October 02, 2018

802.11AX FOR INTERNET OF THINGS - MACHINE LEARNING ASSISTED OPTIMIZED POWER SAVE TECHNIQUES FOR IOT DEVICES USING 802.11AX TARGET WAKE TIME

Vaibhav Lade

Asha Mohan

Santosh Patil

802.11AX FOR INTERNET OF THINGS - MACHINE LEARNING ASSISTED OPTIMIZED POWER SAVE TECHNIQUES FOR IOT DEVICES USING 802.11AX TARGET WAKE TIME

AUTHORS:

Vaibhav Lade
Asha Mohan
Santosh Patil

ABSTRACT

Machine-learning assisted optimized power-saving techniques for battery-operated Wi-Fi Internet of Things (IoT) devices are provided. These techniques use various IoT device information including Manufacturer Usage Description (MUD) data along with a clustering algorithm to provide an automatic and dynamic computation of Target Wake Time (TWT) power save schedule for the devices. These techniques may be used by next generation 802.11ax access points to more efficiently schedule the target wake time for the 802.11ax supported IoT devices, to thereby prolong the lifetime of battery operated IoT devices connected to the 802.11ac access points.

DETAILED DESCRIPTION

The IoT is transforming the world at home, at the office, and at other locations accessible by mobile or other devices. IoT-based products allow the enterprise and consumer industry to better connect things with people. Often such products are portable and located in remote, unreachable environments where it is not possible to provide a constant source of power. Many IoT devices rely on battery power, wherein the life of the IoT device may be extended by lowering power consumption.

Brief Background on previous power-save techniques for Wi-Fi devices

An existing 'Legacy PS' mechanism has been in use as the Wi-Fi standard since 802.11b wherein Wi-Fi clients may sleep between access point (AP) beacons or multiple beacons, waking when data is to be transmitted and for beacons containing the Delivery Traffic Indication Map (DTIM). (Wi-Fi clients may transmit at any time, since the AP does not sleep.) The DTIM is a bit-map indicating that the AP has downlink traffic buffered for transmission to particular clients. If the AP has DTIM bit set for a client, it may retrieve data from the AP by sending a Power-Save Poll (PS-Poll) frame to the AP. This power-save scheme is effective but only allows clients to doze for a small beacon interval. Using this approach, clients still need to wake up several times a second to read DTIM from AP's beacon frame.

The 802.11ax Wi-Fi standard has added power saving enhancements that may benefit both mobile devices and IoT devices. For example, the 802.11ax standard for Wi-Fi emphasizes techniques for power-save battery-powered IoT Wi-Fi devices. In particular, controlling Target Wait Times (TWT), Basic Service Sets (BSS) Coloring, 2 MHz minimum size transmission, and orthogonal frequency-division multiple access (OFDMA) uplink/downlink scheduling may help extend battery life for indoor Wi-Fi IoT devices. With 802.11ax, the 802.11ah Target Wait Time (TWT) mechanism has been modified to support trigger-based uplink transmission. TWT uses negotiated policies based on expected traffic activity between 802.11ax clients and an 802.11ax AP to specify a scheduled wake time for each client that could substantially improve the battery life of the IoT devices. AP and clients negotiate and define a specific time to access the medium. However, the AP needs to efficiently schedule wake times for the clients as the number of clients/IoT devices rapidly changes in the network.

In general, Network-Based Application Recognition (NBAR) techniques for traffic analysis may be used. NBAR can perform packet inspection and recognize a variety of application packets. After application recognition, the network can tune the resource allocation to give higher priority to the critical applications using higher quality of service (QoS). This will guarantee bandwidth allocation to critical applications from a centralized system.

Here, techniques are provided for using machine learning techniques to better understand behavioral patterns of Wi-Fi IoT devices and assist the 802.11ax Wi-Fi access point to deliver a unique power-save solution to IoT devices that can extend the battery life of these devices. Specifically, a clustering mechanism may be used to provide feedback to the AP for power saving in an 802.11ax IoT environment. In particular, a "Hierarchical Agglomerative Clustering Algorithm" is provided that can analyze traffic patterns of the

IoT devices using data collected over a period of time from these devices by the AP that they are connected to. The traffic needs to be analyzed based on time, content, context, neighbor proximity, etc. to identify the group of IoT devices that have a similar behavioral pattern. Behavioral analytics may be performed once the IoT traffic pattern is learned, and may serve as an input to the AP to schedule wake-times of battery powered IoT clients more efficiently based on their behavioral pattern using 802.11ax TWT mechanisms.

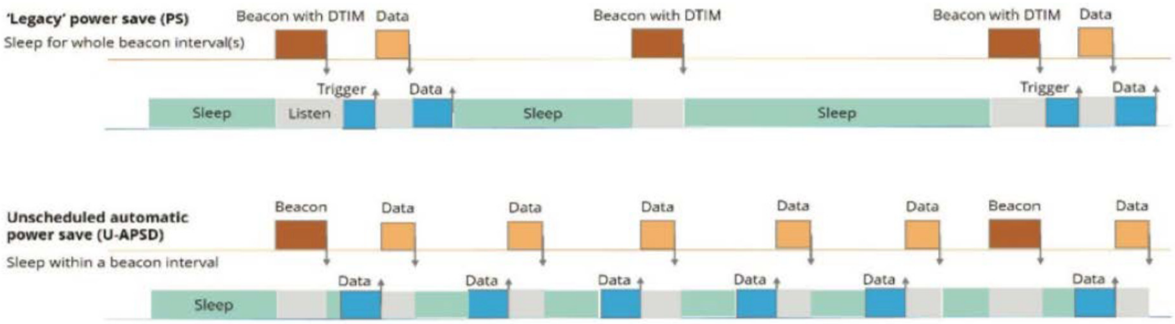

FIG. 1

With 802.11e, the IEEE identified that voice-capable devices needed a new power-save mechanism, as voice packets are transmitted at short time intervals, typically 20 msec. Unscheduled Automatic Power-Save Delivery (U-APSD) allows a power-save client to sleep at intervals within a beacon period. AP buffers downlink traffic until the client wakes up and requests its delivery. So, these existing techniques, as shown in FIG. 1, are not sufficient for IoT devices which typically sleep in minutes, hours, or maybe even days.

Introduction to TWT scheduled power-save scheme (802.11ax Draft 3.0: 27.7)

With 802.11ax, a new power-save mechanism TWT allows more flexible, long-term and even multi-client sleeping arrangements. With TWT, there is no longer a strict agreement between AP beacons and sleep time of IoT Wi-Fi client devices. Generally, the station can request a schedule to wake up at any time in the future. The result is significant power savings for battery operated devices, particularly those in the IoT space. Arrangements are shown in FIG. 2 for individual TWTs and broadcast TWTs.

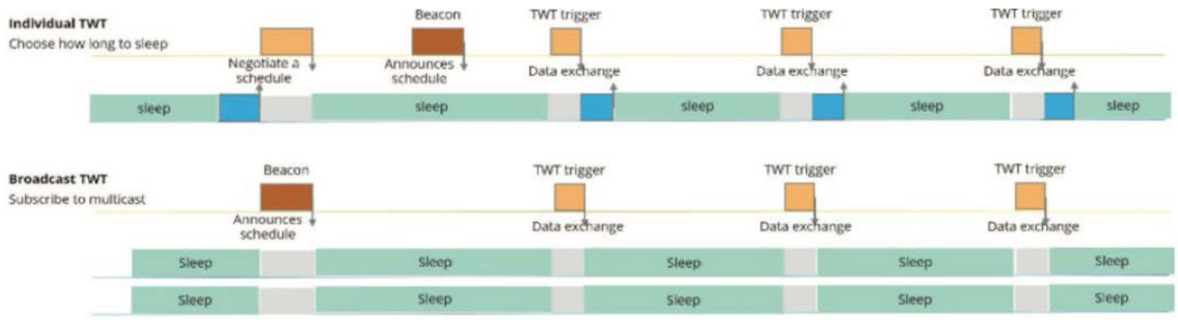

FIG. 2

According to present techniques, a negotiation between the client and AP sets up an "agreed-upon schedule" for the client to wake up and communicate. The schedule is periodic with extended multi-beacon intervals ranging from seconds to minutes, hours or even days between Wi-Fi client device activities. When clients' negotiated/designated time occurs, clients wake up and wait for a trigger from the AP, in the case of multi-user mode, and then exchange data with the AP.

Device-specific information

A sample of device-specific information reported by the AP is provided as follows. The information is collected in several ways. Some data is collected by performing a deep packet inspection, other information is provided from the physical layer and MAC layer header in the packet or from client statistics maintained in the AP's client table. Information may include:

- MAC address of IoT device:
  - This may help with identifying the manufacturer of the IoT device.
- MUD Uniform Resource Identifier (URI) broadcasted by IoT device:
  - MUD is a standard in IETF (https://datatracker.ietf.org/doc/draft-ietf-opsawg-mud/) for describing manufacturing usage description about device traffic and port usage.
  - MUD URI can be captured at an AP to understand device usage pattern in terms of traffic flow and protocol usage.
- Average air time of packets
- A mean time interval between packets

- Destination address/URL that the IoT device is communicating with:
    - This can be captured through Domain Name Service (DNS) query initiated from a device.
- Initial packet information for encrypted communication:
    - IoT devices use Transport Layer Security (TLS), Internet Protocol Security (IPSEC) or similar communication channel towards external servers.
    - Initial packet of TLS. IPSEC can be captured, which will be unencrypted, to understand security algorithms proposed by the IoT device.
- This will help understand security and cypher support on an IoT device.
- Byte distribution of data sent by the IoT device:
    - This will help understand the packet size and spread across the time window during active communication.
- Packet throughput statistic, or the number of packets sent and received.
- Cumulative statistics of different application protocols, servers contacted, multicast/broadcast packets.
- Average packet size.
- Peak to mean ratio of transmission rate.
- The power source of a device.
- RSSI of the device as seen by AP.
- Location of IoT devices.

The MAC and PHY layer statistics collected for each IoT device may be sent to the cloud for further analysis and categorization of IoT devices.

Collaboration between the Access point and Cloud based Machine Learning

A high-level architecture diagram showing the placement of IoT devices is shown in FIG. 3. A group of IoT devices like light bulbs, sensor devices, cameras, printers, etc. are connected to the AP. The AP reports the client information to the controller (in a Control and Provisioning of Wireless Access Points (CAPWAP based AP)) and directly to the cloud for a thin cloud AP. Machine learning is performed on the cloud and provides feedback to the AP in this architecture. The cloud may be public or private and may be configured in any suitable manner.

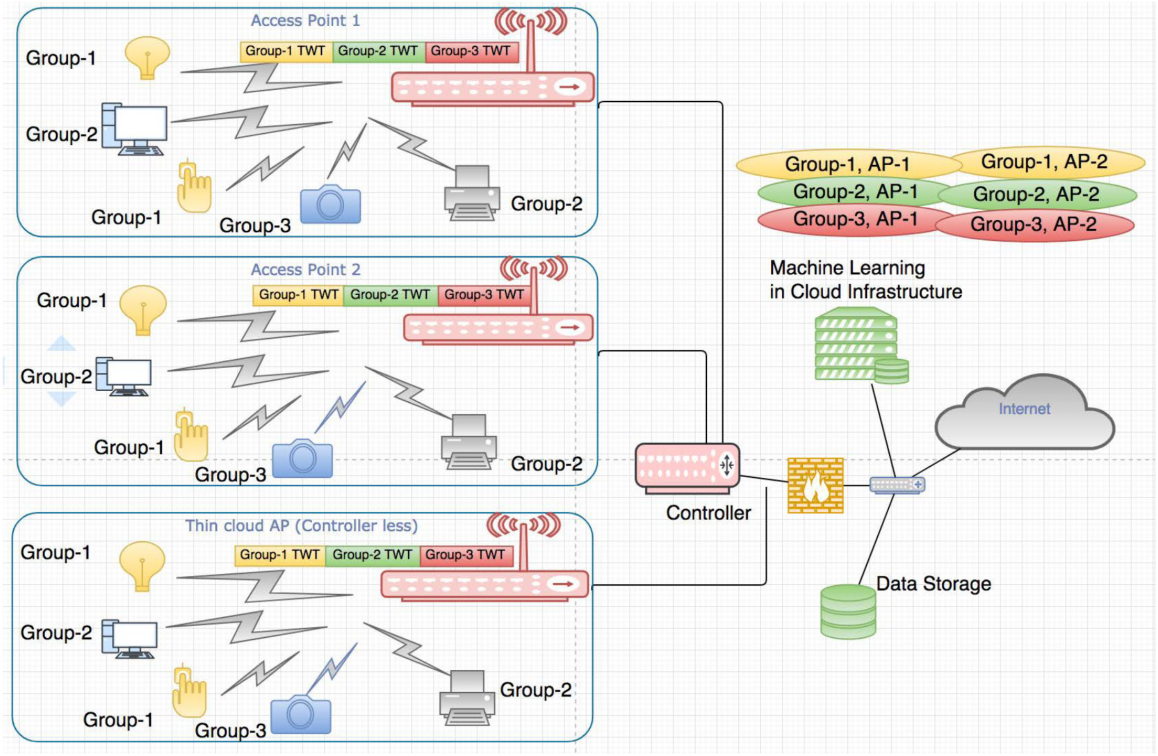

FIG. 3

A comprehensive data mining framework may be developed based on the information collected from the AP per IoT device. IoT device information is collected on an on-going basis and sent to the cloud for machine learning. Information collected from the AP may be used to derive features that can cluster groups of similar IoT devices based on several behavioral patterns.

For example, the mean time interval between packets recorded per IoT device over a period of time is useful in learning the sleep and wake time of the devices. The signaling overhead of the device may be calculated based on a number of different application protocols used, servers contacted, DNS requests, Network Time Protocol (NTP) intervals, etc. The MAC address of the device may indicate the vendor type information (e.g., organizational unique identifier) and may be used to look up a catalog of IoT devices supported by the vendor to classify the type of the device (e.g., a manufacturer usage description can identify the IoT device and its purpose). As more information is learned, the clustering algorithm may be trained to group the set of IoT devices with similar characteristics.

After the features are derived, different weights are assigned to features with a knowledge of how important the features are on identifying similar IoT devices. A 'bottom-up' clustering approach ('Agglomerative') may be used wherein each data point is considered to be a cluster at the first iteration and more points are added to the cluster in subsequent iterations, based on the proximity matrix. An objective function will be defined to find similarity between two clusters based on the increase in squared error when the two clusters are merged.

A definition of similarity for clustering can be defined based on several factors, e.g., traffic patterns of the IoT device over period of time, whether the device encrypts traffic, the mobility of the device, active and passive times of the device, etc. Once the cluster of IoT devices are identified, each cluster can be separately classified into categories of groups of devices. This classification of devices to a group may be sent back to the AP. The AP will use the information as feedback to schedule the TWT of the IoT devices.

The objective of the clustering is to classify each device into one group. The actual implementation may be tuned in various ways to classify the IoT device to provide feedback to the AP. Here is an example grouping that can be created based on device information sent by the AP.

Group-1: Low traffic devices
Group-2: High traffic devices
Group-3: Time critical devices
Group-4: Low-powered devices

It is useful to identify the devices with low power high traffic demands and put them into separate groups. Similarly, the time critical devices can be grouped separately as opposed to devices that are not sensitive to delays in transmissions. The AP can schedule the IoT devices knowing more about its usage and having more context on the capability of the device. Grouping the low powered devices separately helps the AP in scheduling longer sleep times to save battery power for longer periods on these devices.

Every group that is formed from the clustering algorithm will be mapped to a broadcast TWT SP (service period). TWT broadcast SP is uniquely identified by the < broadcast TWT ID, MAC-address of TWT scheduling AP > tuple.

In summary, techniques are provided for automatic and dynamic computation of the TWT power-save schedules for IoT Wi-Fi 802.11ax devices. These computational techniques are assisted/driven based on machine learning of behavioral patterns (e.g., time, content, context, etc.) of the IoT devices.