Security Analysis of the WPA2 KRACK patches

Study Case

Graziano Marallo

KU Leuven

13 December 2018

0 Outline

KU LEUVEN

0 Introduction

- Most of the Wi-Fi networks which are used today are protected and secured by the WPA2
- Used everywhere
- Can we trust WPA2?
- Is really secure as we thought?

1 Outline

1. 4-Way Handshake Protocol
2. KRACK
3. Fuzzing
4. Goal & Findings

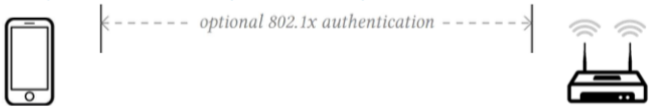
1 4-Way Handshake Protocol [1]

- Provides mutual authentication between client and server
- Negotiates a fresh PTK, proven to be secret
- The protocol itself proven to be secure

1 Functioning

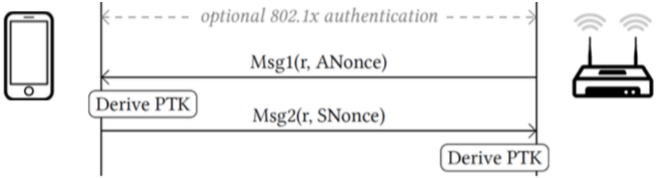
- It is composed of 5 different stages:
  - Stage 1: *Network Discovery*
  - Stage 2: *Authentication and Association*
  - Stage 3: *802.1x Authentication*
  - Stage 4: *4-Way Handshake*
  - Stage 5: *Group Key Handshake*
- Stage 4 is composed of 4 messages exchanged between supplicant and authenticator
- During this exchange the actual protocol is performed

1 Message exchange: Msg 1



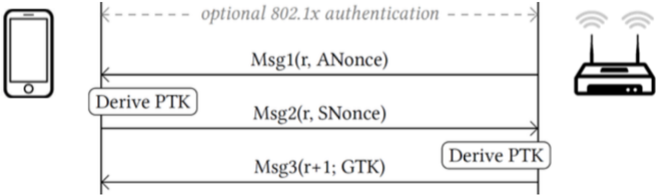
- Sent by the authenticator
- Contains a randomly generated ANonce
- No protection by MIC
- Possible message forging

1 Message exchange: Msg 2



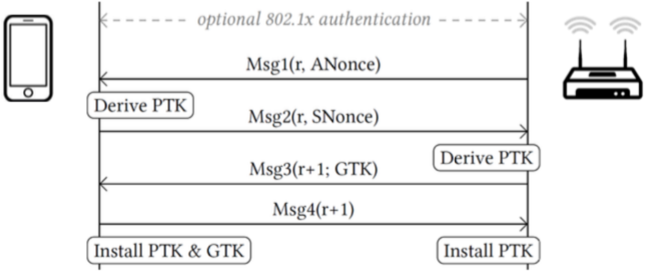
- Sent by the supplicant
- Contains the random SNonce of supplicant
- Protection by MIC
- Computes PTK

1 Message exchange: Msg 3



- Sent from the authenticator in response to the supplicant
- Contains again ANonce
- RSNE is checked with the one received when the protocol takes place for the first time

1 Message exchange: Msg 4



- Last message sent by the supplicant in order to inform the authenticator that the handshake has been completed
- Protection by MIC
- Encrypted data frame can be transmitted

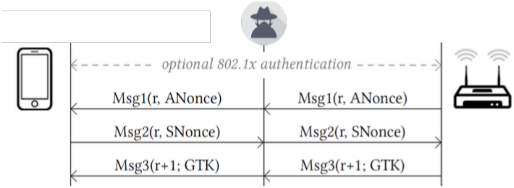
2 Outline

2 KRACK [2]

- The supplicant still accepts retransmissions of message 3
- Possibility to force the reinstallation of th PTK

2 Scenario

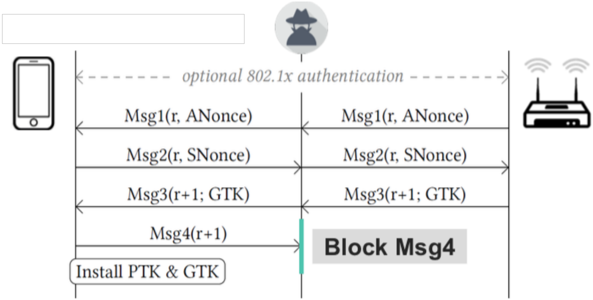
If we consider the victim still accepting retransmission of message 3 after the session key has been installed, the attack is pretty straightforward



In the first stage:

- The attacker manage to set up a channel-based MitM attack
- Able to sniff traffic and manipulate it

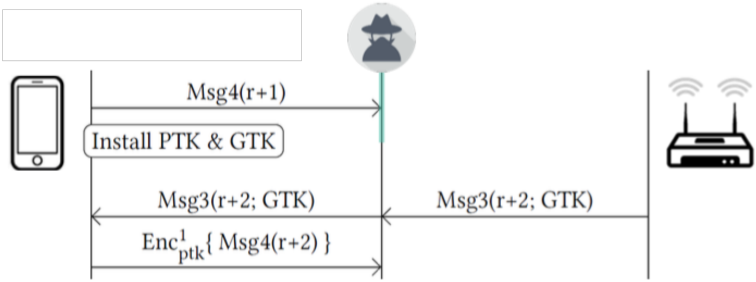
2 Scenario



In the second stage:

- The attacker can prevent message 4 from arriving to the authenticator
- The supplicant will install PTK and GTK as soon as message 4 is sent

2 Scenario



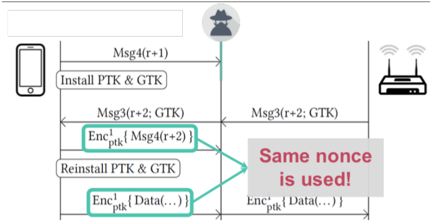
In the third stage:

- Authenticator will resend message 3
- Victim is inducted to reinstall
- Both replay counter and nonce are reset

2 Scenario

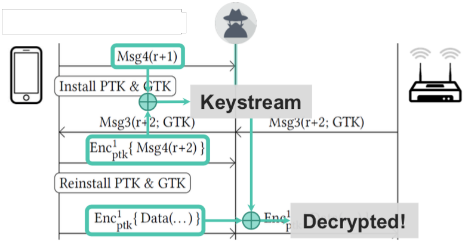
In the fourth stage:

- Supplicant will send an encrypted message since the key has been reinstalled
- Authenticator will accept an old unencrypted message 4 which has a replay counter r + 1
- PTK will be installed and the AP will start sending encrypted unicast data frames to the client



KU LEUVEN

2 Scenario



In the end:

- When the victim retransmit its next data frame, the data-confidentiality protocol will reuse nonces
- The attacker can manages both the forwarding time between messages and amount of nonces
- The client could be de-authenticated by the attacker himself

3 Outline

1. 4-Way Handshake Protocol
2. KRACK
3. Fuzzing
4. Goal & Findings

3 Fuzzing [3]

Various technique can be used to fight hackers' attacks like static analysis, dynamic, symbolic execution and fuzzing. Focus our attention on the latter:

- Fuzzing requires less knowledge of the target,
- Easily adapted and scaled to a large variety of situation and problem.
- The most popular vulnerability discovery solution nowadays

3 Fuzzing process

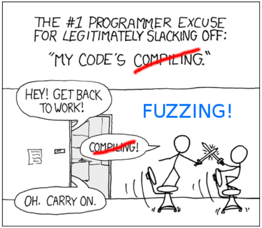


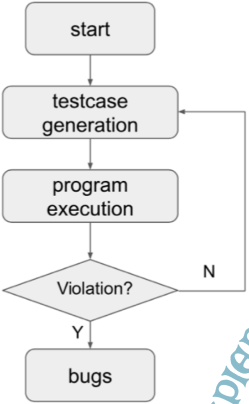
Figure: Fuzzing [4]

- Starts with generating massive normal and abnormal inputs towards application
- Try to detect exception by feeding generated inputs to the target applications
- Monitor each execution states

3 Fuzzing

The main process of the traditional fuzzing process has 4 main stages

- Test case generation stage
- Test case running stage
- Program execution stage monitoring
- Analysis of exceptions



3 American Fuzzy Lop

- **Brute-force Fuzzer**
- Coupled with an exceedingly simple but rock-solid instrumentation-guided genetic algorithm
- Uses a modified form of edge coverage

3 Algorithm

How it works:

1. Load user-supplied initial test cases into the queue,
2. Take next input file from the queue,
3. Attempt to trim the test case to the smallest size that doesn't alter the measured behaviour of the program,
4. Repeatedly mutate the file using a balanced and well-researched variety of traditional fuzzing strategies,
5. If any of the generated mutations resulted in a new state transition recorded by the instrumentation, add mutated output as a new entry in the queue.
6. Go to 2.

4 Outline

4 Goal

Goal of the research:

- Identify the source of the problem
- Perform several analysis
- Debug the source code
- Analyse with a systematic approach the protocol
- Try to solve or mitigate the problem
- Propose a possible solution

4 Ideas

- Analyse source code of IWD (open source implementation of the 4WH)
- Find a possible spot where to inject dummy input
- Create an harness code
- Start fuzzing

4 Findings

- TODO

Thanks for the attention!

4 Bibliography

- Vanhoef, M and Schepers, D and Piessens, F, "Discovering logical vulnerabilities in the Wi-Fi handshake using model-based testing", ASIA CCS 2017 - Proceedings of the 2017 ACM Asia Conference on Computer and Communications Security, 2017, pp. 360-371, DOI 110.1145/3052973
- OPCDE, Dubai, 7 April 2018, Vanhoef https://papers.mathyvanhoef.com/opcde2018-slides.pdf
- Li, Jun and Zhao, Bodong and Zhang, Chao, "Fuzzing: a survey", Cybersecurity, Vol. 1, 2018, pp. 1-13, DOI 10.1186/s42400-018-0002-y
- Fuzzing image https://medium.com/@dieswaytoofast/fuzzing-and-deep-learning-5aae84c20303