Key Reinstallation Attacks

November 27, 2017

1 Outline of the Field

The paper "Key Reinstallation Attacks: Forcing Nonce Reuse in WPA2" from Mathy Vanhoef and Frank Piessens, published in Oct. 2017, reveals a new approach to violate data integrity of WPA2 implementations. For this purpose, it introduces so called key reinstallation attacks (KRACK) against several WiFi handshakes. It is therefore part of the research field of network security, especially targeting wireless and mobile networks like WLAN. In this respect, the paper enhances the set of possible attacks against WPA/WPA2 from only brute-force dictionary attacks [KTT$^+$12] [Hai10] with a completely new method.

2 Basics

When using WiFi networks, data which is sent from or to the user is usually encrypted by a security protocol. The current standard protocol is IEEE 802.11i, widely known as WPA2, where WPA stands for Wireless Protected Access. It was introduced in 2004 [KTT$^+$12] as a consequence of major security problems in the former WPA and WEP protocols. Today, it is widely adapted and is also formally proven secure [HSD$^+$05], which is why only few attempts have been made to design attacks against WPA2. Furthermore, no major weaknesses of WPA2 were found since 2004. [VP17]

When a mobile entity is connecting to a new WLAN access point via WPA2, it usually has to pass 2 stages, the authentication stage and the 4-Way-Handshake. During the authentication stage, WPA2 offers two ways of authentication, either by using a pre-shared-key like a WiFi-password or authentication against a server. [LDS09] Based on the type of authentication, both client and access point will derive a shared secret, called the `Pairwise Master Key (PMK)`.

The initialization of the PMK at the client and the access point, hereafter called `supplicant` and `authenticator` respectively, marks the beginning of the second stage. The so called 4-Way-Handshake is used to create a fresh session key between supplicant and authenticator, called `Pairwise Transient Key (PTK)`. From this PTK, both supplicant and authenticator can derive other important keys used to encrypt data frames, keys, or handshake messages. [VP17] Therefore, reinstalling the PTK will lay the foundation of the attack.

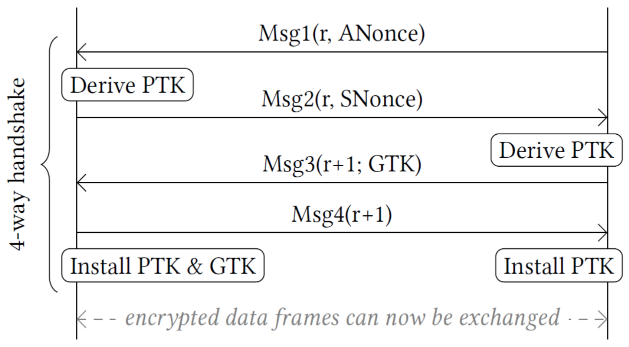

Figure 1: Message exchange during 4-Way-Handshake. [VP17, Figure 2]

Figure 1 shows the messages which are exchanged during the 4-Way-Handshake. After association, the authenticator sends the first message Msg1 to the supplicant. It contains a replay counter `r` in order to prevent replaying messages and as a reference which message the client is answering to. Furthermore, it contains the authenticator nonce `ANonce`. With this information, the supplicant is able to calculate the session key PTK out of PMK, ANonce, `SNonce` (supplicant nonce) and the two MAC addresses. The supplicant then sends its own SNonce to the authenticator in Msg2, which is subsequently also able to calculate the PTK. With Msg3, the authenticator informs the supplicant about the Group Temporal Key (GTK) which is used for multicast traffic. The supplicant acknowledges Msg3 by sending Msg4, therefore completing the handshake. After Msg4, both supplicant and authenticator install the fresh session key PTK which includes resetting its incremental transmit package number, called nonce, and its replay counter. [VP17] These resources are used from one of the data integrity protocols described at the end of this chapter to encrypt data frames that are exchanged between both parties.

Besides the 4-Way-Handshake, the protocol defines two other handshakes which can be manipulated with the later defined attack, the Group Key Handshake and the Fast BSS Transition Handshake (FT). The Group Key Handshake is used to refresh the GTK and distribute it from the access point to all connected clients. The FT handshake on the other side is used, when a client is roaming from one to another access point belonging to the same network. [VP17]

The actual encryption is done by one of the following data integrity protocols. The inital 802.11i amendment included the deprecated Temporal Key Integrity Protocol `(TKIP)` [All15] and the Advanced Encryption Standard (AES) in Counter Mode with CBC-MAC, hereafter denoted as `(AES-)CCMP`. Since 2012, a third protocol, the Galios/Counter Mode Protocol `(GCMP)`, has been added which is currently rolled out. CCMP and GCMP are recommended to use and said to be secure as long as the initialization vector under a specific encryption key is not reused [VP17], whereas TKIP is deprecated due to major weaknesses in the protocol. The attack described in the following will be able to exploit these weaknesses in order to break WPA2 protocol.

3 Summary

The basic idea of the key reinstallation attack lies within resending message 3 during the 4-way handshake. As described in [VP17, sec. 2.4] a secure communication is ensured by using a nonce as an initialization vector which is always incremented before sending a frame. Therefore a nonce is never used twice with the same key. The fact that a supplicant resets this nonce each time it receives Msg3 of the 4-way handshake is used forcing him to use the same nonce with the same encryption key. If an attacker is able to do so he can calculate the session key and decrypt the communication with the access point (AP).

First of all the attacker uses a channel-based man in the middle attack to collect the communication between a supplicant and an authenticator. [VP14] At this point the attacker is able to suppress and resend messages which is sufficient for the attack. Having a deeper look into the different implementations of the 4-way handshakes shows different weaknesses that can be exploited. The easiest way is the plaintext retransmission of message 3 which is only possible if the supplicant will always accept those unencrypted messages. A second version is possible if the supplicant will accept plaintext message 3 under some conditions. And the last one is an attack if the supplicant only accepts encrypted message 3.

The plaintext retransmission of message 3 is the most trivial one of the three versions. The first three messages of the 4-way handshake are just forwarded to the supplicant without any interfering. But message 4 which should finish the handshake is suppressed by the attack. This is why the authenticator thinks Msg3 has not arrived properly. Therefore it resends the message to make sure the supplicant is able to install the PTK. But since the client thinks he has finished the handshake it already has installed the key and starts to encrypt data with it using the first nonces. When the attacker receives Msg3 again with the same GTK he forwards it to the supplicant, who will then install the same PTK again and resets the nonce. As a result the next data sent by the supplicant will be encrypted with a already used combination of key and nonce.

But some implementations will not accept a plaintext Msg3 after they already installed a key. Therefore only the first handshake will use an unencrypted Msg3. Every following handshake uses the old PTK to encrypt all messages. But Android's implementation do accept plaintext retransmissions when they are sent instantly after the original one. In that case the NIC put both message in a processing queue directly after receiving since the NIC (network interface controller) has no PTK installed yet. Now the CPU processes the first message and tell the NIC to install the key. But since only the NIC validates if there is a key installed or not the CPU will process the second message right away. Because the NIC already has a PTK installed it will use the "old" key to encrypt Msg4. After that the CPU tells the NIC to install the "new" key which then resets the nonce.

In order to attack OpenBSD, OS X and macOS both versions will fail because they will only accept encrypted Msg3. Therefore resending a message from the first handshake will not be accepted by the supplicant. To solve this problem the third version will focus on a refresh handshake which will install a new key. Even though the whole communication is encrypted the attacker is able to identify handshake messages by its unique length and destination. As in the attack before two Msg3 are collected and sent immediately one after another. At this point the victim will also install both keys since Msg3 was encrypted. And again he encrypts the second Msg4 with the same nonce as the next data.

A different attack will force a supplicant to reinstall the GTK. As mentioned by [VP17, sec. 4.1] all tested clients resets the replay counter even when they receive an old group key. Therefore an attacker can just suppress the supplicant's response until the authenticator resends Group1 which is also saved and suppressed. Then the attacker finishes the handshake by sending Group2 and collects encrypted data. At some point later the attacker forwards the saved Group1 message to the supplicant

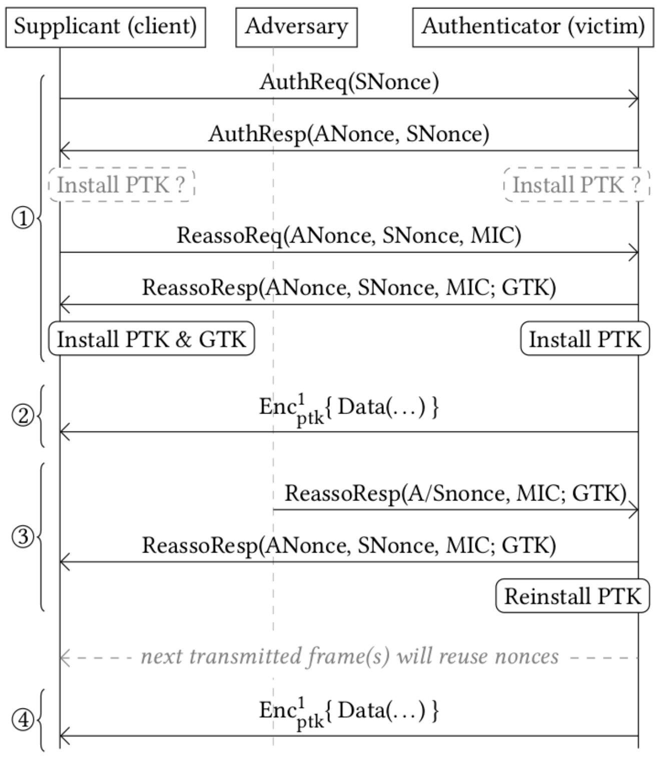

Figure 2: Key reinstallation attack against the authenticator. [VP17, Figure 9]

which then reinstalls the same key and resets the replay counter.

The attacks explained above focuses on the supplicant which is a client connecting to the network. In order to force the authenticator to reinstall the same PTK an attacker can use a similar attack against the Fast BSS Transition (FT) handshake which is shown in figure 2. The messages `AuthReq` and `AuthResp` are similar to message 1 and 2 of the 4-way handshake. Respectively `ReassoReq` and `ReassoResp` are similar in functionality to message 3 and 4. The problem in this protocol is that each time the authenticator receives a reassociation request it reinstalls the PTK. And because this frame does not include a replay counter it will always accept this message and then reuses nonces.

4 Impact

Before the publication of the paper on 16 October 2017 WiFi was considered a secure standard [VP17]. Prior to this there were no known attacks against WPA2 protected WiFi networks. Therefore there was a big reaction in the scientific community and the general media. In addition to the typical media outlets of this field many popular general media outlets like The Guardian [Her17] and Time [Ead17] reported on the vulnerability. Some Vendors whose products were vulnerable were notified on 14 July 2017. After it turned out that the vulnerability was not only present in some implementations of the standard, but was a weakness of the protocol itself, the authors reached out to CERT/CC which notified more vendors of the vulnerability on 28 August 2017 [Uni17], so that they had time to develop a patch which removes the vulnerability [Van17]. The especially critical vulnerability in Android was fixed in the Android Distribution LineageOS on 16 October 2007 [Lin17] and with a security patch in Android itself on 6 November 2017 [Inc17].

The paper influenced its field in the way that it introduced a novel attack technique which exposed a vulnerability in WPA2 protected WiFi networks, that also could be used to attack other protocols in a similar way.

At the moment there is no know published scientific research based on the findings of this paper, but it has to be expected that other protocols are already being researched for similar weaknesses.

Although the technique can be used for attacks with devastating results, there are no known cases where the vulnerability has been exploited. Furthermore are most of the attacks in practice relatively difficult to perform [Van17]. In addition, most online communication is SSL secured and therefore even in case of a successful attack on the WiFi network protected.

References

[All15] WiFi Alliance. Technical note removal of tkip from wi-fi devices. Technical report, 2015.

[Ead17] Lisa Eadicicco. Everything with wi-fi has a newly discovered security flaw. here's how to protect yourself, 2017. `http://time.com/4983720/krack-attack-wpa2-wifi/`; (26/11/2017).

[Hai10] B Haines. 802.11 wireless–infrastructure attacks,". *Seven Deadliest Wireless Technology Attacks, T. Kramer, Ed. New York: Elsevier*, pages 01–05, 2010.

[Her17] Alex Hern. 'all wifi networks' are vulnerable to hacking, security expert discovers, 2017. `https://www.theguardian.com/technology/2017/oct/16/wpa2-wifi-security-vulnerable-hacking` (26/11/2017).

[HSD$^+$05] Changhua He, Mukund Sundararajan, Anupam Datta, Ante Derek, and John C Mitchell. A modular correctness proof of ieee 802.11 i and tls. In *Proceedings of the 12th ACM conference on Computer and communications security*, pages 2–15. ACM, 2005.

[Inc17] Google Inc. Android security bulletin – november 2017, 2017. `https://source.android.com/security/bulletin/2017-11-01`; (26/11/2017).

[KTT+12] Vishal Kumkar, Akhil Tiwari, Pawan Tiwari, Ashish Gupta, and Seema Shrawne. Vulnerabilities of wireless security protocols (wep and wpa2). *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)*, 1(2):pp–34, 2012.

[LDS09] Arash Habibi Lashkari, Mir Mohammad Seyed Danesh, and Behrang Samadi. A survey on wireless security protocols (wep, wpa and wpa2/802.11 i). In *Computer Science and Information Technology, 2009. ICCSIT 2009. 2nd IEEE International Conference on*, pages 48–52. IEEE, 2009.

[Lin17] LineageOS. Lineageos auf twitter, 2017. `https://twitter.com/LineageAndroid/status/920143977256382464`; (26/11/2017).

[Uni17] Carnegie Mellon University. Vendor information for vu#228519, 2017. `https://www.kb.cert.org/vuls/byvendor?searchview&Query=FIELD+Reference=228519&SearchOrder=4`; (26/11/2017).

[Van17] Mathy Vanhoef. Key reinstallation attacks, 2017. `https://www.krackattacks.com`; (26/11/2017).

[VP14] Mathy Vanhoef and Frank Piessens. Advanced wi-fi attacks using commodity hardware. In *Proceedings of the 30th Annual Computer Security Applications Conference*, pages 256–265. ACM, 2014.

[VP17] Mathy Vanhoef and Frank Piessens. Key reinstallation attacks: Forcing nonce reuse in WPA2. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 1313–1328. ACM Press, November 2017.