. Recover a keystream.
2. Reuse the keystream to send arbitrary data.

Keystream-based decryption

- Resend data through the AP to a buddy on the Internet.
- Recover the keystream used for encrypting the packet.

WEP key recovery

Use transmission ability for speeding up weak IV attacks.

Transmission
Recovering a short keystream

If cipher-text & plain-text pair is known, their XOR is a keystream.
Known plain-text (LLC/SNAP headers) in IP packets:

802.11 header	0xAA	0xAA	0x03	0x00	0x00	0x00	0x08	0x00

Transmission
Recovering a short keystream

If cipher-text & plain-text pair is known, their XOR is a keystream.
Known plain-text (LLC/SNAP headers) in IP packets:

802.11 header	0xAA	0xAA	0x03	0x00	0x00	0x00	0x08	0x00

$\oplus$

802.11 header	Cipher-text

$=$

8 bytes of keystream

Can recover 8 bytes of keystream by eavesdropping a packet.

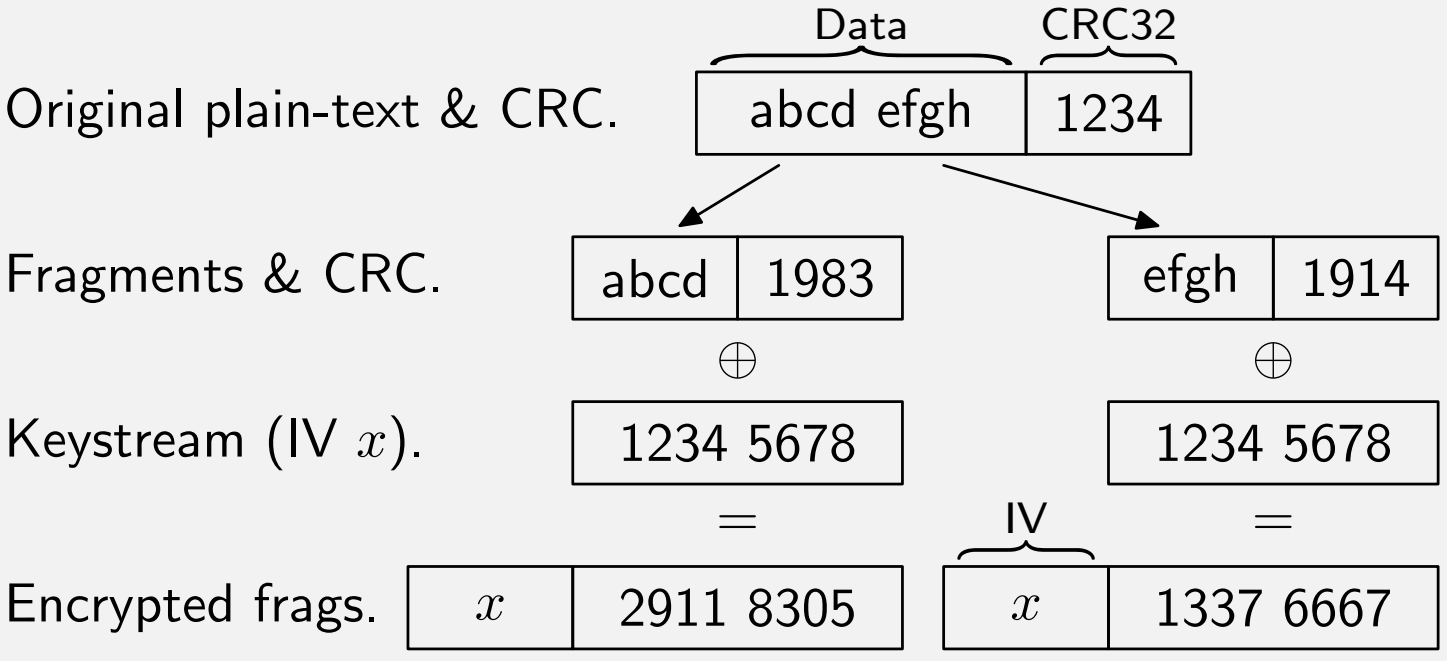
- Can encrypt (and transmit) 8 bytes of arbitrary data.

Transmission
Sending arbitrarily long data

802.11 supports MAC layer fragmentation.

- Transmit arbitrary data in 8 byte chunks.

Fragmentation



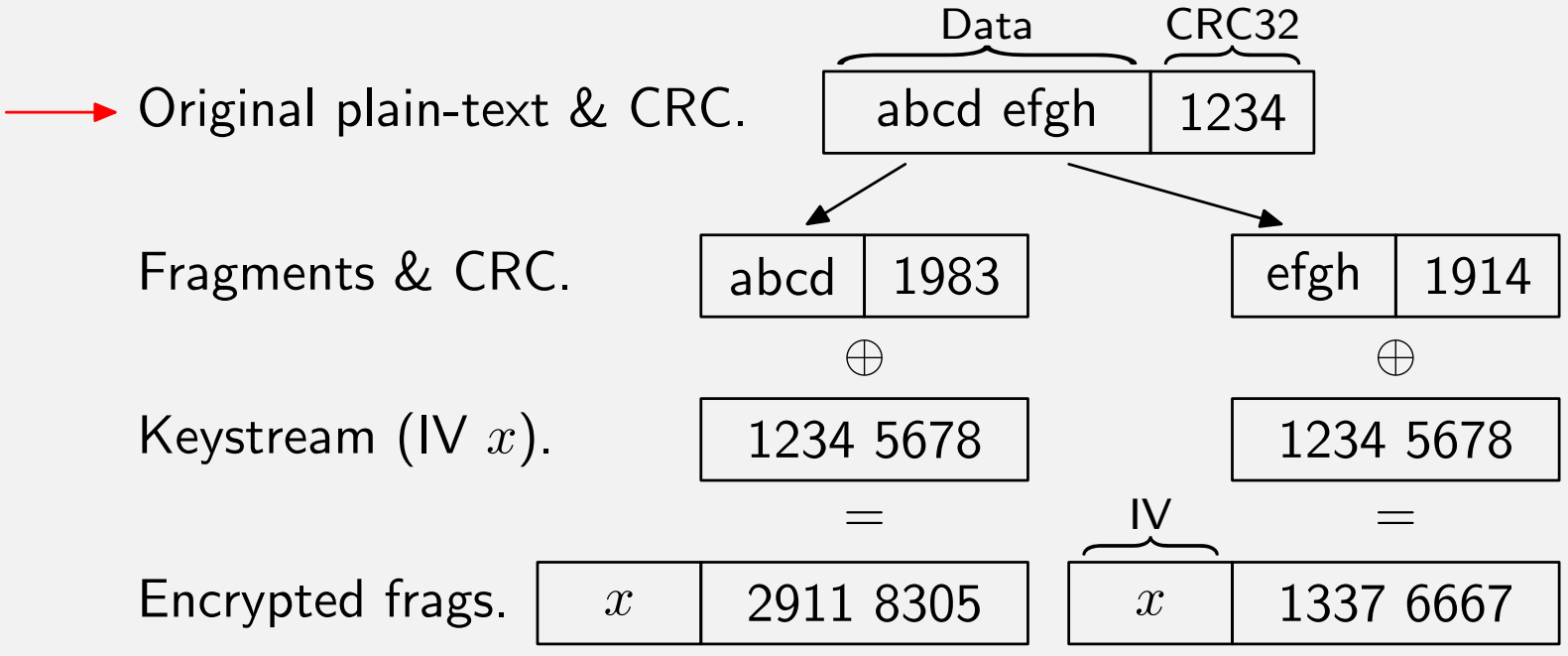
Transmission

Sending arbitrarily long data

802.11 supports MAC layer fragmentation.

- Transmit arbitrary data in 8 byte chunks.

Fragmentation



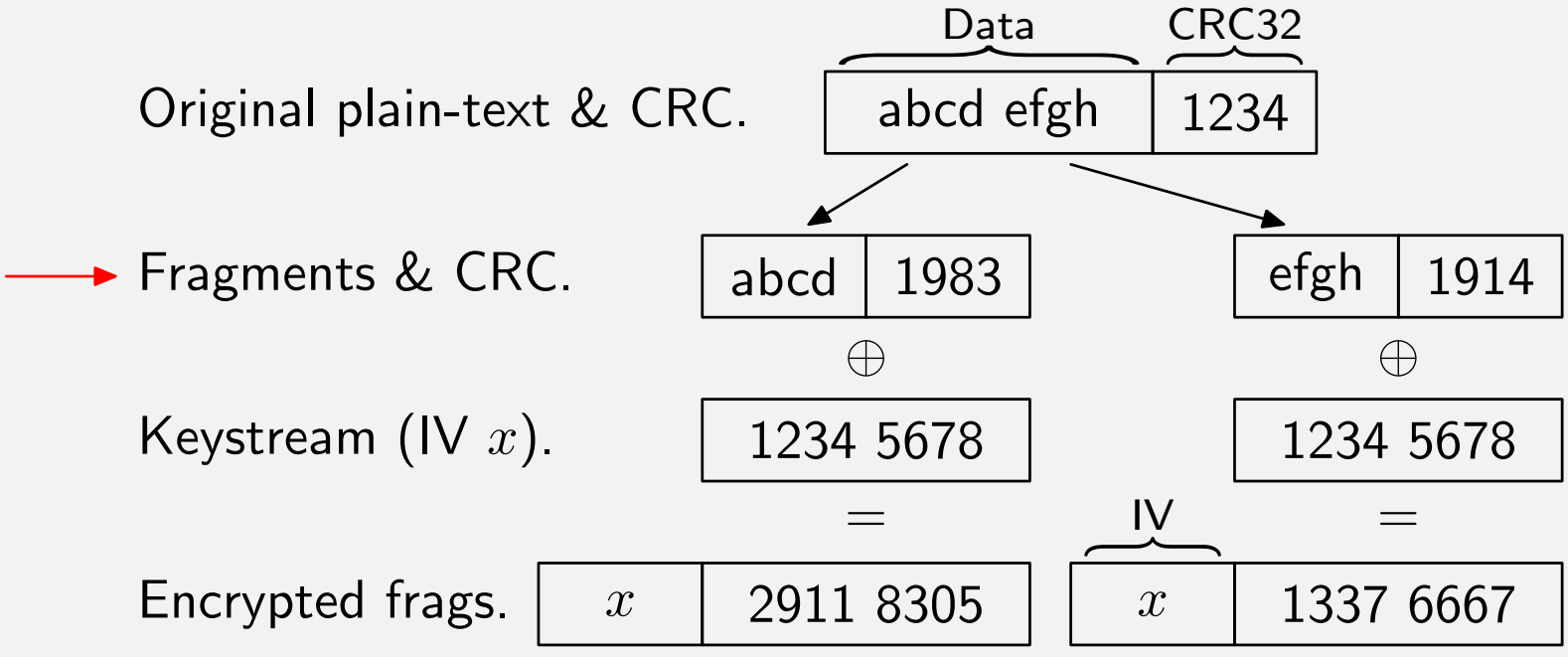
Transmission

Sending arbitrarily long data

802.11 supports MAC layer fragmentation.

- Transmit arbitrary data in 8 byte chunks.

Fragmentation



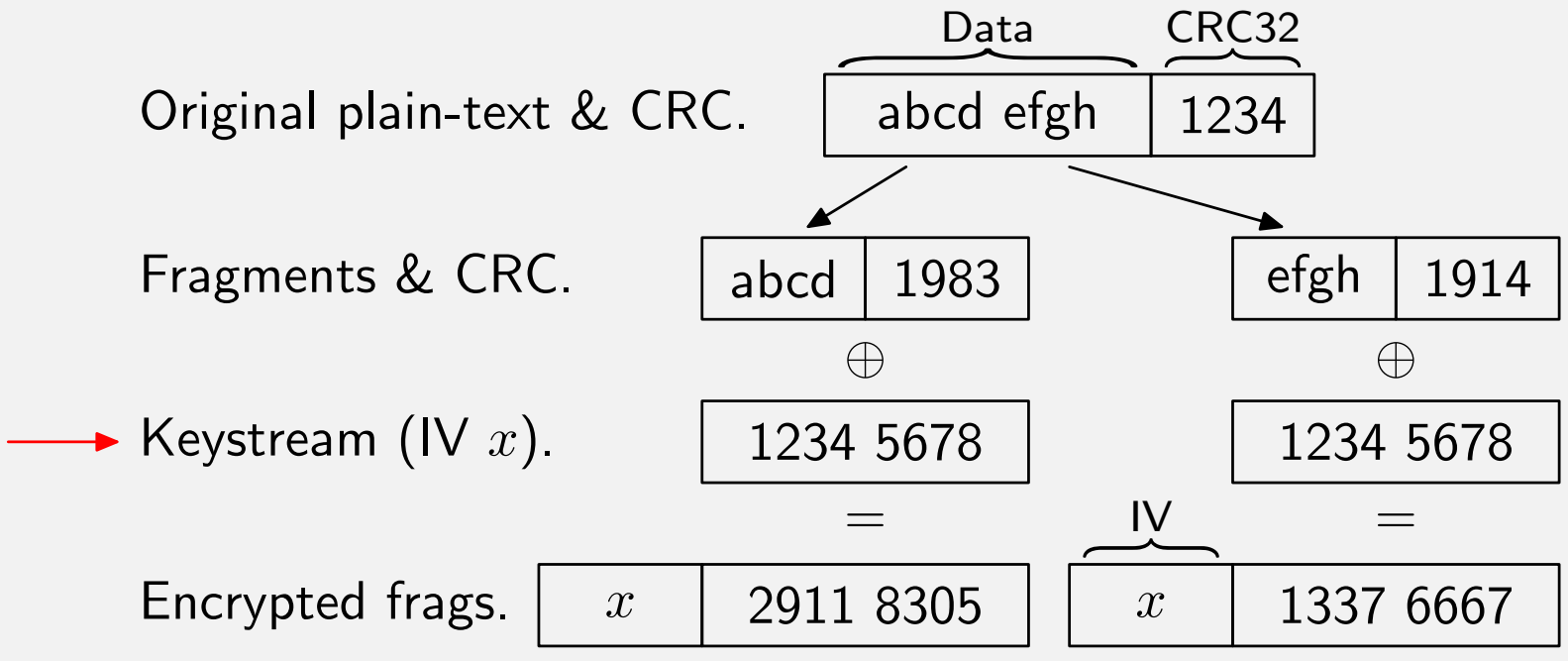
Transmission

Sending arbitrarily long data

802.11 supports MAC layer fragmentation.

- Transmit arbitrary data in 8 byte chunks.

Fragmentation



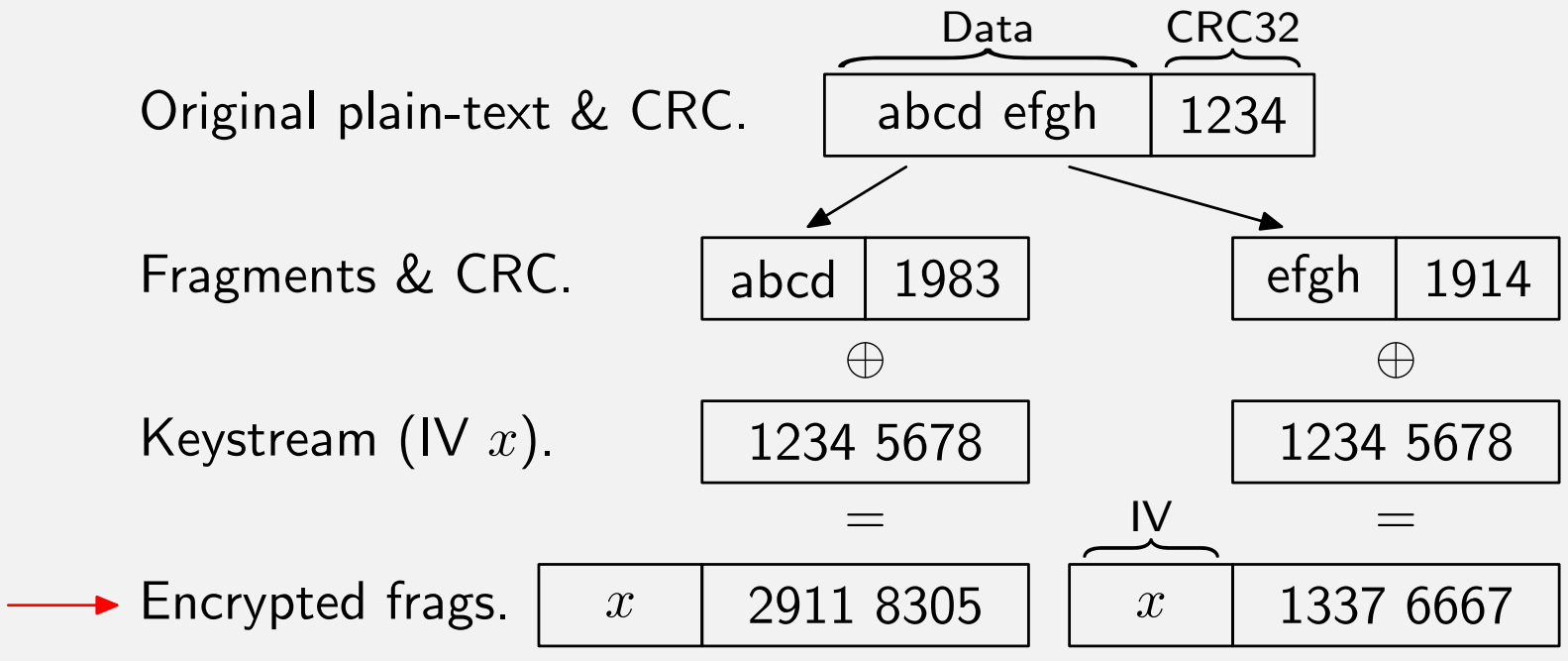
Transmission

Sending arbitrarily long data

802.11 supports MAC layer fragmentation.

- Transmit arbitrary data in 8 byte chunks.

Fragmentation

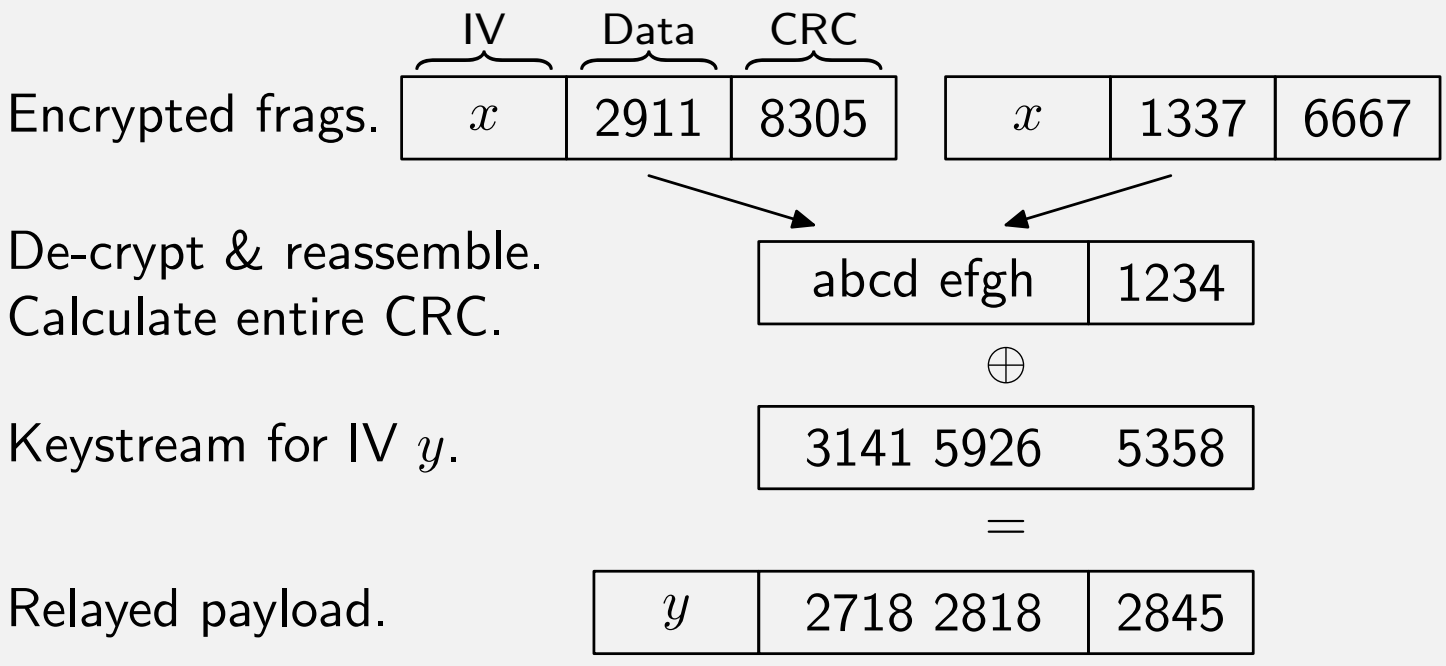


Transmission
Recovering a longer keystream

Discover a longer keystream to avoid sending many tiny packets:

- Send a long broadcast frame via multiple smaller fragments.
- AP relays it as a single packet. (New cipher & plain-text pair.)

Keystream discovery



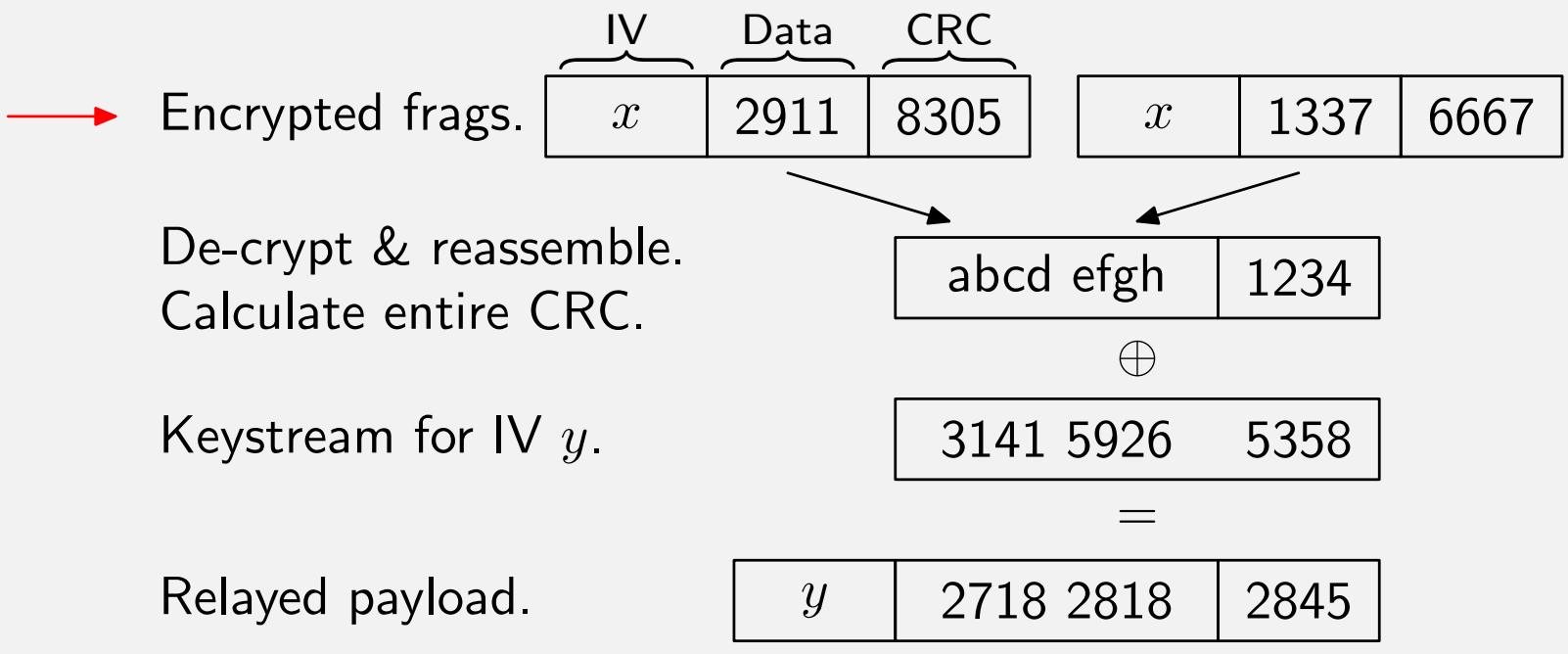
Transmission

Recovering a longer keystream

Discover a longer keystream to avoid sending many tiny packets:

- Send a long broadcast frame via multiple smaller fragments.
- AP relays it as a single packet. (New cipher & plain-text pair.)

Keystream discovery



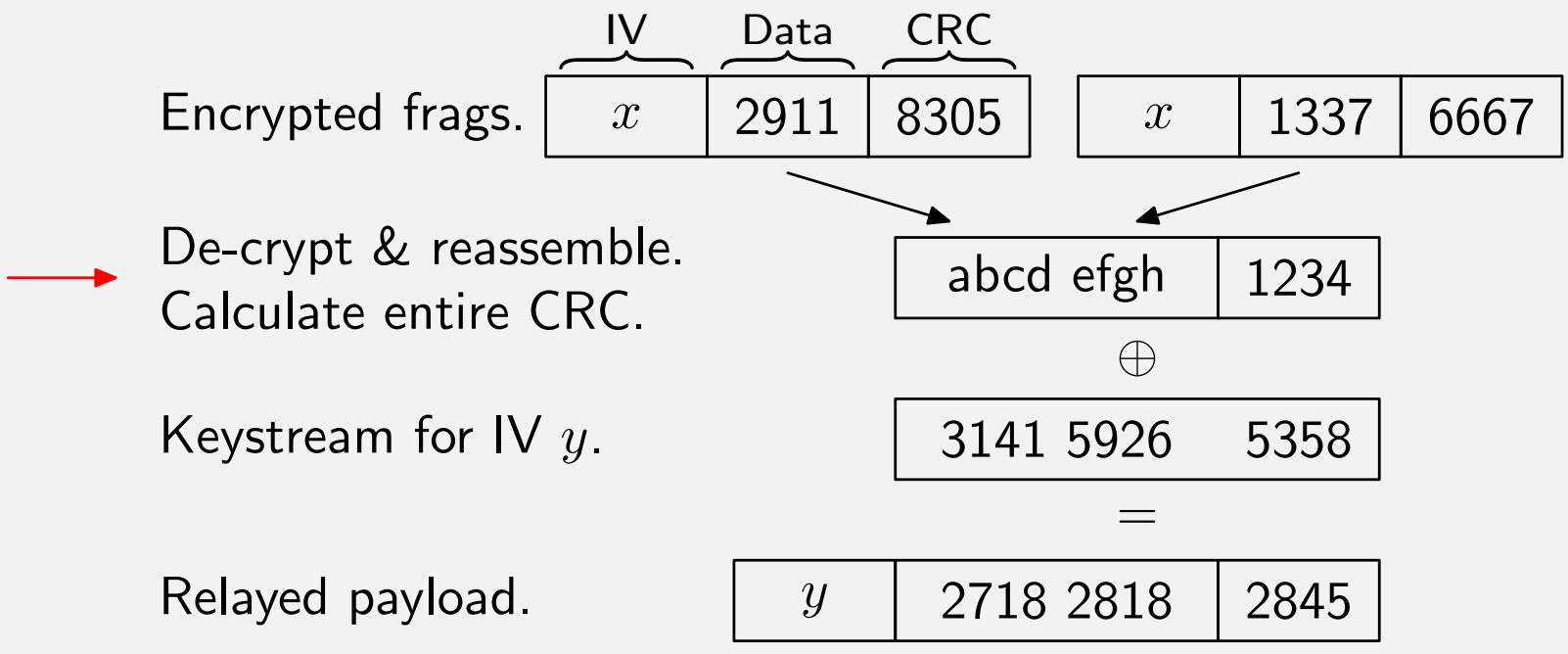
Transmission

Recovering a longer keystream

Discover a longer keystream to avoid sending many tiny packets:

- Send a long broadcast frame via multiple smaller fragments.
- AP relays it as a single packet. (New cipher & plain-text pair.)

Keystream discovery



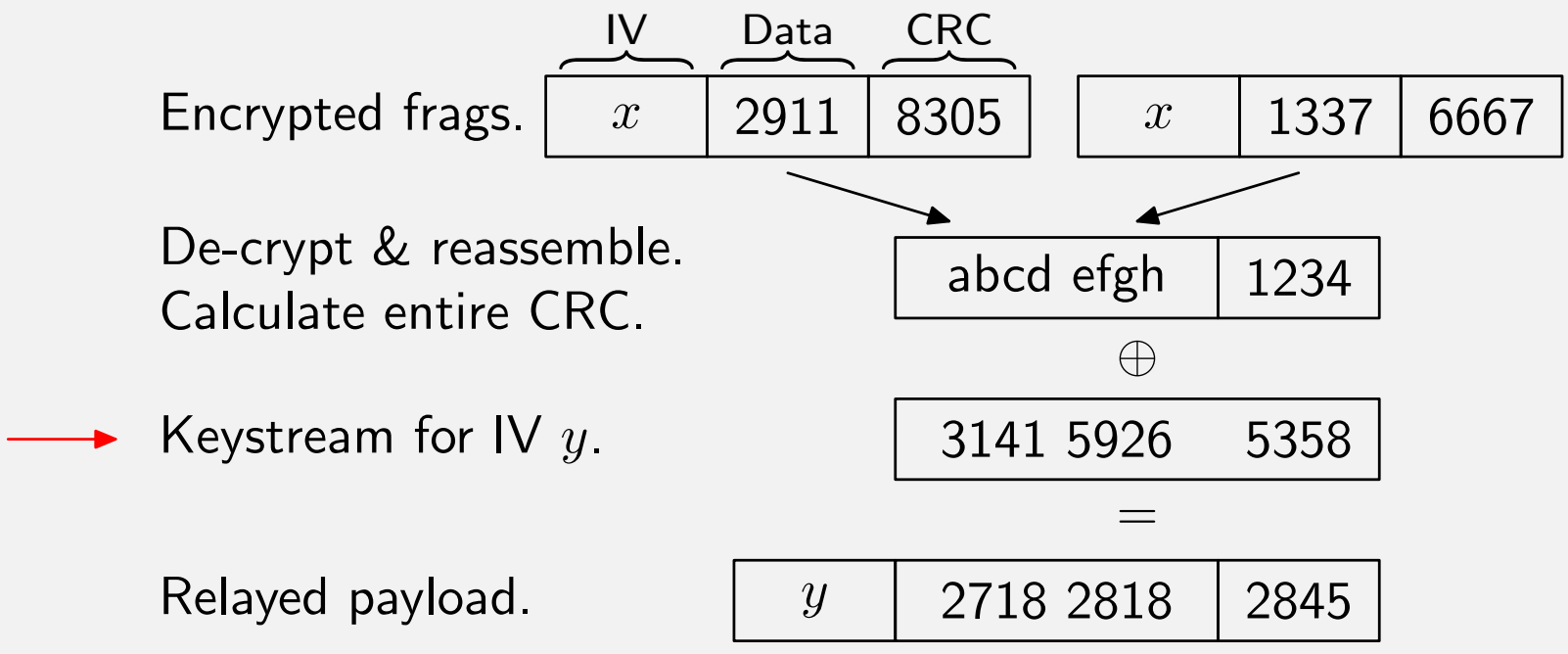
Transmission

Recovering a longer keystream

Discover a longer keystream to avoid sending many tiny packets:

- Send a long broadcast frame via multiple smaller fragments.
- AP relays it as a single packet. (New cipher & plain-text pair.)

Keystream discovery



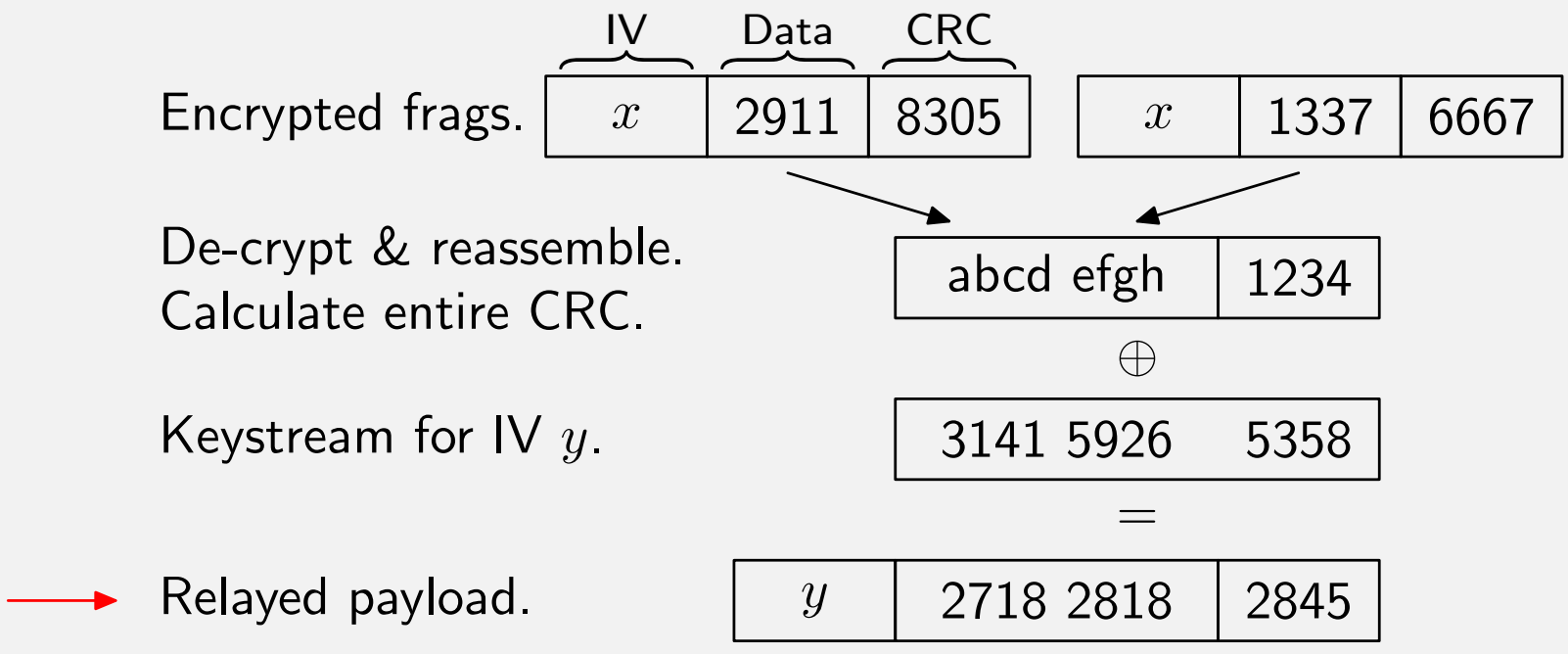
Transmission

Recovering a longer keystream

Discover a longer keystream to avoid sending many tiny packets:

- Send a long broadcast frame via multiple smaller fragments.
- AP relays it as a single packet. (New cipher & plain-text pair.)

Keystream discovery

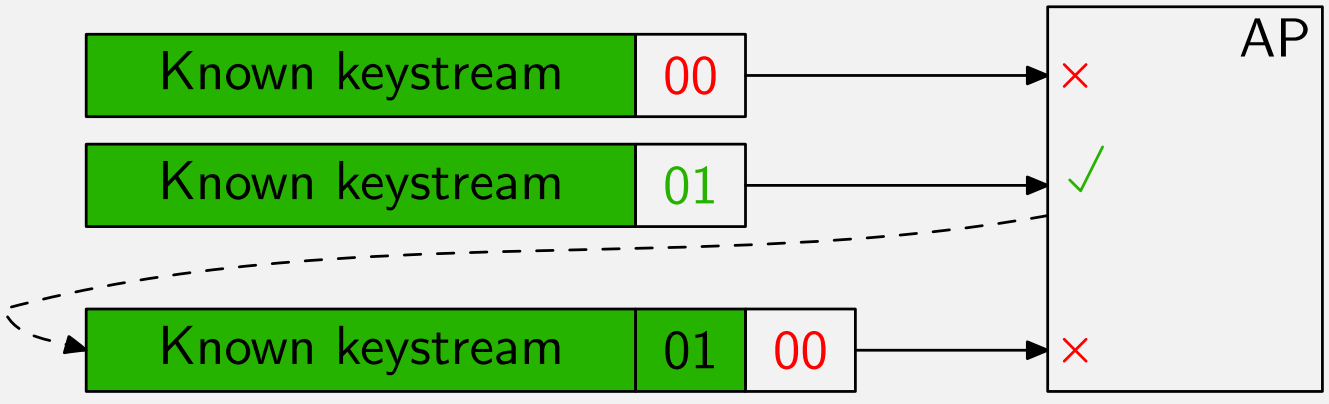


Decryption
Keystream expansion

Decrypt data locally in linear time with respect to its length:

- If n bytes of keystream were known, n bytes of data could be decrypted. Base case: 8 bytes of keystream are known.
- Guess keystream byte $n+1$ and verify it. Send a broadcast using extended keystream. If AP relayed it, guess is correct.
- Continue keystream expansion for whole length of packet.

Decryption example

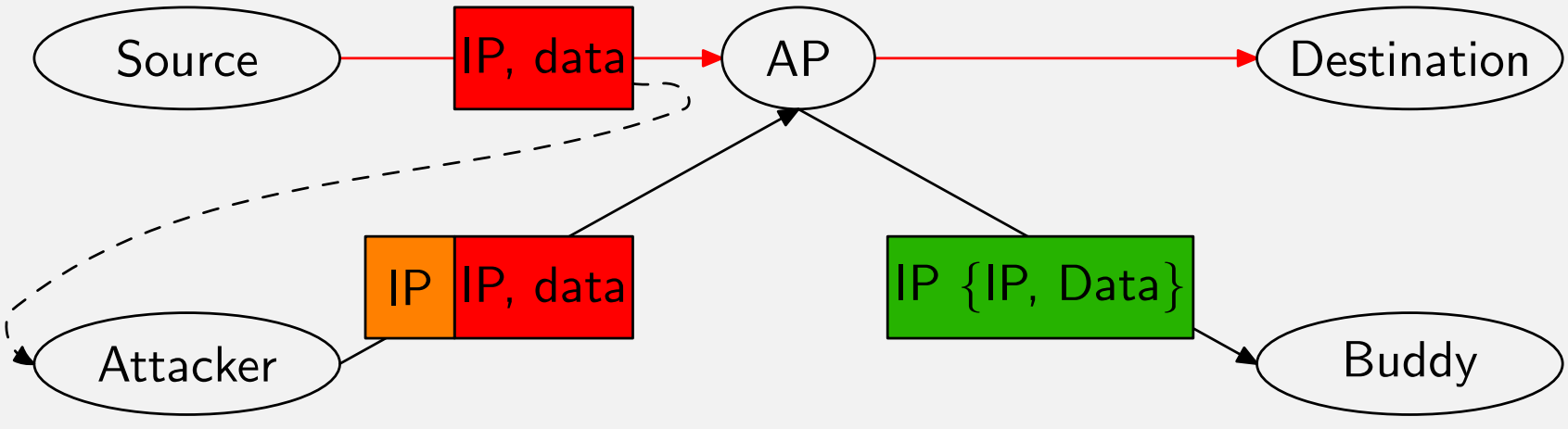


Decryption
Resending data to our Internet buddy

To decrypt data in real-time, resend it to the Internet.

1. Eavesdrop a payload to decrypt.
2. Send two 802.11 fragments: an IP header with our buddy as destination, and the original encrypted payload as a fragment.
3. AP will decrypt and send it in clear-text to our Internet buddy.

Decrypting with an Internet buddy



Fragmentation attack
Summary

To encrypt data:

1. Eavesdrop a data packet.
2. Cipher-text $\oplus$ known plain-text $=$ 8 bytes of keystream.
3. Transmit data in multiple 8 byte fragments.

To decrypt data:

1. Eavesdrop packet to decrypt.
2. Send two 802.11 fragments:
    1. An IP header destined to a buddy on the Internet.
    2. A fragment containing the original eavesdropped payload.
3. Internet buddy will receive the payload in clear-text.

Implementation

Hardware: Atheros chipset.

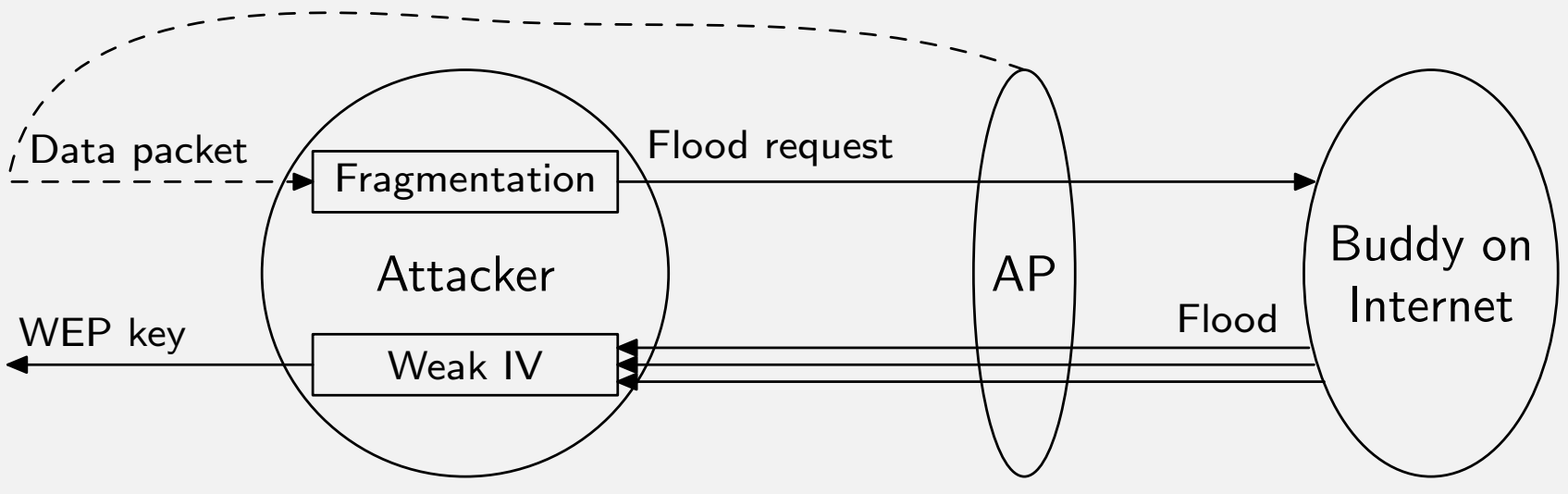
- Software radio. Ideal for packet injection.
- Supports 802.11{a,b,g}.

Software:

- FreeBSD. Added packet injection support to `ath` driver.
- `wesside`. Proof-of-concept fragmentation attack tool.

1. Eavesdrops data packet and uses fragmentation to transmit.
2. Determines the network IP via keystream expansion.
3. Contacts buddy on Internet instructing him to flood the WiFi.
4. Recovers WEP key via weak IV attack (using `aircrack`).

Operation of wesside

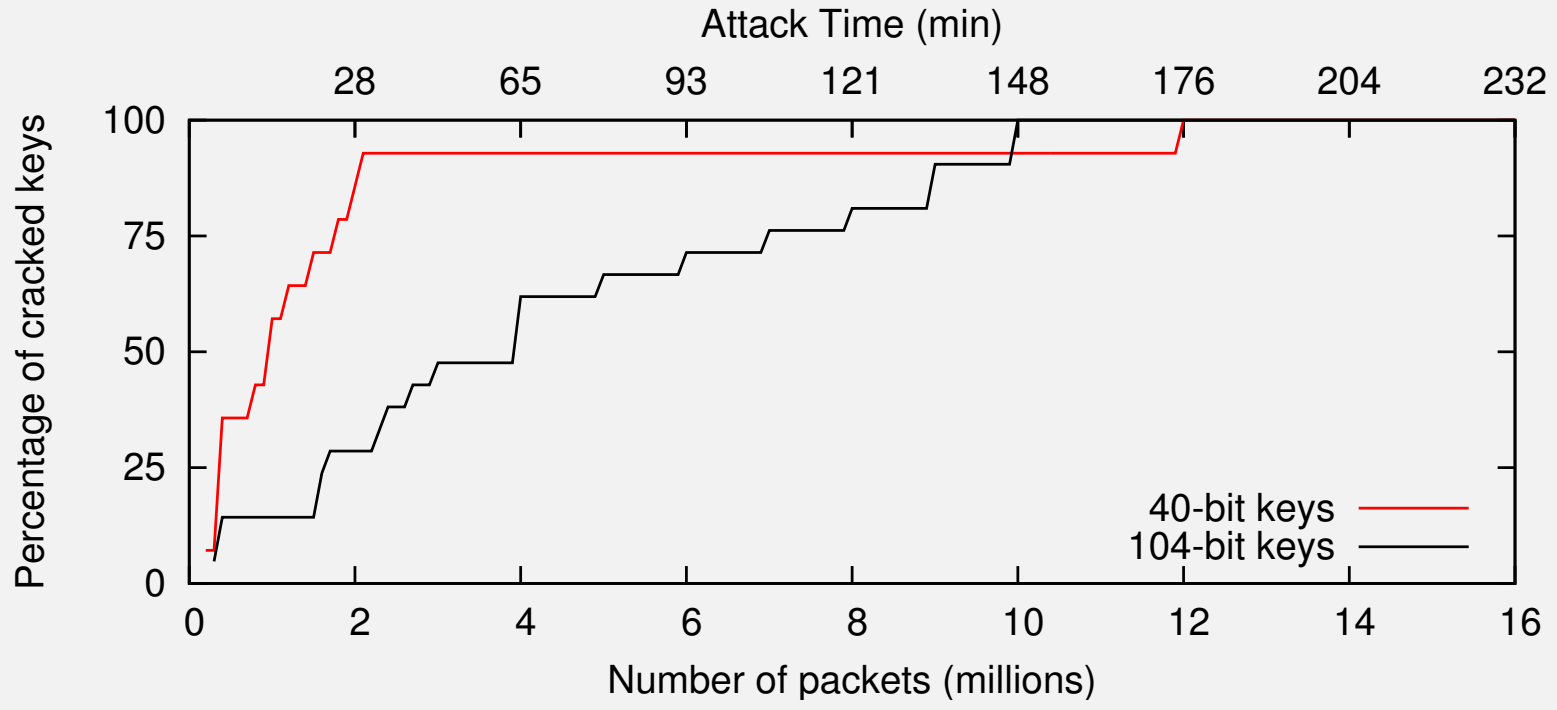


Performance

Fragmentation attack, after eavesdropping one packet:

- Recover 1500 bytes of keystream: < 2 seconds.
- Decrypt network's IP: < 30 seconds.

CDF of cracked keys via weak IV



Lessons

Fragmentation attack:

- May be performed instantly. Frequent re-keying (EAP) does not mitigate the problem. Migrate to 802.11i.
- Non-solution: ship hardware with no fragmentation support.
- Solution: ship hardware with no WEP support.

WEP history:

- Attacks evolve over time. In 2000, theoretical issues were identified. Today, we provide a practical exploit for them.
- Theoretical guidelines must be followed. Perfect example of damage incurred by keystream reuse and no authentication.

Conclusion

Do not use WEP—teach about its failures.

Future Work: The First Nail in WPA's Coffin...