KRACKing WPA2 and Mitigating Future Attacks

Mathy Vanhoef — @vanhoefm

CRYPTO Workshop on Attacks (WAC), Santa Barbara, 18 August 2018
Overview

Key reinstalls in 4-way handshake

Misconceptions

Practical impact

Channel validation
Overview

Key reinstall in 4-way handshake

Misconceptions

Practical impact

Channel validation
The 4-way handshake

Used to connect to any protected Wi-Fi network
› Provides mutual authentication
› Negotiates fresh PTK: pairwise transient key

Appeared to be secure:
› No attacks in over a decade (apart from password guessing)
› Proven that negotiated key (PTK) is secret
› Encryption protocol also proven secure
4-way handshake (simplified)

optional 802.1x authentication
4-way handshake (simplified)

optional 802.1x authentication

Msg1(r, ANonce)

Derive PTK

Msg2(r, SNonce)

Derive PTK
4-way handshake (simplified)

PTK = Combine(shared secret, ANonce, SNonce)
4-way handshake (simplified)

optional 802.1x authentication

Attack isn’t about ANonce or SNonce reuse

PTK = Combine(shared secret, ANonce, SNonce)
4-way handshake (simplified)
4-way handshake (simplified)

- Optional 802.1x authentication
- Derive PTK
- Msg1(r, ANonce)
- Msg2(r, SNonce)
- Msg3(r+1; GTK)
- Msg4(r+1)
- Install PTK & GTK
- Derive PTK
- Install PTK
4-way handshake (simplified)

optional 802.1x authentication

Msg1(r, ANonce)

Derive PTK

Msg2(r, SNonce)

Derive PTK

PTK is installed

Install PTK & GTK

Install PTK
4-way handshake (simplified)

optional 802.1x authentication

Msg1(r, ANonce)

Derive PTK

Msg2(r, SNonce)

Derive PTK

Msg3(r+1; GTK)

Install PTK & GTK

Msg4(r+1)

Install PTK

encrypted data frames can now be exchanged
Frame encryption (simplified)

PTK (session key) → Mix → Nonce (packet number) → Packet key

Plaintext data + Keystream = Encrypted data

→ Nonce reuse implies keystream reuse (in all WPA2 ciphers)
4-way handshake (simplified)

optional 802.1x authentication

Msg1(r, ANonce)

Derive PTK

Msg2(r, SNonce)

Installing PTK initializes nonce to zero

Install PTK & GTK

Install PTK

encrypted data frames can now be exchanged

Install PTK
Reinstallation Attack
Reinstallation Attack

optional 802.1x authentication
Reinstallation Attack

<table>
<thead>
<tr>
<th>Optional 802.1x authentication</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Msg1(r, ANonce)</td>
<td>Msg1(r, ANonce)</td>
</tr>
<tr>
<td>Msg2(r, SNonce)</td>
<td>Msg2(r, SNonce)</td>
</tr>
<tr>
<td>Msg3(r+1; GTK)</td>
<td>Msg3(r+1; GTK)</td>
</tr>
</tbody>
</table>
Reinstallation Attack

optional 802.1x authentication

- Msg1(r, ANonce)
- Msg2(r, SNonce)
- Msg3(r+1; GTK)
- Msg4(r+1)

Install PTK & GTK

Block Msg4

Msg1(r, ANonce)

Msg2(r, SNonce)

Msg3(r+1; GTK)

Msg4(r+1)
Reinstallation Attack

Install PTK & GTK

\[ \text{Msg4}(r+1) \]

\[ \text{Enc}_{\text{ptk}}^1 \{ \text{Msg4}(r+2) \} \]

\[ \text{Msg3}(r+2; \text{GTK}) \]

\[ \text{Msg3}(r+2; \text{GTK}) \]
Reinstallation Attack

In practice, Msg4 is sent encrypted.
Reinstallation Attack

Key reinstallation!
nonce is reset
Reinstallation Attack

Same nonce is used!
Reinstallation Attack

Install PTK & GTK

\[ \text{Enc}^1_{\text{ptk}} \{ \text{Msg4}(r+2) \} \]

Reinstall PTK & GTK

\[ \text{Enc}^1_{\text{ptk}} \{ \text{Data}(\ldots) \} \]

Keystream

\[ \text{Enc}_{\text{ptk}} \{ \text{Msg3}(r+2; \text{GTK}) \} \]

Decrypted!
Key Reinstallation Attack

Other Wi-Fi handshakes also vulnerable (CCS’17)
› Group key, FT, and PeerKey handshake

Lesser-known handshakes also vulnerable (CCS’18)
› TDLS, FILS, and WNM handshake
Overview

Key reinstalls in 4-way handshake

Misconceptions

Practical impact

Channel validation
General impact

Transmit nonce reset

Decrypt frames sent by victim

Receive replay counter reset

Replay frames towards victim
Cipher suite specific

AES-CCMP: No practical frame forging attacks

WPA-TKIP:
  › Recover Message Integrity Check key from plaintext
  › **Forge/inject** frames sent by the device under attack

GCMP (WiGig):
  › Recover GHASH authentication key from nonce reuse
  › **Forge/inject** frames in **both directions**
Handshake specific

Group key handshake:
› Client is attacked (only AP sends real broadcast frames)
› Can only replay broadcast frames to client

4-way handshake: client is attacked → replay/decrypt/forge

FT handshake (fast roaming = 802.11r):
› Access Point is attacked → replay/decrypt/forge
› No MitM required, can keep causing nonce resets
Implementation specific

iOS 10 and Windows: 4-way handshake not affected

› **Cannot decrypt unicast traffic** (nor replay/decrypt)
› But group key handshake is affected (replay broadcast)
› Note: iOS 11 does have vulnerable 4-way handshake

wpa_supplicant 2.4+

› Client used on Linux and Android 6.0+
› On retransmitted msg3 will **install all-zero key**
initial stage of 4-way handshake
Android (victim)
initial stage of 4-way handshake

Derive PTK

Msg3(r+1; GTK) ———> Msg3(r+1; GTK)

Msg4(r+1) ———> Msg4(r+1)
initial stage of 4-way handshake

Derive PTK

Msg3(r+1; GTK)  Msg3(r+1; GTK)

Msg4(r+1)  Msg4(r+1)

Install-key(PTK)

Clear PTK  Install PTK
Install-key(PTK)

Clear PTK

Msg3(r+2; GTK)

Msg4(r+2)

Install PTK

Msg3(r+2; GTK)

Enc^1_{ptk}\{Msg4(r+2)\}
Install-key(PTK)

Clear PTK

Msg3(r+2; GTK)

Msg4(r+2)

Install-key(zeros)

Install PTK

Msg3(r+2; GTK)

Enc_{ptk}^1 \{ Msg4(r+2) \}

Install all-zero PTK
Now trivial to intercept and manipulate client traffic

Install all-zero PTK
Is your device (still) affected?

github.com/vanhoofm/krackattacks-scripts

- Tests clients and APs
- Works on Kali Linux

Remember to:
- Disable hardware encryption
- Use a supported Wi-Fi dongle!
Countermeasures

Many clients won’t get updates…

AP can prevent (most) attacks on clients!
› Don’t retransmit message 3/4
› Don’t retransmit group message 1/2

However:
› Impact on reliability unclear
› Clients still vulnerable when connected to unmodified APs
Overview

Key reinstalls in 4-way handshake

Practical impact

Misconceptions

FACTS
MYTHS

Channel validation
Misconceptions I

Updating only the client or AP is sufficient
› Both **vulnerable** clients & **vulnerable** APs must apply patches

Need to be close to network and victim
› Can use special antenna from afar

No useful data is transmitted after handshake
› Trigger new handshakes during TCP connection
Misconceptions II

Obtaining channel-based MitM is hard
› Can use channel switch announcements

Using (AES-)CCMP mitigates the attack
› Still allows decryption & replay of frames

Enterprise networks (802.1x) aren’t affected
› Also use 4-way handshake & are affected

Image from “KRACK: Your Wi-Fi is no longer secure” by Kaspersky
Overview

Key reinstall in 4-way handshake

Misconceptions

Practical impact

Channel validation
Background: new attacks require MitM

Traffic Analysis
- **Capture all** encrypted frames
- **Block** certain encrypted frames

Attacking broadcast WPA-TKIP
- **Block** MIC failures
- **Modify** encrypted frames

ChopChop
Background: new attacks require MitM

Exploit implementation bugs
› Block certain handshake messages
› E.g. bugs in 4-way handshake

Other attack scenarios
› See WiSec’18 paper [VBDOP18]
› E.g. modify advertised capabilities
Observed threat model

- Attacker manipulates channel and bandwidth
- Exclude low-layer attacks (e.g. beamforming)
- Exclude relay attacks (e.g. AP and client out of range)

Want to make attacks harder, not impossible

≈ stack canaries.

Solution: verify operating channel when connecting
Verifying the current operating channel

Simple, just verify channel number element?
› Say hello to the 802.11 standard
› HT element defines optional 40 MHz bandwidth
› VHT element defines more bandwidths
› And so on …
› Non-trivial to unambiguously encode channel

→ We introduce the OCI element to encode a channel
Problem: Channel Switch Announcements (CSAs)

Unauthenticated CSAs
› Need to verify securely

Authenticated CSAs
› May not arrive → verify reception

Solution: verify CSA using SA query
Limitations

Other (partial) MitM attacks still possible:
› Adversary can act as repeater
› Physical-layer tricks (e.g. beamforming)

So why use this defense?
› Remaining attacks are harder & not always possible
› Straightforward implementation
Standardization & implementation

Part of the upcoming 802.11 standard

<table>
<thead>
<tr>
<th>March 2018</th>
<th>doc.: IEEE 802.11-17/1807r10</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>IEEE P802.11</td>
</tr>
<tr>
<td></td>
<td>Wireless LANs</td>
</tr>
<tr>
<td>Defense against multi-channel MITM attacks via Operating Channel Validation</td>
<td></td>
</tr>
</tbody>
</table>

Implementation is being pushed upstream:

github.com/vanhoefm/hostap-channel-validation
Conclusion

› Flaw is in WPA2 standard
› Proven correct but is insecure!
› Update all clients & check Aps
› New defense: channel validation
Thank you!

Questions?

krackattacks.com