October 2019

SECURE INTERNET OF THINGS ONBOARDING USING PUBLIC KEY CRYPTOGRAPHY AND DIFFIE-HELLMAN INTEGRATED ENCRYPTION SCHEME

Niranjan M. M

Nagaraj Kenchaiah

Ramachandra Murthy

Vijay Kothamasu

SECURE INTERNET OF THINGS ONBOARDING USING PUBLIC KEY CRYPTOGRAPHY AND DIFFIE-HELLMAN INTEGRATED ENCRYPTION SCHEME

AUTHORS:
Niranjan M M
Nagaraj Kenchaiah
Ramachandra Murthy
Vijay Kothamasu

ABSTRACT

Techniques are described for using public key cryptography and blockchain methods to automatically and securely on-board Internet of Things (IOT) devices. This is an improvement over typical approaches in which IOT devices are on-boarded to Wi-Fi® networks with a pre-shared key that could be built-in or configured through out-of-band connectivity (e.g., Bluetooth®, Wi-Fi Protected Setup (WPS), etc.).

DETAILED DESCRIPTION

A significant portion of Internet of Things (IOT) devices today use Wi-Fi access for connectivity. While IOT devices use various access technologies for connectivity such as LoRa®, Bluetooth, Narrowband (NB) IOT radio, etc., Wi-Fi is expected to continue playing a significant role in access connectivity for IOT devices. This is particularly true for IOT devices that require high bandwidth connectivity such as surveillance devices, cameras, printers etc.

IOT devices need to present a set of identity credentials to obtain access to the Wi-Fi network. IOT devices are currently on-boarded through Wi-Fi with an in-built pre-shared key (provided by the IOT device vendors) or secret keys provisioned by connecting via other methods/protocols (e.g., Bluetooth, WPS, Wi-Fi Alliance® Easy Connect®, etc.). There is no scalable well-defined approach to address this problem.

As most IOT devices have little or no flash, it is not possible to have certificate based authentication for on-boarding. Few existing techniques have in-built identity in the devices used for authentication and subsequent encryption. Also, most IOT devices are small embedded devices and, as such, may not have a complete stack to support Internet

Published by Technical Disclosure Commons, 2019

Protocol Security (IPSec), Layer 2 Tunneling Protocol (L2TP), Transport Layer Security (TLS), etc.

Automated on-boarding of IOT devices is a challenge in large deployments. The default pre-shared key may be programmed on the IOT device during the staging process. This is tedious and not scalable.

Accordingly, techniques are described herein to on-board Wi-Fi enabled IOT devices securely and generate secret keys automatically, without the need for a pre-shared key configuration. This may involve public key cryptography, Diffie-Hellman Integrated Encryption Scheme (DHIES) encryption, and blockchain technology for secured IOT on-boarding.

An initial temporary pre-shared key may be generated without the need for out-of-band communication for initial provisioning and association. Public key cryptography may be used to derive asymmetric keys (private and public keys) on both the Access Point (AP) and IOT device. Hashed One-pass Menezes–Qu–Vanstone (HOMQV) protocol may be used along with DHIES encryption scheme to populate a unique symmetric cryptographic key on both the AP and the IOT devices.

Details regarding HOMQV with DHIES encryption scheme is described as follows. In DHIES, the IOT device has a private/secret exponent "a" (secret key), and the corresponding public key is the group element "A = g^a" (where "g" is a generator in a prime-order group). To encrypt a message "M," the AP chooses a random exponent "y," computes "Y = g^y," computes the Diffie-Hellman value "σ = A^y," and uses "K = H(σ, Y)" as a key in a symmetric-key authenticated-encryption scheme to encrypt "M." The AP derives from "K" two symmetric keys "Ka" and "Ke", computes cipher-text "C = EncKe(M)" where "Enc" is some standard CPA - secure symmetric encryption scheme (e.g., Advanced Encryption Standard Cipher Block Chaining (AES-CBC)), and computes an authentication tag "T = MACKa(C)" where "MAC" is some standard authentication code (e.g., Hash-based Message Authentication Code Secure Hash Algorithm 1 (HMAC-SHA1)). The AP sends (Y, C, T) to the IOT device, which recovers the Diffie-Hellman value via "σ = Y^a," computes "K = H(σ, Y)," re-derives "Ka" and "Ke," verifies the authentication tag "T," and decrypts the cipher-text "C" to recover "M." DHIES is an instance of the Key Encapsulation Mechanism / Data Encapsulation Mechanism

(KEM/DEM) paradigm, where "Y" is the KEM portion of the cipher-text and "(C, T)" is the DEM portion. When used for key-wrapping (i.e., when "M" is a cryptographic key), this encryption scheme may also be viewed as a key-exchange protocol where only the IOT is implicitly authenticated. The AP may also be authenticated by the IOT device using a similar method if it has its public key.

Symmetric keys are necessary even though public key cryptography has strong encryption methods. Asymmetric key encryption takes more time for encryption and decryption. Asymmetric key encryption methods cannot be used as security methods in the 802.11i standard. Hence, symmetric keys are generated that can be used with a pre-shared key based security method.

IOT information may be added as Transaction T_IOT = (ID_IOT, PK_IOT, SIG_IOT) to the distributed ledger (e.g., blockchain) by the Wireless Local Area Network (LAN) Controllers (WLC). This may be used to avoid reply and man-in-the-middle attacks and also for key updating in case of re-keying.

Here, ID_IOT is the Media Access Control (MAC) address of the IOT device, PK_IOT is the public key of the IOT device, SK_IOT is the private/secret key of the IOT device, and SIG_IOT is the signature of the IOT device, signed by the device using its private key (SK_IOT).

Maintaining meta-data of transactions on the blockchain may help overcome reply attacks and identity-verification challenges during the key updating phase. For example, a compromised IOT device may launch a reply attack as a legitimate IOT device by frequently sending its information to the AP. By auditing T_IOT on the blockchain, an AP may refuse a replied request when the replied identity (ID_IOT and PK_IOT) is already part of the blockchain which stores timestamps and communication histories among all IOT devices and APs/WLCs. With respect to the key updating phase, only the IOT device having the private key SK_IOT corresponding to the public key PK_IOT can decrypt the cipher-text, carrying the symmetric cryptographic key (iPSK) generated by the AP. Unlike previous approaches, this may be achieved using a single PSK-enabled Service Set Identifier (SSID).

First, an IOT device may connect to the PSK-enabled SSID using the generated temporary PSK. This will help enable a single-SSID solution. As will be explained further

below, secure communication is not mandatory because public key cryptography is used with DHIES to exchange the iPSK between the AP and the IOT device.

The IOT device may connect to the PSK-enabled SSID broadcasted by the Wi-Fi AP. If the symmetric cryptographic key iPSK is not present or its own entry is not in the blockchain, then the IOT device generates a temporary PSK. The temporary PSK is generated using the MAC address of the IOT device along with vendor specific options (e.g., vendor name, serial number, Organizationally Unique Identifier (OUI), etc.) that are part of the association request by using the HashKey function. The AP may also fetch the MAC address of the IOT device along with the vendor specific options in the association request. It may use the same HashKey function to populate the initial temporary PSK. Thus, both the IOT device and the AP have an initial temporary PSK. Before sending back the association response, local or remote Authentication, Authorization, and Accounting (AAA) MAC filtering may be used to verify against the MAC address of the IOT device for MAC authentication. Both exchange messages using this PSK may establish secure communication.

Public key cryptography and blockchain may then be used to on-board the IOT device. As described above, secure communication is established between the IOT device and the AP. This is only for a temporary period. The following key exchange method may occur over this secure connection. Using public key cryptography, both the IOT device and the AP generates a pair of keys: private/Secret Key (SK) and Public Key (PK). For example, SK_IOT and PK_IOT may be generated for the IOT device, and SK_AP and PK_AP may be generated for the AP. The IOT device may send the public key PK_IOT to the AP and the AP may send the public key PK_AP to the IOT device independently. Now both the IOT device and the AP have acquired the other's PK.

The IOT device and the AP may use standard HOMQV along with DHIES to generate a symmetric key on both ends using asymmetric keys. The AP may act as a server and generate a symmetric cryptographic key that is unique per device (e.g., iPSK). The AP may use PK_IOT to wrap the iPSK with DHIES encryption and generate cipher-text using the HOMQV protocol. Only the IOT device having the SK_IOT corresponding to the PK_IOT can decrypt this cipher-text. After decrypting the cipher-text, the IOT device may obtain the symmetric cryptographic key (iPSK) sent by the AP.

Now both the IOT device and the AP have the symmetric cryptographic key (iPSK). The AP may de-authenticate the IOT device to return the iPSK. The IOT device re-connects back to the AP using this iPSK. Once the IOT device connects to the AP, it may update the WLC counterpart, which in turn adds this IOT device information to the blockchain. Once added to the blockchain, it is maintained in the distributed ledger and is used to avoid reply attacks, detect MAC address spoofing, and update keys in case of rekeying (possibly upon factory default).

It will be appreciated that other approaches may also be implemented in accordance with the techniques presented herein. For example, a two-SSID solution is also possible. In this example, the IOT device may initially connect to an open SSID for connectivity. This connectivity may be used to send the PSK-enabled SSID name and exchange public keys to generate the symmetric cryptographic key (iPSK) as described above. After populating the iPSK, the IOT device may use this PSK-enabled SSID to connect to the AP.

Figure 1 below illustrates an example initial IOT device association using a temporary PSK.

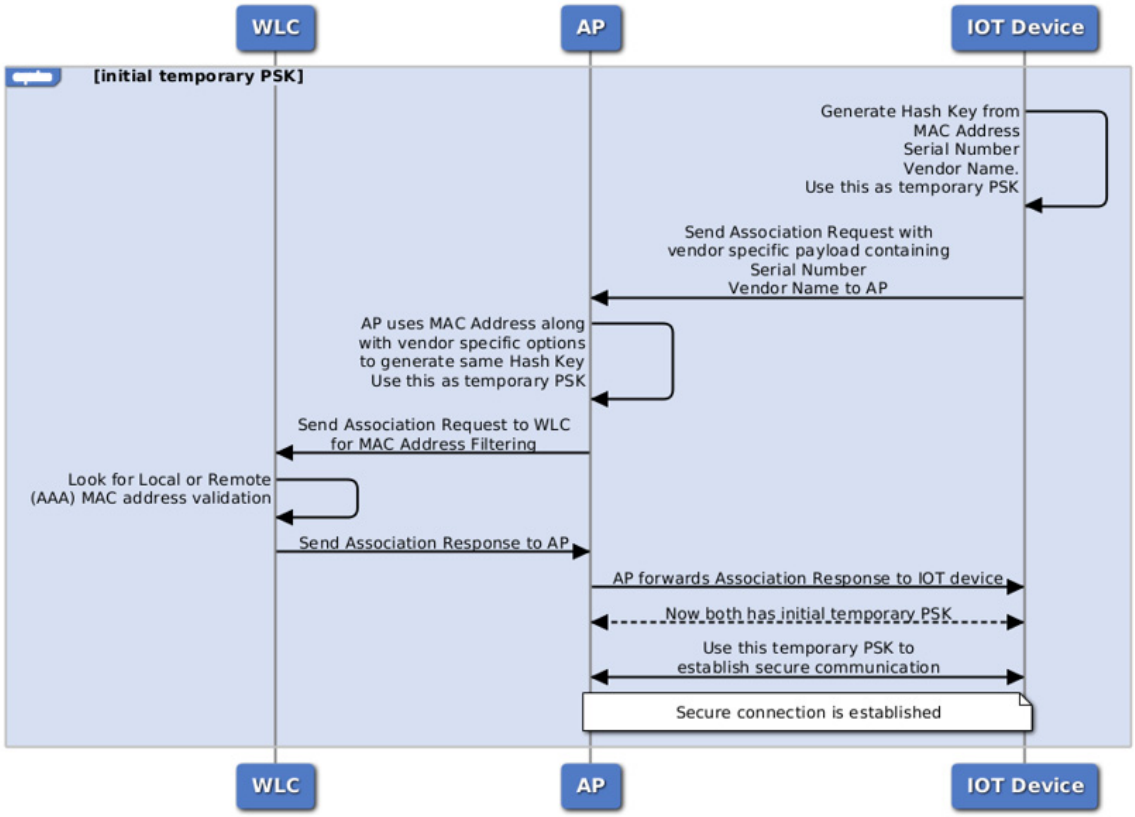

Figure 1

Figure 2 below illustrates a secure IOT on-boarding process using public key cryptography and DHIES encryption.

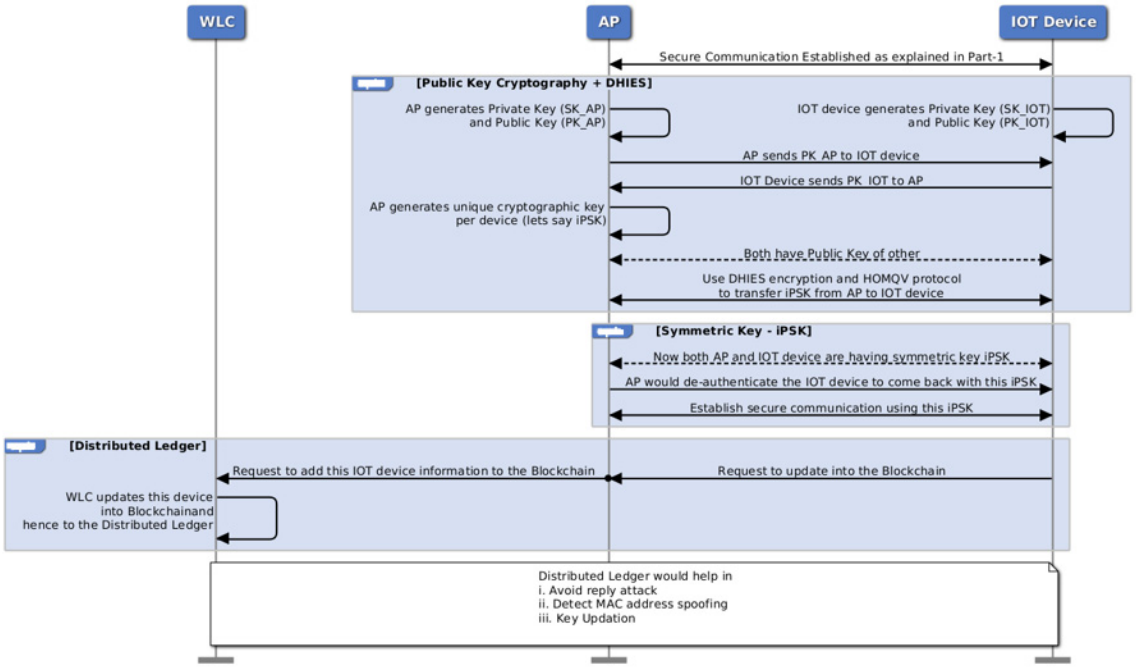

Figure 2

The techniques described herein may enable secure onboarding without a built-in PSK. This may be a single-PSK-enabled SSID solution that avoids using one SSID for onboarding and another SSID for connectivity (e.g., as in the case of hotspots). Furthermore, these approaches are scalable and reduces manual intervention. There is not necessarily a need for other means of connectivity (e.g., via Bluetooth, WPS etc.) for configuring the SSID and pre-shared key required for Wi-Fi connectivity. This may reduce associated costs by avoiding secondary protocols and device stacks. There is also not necessarily a need for a device identity certificate on the IOT device for the authentication. This method may be used to on-board any IOT device with blockchain capability, and/or for enterprise IOT onboarding.

In summary, techniques are described for using public key cryptography and blockchain methods to automatically and securely on-board IOT devices. This is an improvement over typical approaches in which IOT devices are on-boarded to Wi-Fi networks with a pre-shared key that could be built-in or configured through out-of-band connectivity (e.g., Bluetooth, WPS, etc.).