Fast Centralized Authentication in Wi-Fi HaLow Networks

Dmitry Bankov*, Evgeny Khorov†*, Andrey Lyakhov*, and Ekaterina Stepanova*
* Institute for Information Transmission Problems, Russian Academy of Sciences, Moscow, Russia
Email: {bankov, khorov, lyakhov, stepanova}@iitp.ru
† Skolkovo Institute of Science and Technology, Moscow, Russia
E-mail: e.khorov@skoltech.ru

***Abstract*—In early 2016 Wi-Fi Alliance announced a new technology — Wi-Fi HaLow — which aims to master the raising Internet of Things market with its tremendous number of devices. This technology brings revolutionary changes to Wi-Fi, improving transmission reliability and power efficiency in scenarios with thousands of sensor stations being connected to a single access point. However neither novel channel access methods, nor advanced power management schemes can be used until the stations are connected to the access point, because the access point may not know that the stations exist, to say nothing about their capabilities. Nevertheless, it may happen that a large group of sensor stations are simultaneously trying to connect to the access point, which inevitably leads to continuous collisions and finally to extremely long connection process. This problem has been addressed by the technology developers which designed the Centralized Authentication Control protocol. The protocol provides the access point with an opportunity to hasten the process by managing the ratio of stations which are allowed to start establishing a connection. In this paper, we study how the number of successes depends on this ratio and develop two algorithms, which hasten connection process close to the theoretical limit.**

***Index Terms*—Internet of Things, IEEE 802.11ah, Wi-Fi HaLow, Authentication, Link Set-Up**

I. Introduction

Being the most widely used wireless technology operating in license-exempt bands, Wi-Fi tries to expand its area of applicability to typical Internet of Things scenarios [1]. Specifically, in early 2016 Wi-Fi Alliance announced a new technology — Wi-Fi HaLow — which is based on the novel IEEE 802.11ah standard. Originally planned as a set of small refinement of Wi-Fi operation, the standard turned out to be a heavy document touching almost all Wi-Fi functionality. In particular it provides a palette of new methods improving transmission reliability, spectrum efficiency and power consumption in a scenario when up to 8000 of power-limited sensor stations (STAs) are connected to a single access point (AP). However novel channel access or power management techniques cannot be used before the STAs are connected to the AP. In other words, the AP needs to know that a particular STA is present, and which techniques it supports.

Consider a scenario with a large number of STAs trying to establish a connection with the AP. For example, such a situation may occur after a power outage, or when a group of STAs enters the coverage area of the AP. In any case, the AP periodically broadcasts special management frames, called beacons, to advertise the network. Having received such a beacon, all the STAs start contend for the channel to send a request to establish a connection. When the number of STAs is small, even the legacy CSMA/CA method with the default parameters provides low probability of request collisions, and the process finishes quickly. However, the large number of STAs leads to multiple frame collisions, frequent retransmissions, which extremely increases the time and the energy, needed to establish the connection.

To improve the situation, Wi-Fi developers designed a special protocol, which provides the access point with an opportunity to hasten the process by managing the ratio of STAs which are allowed to start establishing a connection.

Establishing of a connection is a multi-stage process, which starts with *Authentication*. By *authentication* handshake, the AP and the connecting STA get information about the presence of each other and check the validity of security keys. During authentication the STA sends the AP an authentication request (AuthReq), and the AP responds with an authentication response (AuthReq). This step is mandatory even when the open system authentication is used, i.e. when the network is insecure, because the devices have to ensure that such authentication is supported. Dynamical control of the ratio of STAs which are allowed to initiate Authentication provides an opportunity for the AP to decrease the collision probability and to speed up the connection establishing process. So the corresponding protocol is called Centralized Authentication Control (CAC)[1]. Naturally, the standard is a framework: it describes CAC, but says nothing about how to dynamically select its parameter values, i.e. an appropriate value for the ratio.

In this paper, we propose two algorithms that can be used to select CAC parameters. We perform extensive simulation experiments and show that their usage allows significantly decreasing the authentication time for large groups of simultaneously connecting STAs even in case of heavy background traffic.

The reported study was partially supported by RFBR, research projects 15-07-09350 a and 15-37-70004 mol_a_mos.

[1]Note that this protocol is related to channel access rather than to the security issues [2].

978-1-4673-8999-0/17/$31.00 ©2017 IEEE

The rest of the paper is organized as follows. In Section II, we describe a considered scenario and formulate the problem statement. Section III briefly reviews related papers. We design algorithms which adaptively select CAC parameters in Section IV. Simulation results are given in Section V. Section VI concludes the paper.

II. Scenario and Problem Statement

In the paper, we consider a Wi-Fi HaLoW network with an AP and a large number of sensor STAs. These STAs simultaneously appear in some time instant and start trying to set up the link with the AP, using the CAC protocol. We assume that all the STAs are located in transmission range of each other. Moreover, apart from the sensor STAs, there are several interfering STAs that transmit in the saturated mode.

A. Centralized authentication control

Let us describe in detail the CAC protocol. It works in the following way. The AP periodically broadcasts beacon frames which carry the authentication threshold, hereafter referred to as *threshold*. The interval between two consequent beacons is called beacon interval (BI). Threshold is an integer number from 0 to 1023 and is set by the AP. On initialization, a STA equiprobably draws an integer value from 0 to 1022. If the threshold is greater than the drawn value, the STA enqueues an authentication request (AuthReq). Otherwise, it waits for the next beacon, receives the new threshold and performs the comparison again.

When an AuthReq is enqueued, the STA starts an $AuthenticationRequestTimeout$ timer, during which it expects to receive the response (AuthRep) from the AP. If no AuthRep is received when the timer expires, the STAs compares the random value with the latest received threshold and if the threshold is greater than the random value, adds a new AuthReq to the queue. Note that the random value is drawn only once on initialization and can be re-drawn only after receiving an AuthRep from an AP.

B. Channel access.

To transmit AuthReq and AuthRep frames (see Fig. 1), the STAs and the AP use EDCA, which can be briefly explained as follows. Each STA has a transmission queue. When a frame is added to an empty queue, the STA senses the channel and, if it is idle, the STA transmits this frame. If the channel is busy, the STA randomly picks a backoff counter initialized with an integer number from an interval from 0 to $CW_r - 1$, where CW_r is the contention window and r is the retry counter. Initially, r equals zero and CW_0 equals the CW_{min} parameter (by default, equal to 16). The backoff counter is suspended when the channel is busy. Once the channel stays idle for time interval $AIFS$, the STA resumes the backoff countdown, decrementing it every empty slot σ. When the backoff reaches zero, the STA transmits the frame. $SIFS$ after a STA has received a frame, it shall send an acknowledgment (ACK) to the sender. If the ACK is received, the STA considers the frame successfully transmitted and starts serving a new frame, if the queue is not empty. If no ACK arrives during $AckTimeout$, the STA considers that the frame is lost. In this case, the STA increases the retry counter and selects a new value of backoff counter from a new window defined as:

$$CW_r = \begin{cases} CW_{min}, & r = 0, \\ \min\left\{2CW_{r-1}, CW_{max}\right\}, & r > 0, \end{cases}$$

where CW_{max} is the maximal value of the contention window (by default, equal to 1024). If the retry counter reaches the retry limit RL (by default equal 7), the frame is dropped from the queue.

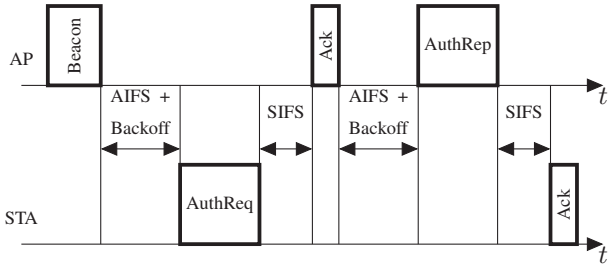

Figure 1. An Example of Authentication Handshake.

When several STAs work in the same channel and their backoff counters reach zero simultaneously, their frames collide, which results in unsuccessful transmission attempt. The more STAs try to transmit their data, the greater is the collision probability and the time, required to resolve the collisions. As a result, STAs can fail to authenticate before $AuthenticationRequestTimeout$ and have to send a new AuthReq. The centralized authentication control protocol allows the AP to manage the number of STAs simultaneously trying to request authentication and thus to increase the probability that authentication is completed within one beacon interval. So we need to define an algorithm to control the authentication threshold in such a way that minimizes the average authentication time for the STAs.

III. Related Papers

Previously, hastening the link set-up process was not considered as an issue for Wi-Fi networks. However, in many state-of-the-art scenarios a large number of STAs are located close to each other and work with the same AP. When many STAs simultaneously try to connect to an AP, their authentication attempts collide, which significantly increases the connection set-up time. A way to solve this problem, known as Fast Initial Link Set up, has been introduced in the IEEE 802.11ai [3] standard amendment. Firstly, it describes a solution to decrease the time needed to detect a nearby network, the idea of which is that APs provide the connecting STAs with information about other networks in vicinity. Secondly, it provides a way to decrease the collision probability at link set-up by limiting the number of contending STAs. To do it, the AP divides STAs into groups and allocates special time intervals, during which only one group can access the channel. The division

into groups is done either by the traffic priority or by the MAC address.

Another solutions to limit contention is described in IEEE 802.11ah standard amendment [4]. This amendment introduces to authentication control protocols: the centralized (CAC) and the distributed (DAC). The DAC is studied in [5]. The authors propose a mathematical model of link set-up process with DAC and show that with DAC the link set-up time depends linearly on the number of connecting STAs, while without it the dependency is almost exponential.

Studied in the paper the centralized authentication control protocol was initially described in presentation [6]. This presentation contains also the idea of a threshold selection algorithm, which is based on tracking the length of the AP queue with AuthReps. Specifically, every beacon interval the value of the threshold is increased or decreased by Δ that has a fixed value. If the queue is longer than some parameter, the threshold is decreased by Δ. Otherwise, it is increased.

A simplified analysis of CAC is given in [7]. In that paper, the authors do not consider any specific threshold selection algorithm. Instead, they estimate the association time for a large group of STAs by dividing it into equal sub-groups, calculating the association time for a sub-group without CAC and multiplying this time by the number of groups. Thus they show that the association time can be minimized by selecting a specific number of sub-groups, but do not propose a way to set this number by choosing the authentication threshold, when the number of STAs is unknown.

Another important result is given in [8], where the authors describe their implementation of the IEEE 802.11ah MAC and PHY in the ns-3 [9]. To evaluate performance of an IEEE 802.11ah network, they use an algorithm for choosing the authentication threshold, proposed in [6], and show that the link set-up time grows linearly with the number of connecting STAs. However, they do not describe a way to select Δ, but, as it is shown in Section V, the authentication time and the rate of its growth with the number of STAs significantly depend on Δ. Since such an implementation of IEEE 802.11ah is being actively used to evaluate network performance in a variety of scenarios [10], we also run experiments using the model developed in [8], changing only the implementation of CAC.

Our previous study of CAC is given in [11], where we have solved a similar problem. Specifically, we evaluate performance of CAC and propose several algorithms to choose the authentication threshold. It is shown that with proper threshold selection, 8000 STAs can connect to an AP in less than 2 minutes. However, this study has a significant drawback. In that paper, the STAs re-pick the random value before receiving every new beacon. But the current version of the IEEE 802.11ah standard explicitly states that the random value is picked on initialization and can be re-picked only after successful authentication. Such changes significantly affect the efficiency of the designed solutions, and in Section IV, we design novel algorithms which take into account the aforementioned peculiarity.

Also in [11] we have proposed an algorithm based on virtual carrier sense to provide contention-free access for the AuthRep, AReq and ARep frames. This idea was developed in [12]. In this paper it is proposed that STAs use EDCA to transmit AuthReq frame, and then transmit AuthRep, AReq, ARep and Ack frames separated by SIFS. For this purpose STA specifies the time until the end of the Ack for ARep in the Duration/ID field of AuthReq. STAs that receive AuthReq consider the channel virtually busy for the time specified in Duration/ID field. If the AP refuses to associate the STA, the AP sets 0 in Duration/ID field of AuthRep. Implementation of this scheme can face the following problems. Firstly, this scheme does not correspond to the standard because it requires sending AuthRep, AReq and ARep instead of acknowledgements. Secondly, the AP and STAs need time to process AuthReq, AuthRep, AReq and ARep frames, therefore they might not be able to send these frames with interval SIFS. Finally, the AP can stop virtual occupancy of the channel only with a CF-end frame, in which it is impossible to send the information about a cause of association failure. Other mistake made in this paper is that during authentication the STAs re-pick the random value after every beacon.

IV. Authentication Threshold Selection Algorithms

A. Preliminary analysis

Before developing the algorithms, let us do some auxiliary experiments. Consider one BI, during which k STAs are allowed to request authentication, and no other STAs are present. If k is low, all STAs succeed to send AuthReqs and receive AuthReps during the BI. If k is high, most AuthReqs collide, as well as the corresponding AuthReps. As the result, the AP does not succeed to respond to all AuthReqs during the BI.

Obviously, there is a maximal number of STAs k_{opt} (see Fig. 2, which presents results averaged over 100 runs) which, when allowed authentication in a BI, all succeed to send AuthReqs and to receive AuthReps. An algorithm for authentication threshold selection should authenticate such a maximal number of STAs each BI and thus reach minimal average link set-up time for the group of STAs. However, the AP initially knows neither k_{opt}, nor the number of connecting STAs.

The problem of finding k_{opt} can be solved approximately by tracking the length of the AP queue of AuthReps. When the number of simultaneously connecting STAs exceeds k_{opt}, by the end of the BI the queue is non-empty, which means that at the next BI the AP should allow authentication to a lesser number of STAs.

Here, we see the following issue. Consider a situation, when a STA sends an AuthReq and the AP adds the corresponding AuthRep to the queue, but does not succeed to deliver it before $AuthenticationRequestTimeout$ at the STA expires. If the timeout is less than the BI duration, the STA can transmit a new AuthReq before the end of the BI. As a result, by the end of the BI the AP's queue can contain several AuthReps for the same STA. This issue has negative effect on the network performance because of two reasons. Firstly, it increases the

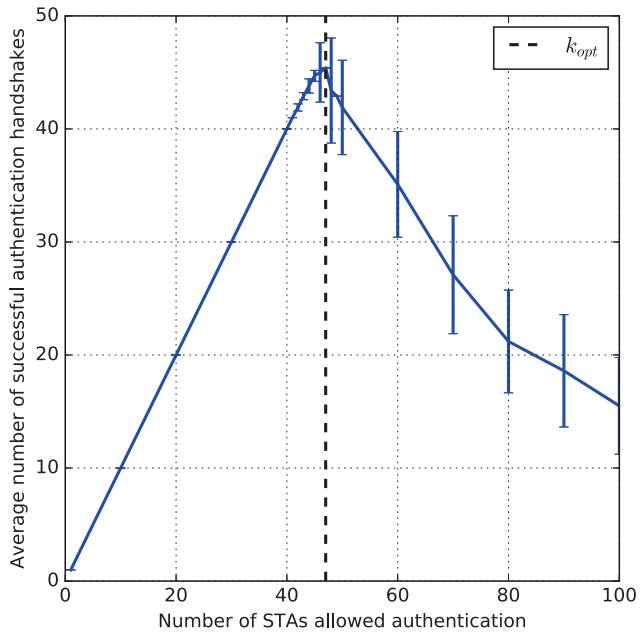

Figure 2. Dependency of the average number of successful authentication attempts in a BI on the number of STAs that are allowed authentication

channel load. Secondly, the AP wastes time sending unnecessary AuthReps, thus increasing the delay for other frames. To solve this problem, we set $AuthenticationRequestTimeout$ equal to the BI duration.

Now let N unassociated STAs appear at the beginning of the experiment. Their random values are equiprobably distributed between 0 and 1022. By setting the threshold value to v, the AP allows authentication to the average number of $\frac{v}{1023}N$ STAs. Increasing the threshold by Δ, the AP allows authentication on average to additional $\frac{\Delta}{1023}N$ STAs and to those STAs that have not succeeded to authenticate during the previous BIs. In order to connect the STAs as quickly as possible, the AP should select such Δ, that $\frac{\Delta}{1023}N = k_{opt}$ holds, i.e., to authenticate the maximal possible number of STAs in each BI and not to leave unauthenticated STAs from the previous BIs.

This explains two drawback of the algorithm proposed in [6]. The first one is that Δ is fixed and is not set according to the number of STAs. For example, if k_{opt} equals 50, Δ should be set to 6 if 8000 STAs try to authenticate, because $\Delta = \frac{k_{opt}}{N}1023 = \frac{50}{8000}1023 \approx 6$. At the same time, if the number of STAs is 50, Δ should be 1023, but if we keep Δ equal to 6, the STAs have to wait up to 170 BIs to request authentication. The second drawback is that this algorithm has no waiting mode. If there are no new STAs, the AP has no AuthReps to send and increases the threshold up to 1023. If suddenly a group of STAs appears, they all send AuthReqs, the queue at the AP grows and the AP decreases the threshold. In this case, the AP wastes much time, while decreasing the threshold to such a value that allows authentication to less than k_{opt} STAs and the queue does not grow by the end of BI.

B. Proposed algorithms

Taking into account the drawbacks of the existing solutions, we develop two novel algorithms how to select the threshold.

Both algorithms can work in three modes (waiting, studying, working) and select Δ adaptively according to the queue size.

Let us consider the first algorithm, called “Up”.

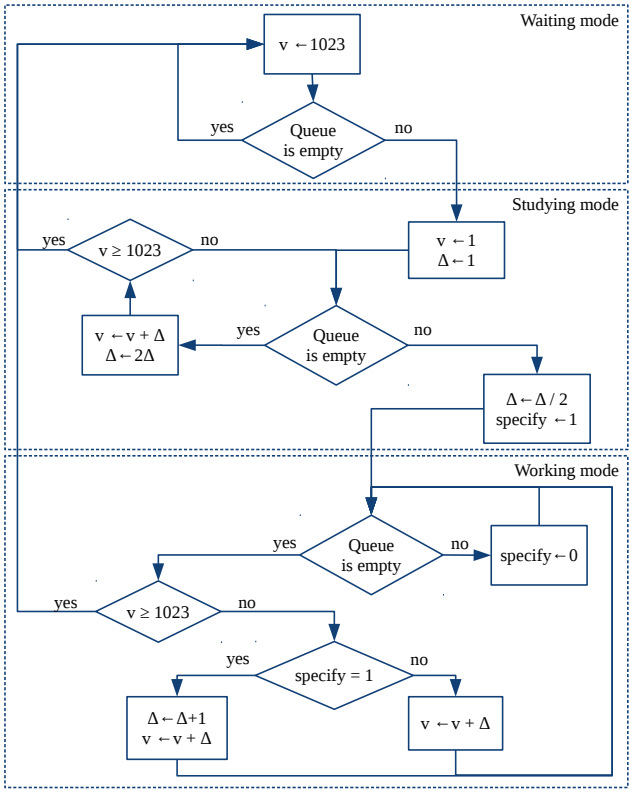

Figure 3. Algorithm “Up”

Initially, the AP works in the *waiting* mode. In this mode, the AP maintains the maximum threshold.

When AuthReps appear in the queue, the AP sets threshold to 1 and switches to the *studying* mode. In the studying mode, each BI the AP increases the threshold value by Δ, which is initially equal to 1. The AP tries to find an optimum Δ to authenticate each BI as many STAs as possible. For this purpose, the AP doubles Δ each BI until the AuthRep queue becomes nonempty by the end of some BI. Then the AP understands that the current value Δ is too large, and the previous one is less than or equal to the optimum one. In this case, the AP halves Δ and passes to the working mode.

In the *working* mode, the AP increases the threshold value by Δ if the queue is empty, otherwise the threshold value does not change. Also, the AP tries to find Δ more precisely, increasing its value by one until AuthReps remain in the queue at the end of BIs again. The AP switches back to the waiting mode when the threshold reaches its maximal value.

The second algorithm, called "Down", has the same waiting and working modes, but the studying mode is arranged differently.

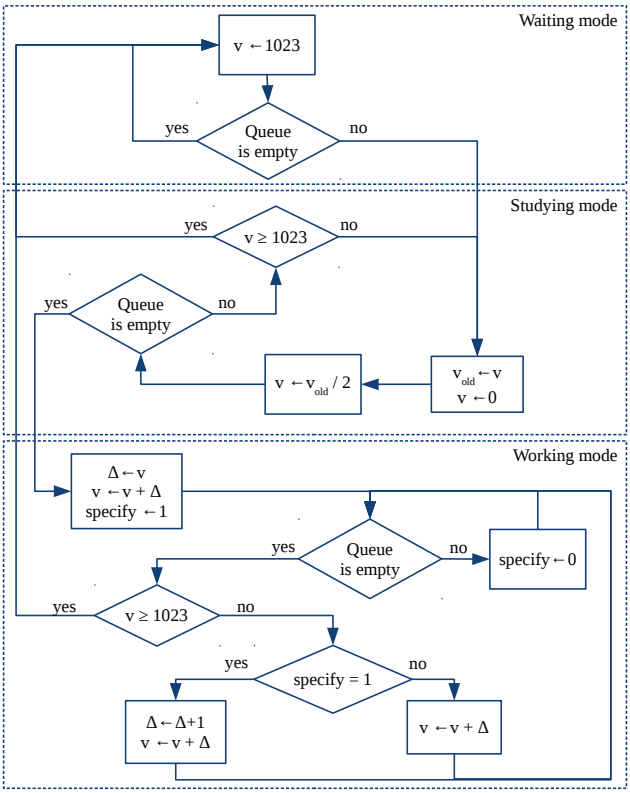

Figure 4. Algorithm "Down"

At the beginning of the *studying* mode, the threshold is maximum. Every time the queue becomes nonempty at the end of a BI, the AP sets the threshold value to 0 and does not change it until the queue becomes empty. Then the AP sets the threshold value to half of the previous nonzero value. If after reduction of the threshold, the queue remains empty, the AP sets Δ to the current threshold value and passes to the working mode. As in algorithm "Up", the AP switches back to the waiting mode when the threshold becomes 1023.

The algorithm block diagrams are shown in Fig. 3 and 4. Transitions between the rectangular blocks are made at sending a beacon. Current threshold value is marked as v and v_{old} is a variable used to temporarily store the previous threshold value. The assignment operator is denoted as $\leftarrow$ and "specify" is the flag that indicates whether the AP should try to find optimal Δ more precisely.

V. Experimental Results

To evaluate efficiency of developed algorithms, we run a number of experiments in the scenario described in Section II. The simulation has been done with the ns-3 simulator [9].

All STAs are placed randomly around the AP in a circle with the radius of $100\,\mathrm{m}$. The AP and STAs transmit in $1\,\mathrm{MHz}$-wide channel using the most reliable modulation and coding scheme MCS0. The number of interfering STAs is 20, and they transmit 100-byte data frames in the saturated mode. The BI duration and $AuthenticationRequestTimeout$ equal $512\,\mathrm{ms}$.

Fig. 5 presents numerical results averaged over 10 runs.

Specifically, we can see how the average authentication time of a group of STAs depends on the the number of STAs in this group for various threshold selection algorithms. From the Figure we can see that no fixed value of Δ can provide high efficiency for an arbitrary number of STAs. For example, if the number of STAs is about 4000, the best choice among all other choices with the constant Δ is $\Delta = 30$. However for a small number of STAs, e.g. 500 or less, selection $\Delta = 100$ provides manifold decrease of the average authentication time.

We also have designed an "Oracle" algorithm that knows the exact number of STAs which are going to authenticate and selects Δ in such a way that each BI the average number of STAs that successfully authenticate is maximal, specifically, $\Delta = \frac{k_{opt}}{N}1023$. As one can see, the average authentication time grows linearly for developed algorithms and for "Oracle", but grows much faster for algorithms with a specified Δ, regardless of the Δ value.

Comparing algorithm "Up" and "Down", we can conclude that algorithm "Down" is the best of all the presented realizable algorithms almost for any number of STAs. Such a result is explained by the fact that a considerable part of STAs authenticate in the course of waiting and studying modes while the threshold value is large. If the number of STAs is small, the algorithm "Down" authenticates them maintaining a large threshold value, and if there are many STAs, it quickly finds the necessary Δ. At the same time, more STAs authenticate during the studying mode of algorithm "Down" than during the studying mode of algorithm "Up". That is why algorithm "Down" outperforms the algorithm "Up". Even though algorithm "Down" is an heuristic approach to control the authentication threshold, the authentication time for it exceeds the authentication time for the "Oracle" algorithm only by 20%.

The algorithm "Up" still should not be dismissed, because although it is slightly less efficient than "Down", it consumes channel resources in a less aggressive way. It could be kept as a variant to use in a loaded network, where the authentication time is less critical than the throughput available for already associated STAs.

VI. Conclusion

We have studied the Centralized Authentication Control protocol and its usage to hasten connection set-up in IEEE 802.11ah networks. The efficiency of this protocol requires the correct choice of the authentication threshold, however, the standard does not specify an authentication threshold management algorithm. We have shown that during connection set-up the authentication threshold shall be gradually increased from its minimal to its maximal value, the increment depending on the number of connecting stations. We have designed two algorithms that adaptively select the increment for authentication

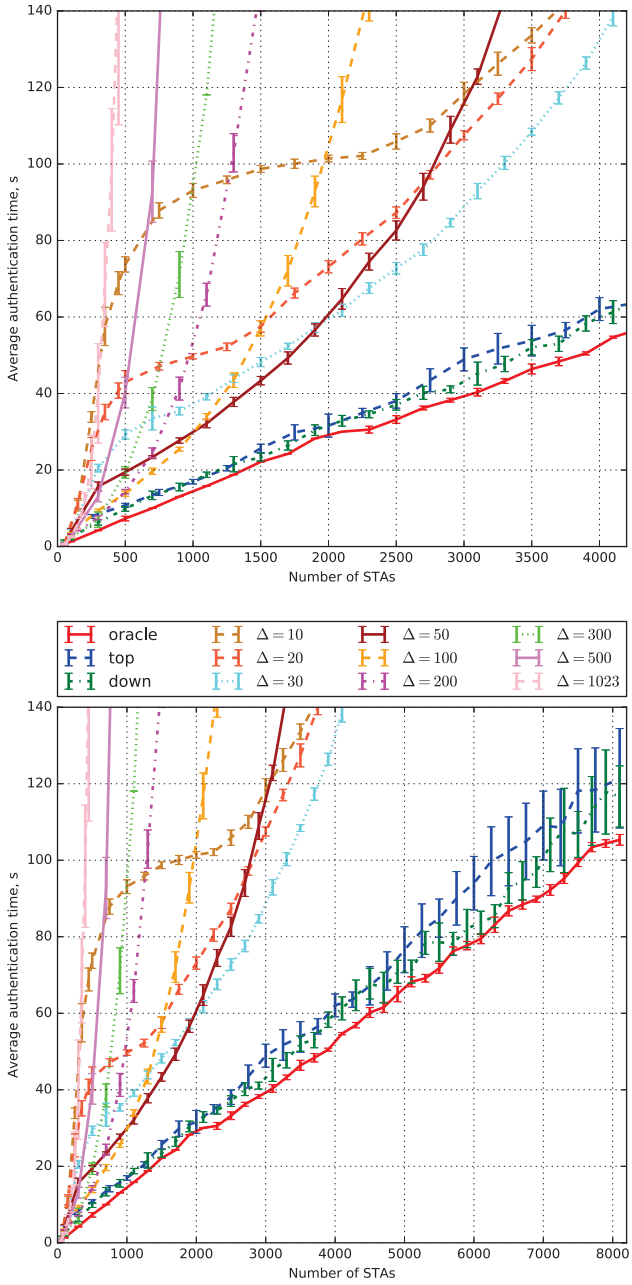

Figure 5. Dependency of the average authentication time on the number of STAs

threshold and have shown that, firstly, they outperform an existing solution that uses a fixed increment. Secondly, we have shown that for the developed algorithms the authentication time just slightly exceeds the time for an "Oracle" algorithm that has a priori knowledge about connecting stations and uses this knowledge to set the authentication threshold in such a way that guarantees the fastest authentication.

We have implemented our algorithms in ns-3 simulator and have modeled a scenario, when the network is used not only by connecting stations but also by stations that generate heavy background traffic. According to experimental results our algorithms are efficient even in such a complex scenario.

References

[1] D. Singh, G. Tripathi, and A. J. Jara, "A survey of internet-of-things: Future vision, architecture, challenges and services," in *Internet of things (WF-IoT), 2014 IEEE world forum on*. IEEE, 2014, pp. 287–292.

[2] M. P. Pawlowski, A. J. Jara, and M. J. Ogorzalek, "Extending extensible authentication protocol over ieee 802.15. 4 networks." in *IMIS*, 2014, pp. 340–345.

[3] *IEEE P802.11ai™/D4.0 Draft Standard for Information Technology—Telecommunications and information exchange between systems— Local and metropolitan area networks— Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment to IEEE P802.11-REVmc™/D3.2: Fast Initial Link Setup*, February 2015.

[4] E. Khorov, A. Lyakhov, A. Krotov, and A. Guschin, "A Survey on IEEE 802.11ah: An Enabling Networking Technology for Smart Cities," *Computer Communications*, vol. 58, pp. 53–69, 2015.

[5] D. Bankov, E. Khorov, and A. Lyakhov, "The Study of the Distributed Control Method to Hasten Link Set-up in IEEE 802.11ah Networks," in *Problems of Redundancy in Information and Control Systems (REDUNDANCY), 2016 XV International Symposium*. IEEE, 2016, pp. 13–17.

[6] H. Wang. (2012) Supporting Authentication/Association for Large Number of Stations. [Online]. Available: http://mentor.ieee.org/802.11/dcn/12/11-12-0112-04-00ah-supporting-of-the-authentication-association-for-large-number-of-stations.pptx

[7] P. Sthapit, S. Subedi, G.-R. Kwon, and J.-Y. Pyun, "Performance Analysis of Association Procedure in IEEE 802.11ah," in *ICSNC 2015 : The Tenth International Conference on Systems and Networks Communications*. IARIA, 2015, p. 4.

[8] L. Tian, S. Deronne, S. Latré, and J. Famaey, "Implementation and Validation of an IEEE 802.11ah Module for ns-3," in *Proceedings of the Workshop on ns-3*. ACM, 2016, pp. 49–56.

[9] The ns-3 Network Simulator. [Online]. Available: http://www.nsnam.org/

[10] L. Tian, J. Famaey, and S. Latré, "Evaluation of the IEEE 802.11ah Restricted Access Window Mechanism for Dense IoT Networks," in *IEEE 17th International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM), Accepted*. IEEE, 2016.

[11] D. Bankov, E. Khorov, and A. Lyakhov, "The Study of the Centralized Control Method to Hasten Link Set-up in IEEE 802.11ah Networks," in *European Wireless 2015; 21th European Wireless Conference; Proceedings of*. VDE, 2015, pp. 1–6.

[12] N. Shahin, L. Tann, and Y.-T. Kim, "Enhanced registration procedure with nav for mitigated contentions in m2m communications," in *Network Operations and Management Symposium (APNOMS), 2016 18th Asia-Pacific*. IEEE, 2016, pp. 1–6.