Article

Stateless Reassociation in WPA3 Using Paired Token

Byoungcheon Lee [1,†,‡]

[1] Department of Information Security, Joongbu University, 305 Dongheon-ro, Goyang-si, 10279 Korea; sultan@joongbu.ac.kr

Abstract: In WPA3 secure connection is executed in two sequential stages. Firstly, in authentication and association stage a pairwise master key (PMK) is generated. Secondly, in post-association stage a pairwise transient key (PTK) is generated from PMK using the traditional 4-way handshake protocol. To reduce the heavy computation of the first stage PMK caching can be used. If client and AP are previously authenticated and has PMK cache, client can skip the first heavy stage and reuse the cached PMK to directly execute the 4-way handshake. But PMK caching is a very primitive technology to manage shared key between client and AP and there are many limitations; AP has to manage stateful cache for multiple clients, cache lifetime is limited, etc. Paired token (PT) [14] is a new secondary credential scheme that provides stateless pre-shared key (PSK) in client-server environment. Server issues paired token (public token and secret token) to authenticated client where public token has the role of signed identity and secret token is a kind of shared secret. Once client is equipped with PT, it can be used for many symmetric key based cryptographic applications such as authentication, authorization, key establishment, etc. In this paper we apply the PT approach to WPA3 and try to replace the PMK caching with the one-time authenticated key establishment using PT. At the end of the authentication and association stage AP securely issues PT to client. Then in reassociation stage client and AP can compute the same one-time authenticated PMK from PT in stateless way and compute PTK using the traditional 4-way handshake protocol. Using this kind of stateless reassociation technology AP can provide high performance service to huge number of clients.

Keywords: Wi-Fi; WPA3; PMK caching; Stateless reassociation; Paired token; Secondary credential; JSON web token; One-time authenticated key establishment

1. Introduction

There have been many criticism on the limitations of WPA2-PSK [9]. The password can be cracked offline. If the password is known to attackers, they can sniff or spoof the user. Anyone can try to disconnect other's connections easily. There is no security service in open connection.

WPA3 [1] released by Wi-Fi Alliance in 2018 provides several security improvements over WPA2. Open connection which does not use password protection is also protected with the opportunistic wireless encryption (OWE). Password in personal mode is protected from offline crack with simultaneous authentication of equals (SAE). Using the device provisioning protocol (DPP) it provides easy connectivity to devices which do not have display. It provides improved security using 192-bit security suite.

In WPA3 secure handshake is executed in two sequential stages. The first stage is authentication and association which results to share pairwise master key (PMK) between client and AP. PMK is generated from OWE in enhanced open connection and from SAE in personal mode. In enterprise mode authentication server checks the authenticity of client using various extensible authentication protocols (EAP), and then generates PMK and distributes it to both client and AP in secure communication.

The second stage is to generate pairwise transient key (PTK) from PMK using the traditional 4-way handshake protocol. The first stage is heavy in computation and communication with the Diffie-Hellman key exchange in personal mode or EAP in enterprise mode, while the second stage is reasonably efficient. In enterprise mode full authentication sometimes takes more than 1 second and it is critical for time sensitive applications like voice. Thus reducing the latency of the first full authentication and association stage is a very practical requirement for better performance.

PMK caching has been used as a fast roaming technology. If client and AP are previously authenticated and have PMK cache, they can skip the first heavy stage and reuse the cached PMK to directly execute the 4-way handshake. If PMK caching is enabled, client and AP keep the previous PMK and PMKID in cache. In subsequent connection request client can request reassociation by presenting a valid PMKID, and then AP finds the corresponding PMK in cache. If it is successful, heavy authentication of the first stage is skipped and only the 4-way handshake using the cached PMK is executed. In WPA2-Personal PMK caching has no advantage in performance, since PMK is computed from shared passphrase with simple hash computation. But in WPA3-Personal PMK is computed from the SAE (Diffie-Hellman key exchange) that PMK caching can enhance the performance a lot. But PMK caching is a very primitive technology to manage shared key between client and AP and there are many limitations; AP has to manage cache, connection process requires stateful service in AP, available cache lifetime is limited, the number of serviced clients will be limited, etc.

Paired token (PT) is a new secondary credential scheme that provides stateless pre-shared key more efficiently in client-server environment, specially manage it in stateless way in server side [11–14]. Assume that there is an independent authentication system between client and server using some primary credential. Server authenticates the client using the primary credential and then issues PT (public token and secret token) to authenticated client as a secondary credential. Public token has the role of signed identity that represents the authenticated state of the client. Secret token is a kind of shared secret between client and server with a special property that server can compute secret token anytime from given public token, thus server does not need to save client tokens issued by itself. This feature provides the stateless property in server side. PT can be applied to many symmetric key based cryptographic applications such as authentication, authorization, secure communications, etc.

In this paper we apply the PT approach to WPA3 and try to replace the PMK caching to one-time authenticated key establishment using PT. If a full authentication and association is finished successfully, AP issues PT to client securely and client saves it. In subsequent connection request client can request quick reassociation using PT. In this stage client and AP can compute the same one-time authenticated PMK from PT and use it to compute PTK in the following 4-way handshake protocol. The proposed reassociation protocol using PT has the following advantages.

1. Reassociation request by client provides one-time authentication of client.
2. AP can compute one-time authenticated PMK in stateless way.
3. Same PT can be used for quick reassociation multiple times for the lifetime of PT.

Once client is equipped with PT, reassociation process is the same in heterogeneous authentication scenarios such as enhanced open connection, personal authentication, and enterprise authentication.

This paper is organized as follows. Section 2 reviews WPA2, WPA3 and paired token. Section 3 presents the proposed key establishment protocol applied to WPA3. Section 4 provide security and performance analysis. Finally section 5 concludes the paper.

2. Related works

2.1. WPA2

Wi-Fi security protocols are started from WEP in 1997 and evolved into WPA in 2003, WPA2 in 2004, and WPA3 in 2018 [10]. There are two modes of authentication in WPA. WPA-Personal, or referred to as WPA-PSK (pre-shared key) mode, is designed for home and small office networks with single access point (AP). It's security depends on the pre-shared key between client and AP and doesn't require an authentication server. WPA-Enterprise, referred to as WPA-802.1X mode, is designed for enterprise networks with multiple APs. It requires a RADIUS authentication server and various kinds of extensible authentication protocols (EAP) are used for authentication.

In WPA2-Personal mode client and AP share a static password PSK. A pairwise master key (PMK) is computed from PSK

$$PMK = PBKDF2(HmacSha1, PSK, SSID) \tag{1}$$

and then 4-way handshake is followed. In WPA2-Enterprise mode client is authenticated by a RADIUS server with various extensible authentication protocol (EAP) and then authentication server generates PMK and distributes it securely to client and AP. After that 4-way handshake is followed between client and AP.

The core component used in Wi-Fi security protocol is the 4-way handshake protocol. Client and AP are sharing the same static PMK, but it is not recommended to use the static PMK for encryption of communications. The traditional 4-way handshake protocol is used to establish dynamic session key called pairwise transient key (PTK) from PMK. Using this protocol client and AP proves the possession of same PMK each other without exposing PMK over the communication channel, and then computes PTK from PMK and other information. Client and AP exchanges AP nonce (AN) and STA nonce (SN) and the PTK is computed from the attributes PMK, AN, SN, AM(AP MAC address), and SM (STA MAC address).

$$PTK = PRF(PMK, AN, SN, AM, SM). \tag{2}$$

The handshake also yields the group temporal key (GTK) which is used to decrypt multicast and broadcast traffic.

There have been many criticisms on the security of WPA2-PSK. The static shared password PSK can be cracked offline, thus using strong password is highly recommended. Although a strong password is used, there are so many misuse cases in the real world that password is shared to public. For example, Wi-Fi password is announce to public in cafe, restaurant, etc. If the password is known to attackers, they can sniff or spoof the communications of other users easily. Management frames are not protected that attackers can disconnect other's connections with de-authentication attack. In public Wi-Fi services using WPA2-Open there is no security services such as password protection and encrypted communications.

2.2. *WPA3*

WPA3 released in 2018 [1] was designed to strengthen security in Wi-Fi networks. It provides several security improvements over WPA2.

1. WPA2 open connection uses plaintext communication with no password protection. To strengthen user privacy even in open connection WPA3 provides individualized encryption using opportunistic wireless encryption (OWE, RFC 8110) [2]. In OWE client and AP executes unauthenticated Diffie-Hellman key exchange to create one-time PMK and then 4-way handshake is followed to derive PTK from the PMK.
2. Ordinary home networks in personal mode use simultaneous authentication of equals (SAE) [3,4] in password authentication. This handshake is resistant against offline dictionary attacks and the resulting PMK is changing dynamically depending on Diffie-Hellman key exchange in SAE. The PMK is then used in 4-way handshake to generate PTK.
3. Using new device provisioning protocol (DPP) WPA3 provides easy connectivity of devices that do not have display. It provides a simple and secure way to add these devices to a Wi-Fi network using QR codes. It provides concrete mutual authentication using public key cryptography and easy configuration of security.
4. WPA3 has improved security using 192-bit security suites.

5. Using protected management frame client and AP exchange management frames in encrypted form, which can prevent attacker's misbehavior.

2.3. *PMK caching for fast roaming*

Full handshake in WPA3 are heavy in performance. OWE in WPA3-Open requires unauthenticated DH key exchange. In WPA3-Personal SAE requires not only DH key exchange but also computation of password element (PE) from password which uses expensive hunting-and-pecking technique [4]. In WPA3-Enterprise extensive authentication protocol (EAP) with RADIUS server takes some time with interaction with remote RADIUS server. If it should be repeated in every connection requests, it will be very time consuming both for client and for AP. To reduce the latency of full handshake quick reassociation technologies have been introduced.

PMK caching is a quick reassociation technology that client and AP reuse the previously shared PMK in next connection requests. It has been mainly used in WPA2-Enterprise networks as a fast roaming technology, since full authentication by the central RADIUS server using EAP is heavy in performance, sometimes takes several seconds. If client and AP are previously authenticated and has PMK cache, client can skip the heavy full handshake and reuse the cached PMK to directly execute the 4-way handshake. If PMK caching is enabled, client and AP keep the previous PMK and PMKID in cache. PMK is computed as (1) and PMKID is a HMAC value computed from PMK as follows.

$$PMK_{ID} = H(PMK, PMK_{Name}|AM|SM) \tag{3}$$

PMK_{ID} is used as an index to identify PMK. If client requests connection using PMKID in subsequent request and AP finds corresponding PMK in cache, the full authentication is skipped and the cached PMK is reused. Thus client can immediately execute the 4-way handshake process ensuring a minimal latency.

Opportunistic key caching (OKC) is an extended version of PMK caching in roaming scenario in multiple AP enterprise environment. Once a client completes the full handshake with an AP, the PMK is synchronized automatically among all the APs on the network. Now if the client roams to any other AP in the same network, that AP would also have the PMK and the expensive EAP can be skipped, making the roam a lot faster.

If PMK caching is applied to WPA2-Personal, it has no performance gain since PMK can be computed easily from PSK with hash computation (1). Moreover brute-force offline attack called PMKID attack is possible [5]. Since PMKID is computed from PSK using two equations (1) and (3) and it is transported over the air, attacker can launch offline dictionary attack to match dictionary password and eavesdropped PMKID. Thus, PMK caching is not recommended in WPA2-Personal mode.

2.4. *Stateless key establishment using paired token*

Paired token (PT) is a new secondary credential scheme that provides stateless pre-shared key (PSK) more efficiently in client-server environment, specially can manage it in stateless way in server side [11,12,14]. Assume that there is an independent authentication system between client and server using some primary credential. Server authenticates the client and then issues paired token (public token and secret token) to authenticated client as a secondary credential. Public token has the role of signed identity of the client that represents the authenticated state of client. Secret token is a kind of shared secret between client and server with a special property that server can compute secret token anytime from a given public token, thus server does not need to save issued client tokens. Here we describe the scheme in the following two stages.

2.4.1. Initial authentication and issuing paired token

Let's consider a simplified authentication model between client and server. Client is registered to the server and has some primary credential for initial authentication. Assume that server has a master secret key K which is used for issuing tokens. It is used only inside the server and never be exposed outside.

In initial authentication client logs into the server using primary credential, for example, using ID and password. If initial authentication is successful, server computes two tokens as follows.

1. Public token $T_p = G_{JWT}(K, Info)$: a normal JSON web token (JWT) on user's authorization information $Info$.
2. Secret token $T_s = G_{JWT}(K, T_p)$: a recursive JWT on the above public token T_p.

Here $G_{JWT}(K, Info)$ is an abstract notation of issuing process of a JWT [6–8,14]. It represents that server prepares user-specific authorization information $Info$ and puts it in the Payload, prepares proper Header, and generates a HMAC value of the header and payload using the server's secret K,

$$Signature = HMAC(K, Header||Payload).$$

Then $Token = [Header.Payload.Signature]$ is a valid JWT issued to the user by the server. To issue JWT with limited lifetime, $Info$ can have an expiration value.

Server sends $< T_p, T_s >$ to client through a secure communication channel. In the issuing stage of paired token, secure communication channel is required to send PT to client securely. Note that initial authentication requires secure communication channel to send password securely and issuing paired token can use the same secure communication channel. As a secure communication channel we can use https, or use other custom secure channel. Client stores paired token securely in application or key storage. In web security environment paired token can be stored in browser storage such as local storage.

Public token T_p represents a signed identity of the user and can be sent to the server to provide identify of client. Secret token T_s is a kind of shared secret between client and server, and it will never be sent to server directly. Server does not need to save $< T_p, T_s >$ in DB, since T_p will be presented by the client and T_s can be computed anytime from T_p. Therefore T_s is an inherently shared secret with the server in a stateless way. Maybe server can decide to store T_p for logging purpose, but it will not be used in authentication stage.

2.4.2. One-time authenticated key exchange using paired token

If client is equipped with paired token as shown above, single message quick one-time authenticated key transport is possible using paired token. Now client equipped with paired token $< T_p, T_s >$ wants to establish a secure shared key with the server.

Client gets current time t, computes a time-based one-time authentication value $auth$, computes one-time authenticated key k as follows.

$$auth = HMAC(T_s, t||T_p), \tag{4}$$

$$k = HMAC(T_s, t||T_p||\text{“}key\text{”}). \tag{5}$$

Here “key” is a pre-agreed label for key generation. Client sends $< T_p, t, auth >$ to server.

Upon receiving $< T_p, t, auth >$, server first verifies the validity of $auth$ as follows.

1. Verifies the validity of T_p and identifies who is requesting authentication.
2. Gets his own current time and checks that client's request time t is within allowed limit (checking liveness of request to defend against replay attack).
3. Computes the secret token $T_s = G_{JWT}(K, T_p)$ from T_p and then verifies the validity

$$auth \overset{?}{=} HMAC(T_s, t||T_p). \tag{6}$$

If it is valid, server computes the same one-time authenticated key k in (5) using T_s. Here $auth$ is a time-based one-time authentication of client and proves the possession of T_s. It is an application of time-based one-time password (TOTP) scheme to paired token scenario.

3. Stateless reassociation in WPA3 using paired token

In this section we show how paired token can be incorporated with WPA3 to enhance the reassociation function. We will replace the PMK caching-based reassociation with PT-based reassociation. If the PT-based reassociation function is enabled, AP will issue paired token to authenticated client and use it for quick reassociation in subsequent connections. In the following we describe the PT-based reassociation.

3.1. Full authentication and association (issuing paired token)

We consider 3 authentication scenarios; WPA3-Open, WPA3-Personal, and WPA3-Enterprise. In every 3 cases, client and AP will share the same PMK after the full handshake is finished. After that AP prepares client's authorization information $Info$ and computes the following paired token.

1. Public token $T_p = G_{JWT}(K, Info)$
2. Secret token $T_s = G_{JWT}(K, T_p)$

And then AP encrypts $< T_p, T_s >$ using the PMK and sends it to client. Now client decrypts it using the same PMK, recovers $< T_p, T_s >$ and save it in client system. If AP wants to distinguish 3 different authentication methods, AP can prepare $Info$ differently according to AP's policy on authentication methods. For example, in the case of WPA3-Enterprise client is explicitly authenticated by the RADIUS server that $Info$ can be prepared in privacy preserving way. In the case of WPA3-Open AP can include more client-specific information in $Info$ such that AP can distinguish the client in subsequent connections.

3.2. Quick reassociation using paired token

Now client is equipped with paired token $< T_p, T_s >$ and PT-based reassociation function is enabled. If client wants to connect to the same AP again, client gets current time t, computes a time-based one-time authentication value $auth$ and computes one-time authenticated key PMK as follows.

$$auth = HMAC(T_s, t||T_p), \tag{7}$$

$$PMK = HMAC(T_s, t||T_p||\text{“}key\text{”}). \tag{8}$$

Client requests reassociation connection to the AP by sending $< T_p, t, auth >$. Upon receiving $< T_p, t, auth >$ AP checks the authenticity of client as follows.

1. Verifies the validity of T_p and identifies who is requesting reassociation connection.
2. Gets his own current time and checks that client's request time t is within allowed limit.
3. Computes the secret token $T_s = G_{JWT}(K, T_p)$ from T_p and then verifies the validity

$$auth \stackrel{?}{=} HMAC(T_s, t||T_p). \tag{9}$$

If all verification are valid, AP computes the same one-time authenticated key PMK (8) using T_s. Note that AP computes PMK in stateless way without using any client-specific stored information.

Now client and AP have the same one-time PMK. Client and AP execute the 4-way handshake protocol to compute PTK from PMK. Note that $auth$ is a time-based one-time authentication and one-time PMK is changing depending on t. So same PT can be used multiple times for reassociation for longer period of time.

3.3. *Forward secure reassociation using paired token*

The above quick reassociation protocol is efficient, but does not provide forward security. If an attacker gets a knowledge of T_s, then he can decrypt every previous encrypted traffic using the same PT. Since PT is a secondary credential that is intended to be used multiple times during its lifetime, providing forward security is important.

To provide forward security DH key exchange can be incorporated into the protocol. If client wants to connect to the same AP again, client prepares current time t and DH key share g^x and computes

$$auth1 = HMAC(T_s, t||T_p||g^x). \tag{10}$$

Client sends $< T_p, t, g^x, auth1 >$ to AP.

Upon receiving $< T_p, t, g^x, auth1 >$, AP verifies the validity of $auth1$ in the following steps.

1. Verifies the validity of T_p and identifies who is requesting reassociation connection.
2. Gets his own current time and checks that the time difference from client's request time t is within certain limit.
3. Computes the secret token $T_s = G_{JWT}(K, T_p)$ from T_p and then verifies the validity

$$auth1 \overset{?}{=} HMAC(T_s, t||T_p||g^x). \tag{11}$$

If the above verification is successful, AP prepare its DH key share g^y and computes

$$auth2 = HMAC(T_s, t||T_p||g^{xy}), \tag{12}$$

$$PMK = HMAC(T_s, t||T_p||g^{xy}||\text{“}key\text{”}). \tag{13}$$

AP sends $< T_p, t, g^x, g^y, auth2 >$ to client.

Then client can compute g^{xy} and verify the validity of $auth2$. If it is valid, client computes the same PMK (13). Now client and AP share the same PMK and can execute 4-way handshake to derive PTK.

3.4. *Fast roaming in enterprise environment*

Let's consider the fast roaming scenario in enterprise environment with multiple APs. If PMK caching is used, multiple APs have to share the real, dynamically changing, PMK cache for fast roaming. If PT-based reassociation is used for fast roaming, it is enough for multiple APs to share the static master secret key K. If all APs share K, any AP can provide fast roaming service by itself without any help of neighbor APs. In enterprise environment the RADIUS server and multiple APs are connected with a secret communication channel that sharing K secretly is quite practical assumption.

3.5. *Comparison of features*

We compare PT-based reassociation with PMK caching-based reassociation.

In the case of PMK caching client and AP share PMK as a long-term secret and use it also as a session secret. Thus using a PMK for long period of time should be very careful though real session key PTK is changing because of the randomness in 4-way handshake. AP authenticates client if it presents a valid PMKID that is present in AP's cache. Client sends PMKID to AP to start reassociation, then AP has to find the corresponding PMK in the cache. PMK is a random looking information that it does not provide any information that can identify the client. Because of the characteristics of cache memory the lifetime of PMK is limited and the number of serviced client is limited. The overall service of AP is stateful.

On the other hand, in the case of PT-based reassociation client and AP share the secret token as a long term secret. Session secret is a one-time PMK computed from secret token and current time, thus same PT can be used for longer period of time during its lifetime. AP can decide proper expiration time of T_p according to its policy. Client sends $< T_p, t, auth >$ to AP to start reassociation. Then AP can identify client and its authorization information from $Info$ in T_p. AP also verifies the time-based one-time authentication of client in $auth$. AP can compute one-time PMK with 3 hash computations, which is very efficient compared with the stateful service of PMK caching. Since AP does not need to keep any client specific information, there is no limit on the number of clients that can be serviced. AP can provide service efficiently in a stateless way for large number of clients.

In terms of fast roaming in enterprise environment, PT-based reassociation is more efficient than PMK caching. In PMK caching multiple APs should share the dynamically changing PMK cache in real time. On the other hand, in PT-based reassociation multiple APs can share the master secret key K for fast roaming. Then any AP can provide fast roaming service very easily by itself.

Table 1. Comparison of features; PMK caching vs. PT-based reassociation.

	PMK caching	PT-based reassociation
long-term secret	PMK	secret token
session secret	PMK	one-time PMK
authentication	possession	one-time $auth$
request info	PMKID	$< T_p, t, auth >$
identify client	no info	$Info$ in T_p
lifetime	cache limit	lifetime of PT
no. of clients	limited	unlimited
service type	stateful	stateless
enterprise roaming	share cache	share K

4. Analysis

4.1. Security analysis

Unforgeability. Public token and secret token are JWTs signed by AP that they cannot be forged by other entities than AP. Attackers can try to collect public tokens and authentication protocol messages, and then try to compute secret token, or even AP's secret key. Attackers can also try to forge another authentication messages without having secret token. The security of this kind of attacks will depend on the security of the underlying hash function.

Resistance to replay attack. Any kind of eavesdropping and replaying attack will be difficult since time-based one-time authentication $auth$ was used in the first move of request. Simple replay attack will not work at another time. Attackers should be able to compute fresh protocol messages working on current time.

Resistance to DOS attack. Attackers can try to attack the availability of service by sending incorrect messages to AP. But AP can detect this kind of attacks very early in the first move of request. Client's request message contains one-time authentication $auth$ and the verification process is very efficient with just few hash computations. AP can stop invalid connection requests from attackers very early and the attackers will be requested to start from the full authentication again.

Resistance to MITM attack. Man-in-the-middle attack is an issue related with the full authentication. Client has to be able to verify the authenticity of AP. Once client is equipped with paired token issued by AP, client and AP have a special 1-to-1 secure communication channel. Although multiple clients share the same PSK in WPA3-Personal, they will have different PTs issued by the same AP. Any attacker in the middle cannot intrude into the secure connection established using PT.

Privacy and untraceability. In the PT-based reassociation stage public token is sent to AP in plain communication channel as an identification of client, therefore network attacker can identify the client from the communication traffic. If privacy is a prime issue, AP can issue anonymous opaque PT with no client-specific information in public token. If AP still wants to identify the client, AP can keep the record of issued public token. It will depend on policy.

If fixed anonymous PT is used for long period of time, network attacker can try to trace the activity of the same client. To provide untraceability, AP can issue renewed anonymous PT according to its policy. Issuing renewed PT to already authenticated client is not heavy in performance. Network attacker cannot trace the renewed PT, but AP can trace the identity of client.

Forward security. Since PT is a secondary credential that is intended to be used multiple times during its lifetime, providing forward security is important. We have shown the forward security version of reassociation protocol, though it requires more communication and computation.

System security. PT-based reassociation uses time-dependent one-time authentication and key establishment using PT. Therefore, any network attacker who does not have the knowledge of secret token cannot generate fresh protocol messages and cannot continue attack. Since same PT is used multiple times during its lifetime, attackers will be more interested in system attacks that can get PT itself.

Since secret token is a secondary credential that has to be stored and used in the client system, its security will highly depend on the system security, key storage security, or application security. If an attacker can get the secret token itself by hacking the client operating system or using some malicious software, then he will be able to sniff or spoof other legitimate users. Therefore, we need to use the proposed PT-based reassociation system in a secure way in the point of system security (secure operating system, defense against malwares, secure storage, application security, etc).

4.2. *Performance analysis*

We compare the performance of PT-based reassociation with PMK caching.

Table 2. Performance; PMK caching vs. PT-based reassociation.

	PMK caching	PT-based reassociation
service type	stateful	stateless
no. of clients	limited	unlimited
enterprise roaming	share cache	share K

PMK caching is a cache-based stateful service. AP has to manage current PMKs and PMKIDs in cache. When client presents a PMKID, AP has to find the corresponding PMK from cache that it is a stateful service. It's hard for AP to provide PMK caching service to large number of clients, since AP is a lightweight computing device. To provide fast roaming service in enterprise environment multiple APs have to share dynamically changing PMKs in cache.

PT-based reassociation is a stateless service to unlimited number of clients. If client sends a connection request message $< T_p, t, auth >$, client and AP share the same one-time PMK and can start the 4-way handshake immediately. All the computations for the verification of one-time authentication and computing one-time PMK are 3 hash computations. It is a huge performance gain compared with the stateful service of PMK caching. In the point of fast roaming, multiple APs who share the master secret key K can provide roaming service by themselves without any prior arrangement.

If efficient PT-based reassociation is used more extensively, we can expect huge performance gain with reduced usage of full handshake.

5. Conclusion

In this paper we have proposed a new stateless reassociation scheme using paired token that can be applied to WPA3. It provides better performance than the traditional PMK caching in the point of stateless service, number of clients, fast roaming, etc. If PT-based reassociation technology is incorporated with WPA3, it will enhance the overall performance of wireless security protocol in client and AP. Once client is equipped with PT after fully authenticated and associated with an AP, then reconnection to the same AP will be much faster and safer.

This paper is the first draft of PT-based wireless security protocol. We need to invest more effort to analyze the security of the proposed scheme in more detail. For real application we also need to investigate every technological details required in wireless security protocol.

Funding: This research received no external funding.

Acknowledgments: This work was conducted while the author visited the University of Alabama at Birmingham as a sabbatical research fellow with the support of Joongbu University in 2020. Special thanks to professor Yulian Zheng for the favor of invitation and many helpful discussions on this work.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Wi-Fi Alliance, "WPA3 specification version 2.0," (2018)
2. Dan Harkins and Warren Kumari, "Opportunistic Wireless Encryption,", RFC 8110 (2017)
3. Dan Harkins, "Simultaneous Authentication of Equals: A Secure, Password-Based Key Exchange for Mesh Networks," 2008 Second International Conference on Sensor Technologies and Applications (sensorcomm 2008), Cap Esterel, 2008, pp. 839-844. (2008)
4. Dan Harkins, "Dragonfly Key Exchange," RFC 7664 (2015)
5. Steube, J., "New Attack on WPA/WPA2 Using PMKID," Available online: https://hashcat.net/forum/thread- 7717.html (accessed on 10 December 2020).
6. Dick Hardt, "The OAuth 2.0 authorization framework," RFC 6749 (2012)
7. Michael B. Jones and Dick Hardt, "The OAuth 2.0 authorization framework: bearer token usage," RFC 6750 (2012)
8. Michael B. Jones, John Bradley, and Nat Sakimura, "JSON web token (JWT)," RFC 7519 (2015)
9. Christopher P. Kohlios and Thaier Hayajneh, "A Comprehensive Attack Flow Model and Security Analysis for Wi-Fi and WPA3," Electronics 2018, 7, 284 (2018)
10. Kevin Benton, "The Evolution of 802.11 Wireless Security," UNLV Informatics-Spring (2010)
11. Byoungcheon Lee, "Strengthening of token authentication using time-based randomization," Journal of Security Engineering, vol. 14, no. 2, pp. 103-114 (2017)
12. Byoungcheon Lee, "Stateless Randomized Token Authentication for Performance Improvement of OAuth 2.0 MAC Token Authentication," Journal of The Korea Institute of Information Security & Cryptology, VOL.28, NO.6, pp. 1343-1454 (2018)
13. Byoungcheon Lee, "Efficient Wi-Fi Security Protocol Using Dual Tokens," Journal of The Korea Institute of Information Security & Cryptology, Vol. 29, No. 2, pp. 417-429 (2019)
14. Byoungcheon Lee, "Paired Token: A New Secondary Credential Providing Stateless Pre-Shared Key," Manuscript (2020) (Submitted to Journal)