April 2022

METHOD TO ADDRESS SECURITY VULNERABILITIES WITH RESPECT TO OFFLINE AND ONLINE DICTIONARY ATTACKS ON WPA2-PSK

Niranjan M M

Vijay Kothamasu

Nagaraj Kenchaiah

METHOD TO ADDRESS SECURITY VULNERABILITIES WITH RESPECT TO OFFLINE AND ONLINE DICTIONARY ATTACKS ON WPA2-PSK

AUTHORS:
Niranjan M M
Vijay Kothamasu
Nagaraj Kenchaiah

ABSTRACT

The techniques presented herein is to enhance the security of the WPA2-PSK methods to combat both Offline and Online Dictionary Attacks by generating independent random keying parameters on both Supplicant and Authenticator, which are not exchanged explicitly or in any form between them. These parameters are used in conjunction with "Password Key Element" which is generated from PSK using known transformation. This would overcome the offline dictionary attacks faced by current WPA2-PSK method. Also, using "Cookie Loop", where-in Cookie would be initially generated by Authentication Server (AAA Server) and later passed in encrypted form in all the transactions (M1-M4) between Authenticator (WLC) and Supplicant (Client), and also in Access-Request & Access-Accept messages. This would overcome the online dictionary attacks.

DETAILED DESCRIPTION

With the pandemic situation for years together the employee work environment is like never before. Work From Home (WFH) is still in-place for many organisations while hybrid work environment is other option in some countries. When it comes to WFH and hybrid models, Wi-Fi plays crucial for the proper functioning and getting the right connectivity to office network. And most used security policy for these home and small business deployments is WPA2-PSK. Also, in some of the deployments, onboarding of IoT devices uses WPA2-PSK due to unavailability of flash to store certificates required for EAP-TLS etc.,

WPA2-PSK uses Pre-Shared-Key (PSK) for the client authentication and getting the Wi-Fi connectivity. But WPA2-PSK can still be exploited, and it can be cracked with Dictionary attacks. Lot of efforts went in this area to strengthen its security with mechanisms like iPSK, mPSK and easyPSK, which are used in conjunction with WPA/WPA2/WPA3 to establish secure connection. Initially, defaultPSK was being used for all the devices, where-in if one device is compromised, then whole network of devices would get compromised. Later iPSK introduced, which provides unique identity PSK for each device, where-in administrator needs to maintain iPSK DB containing unique PSK for all the devices in the network. Also, vendors are provided with mPSK, where-in devices can use any one of the PSKs configured for the SSID (on the AAA server). It reduced the burden on the administrator for maintaining huge iPSK DB. Similarly, EasyPSK implemented, where-in administrator can give any one of the PSK configured on the AAA server (without having one-to-one mapping for the devices), but it is implemented using brute-force method on the AAA server to determine which PSK is used by the client. While these methods improve security aspect, they still have their own challenges and drawbacks such as Offline Dictionary Attacks and Online Dictionary Attacks. All existing WPA2-PSK methods are vulnerable to Offline Dictionary Attack. To overcome this WPA3 introduced which solves the Offline Dictionary Attacks, by using Intermediate Keying Material, but still, it is vulnerable to Online Dictionary Attacks.

The techniques presented herein is to enhance the security of the WPA2-PSK methods to combat both Offline and Online Dictionary Attacks by,

- Generating independent random keying parameters on both Supplicant and Authenticator, which are not exchanged explicitly or in any form between them. These parameters are used in conjunction with "Password Key Element" which is generated from PSK using known transformation. This would overcome the offline dictionary attacks faced by current WPA2-PSK method.
- Using "Cookie Loop", where-in Cookie would be initially generated by Authentication Server (AAA Server) and later passed in encrypted form in all the transactions (M1-M4) between Authenticator (WLC) and Supplicant (Client), and in Access-Request & Access-Accept messages. This would overcome the online dictionary attacks.

As per this method, Password Key Element (called PQPSK from now) is generated from PSK using negotiated P and Q, as PQPSK = Function (P, Q, PSK). Further, the generated PQPSK is used in the message exchange flows instead of PSK. Two independent random keying parameters X1 (on Supplicant) and X2 (on Authenticator) are generated and used in conjunction with PQPSK to exchange messages between Supplicant and Authenticator as below:

- Product of X1 and PQPSK (i.e., X1 * PQPSK) (in EAPOL-Start message from Supplicant (Client) to Authenticator (WLC)).
- Product of X2 and PQPSK (i.e., X2 * PQPSK) (in EAPOL-Key Message M1 from Authenticator (WLC) to Supplicant (Client)).

After exchanging these messages, Supplicant and Authenticator, independently generates X = X1 * X2 * PQPSK. Further, Supplicant and Authenticator uses existing known functions PBKDF2 and KBKDF along with newly introduced known function which takes X as input to generate PTK = {KEK, KCK, TK}. Note here, X1 and X2 are never shared between Authenticator (WLC) and Supplicant (Client) in any manner. Hence it overcome Offline Dictionary Attack encountered by the existing WPA2-PSK flow, where-in ANonce, SNonce and MIC are exchanged in M1 and M2, attacker (e.g., MITM attack) can use these to deduce PSK.

Further EAPOL-Key Message M2 and M3 exchanged between Supplicant (Client) and Authenticator (WLC) contains MIC generated using KCK as well as Encrypted Cookie using KEK. The Cookie generated by the Authentication Server is passed across Authenticator and Supplicant in encrypted form and later validation back in Authentication Server. This will ensure closed loop behaviour to overcome the Online Dictionary Attacks.

Figure-1 depicts the detailed flow of the proposed method to combat Offline and Online Dictionary Attacks on WPA2-PSK.

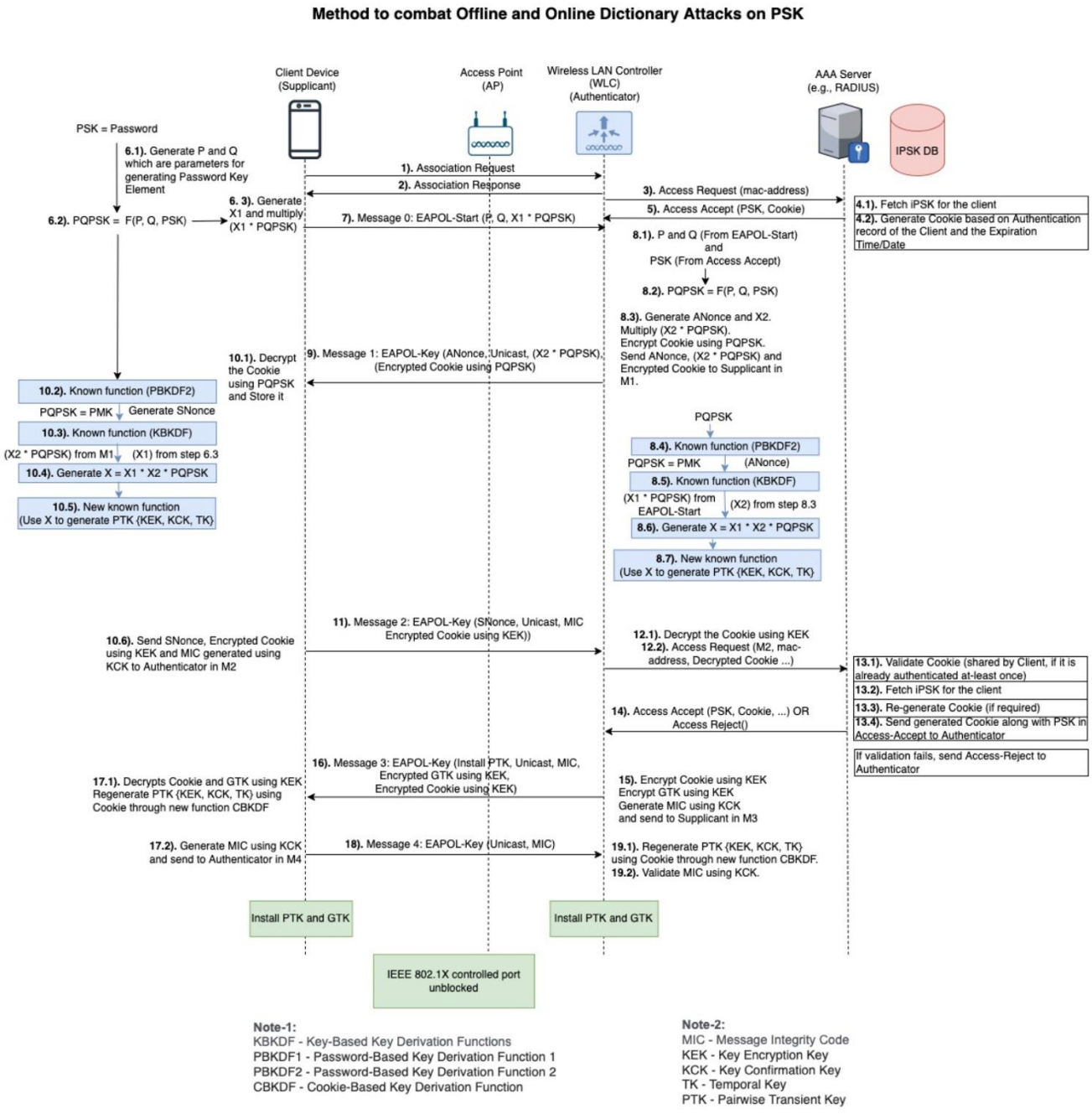

Figure-1

The techniques presented herein is explained end-to-end as below in steps.

- AAA Server (e.g., RADIUS) would have configured to support PSK (viz, iPSK, defaultPSK).
- SSID on the WLC is configured with WPA2-PSK and pushed to the AP. For simplicity, considered "Central Authentication" scenario, where-in WLC does the complete client authentication flows. Another scenario would be "Local Authentication", where-in AP does the complete client authentication flows. This method holds good for both the scenarios.
- Client would be configured to connect to this SSID with the WPA2-PSK (along with configuring iPSK as provided by the administrator).
- Client tries to associate to this SSID by sending association request. Upon receiving association request, WLC sends the Access-Request along with MAC address to the AAA server. If MAC address is legitimate and MAC address to iPSK mapping found, AAA server would respond with Access-Accept having iPSK as well as uniquely generated Cookie (having authenticated record of the client as well expiration time/date). This Cookie is passed in encrypted form within all transactions (M1-M4) between Authenticator and Supplicant. Otherwise, AAA Server respond with Access-Reject.
- As part of authentication, Authenticator (WLC) and Supplicant (Client) exchanges EAPOL messages (Starting from EAPOL-Start to M1-M4) as below:
    - On Supplicant: Supplicant generates parameters P and Q, which are used in conjunction with configured PSK to derive password key element which is further used for computing keys. For deriving password key element, known function is used (e.g., Password Key Element = PQPSK = Function (P, Q, PSK)).
    - Handling of EAPOL-Start and EAPOL-Key Message M1:
        - On Supplicant (Client):
            - Supplicant (Client) generates X1, a random keying parameter, which would be multiplied by the PQPSK (i.e., X1 * PQPSK) and sent in EAPOL-Start message to Authenticator (WLC). EAPOL-Start would also carry supplicant generated P and Q parameters.
            - Note here, X1 is never shared between Supplicant and Authenticator in any form.

- On Authenticator (WLC):
    - Upon receiving EAPOL-Start, Authenticator (WLC) derive Password Key Element using parameters P and Q shared by the Supplicant. For deriving Password Key Element, known function is used (e.g., Password Key Element = PQPSK = Function (P, Q, PSK). Authenticator (WLC) generates ANonce as well as X2, a random keying parameter.
    - X2 would be multiplied by the PQPSK (i.e., X2 * PQPSK) and send to the Client as part of EAPOL-Key message M1 along with ANonce. The message M1 also carries Cookie received earlier from the AAA server. This Cookie is encrypted using PQPSK generated above.
    - Further, Authenticator generates PTK as below:
        - PQPSK generated as above using P, Q and PSK will be fed as input to the existing Known Function "Password-Based Key Derivation Function 2 (PBKDF2)"
        - The output is passed through another Known Function "Key-Based Key Derivation Function (KBKDF)" with input parameters as ANonce and PQPSK.
        - Using product (X1 * PQPSK) received from the Supplicant in EAPOL-Start and the X2 generated locally on the Authenticator are used to generate X = (X1 * PQPSK) * X2 = X1 * X2 * PQPSK.
        - This X is used as input to the New Known Function to generate PTK (i.e., PTK = {KEK, KCK, TK})
    - Note here, X2 is never shared between Authenticator and Supplicant in any form.
- On Supplicant (Client):
    - Upon receiving EAPOL-Key Message M1, which carries product (X2 * PQPSK) and encrypted Cookie.

- The encrypted Cookie will be decrypted using PQPSK generated earlier. The Cookie will be stored on the Supplicant for later use.
- Further, Supplicant generates PTK as below:
    - PQPSK generated earlier will be fed as input to the existing Known Function "Password-Based Key Derivation Function 2 (PBKDF2)".
    - The output is passed through another Known Function "Key-Based Key Derivation Function (KBKDF)" with input parameters as SNonce and PQPSK.
    - Using product (X2 * PQPSK) received from the Authenticator in EAPOL-Key Message M1 and the X1 generated locally on the Supplicant are used to generate X = X1 * (X2 * PQPSK) = X1 * X2 * PQPSK.
    - This X is used as input to the New Known Function to generate PTK (i.e., PTK = {KEK, KCK, TK}).

- Handling of EAPOL-Key Message M2, Access-Request, Access-Accept/Access-Reject:
    - On Supplicant (Client):
        - Supplicant sends SNonce, MIC generated using KCK and Encrypted Cookie using KEK to the Authenticator as part of EAPOL-Key Message M2.
    - On Authenticator (WLC):
        - Upon receiving M2 containing Encrypted Cookie, it decrypts the Cookie using KEK. Authenticator sends the Access-Request containing M2 information and Cookie to the Authentication Server (AAA Server/RADIUS).
    - On Authentication Server (AAA Server):
        - Upon receiving Access-Request, it validates the Cookie against the one it has sent earlier. If validation is success, it fetches the iPSK for this client. For the new client onboarding scenario (first-time

association), Authentication Server generate Cookie based on client authentication record and expiration time/date.

- Cookie will be having expiration date and if expired, new Cookie will be generated.
- Both iPSK and Cookie will be sent to Authenticator as part of Access-Accept. If any of the validation fails, it sends Access-Reject.

- Handling of EAPOL-Key Message M3:
  - On Authenticator (WLC):
    - Upon receiving Access-Accept, Authenticator Encrypts the Cookie using KEK.
    - It generates MIC using KCK.
    - Authenticator sends Encrypted GTK, Encrypted Cookie as well as MIC as part of EAPOL-Key Message M3 to the Supplicant.
  - On Supplicant (Client):
    - Upon receiving M3 with Encrypted Cookie, it decrypts Cookie using KEK. Similarly decrypts the GTK using KEK.
    - Supplicant regenerate PTK (KEK, KCK, TK) using Cookie through new Known Function "Cookie-Based Key Derivation Function (CBKDF)".
    - Supplicant installs PTK and GTK.
- Handling of EAPOL-Key Message M4:
  - On Supplicant (Client):
    - Supplicant sends Message Integrity Code (MIC) generated using newly generated KCK to the Authenticator as part of EAPOL-Key Message M4.
  - On Authenticator (WLC):
    - Authenticator regenerate PTK (KEK, KCK, TK) using Cookie through new Known Function "Cookie-Based Key Derivation Function (CBKDF)".
    - Authenticator validates the MIC using newly generated KCK.
    - Authenticator installs PTK and GTK.

The techniques presented herein propose method wherein the actual key used for data encryption is never sent or shared in any form in the handshake. This will help to combat Offline Dictionary Attacks. If we closely look at the existing mPSK and easyPSK implementation, they solve the weakness of PSK only to an extent but causes the additional burden on AAA server to verify the hashed string in the brute-force manner, in fact, existing mPSK and easyPSK increases the chances of Offline Dictionary Attacks. Moreover, the "Cookie Loop" between Authentication Server, Authenticator and Supplicant solves the Online Dictionary Attack. Cookie always sent in encrypted form between Authenticator and Supplicant, hence through Cookie validation, we are validating the messages exchanged. If validation failed, then Access-Reject is triggered. Additionally, in this proposed method, no additional configuration needed for this mechanism and keeps it simple just as PSK but much more secure.