Available online at www.sciencedirect.com

ScienceDirect

Procedia Technology 17 (2014) 705 – 712

Procedia
Technology

Conference on Electronics, Telecommunications and Computers – CETC 2013

FairWLAN - IP level QoS mechanism for large Wireless LANs

Simão Silva[a,b], Ricardo Lopes Pereira[a,b], Rui Valadas[a,c]

[a] *Instituto Superior Técnico-Universidade de Lisboa, Lisboa, Portugal*
[b] *INESC-ID, Lisboa, Portugal*
[c] *Instituto de Telecomunicações, Lisboa, Portugal*

Abstract

Without traffic classification, WiFi access points try to distribute instantly available bandwidth equally among the several stations. Stations which perform sporadic use of the network, such as those used for browsing the WWW, will have to share bandwidth when they need it, even though they consume no resources for most of the time. We believe that a fairer method for resource allocation should take into account the usage history, both recent and historical, of each station. Also, in order to maximise the global capacity of a WiFi network, radio conditions for each station may be taken into account.

In this paper, we present FairWLAN, a QoS management system for large Wireless LANs, that retrieves information on each station associated to each access point to drive the configuration of the QoS mechanisms on the upstream router.

Peer-review under responsibility of ISEL – Instituto Superior de Engenharia de Lisboa, Lisbon, PORTUGAL.

Keywords: QoS, WLAN, SNMP, campus WLAN, operator WLAN

1. Introduction

Nowadays, wireless access is one of the easiest and most convenient ways available to use a network. However, by using a shared medium (radio channel) with limited capacity, the wireless link is often the bottleneck in the data transmission path [1]. Traffic prioritisation in Wireless Local Area Network (WLAN) is limited by the capability to classify different traffic flow with fine granularity. Many modern Access Points (AP) try to combat this problem by implementing basic Quality of Service (QoS) mechanisms that strive to share available capacity equally among the several stations (for the same priority). The fairness of such solutions is limited, as knowledge of previous interactions is not kept: e.g. a user who previously downloaded a large amount of data will be awarded the same share of bandwidth as another who has downloaded nothing yet.

In this paper we proposed FairWLAN, a novel, centralised QoS management system, which uses performance and historical traffic statistics for each station associated with each AP on a campus WLAN to configure QoS mechanisms on the upstream router. This solution provides fair sharing of WLAN capacity over extended periods of time.

E-mail address: simao.silva@ist.utl.pt

Peer-review under responsibility of ISEL – Instituto Superior de Engenharia de Lisboa, Lisbon, PORTUGAL.
doi:10.1016/j.protcy.2014.10.272

In the next Section we provide an overview of FairWLAN. Section 3 details FairWLAN's architecture, that is experimentally validated in Section 4. Section 6 presents the conclusions.

2. FairWLAN

In a campus WLAN, all APs are connected to the same router using an ethernet Local Area Network (LAN). At the IP layer (L3), all the stations are on the same subnetwork as the router, as depicted in Figure 1. Through Simple Network Management Protocol (SNMP) requests, it is possible to retrieve information about the APs and associated stations. Based on this information it is possible to act on the router in order to improve the QoS of some users at the expense of restricting traffic to stations with heavier traffic history.

Traffic throttling is performed on individual stations on each AP. Throttling will also result in a decrease of collisions at the wireless medium, meaning that less packets will be dropped. FairWLAN periodically reconfigures the QoS parameters for each station using information recently acquired from the APs.

The goal of FairWLAN is the improvement of the perceived QoS for most users. This is done by assessing different metrics, retrieved from the APs for each station. The intended equilibrium between stations is accomplished by imposing bandwidth limits to misbehaving stations. In order to achieve this proposed level of control, the FairWLAN must fulfil the following requisites:

- **Agnostic to traffic** All users must be treated equally. For this equality to be achieved, FairWLAN must be completely independent from the type of applications used by the user or its purpose. The only information that matters is the amount of bandwidth that they are using. A user which has used little of the network in the past will experience better QoS than another which has used the network intensively before, regardless of the applications used by each one.
- **Scalability** In today's *Campus* networks it is normal to have a large number of APs connected at any time. It is therefore important for the proposed solution to be capable of handling large amounts of data about the users connected to WLAN with a very low delay.
- **Adaptability** FairWLAN needs to take into account the changing circumstances of the *campus* network, which means that it needs to adapt to various number of APs and various network topologies.
- **Interoperability** FairWLAN must be capable of accessing information from different vendors, taking into consideration the way different vendors make available the required information. Information like the amount of packets that are being dropped, the amount of bandwidth that is being used or simply the state of the association of the station to the AP.

3. Architecture

In order to fulfil the requisites, FairWLAN must be capable of retrieving information, independently of the number of APs present on the campus network. To achieve this goal, we designed a system composed by multiple modules, each one providing a specific service. Modules communicate in order to exchange the results of their computations. The proposed modules are the Collector, Recorder and Controller. The Collector is the module responsible for gathering station association related information, such the data rate of a specific client at PHYsical Layer (PHY), MAC and IP address, the number of octets exchanged with the station and MAC Service Data Unit (MSDU) retries and fails. With this information the collector module is capable of calculating the current amount of bandwidth being used and relate it to the station's MAC address. That information is then transmitted to the Recorder. The Recorder is the module responsible for storing the retrieved data and the processed information. It is also capable of performing several accounting operations, such as calculating the amount of data downloaded and uploaded by a station to the network during a time period, and then relay that information to the controller. The Controller is the module responsible for deciding and imposing limits to the various stations. Those limits are calculated using the information gathered by the Collector and stored by the Recorder.

There are two possible deployment configurations supported by this solution. A single machine configuration where all the modules are run in the same physical machine, a multi-machine configuration where the distributed capability of this architecture is used. The former configuration uses less hardware, but in other hand is unable to control a high number of APs. The latter solution is more scalable, as each module has its one dedicated machine, which means that it is possible to use several Collectors. Those collectors should be deployed proportionally to the number of APs present on the network. The fact that each module works independently from the others, allows for the solution to be interoperable with different APs, by having different collectors working in the same deployment, each responsible for a different type of APs. Each of the FairWLAN's three modules, will be detailed in the following subsections.

3.1. Collector

The Collector module is responsible for collecting metrics and information from the APs. With those metrics and information it is possible to characterise the association between stations and AP. The Collector is divided into four components: Equipment Interface (EI), Retriever, Inter-Process Communication (IPC) component and Logger.

The EI is the component responsible for the direct interaction with APs. Through a simple SNMP request is possible to retrieve Object IDentifiers (OID)s that contain the data needed. The EI component provides a Application Programming Interface (API) to the Retriever component. It is through this API that the retriever module is able to access the information gathered from the APs. Basically, EI component is an interface that masks the details of communicating with the network devices to retrieve data, enabling the collector scalability and adaptability. In our currently implementation, we have a single EI, that provides SNMP communication.

The Retriever component's main function is to process the information about the various APs and their associations. The information used is the maximum data rate at PHY, the number of octets send and received allowing the calculation of the occupied bandwidth, the number of MSDU fails and retries, IP and MAC address from every station and the number of station associated to the AP.

The IPC component delivers all data processed by the Collector, ensuring the communication between the Collector and the Recorder. The IPC component accesses the information gathered by the retriever through a queue. The communication between Collector and Recorder is accomplish through TCP.

The Logger component is also very simple in design. It is solely responsible for storing into a log file the information that the Retriever Module produces, for debug purposes.

The Collector module uses multi-threading, which permits simplicity in design, while being able to support a larger number of APs than a sequential solution. This means that multiple retrievers can exist, with each one being responsible for a single AP. Those multiple Retrievers produce data to a First-in-First-out (FIFO) queue. The queue is then processed by the IPC thread that then sends the information in queue to the Recorder. Figure 2 illustrates the Collector architecture in conjunction with the other two modules of FairWLAN.

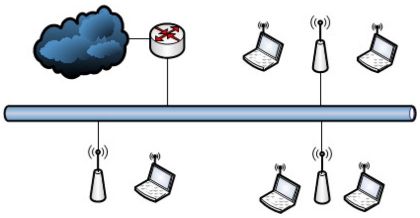


Fig. 1: Campus network representation

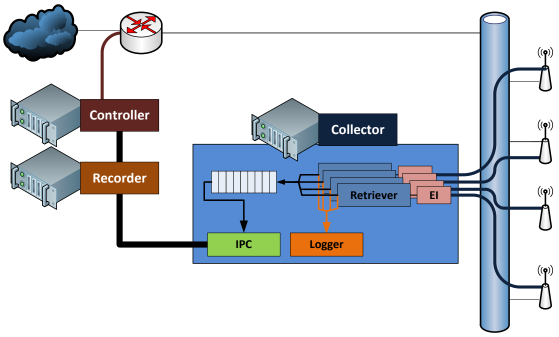


Fig. 2: Collector - Architecture and components' detail

3.2. Recorder

This module is responsible for keeping all relevant information for each AP. That information is gathered by the Collector module, described on Section 3.1. For each station, the Recorder keeps its MAC and IP addresses, maximum data rate at PHY level, occupied bandwidth upstream and downstream, if there is any packet loss and to which APs it is connected to. A station can only be connected to one AP at a given time. This module is also able to associate the information about the station to the respective user (the owner of the device). The Recorder module consists of five components: User Information Retrieval (UIR), Long-Term User Accounting information (LTUA), Accounting, IPC and Logger.

The UIR is capable of fetching all the information required to relate an user with the stations that he is using from the authentication system implemented on the *campus* network. The Instituto Superior Técnico wireless network authentication system uses the FreeRadius server. We use it to associate a user's identifier to a station's MAC address by simply accessing the FreeRadius log file.

The LTUA is responsible for storing the user accounting information in a more permanent way than the application's memory at runtime. The solution chosen was to store the information in a MySQL database server[1].

The Accounting component is the one responsible for keeping track of the traffic consumed by each station, in a transient fashion (last 60 minutes). It is also capable of relating the station identifier with the user identifier, thus accounting traffic for all the stations used by the same user, allowing for the identification of intensive users, independently of the stations being used at the moment. This component gathers various information about each station: the IP and MAC addresses at that time instant or in the past, the first and last time the station was seen in the network, which APs it as been associated to, the maximum data rate at PHY level at the moment, the occupied bandwidth upstream and downstream for that time instant, if there is packet loss and finally to which AP the station is connected to.

The IPC component is responsible for ensuring the communication of this module with the other modules. The Recorder IPC component is more complex than the Collector IPC component because it has to simultaneously communicate with the controller and with multiple Collector instances. In order to do this, the Recorder IPC component is divided into two smaller parts: the Controller Handler assures the communication with the single Controller; the Collector Handler handle communication with multiple Collectors. All communication between Recorder and Collector are accomplished through TCP. The Collector Handler is more complex as it must be capable of supporting multiple Collectors. It implements multiple threads which feed data from the Collectors into a FIFO queue. This queue is consumed by the Accounting component.

The Logger component is simple in design because his only task is to log problems with the other components. The other components report their occurrences and the Logger component logs them to file for debug.

Figure 4 illustrates the Recorder modular design and also its role in FairWLAN's architecture. This module uses a multi-thread design which allows it to accommodate multiple Collectors. Every Collector delivers information about APs and associated stations. This information is then correlated to a network user using the log from the authentication system installed at the *campus* network. The only information important to this process present in the logs is the relation between the user and the MAC address of the stations where he is currently authenticated. With this relation between user and stations it is now possible to account the network usage by user.

3.3. Controller

The Controller module is the main module in the proposed architecture, as it collects the information stored in the Recorder module and according the predefined rules triggers actions on the router. The controller is capable of generating rules specific to each station. This module is also able to calculate the bandwidth that each equipment has a right to without compromising the rights of others. In order to provide the functionality described, this module implements the following components: Enforcer, Coordinator, IPC, Logger, Stats and Ruler.

The Enforcer is the component responsible for applying rules to the *campus* network router. This router, as mentioned before, must be the router upstream from all the APs. In our current implementation, the router must be a Linux

[1] `http://dev.mysql.com` (retrieved in September 2013)

system as the Enforcer only generates instruction for this operating system [2]. The Enforcer converting rules created by the Ruler into system commands. Through those simple system commands, the Enforcer is capable of configuring a class based scheduling mechanisms in real-time. The FairWLAN enforcer assigns a class to each station. Each class has a set of parameters, being the most important one the maximum authorised bandwidth. Those classes are define through the *tc* command, available in Linux systems.

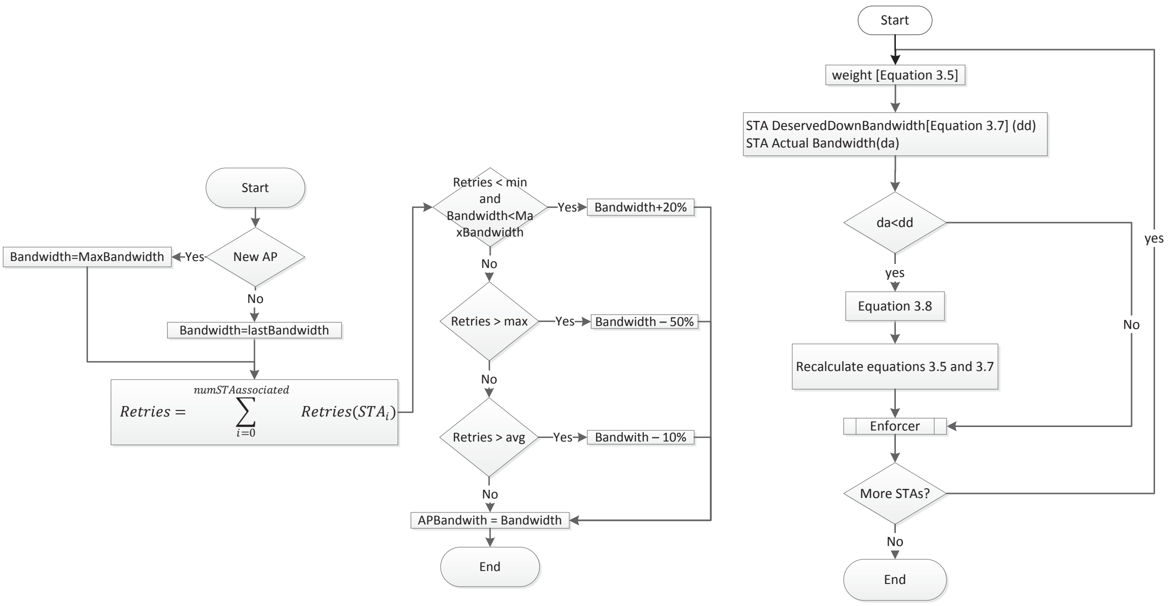
The Coordinator is the component responsible for controlling the other components. This component also has the responsibility of making the required information available to the other components.

The IPC component is similar to the IPC components of the other modules. In this case the component assures the communication with the Recorder. This IPC component only has the capability to communicate with a single Recorder.

The Logger component's main function is to store all the rules applied to the router, for debugging purposes. It is also responsible for storing all data regarding the inner workings of the module.

The Stats component not only generates all the statistics about the stations and APs but also makes them available to others applications. This means that it is possible to use an external, independent application to present statistics.

The Ruler is the component capable of generating rules specific to each station at a regular time interval. Those rules specify the bandwidth to which the station is entitled, guaranteeing that all other users have a minimum of bandwidth. The rules, after being created, are passed along to the enforcer, that applies them to the router. The rules are created according to a method whose goal is to divide the wireless medium capacity in the fairest way possible. This division is made at the IP level. The maximum IP bandwidth capacity of each AP is important as it is at this level that the rules are applied. The maximum bandwidth supported by 802.11a and 802.11g is 54 Mbps at the PHY level, which means that the maximum bandwidth expected at the third level of the IP stack is 27 Mbps when using TCP [3]. In order to calculate a fair distribution of resources it is necessary to take into consideration not only the variables already mentioned but also the past usage of the medium. In order to calculate that fair division, the flowchart present in Figure 3b is used. It adjusts the total bandwidth to make available at each AP according to the level of retransmissions occuring at that AP.



(a) Flowchart for AP Bandwidth available

(b) Flowchart for station Bandwidth

Fig. 3: Controller - Flowcharts

In order to determine the bandwidth to assign to each station, we start by calculating the part (*weight*) of the wireless medium that a station as the right to use, using Equation 1, which takes into account the number of MSDU retries (STARetries). The bandwidth occupation history is that calculated using Equation 2 ($STAHistory$). The weight given to the traffic history and retransmissions is defined through β, varying between 0 and 1. The second fraction of this equation accounts for the station's ability to sense the wireless medium, where STAPHYRates represents the current physical rate of the station and APmaxPHYRate symbolizes the maximum physical rate supported by the AP.

$$weight = \frac{1}{\beta \times (STARetries + 1) + (1 - \beta) \times (STAHistory)} \times \frac{STAPHYRates}{APMaxPHYRate} \tag{1}$$

Equation 2 is used to calculate the component known as $STAHistory$. This component is a weighted moving average using the last bandwidth measurement. The value of α can be changed to account for a longer or shorter analysis period.

$$STAHistory = \alpha \times DownAveBandwidth + (1 - \alpha) \times actualDownBandwidth \tag{2}$$

Equation 3 is the one that determines the downstream bandwidth that the station is entitled to use. This equation calculates the ratio of *APbandwidth* that can be occupied by the station minus the amount of bandwidth that is already being used to upload data by that station.

$$DeservedDownBandwidth = \frac{weight}{\sum weight} \times APbandwidth - STAupBandwidth \tag{3}$$

The available bandwidth (*APbandwidth*) is determined according to the flowchart in Figure 3a. The AP available bandwidth is calculated taking into account the total number of MSDU retries.

Equation 3 only takes into account the amount of bandwidth that each station has the right to use. It is also necessary to determinate if the station is expected to use the totality of the assigned bandwidth. Basically, if the current bandwidth of a station is less than the assigned bandwidth (*DeservedDownBandwidth*), the station is not expected to make use of the assigned bandwidth. The bandwidth expected to remain unused can be redistributed by the other station which are expected to use it. However, in order to guarantee that, if the station wants to start using more bandwidth between iterations, it will be able to do so, a percentage ($bookingfee$) of the bandwidth will not be redistributed, but reserved for the station, according to Equation 4, which recalculates the bandwidth available at each AP. In it, the variable *dd* represent the *DeservedDownBandwidth* of the station and the variable *da* depicts the actual occupied bandwidth.

$$newAPBandwidth = newAPBandwidth - bookingfee \times (dd - da) + da \tag{4}$$

Figure 5 illustrates the Controller design and also its role in the global architecture. This module also follows a multi-thread design. One of its most important components is the Ruler, that is responsible for the creation of rules. Each one of these rules, specific for each station, will be applied to the router by the Enforcer. Although the router in this work was linux based some enterprise routers are implemented with other systems so the Enforcer is in fact the component that are capable of translating the rules defined by the ruler to the QoS system implemented on the network.

4. Experimental Evaluation

In order to validate FairWLAN, a test-bed was implemented using GNU/Linux boxes. This test-bed consists of a computer acting as router, an AP and three equal stations, at the same distance from the AP. Figure 6a shows what happens when three stations try to achieve a downstream rate of 20Mbps (UDP using iperf) in a standard WLAN implementation. Stations start their downloads 20 seconds apart. When a new station starts downloading, it has to share the wireless medium with the other stations already downloading. The capacity that each station makes use of does not depend on its usage history. Only their radio conditions define the amount of traffic each station receives, explaining the uneven bandwidths.

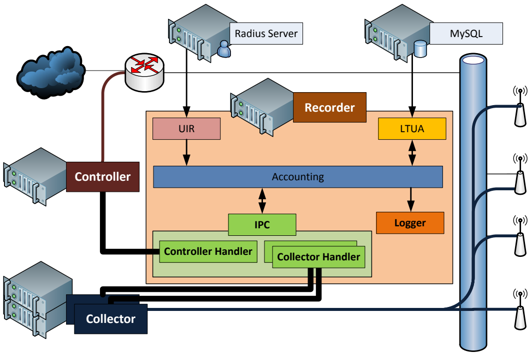


Fig. 4: Recorder - Architecture and components' detail

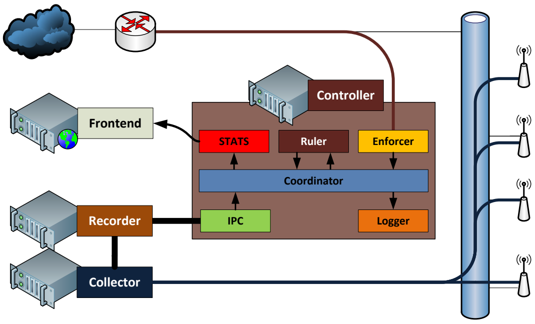


Fig. 5: Controller - Architecture and components' detail

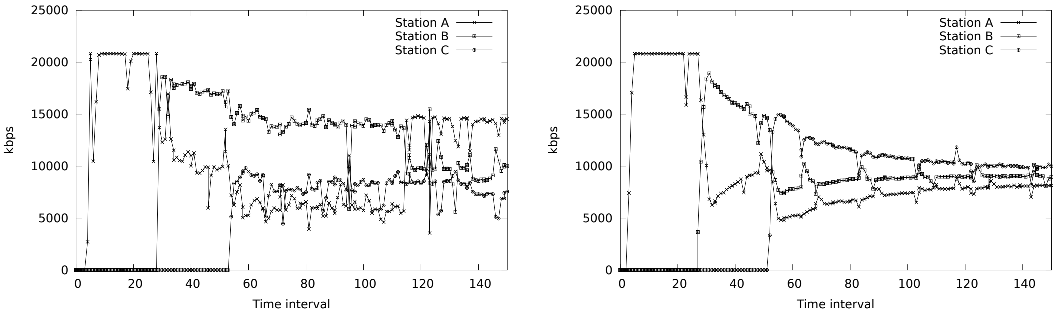
Figure 6b shows what happens in the same scenario using FairWLAN. When a new station starts downloading, the other stations that were already downloading see their bandwidth reduced, allowing the allocation of more bandwidth to the new station. As time goes by, the amount of traffic downloaded by each station starts to approximate, therefore the bandwidth assigned to each station by FairWLAN converges to the same value.

By providing more capacity to stations which have received less traffic, FairWLAN allows sporadic users, such as WWW browsers, to be less impacted by heavy users, such as Peer-to-Peer file sharing downloads. Other tests showed that webpage download time in the presence of heavy downloaders was reduced. We have also validated FairWLAN's Collector module's performance using a large WLAN with almost a thousand stations and about 50 APs [4].

5. Related work

Earlier proposals have also used information on the quality of the wireless channel for each station in order to achieve better WLAN resource distribution. Hierarchical Token Bucket (HTB) is a Class Based Scheduling Discipline very similar to Class Based Queueing (CBQ) and it is also very precise in wired networks [5,6] and in WLAN [7,8]. However, HTB is unable to cope with different channel quality for each station, cause be factors such as their distance or mobility.

There are two other proposals to solve this HTB handicap [9]. These are referred to as Wireless Hierarchical Token Bucket (WHTB) and Time-based Hierarchical Token Bucket (TWHTB). WHTB works by sensing the channel quality between an AP and a particular station [9]. TWHTB makes its analysis of the channel based on the time needed to



(a) Standard 802.11 scenario

(b) 802.11 scenario with proposed solution

Fig. 6: Tests results

successfully transmit (including retransmissions and reception of the ACK message). FairWLAN's behaviour is closer to that of WHTB. FairWLAN detects the degradation of the wireless channel between the AP and each station by the retransmission count from the AP.

As far as we can tell, FairWLAN is unique in that it takes historical and short term traffic information into account, providing a different concept of fairness. More traditional approaches to this problem tend to search for an instant fairness, this means that Weighted Fair Queuing (WFQ) allocation type schemes only take into consideration traffic flows that are currently active, achieving a proportional fairness. These types of schemas do not take into consideration the previous behaviour of the traffic flows [10].

FairWLAN also differs from the described works in that it controls the upstream router instead of being implemented in the APs, thus being able to continue to manage each station even as they roam among APs.

6. Conclusion

In this paper we proposed FairWLAN, a centralised, IP level QoS management system for campus WLAN which provides a fair share bandwidth to each station by weighing current and historical traffic data as well as physical layer (station data rate) and data link (transmission retries) information. Our solution maximises capacity use at each AP while being fair over long periods among stations at each AP. All data necessary for the decisions is retrieved from the APs using SNMP. The solution also incorporates RADIUS information so that stations can be associated with users, preventing users with multiple devices from hording bandwidth.

FairWLAN uses a different concept of fairness. Instead of an instant equal distribution of bandwidth, it achieves an equal distribution of traffic over a long time. This enables a better Quality of Experience for intermittent users at the expense of lower quality of service for heavy users. In a large WLAN with hundreds of users, the cost paid by each heavy user will be little, and should not impact applications such as streaming media.

In the future we intend to deploy FairWLAN in a real campus WLAN.

Acknowledgements

This work was partially supported by national funds through FCT Fundação para a Ciência e a Tecnologia, under project PEst-OE/EEI/LA0021/2013.

References

[1] Keranidis, S., Korakis, T., Koutsopoulos, I., Tassiulas, L.. Contention and traffic load-aware association in IEEE 802.11 WLANs: algorithms and implementation. In: Modeling and optimization in mobile, ad hoc and wireless networks, 2011 international symposium on. 2011, p. 334 –341. doi:10.1109/WIOPT.2011.5930036.

[2] Hubert, B., Maxwell, G., van Mook, R., van Oosterhout, M., Schroeder, P.B., Spaans, J.. Linux advanced routing & traffic control howto. 2012. URL: http://www.tldp.org/HOWTO/Adv-Routing-HOWTO/.

[3] Bing, B.. Emerging techologies in Wireless LANs. Cambridge University Press; 2008.

[4] Silva, S., Pereira, R.L., Valadas, R.. Experimental evaluation of FairWLAN. In: The International Conference on Information Networking. 2014,.

[5] Balan, D., Potorac, D.. Linux HTB queuing discipline implementations. In: Networked digital technologies, 2009. (NDT 2009). first international conference on. 2009, p. 122 –126. doi:10.1109/NDT.2009.5272182.

[6] Ivancic, D., Hadjina, N., Basch, D.. Analysis of precision of the HTB packet scheduler. In: Applied electromagnetics and communications, 2005. ICECOM 2005. 18th international conference on. 2005, p. 1 –4. doi:10.1109/ICECOM.2005.204958.

[7] Grewal, J., DeDourek, J.. A framework for Quality of Service in Wireless networks. In: Communication Networks and Services Research Conference, 2005. Proceedings of the 3rd Annual. 2005, p. 231 – 236. doi:10.1109/CNSR.2005.7.

[8] Valenzuela, J., Monleon, A., San Esteban, I., Portoles, M., Sallent, O.. A Hierarchical Token Bucket algorithm to enhance QoS in IEEE 802.11: proposal, implementation and evaluation. In: Vehicular technology conference, 2004. (VTC2004-fall 2004) IEEE 60th; vol. 4. 2004, p. 2659 – 2662 Vol. 4. doi:10.1109/VETECF.2004.1400539.

[9] Garroppo, R., Giordano, S., Lucetti, S., Risi, G.. A comparison of HTB based channel-aware schedulers for 802.11 systems. In: Wireless Internet, 2005. proceedings. first international conference on. 2005, p. 2 – 9. doi:10.1109/WICON.2005.1.

[10] Minagawa, T., Ikegami, T.. Controlling user flows with RIO and WFQ. In: Communications and Information Technologies (ISCIT), 2010 International Symposium on. 2010, p. 87–92. doi:10.1109/ISCIT.2010.5664907.