March 2022

RESILIENT METHOD TO ENHANCE EFFICIENCY OF INTERNET OF THINGS IN WI-FI6/WI-FI7 DEPLOYMENTS

Niranjan M M

RESILIENT METHOD TO ENHANCE EFFICIENCY OF INTERNET OF THINGS IN WI-FI6/WI-FI7 DEPLOYMENTS

AUTHOR:
Niranjan M M

ABSTRACT

Wi-Fi6 has arrived and is being fast adopted than expected and its successor low-latency Wi-Fi7 development is in progress. While these technologies have brought great features, TWT is a real boon for the success of IoT devices by enabling them to increase their battery life. This is based on the mechanism that Access Point (AP) and the IoT device will negotiate for time interval for accessing the medium. Since many IoT devices are battery driven and are meant to operate for days, weeks, months, or years, the low power consumption becomes a crucial aspect to increase the battery life. Hence AP need to keep IoT device information along with TWT details for long period to allocate the radio resource for the device to readily communicate when they wake up. But what if this information is lost on the AP due to critical scenarios such as AP crash, re-joins to another WLC, reboots due to software upgrades etc. The whole purpose of defining TWT mechanism does not serve the purpose due to these situations. The technique presented herein is to cache the IoT device TWT information on a suitable peer node in the network. So that for some reason AP loses this information, it can fetch all the details from this peer node and reconcile the information and start serving the IoT devices as and when they wake-up next time.

DETAILED DESCRIPTION

Wi-Fi6 (802.11ax) has arrived and is being fast adopted than expected and its successor low-latency Wi-Fi7 (802.11be) development is in progress. While these technologies have brought great features, TWT is a real boon for the success of IoT devices by enabling them to increase their battery life. This is based on the mechanism that Access Point (AP) and the IoT device will negotiate for time interval for accessing the medium.

IoT devices could potentially sleep for hours or days at a time to conserve battery life. Hence AP need to keep IoT device information along with TWT details for long period to allocate the radio resource for the device to readily communicate when they wake up. But what if this

information is lost on the Access Point (AP) due to critical scenarios such as AP crash, re-joins to another WLC, reboots due to software upgrades etc. The whole purpose of defining TWT mechanism does not serve the purpose due to these situations. Hence, we need a reliable method to maintain the TWT information of IoT devices in the given deployment. So that we shall re-build this information which would avoid re-association time and in-turn improves battery life.

Since many IoT devices are battery driven and are meant to operate for days, weeks, months, or years (depending on the application), the low power consumption becomes a crucial aspect to increase the battery life. IoT devices are equipped with embedded Network Interface Card (NIC) and thus can communicate autonomously within the network they belong to. The wireless NIC represents a large portion of the energy consumed by the device and hence an efficient power management for the NIC is important. This is achieved using different wake up and doze timers.

In legacy IEEE 802.11, the specified maximum idle period allows any station to maintain its association state is for up to 18.64h of inactivity, while TWT mechanism in IEEE 802.11ax/802.11be aims to utilize different periods for different applications, up to a year scale. As IoT devices can sleep for hours/days, AP needs to maintain device information along with negotiated TWT details for long period. When IoT device wakes up and attempt to communicate, AP will use the device information along with TWT details to allows communication without association again. Below are some of the scenarios where AP may lose this information. In which case, device must re-join like a new client, and which effectively reduce the battery life of IoT device.

- AP radio restart due to change in radio configurations (IoT devices may disconnect and associate back).
- AP crash, reboot and boot-up from scratch.
- AP disconnect from the WLC (due to keep-alive loss, WLC crash etc.,).
- AP rebooted due to image upgrade/downgrade.

In all the above cases, IoT device information along with TWT details is lost from the AP which could trigger following drawbacks:

- IoT device may need to re-associate optionally followed by re-authentication (increase in wake-up time and hence reduces battery life).

- As TWT information is lost, data which IoT device supposed to send may also lost (and hence it may affect application which use these data for statistical purpose).

The first component of the technique presented herein is to cache the IoT device TWT information on a suitable peer node in the network. So that for some reason AP loses this information, it can fetch all the details from this peer node and reconcile the information and start serving the IoT devices as and when they wake-up next time.

The second component of this technique presented herein is to specify peer node and fetching details in case of central switching deployment. In this case, IoT device's TWT information present on the AP shall be synced and cached to the WLC as and when IoT devices join and negotiate. If AP goes down and loses this information, it can fetch all these details from the WLC and reconcile the information when AP comes back and re-join WLC.

The third component of this technique is to specify the caching mechanism in case of local switching (also called flex-connect) deployments. In this case APs will also store all IoT device TWT information on its Primary and secondary peer APs. Primary and secondary peer APs are elected based on hashing of AP MAC address as a key. In-order to optimize and provide redundancy to this approach, incorporated consistent hashing mechanism (say KETAMA) where MAC address of AP is hashed to find the primary and secondary peer APs where IoT device information will be cached. When AP has lost IoT device information, it will use same mechanism to fetch and reconcile the information of all IoT devices that belong to the given AP. Local switching is prominent in Cloud (private and public) deployments.

With respect to TWT, there are existing techniques which try to improve and extend the battery life of the IoT devices using machine learning, grouping of IoT devices based on the application. But didn't encounter any technique which consider the case of providing the high availability to IoT device TWT information to serve the IoT devices seamlessly.

Dependency on IoT device (client) context and aspects of Scale:

To reconcile and re-use TWT information to work seamlessly, there is every need to maintain complete client context in-order to identify an authenticated IoT device. Thus, it is not practical to support the proposed mechanism for unlimited number of IoT devices. Hence, consider some of the below given aspects as criteria for caching the client context and TWT information:

(a) Type of the IoT device - IoT device based on the service they provide shall be given preference.

(b) TWT interval - IoT devices with long TWT interval shall be given preference.

(c) Admin configured priority for the IoT devices.

(d) Location of IoT deployments.

Based on these aspects, one shall decide maximum how many IoT devices can be cached on a given platform.

The techniques presented herein is explained in-detail as below:

- When IoT device comes up first time, it will associate with AP and negotiates the TWT interval.
- AP and IoT device store the negotiated TWT interval details.
- AP, apart from storing IoT device and TWT details locally, it will also sync to WLC/AP as explained below:
    - In case of central switching deployments, AP will sync these details to WLC.
    - In case branch/flex deployments (Cloud deployments) AP will sync this information to its primary and secondary peer APs.
- Next time when IoT device wakes up and it can seamlessly send data as the respective AP would have already have device and TWT details.
- If AP goes for a reboot (due to any of the scenarios explained above), naturally the IoT device information would have lost. But soon it comes up, will communicate with WLC or primary/secondary APs and will re-build this information.
- Next time when IoT device wakes up, AP has the required information in-place to serve the client.

Figure-1 explains Central Switching scenario wherein IoT TWT details are cached on WLC, with High Availability (HA) support.

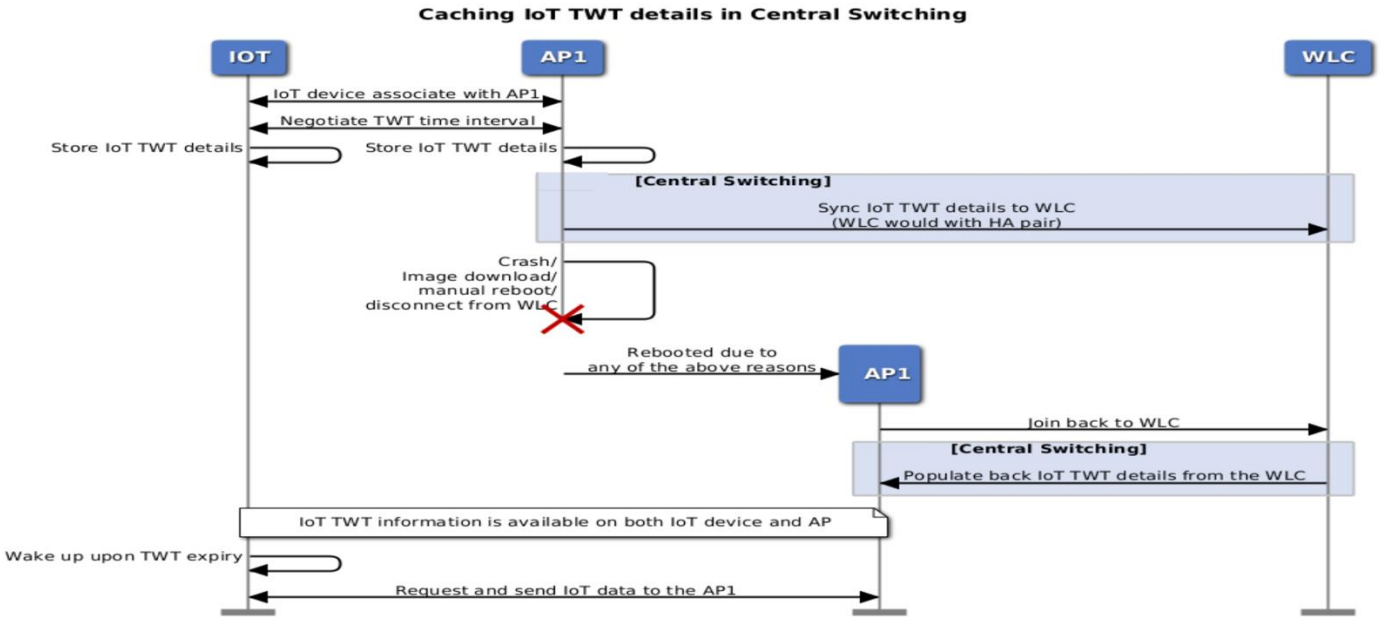

Figure-1

Figure-2 explains Local Switching scenario wherein IoT TWT details are cached on the AP(s), which are selected based on consistent hashing algorithm.

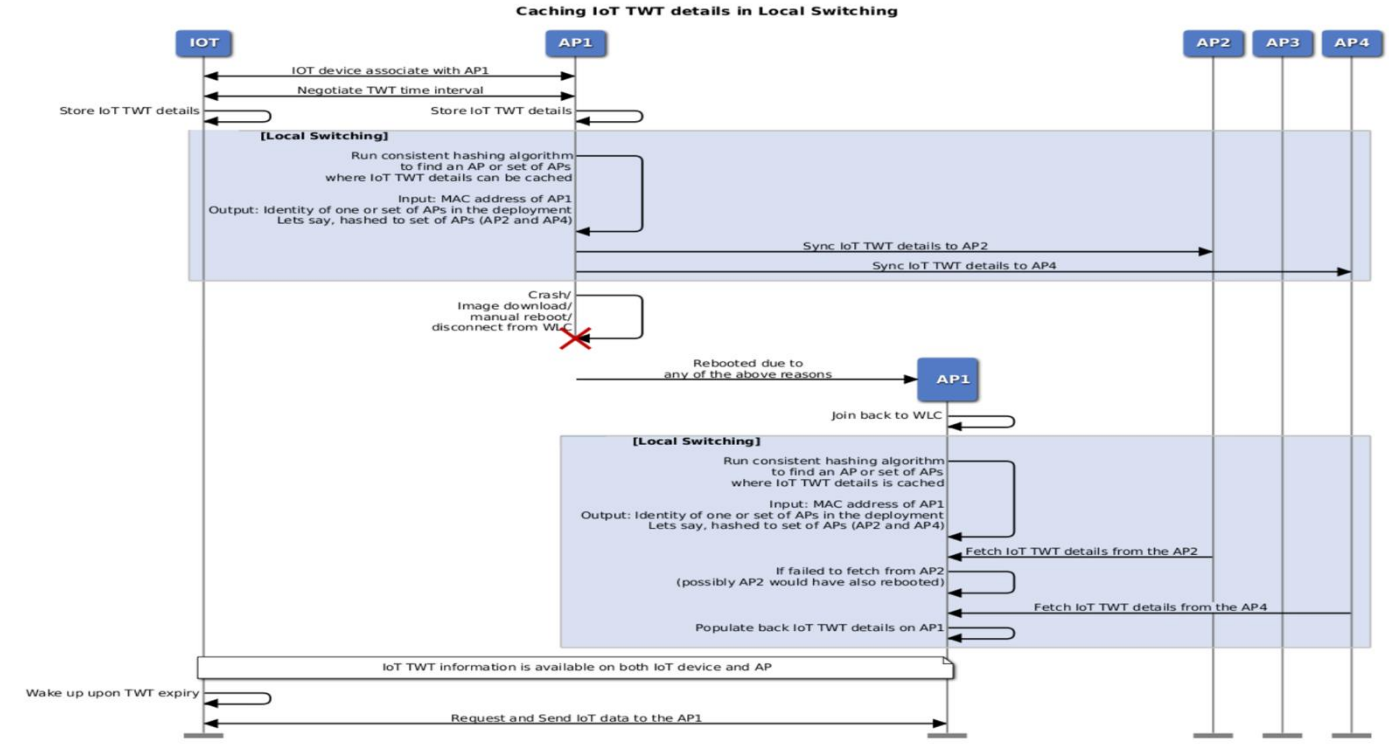

Figure-2

The techniques presented herein propose method to avoid re-association time and hence improved battery life of IoT devices. In this method, TWT information is cached, even in the absence of temporary unavailability of AP where IoT device associated, it makes sure consistent periodic data from the IoT devices which may be used for data mining and assurance. Number of IoT devices and application areas are increasing and lot of them are deployed in mission critical and life-threatening applications. These IoT devices play a crucial role in safeguarding the critical equipment. So IoT devices Battery life is extremely important. This is one of the reasons why TWT specification got a place in the new 802.11ax/802.11be standards. Hence this method shall help and align in this direction and compliment the standards. There are wide variety of IoT devices which can sleep for days, weeks or more and chances of scenarios where IoT context information getting lost (from APs) are quite possible and this incorporating this method is necessary.

Appendix-A

Target Wake Time (TWT):

TWT allows devices to negotiate when and how often they will wake up to send or receive data. TWT increases device sleep time and, in turn, substantially improves battery life. The TWT mechanism in IEEE 802.11ax is based on the implementation in IEEE 802.11ah. TWT will be very useful for both mobile devices and IoT devices. TWT uses negotiated policies based on expected traffic activity between 802.11ax/802.11be clients and an 802.11ax/802.11be AP to specify a scheduled wake time for each client. 802.1ax/802.11be IoT clients could potentially sleep for hours or days at a time to conserve battery life. With the TWT mechanism, stations can agree with the AP on a common wake scheduling, allowing them to wake up only when required, hence, to minimize energy consumption and contention within the Basic Service Set (BSS), that is, the wireless network formed by the AP and the associated stations.