Automated Traffic Class Prediction and Prioritization on 802.11 using Machine Learning

Christiaan Brand, Riaan Wolhuter
Department of Electrical and Electronic Engineering
University of Stellenbosch, South Africa
{*cbrand, wolhuter*}*@sun.ac.za*

***Abstract*—With the steady increase of Internet access rates (DSL, LTE, FTTH) and aging local access technologies (802.11) QoS problems are starting to manifest themselves on the inside of SOHo (Small Office / Home) networks. Bandwidth bottlenecks are slowly moving to the inside of our networks and we need a solution to solve this problem without investing in other access technologies. Substantial capital investment has been made into some of these (802.11 WiFi Access Points at airports and coffee shops) because of their availability in consumer computer equipment. Our solution should be able to solve the problems experienced when dealing with multiple traffic classes from the same node in a single cell. In this paper we will examine the impact of implementing a traffic class prediction module on a congested 802.11 network. A novel node prioritizing scheme for wireless networks operating in DCF mode will also be presented.**

I. Introduction

OVER the last decade the Internet irrevocably changed from a Technology Push, to a Market Pull environment [1]. Network technologies are struggling to keep up with the myriad of applications being developed and infused into the Internet fabric. In addition to "traditional" services such as e-mail, http and ftp, interactive media-related services such as gaming, VoIP and Video-over-IP have grown tremendously and are now all competing for the same resources these traditional applications once had exclusive access to. These applications all have different metrics in terms of minimization of packet loss, jitter and latency [2]. The proper management of bandwidth is very important in order to maximize gain in terms of user experience and profit margins [3]. Being able to differentiate between different services on a hardware level will allow systems to optimize for the unique metrics and specific requirements of that application. This will allow QoS levels to be dynamically configured where required. A user is no longer bound to using just one service at a time, which means that rapid changes in QoS levels need to be considered. Identifying and classifying traffic is at the center of this arena and some major advances in this field have been made over the last three years [4].

Historically, IP traffic on a network was identified by examining the port numbers on which the services were offered. The Internet Assigned Numbers Authority (IANA) assigns all "well known" ports from 0-1023 to specific protocols, and also keeps a register of application-to-port mappings for ports 1024-49151. This solution was a simple, yet effective one, where only the packet header required inspection. Over the last few years this approach was abandoned, since many applications do not have any IANA assigned ports.

A more recent approach was to turn to Machine Learning (ML) techniques (a subset of the wider Artificial Intelligence discipline) to help with IP traffic classification. Many papers on this subject have been published over the last few years [5][6][7]. The ML approach works by firstly identifying *features* by which unknown IP packets might be differentiated. Features are attributes calculated over many packets, and include statistics such as mean packet length, inter-packet arrival times and flow duration.

II. Problem Statement and Proposal

All the techniques in the previous section assumed that the packets to be classified, have already arrived at our Classifying Station (CS) or router. There are numerous reasons why we may want to do these classifications, but for the scope of our research QoS is the major one. Once incoming packets have been correctly classified, a QoS module should be queried to establish which one to schedule on the router egress port first. The previous scenario assumes that the bandwidth bottleneck lies at the local router's egress port - the connection to the Internet. The majority of users gain access to the Internet via a local multiple access medium (eg. WiFi, Wired Ethernet, WiMAX). Historically, with local network access speeds being multiples of our Internet bandwidth, the QoS problem really only started to manifest itself at the local router's (or gateway's) interconnection to the Internet. With the momentous leaps in Internet access technology (DSL, LTE, FTTH) [8] on the one hand, and traditional SOHo (Small Office / Home) networks still relying on aging 802.11b/g access technologies, QoS requirements are slowly moving to the inside of the network as well.

A typical SOHo network consists of several users all attempting to gain access to the same Internet link - but before access to this link can be negotiated they must first traverse their SOHo network; a fact ignored in many QoS solutions. The solution is not as straightforward as one might assume. In order for the QoS system on the gateway to prioritize a specific packet on the CSMA channel it will need to have some information about the contents of the packet to be received. There is no way for it to gain direct knowledge about the

contents, since any communication from the node (wishing to transmit the packet) to the QoS-enabled router is subject to the same issues as the full packet waiting to be transmitted.

In this paper, the authors will propose a novel technique, where the QoS manager attains knowledge about packets *to be transmitted* by an indirect means: Predicting the class of the next packet to be transmitted by each node, by using some *learned* knowledge. Note the distinction between classification, identification and *prediction*. Once the class is known this information will be passed to a QoS scheduler running on the network controller - in this paper this will be the WAP (Wireless Access Point). Our simulation network is shown in Figure 1. The aim of our work was to develop a technique that will make correct QoS decisions and also being able to implement this on a network without any changes (in hardware or software) to the clients. It is only necessary to update the WAP to support our model. This update consists of two distinct parts:

1) The ML algorithm predicting the next-packet-class to be transmitted by the stations.
2) The QoS module acting on commands received from the ML algorithm.

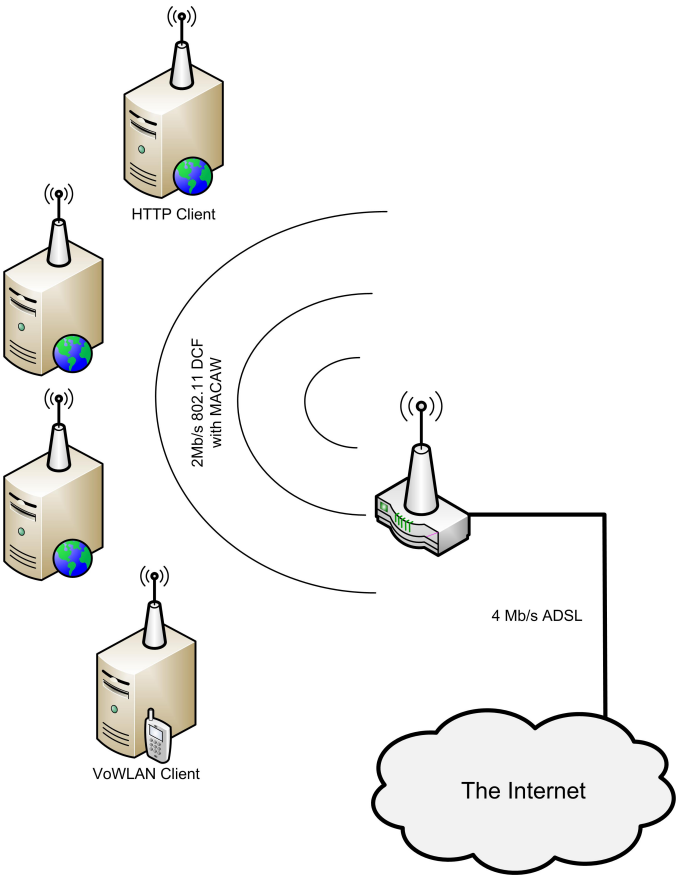

Fig. 1. Experimental simulation setup for QoS assignments.

III. Related Work

Using ML techniques for packet classification was first introduced in the context of intrusion- and denial of service detection [9]. Lately it has also been expanded to include QoS applications and form part of ISP obligations with respect to "Lawful Interception" of IP data [10].

The next section will briefly outline our novel ML Packet Prediction algorithm and the physical testbed used to verify the work.

A. ML Packet Prediction

As there are a multitude of different ML techniques available, we need to consider some criteria before attempting to find the approach most suited for our application:

- As this solution will be implemented on a CPE (Customer Premises Equipment) level, computationally intensive techniques have to be excluded.
- After making a prediction about the class of traffic a specific node might transmit, we will almost immediately (when that node indeed transmits the packet) know whether that prediction was correct. The algorithm should be able to use this information to retrain itself continuously.
- Due to privacy laws and the lack of CPE resources, the algorithm should only examine IP packet headers.

The data contained only in the packet headers might seem insufficient at first glance, but certain patterns start to emerge when we run these headers through a statistical preprocessor. Common *flow statistics processing* [11] on packet headers involve calculating mean packet inter-arrival time, mean packet length and flow duration. For this model we will only focus on packet lengths. For flow duration to be considered, we have to keep track of IP flows - for prediction we focus on individual packets. Packet inter-arrival times are skewed by lower layer multiple access links - the packet may be ready for transmission for a while before it is given a slot in which to transmit. We cannot assume that packet inter-arrival times (the time between packet arrivals from a certain node at the router) and packet-transmit-ready times (the time between packets are ready to be transmitted by the node) are equal. A moving average of the last 100 packets received from a specific node for each protocol class will be held.

A qualitative ML analysis showed that the last packet in a sequence of the same protocol class packets received from a host has a high probability of starting to differ in size from the rest of the packet stream. Practical experience suggests that applications send data to be transmitted over the Internet to the networking API and this data is then segmented into packets to be transmitted as soon as the channel allows. The characteristics of the last packet is often different from the rest of the serialized stream. As the packet lengths are received by our module, the system records the current protocol's class and adds the previous packet's length to the mean calculation for the current class. Before being added, however, the system first calculates the current mean for the protocol. If the current

mean, and the size being added differ more than 30 %, the packet is discarded as an anomaly. Slow migrations in packet sizes are thus permitted, but no single packet will be able to skew the results. We also considered the bimodal distributions of packet sizes on the Internet [12]: IP Packets are normally distributed around acknowledgments (40 bytes) and full packets (1500 bytes) because of the large scale use of Ethernet-based link layer technologies. By extending this observations to our model we included a second mean per protocol class per host. If the packet length differs more than 30 % compared to the current mean, a second mean is created. From here on, packets are added to the mean they most closely resemble. If they differ more than 30 % from both means, the packet is not considered. The prediction consists of examining the size of the current packet. It is matched to the mean of the protocol class it is closest to. It is clear that our packet size mean values resemble transient characteristics for the protocol class types. Figure 2 graphically depict the performance of our prediction algorithm sized to illustrate the importance based on number of transitions. The horizontally larger bars represent frequent transitions and is more significant for overall accuracy.

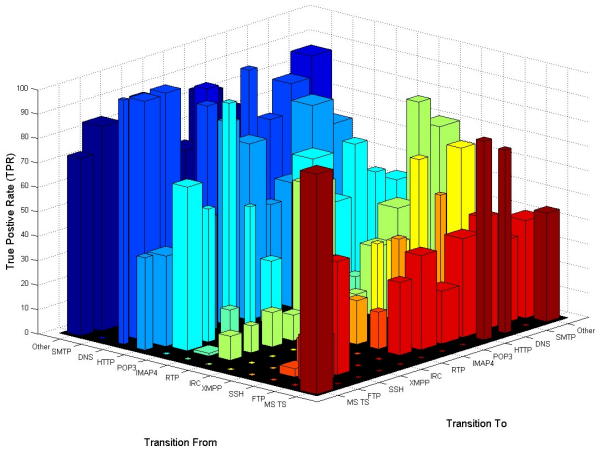

Fig. 2. Statistical Model TPR with Importance

B. QoS on 802.11 WiFi Networks

The original 802.11 [13] specification defines two distinct means of operation. PCF (Point Coordination Function) and DCF (Distributed Coordination Function). PCF allows the WAP to control all activity in a specific cell whereas DCF does not use any kind of central control. Support for QoS could most easily be realized using PCF, but the availability of this is scarce as PCF is optional in any standard 802.11 implementation. A specific QoS extension to standard WiFi was released in the form of 802.11e [14]. This implementation was too big for the most manufacturers and as such, a subset of 802.11e, called WMM (Wireless Multimedia Extensions) was born. WMM provides four traffic classes and priority management is realized through the use of EDCA (Enhanced Distributed Channel Access) [15]. For a WiFi network to support WMM the WAP and all the stations need to have this extensions as part of its MAC layer. For this reason we opted for our own design, that is able to provide the necessary QoS requirements by only adapting the transmission scheme at the WAP.

IV. Our proposal

In order to reap the benefits of our ML algorithm, a QoS prioritization scheme is needed. As mentioned in the previous section, we will build this on top of the existing 802.11 infrastructure as this is one of the most commonly used shared Internet access mediums in the SOHo environment. Once our ML module makes a prediction about a certain traffic class to be transmitted from a node, that information is sent to the QoS module. This module has been built upon MACAW (Multiple Access with Collision Avoidance for Wireless) which is already present in the majority of 802.11 wireless equipment. A minor modification to the WAPs MAC (Media Access Control) layer was necessary in order to facilitate our QoS priorities. It is important to note that no changes other than the enabling of MACAW is necessary at any of the clients.

A description of the QoS module will now be presented.

MACAW was originally implemented to solve the hidden terminal problems found in wireless networks. On a DCF (Distributed Coordination Function) wireless network with MACAW, each packet transmission is prefaced by a RTS (Request to Send) and CTS (Clear to Send) packet. In our system, if a certain node needs prioritizing, the number of active nodes on the network are counted and recorded as *n*. The number of packets sent by *other* nodes since you have last transmitted a packet, is given as i_N. If the *to-be-prioritized* node (*N*) has not sent a packet in *n*-packets, as can be seen by satisfying equation 1, a dummy CTS frame is sent from the network controller (WAP). The CTS frame will be set as to allow enough time for *N* to issue an RTS. The CTS frame's destination address will be set as that of *N*. This will cause all the stations in the vicinity of the WAP to hold off on transmitting data, since they assume *N* will send a data frame. *N* will simply mark this frame as invalid and discard it, since it did not issue a corresponding RTS. While all the other stations are backing off, *N* will have time to issue a *real* RTS since the channel is now idle. If it has no data to send, the dummy CTS will simply time out on all the stations, and the channel will again be available to any station. At this point, the local packet counter i_N is set to 0 (as if *N* has just transmitted a packet). This simulates a polled environment for stations with a higher priority and guarantees them a transmission slot at least once every cycle. A frame transmission summary can be seen in Table I.

$$\sum_{i=1}^{n} i_N \geq n \tag{1}$$

It is also quite clear that the WAP will roughly have *n* times as many frames to transmit as any of the stations. This

TABLE I
QOS MAC LAYER

Station	Frame Source	Direction	Frame Type	Description
Station 1 (HTTP)	Station 1	$\overset{WAP}{\rightarrow}$	RTS [x μs]	Station 1 sends an RTS frame to the AP to request an up link for x μs.
Station 1 (HTTP)	WAP	$\overset{station}{\leftarrow}$	CTS [y μs]	The WAP grants the RTS by issuing a CTS for y μs ($x - y$ = MACAW overhead).
Station 1 (HTTP)	Station 1	$\overset{WAP}{\rightarrow}$	Data Frame	The station transmits the data frame.
Station 1 (HTTP)	Station 1	$\overset{WAP}{\rightarrow}$	RTS [x μs]	Station 1 sends an RTS frame to the AP to request an up link for x μs.
Station 1 (HTTP)	WAP	$\overset{station}{\leftarrow}$	CTS [y μs]	The WAP grants the RTS by issuing a CTS for $y + c$ μs, manufacturing silence for c μs.
Station 1 (HTTP)	Station 1	$\overset{WAP}{\rightarrow}$	Data Frame	The station transmits the data frame.
Station 4 (VoIP)	WAP	$\overset{station}{\leftarrow}$	CTS [i μs]	This will force all stations to stay silent for i μs. Station 4 will ignore this frame.
Station 4 (VoIP)	Station 4	$\overset{WAP}{\rightarrow}$	RTS [x μs]	Station 4 sends an RTS frame to the AP in the manufactured silent slot.
Station 4 (VoIP)	WAP	$\overset{station}{\leftarrow}$	CTS [y μs]	The WAP grants the RTS by issuing a CTS for y μs.
Station 4 (VoIP)	Station 4	$\overset{WAP}{\rightarrow}$	Data Frame	The station transmits the data frame.

is due to the centralized nature of infrastructure mode WiFi and the symmetric traffic distribution of VoWLAN (Voice over Wireless LAN). The WAP contends for channel access is exactly the same way as stations do. In order to *favour* transmission from the WAP, our model allow the WAP to slightly over-estimate the *back off*-time it acknowledges in a CTS frame as a result of an RTS being issued. This allows the WAP sole access to the channel for this manufactured silent period on the network. This overestimation is only done in case the WAP has high priority frames that need transmission and as such does not cause silent intervals on the medium.

V. EVALUATION

For evaluation of our complete QoS offering, a version of our prediction module was implemented in OMNeT++ [16]. We implemented 4 nodes connected to an 802.11b WAP running at 2Mb/s as our SOHo network. A *server*-node connected to the SOHo network at 100Mb/s serves as our model of a uncongested link to the Internet. Our simulation starts off by establishing a VoWLAN connection from one of our nodes to a node on the Internet. At $t = 5$ seconds, three other nodes join in and starts uploading data to the Internet (via TCP). By examining Table II the reader can clearly see that this has an adverse effect on the VoWLAN session and that it becomes unusable in this state. At *t*=20 seconds, we enable our packet predictor's QoS module as described in section IV. The accuracy of the predictor was calculated beforehand for a standard network topology and averages at around 89 %. Within a few moments the jitter and packet loss of the VoWLAN session is restored to an acceptable state, without having any large negative impact on the unprioritized nodes. Figure 3 summarizes the packet roundtrip times and graphically displays lost packets. Jitter is calculated as $\frac{RT_{max} - RT_{min}}{RT_{avg}} \times 100$, where *RT* denotes the CTCP round trip times. From our simulation results it is clearly visible that our QoS technique enhances user experience and efficiency on the wireless network quite dramatically when coupled with our packet prediction algorithms.

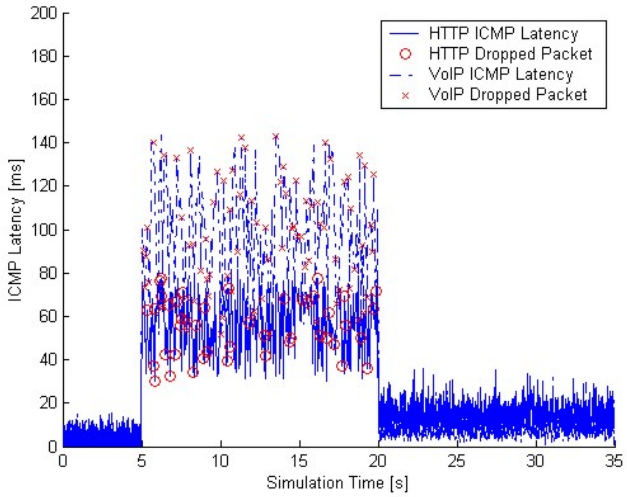

Fig. 3. WiFi Prioritization Results

VI. CONCLUSIONS AND FUTURE WORK

Users are no longer bound to using just one service at any given time: A Multitude of applications running on a single node is a complex QoS problem that is difficult to solve. The ability to predict packets on a network opens up the possibility of updating QoS levels as the user's bandwidth profile changes. This means that we can adapt our criteria for scheduling packets (QoS) even before the network is aware of the impending congestion due to a user suddenly changing his profile.

In this article we have shown that our packet predictor in conjunction with our own 802.11 scheduler solves the QoS problem in a unique way. Some more detailed simulation results involving throughput is necessary and currently in the implementation phase. Work is currently underway to implement all our modules on 802.11 hardware. All results given in this paper were obtained using network simulation

TABLE II
OMNeT++ SIMULATION RESULTS

***t* [s]**	**State**	**Node Type**	RT_{avg} **[ms]**	**Packet Loss [%]**	**Jitter [%]**
0 - 5	Only VoIP Stream currently active	HTTP	2.73	0	4.53
		VoIP	2.73	0	2.63
5 - 20	Three HTTP Uploads started	HTTP	77.81	16.0	175.63
		VoIP	144.17	51.06	178.75
> 20	QoS Module Activated	HTTP	14.2	0	9.2
		VoIP	9.2	0	5.8

through OMNeT++ and the INET framework. Providing QoS for other traffic types and network topologies are also currently being investigated.

REFERENCES

[1] D. Bollier, *When Push Comes to Pull: The New Economy and Culture of Networking Technology*, Aspen Institute Std., 2006.

[2] M. Marchese, *QoS over heterogeneous networks*, 1st ed. West Sussex, England: Wiley, 2007.

[3] Nortel Networks, *Introduction to Quality of Service*, Std., 2003.

[4] T. T. Nguyen and G. Armitage, "A survey of techniques for Internet traffic classification using machine learning," *IEEE Communications Society*, vol. 10, no. 4, pp. 56–76, 2008.

[5] S. Zander, T. Nguyen, and G. Armitage, "Automated traffic classification and application identification using machine learning," in *Proc. IEEE Conference on Local Computer Networks*, Sydney, Australia, Nov. 2005.

[6] A. McGregor, M. Hall, P. Lorier, and J. Brunskill, "Flow clustering using machine learning techniques," in *Proc. Passive and Active Measurement Workshop*, Antibes Juan-les-Pins, France, Apr. 2004.

[7] T. Auld, A. W. Moore, and S. F. Gull, "Bayesian neural networks for Internet traffic classification," *IEEE Transactions on Neural Networks*, vol. 18, no. 1, pp. 223–239, January 2007.

[8] P. Ödling, T. Magesacher, S. Höst, and P. Börjesson, Eds., *The Fourth Generation Broadband Concept*, IEEE Communications Magazine, January 2009.

[9] V. Paxson, "Bro: A system for detecting network intruders in real-time," *Computer Networks*, no. 31, pp. 2435–2463, 1999.

[10] F. Baker, B. Foster, and C. Sharp, "Cisco architecture for lawful intercept in IP networks," *Internet Engineering Task Force*, RFC 3924, 2004.

[11] K. Claffy, "Internet traffic characterisation," Master's thesis, University of California, USA, 1994.

[12] W. John and S. Tafvelin, "Analysis of Internet backbone traffic and header anomalies observed," Chalmers University of Technology, Göteborg, Sweden, Tech. Rep., 2008.

[13] A. S. Tanenbaum, *Computer Networks*, 4th ed. New Jersey, USA: Prentice-Hall, 2003.

[14] "Amendment: Medium access control quality of service enhancements," 802.11e. [Online]. Available: http://standards.ieee.org/getieee802/download/802.11e-2005.pdf

[15] J. Lee, W. Liao, J.-M. Chen, and H.-H. Lee, Eds., *A Practical QoS Solution to Voice over IP in IEEE 802.11 WLANs*, IEEE Communications Magazine, April 2009.

[16] "OMNeT++: Discrete event simulation system." [Online]. Available: http://www.omnetpp.org/

Christiaan Brand received his M.Sc Eng and B.Eng and is currently completing his PhD at the University of Stellenbosch. His research interests include wired- and wireless digital telecommunication systems and IP networks.