Incentivising fairness and policing nodes in WiFi

Ian Dangerfield, David Malone, Douglas J. Leith

***Abstract*—In this paper we propose a feedback-based scheme to penalise misbehaving nodes in an 802.11 network based on gains achieved due to their (mis)configuration. Only the access point (AP) is modified and the scheme requires no additional communication or cooperation from other nodes. We achieve this by failing to send MAC-level ACKs with a probability determined by the online feedback scheme. The scheme is designed so that it can incentivise nodes to configure themselves correctly and avoids the need to explicitly detect misbehaving nodes.**

I. Introduction

In current 802.11 networks, individual nodes have little incentive not to behave in a selfish manner. Further, the widely deployed 802.11e extensions provide the mechanisms needed for nodes to reconfigure themselves to gain a significant advantage. We consider situations where the node aims to benefit from its behaviour, creating the possibility of incentivising good behaviour[1]. The existing literature on misbehaving 802.11 nodes primarily focuses on determining the mechanism of cheating (e.g. shorter back-off counter), so that a specific action can then be taken to correct for the behaviour (e.g. [1], [2]). In this paper we take a different approach, exploiting the fact that an appropriate response to misbehaviour is to impose a penalty related to the gain made by the misbehaving node. We construct the penalty so that it can provide an incentive for the node to behave correctly.

Suppressing the generation of MAC ACKs has been suggested as a way to allocate bandwidth for traffic prioritisation in a network of well-behaved nodes [3]–[5]. Packets for which no ACK is generated are also dropped by the AP, just as if the packet was incorrectly received. We adapt the above idea of not sending ACKs and show how it can be used to control misbehaving nodes. The AP chooses not to send MAC-level ACKs with some probability, P_{ack}. If the sender follows the standard 802.11 CSMA protocol this causes the sender to back-off by increasing its MAC-level contention window. In addition, if the sender is not following these rules our scheme implicitly detects this allowing action to be taken. This approach has a number of advantages. Firstly, it operates on a fundamental part of 802.11 so it is difficult to counter: if we do not generate a MAC-layer ACK the sender knows that its packet has not been forwarded; even a misbehaving sender must retransmit to ensure delivery. Consequently, we do not need to make any assumptions about any possible transport layer. Secondly, since the scheme requires changes to the AP only, no modification needs to be made to the sender, which is an obvious advantage if selfish behaviour is suspected. Thirdly, this scheme provides a mechanism to penalise a node without resorting to jamming [6]. We focus on uplink traffic as downlink traffic can be policed using more traditional techniques by the AP.

This material is based upon work supported by Science Foundation Ireland under Grant No. 08/SRC/I1403 and 07/SK/I1216a.

[1]A node which simply wishes to disrupt the network can do so in any case, simply by transmitting packets and causing collisions.

II. Throughput Model

Since we propose dropping MAC ACKs in order to cause a node to back-off its contention window, it is important that we model the effects that both finite retries and back-off stages have on the network. The model used is a Bianchi-type model which takes into account finite retries R as well as m back-off stages, with $R \geq m$. For simplicity, we consider packets of constant size. Combining the approachs in [7], [8] we have the probability of attempting a transmission in a slot $\tau(p)$, with p being the probability that the transmission fails, can be expressed as $\tau(p) = E[\text{\# attempts}]/E[\text{\# slots}]$, with the i^{th} node having $\tau = \tau_i$ and $p = p_i$. Using the standard 802.11 back-off mechanism, whereby the window size is doubled at each of the first m back-off stages and where a total maximum of $R \geq m$ retries are permitted, yields

$$\tau(p) = \frac{\frac{1-p^R}{1-p}}{(\frac{1}{1-2p} - \frac{2^m p^m}{(1-p)(1-2p)} - \frac{2^m p^R}{1-p})\frac{W_0}{2} - \frac{1-p^R}{2(1-p)}}. \quad (1)$$

This can be solved for a particular p_i and τ_i by noting that

$$1 - p_i = (1 - P_{ack,i}) \prod_{j=1, j\neq i}^{n} (1 - \tau_j). \quad (2)$$

Here, $P_{ack,i}$ is the probability that the MAC-level ACK packet is dropped by our scheme. Then the throughput S can be calculated using $S = E[\text{packet length}]/E[\text{slot time}]$, yielding

$$S_i = \frac{P_{s_i} L}{\delta P_{sl} + T_f P_f + \sum_{i=1}^{n} T_s P_{s_i}}. \quad (3)$$

with P_{s_i} and T_s being the probability and duration of a successful transmission, P_{sl} and δ the probability and duration that no packet is transmitted and P_f and T_f the probability and duration that a transmission is attempted and fails. The idle time δ corresponds to one PHY slot. Specifically

$$P_{s_i} = (1 - P_{ack,i})\tau_i \prod_{j=1, j\neq i}^{n} (1 - \tau_j), \quad (4)$$

$$P_f = 1 - P_{sl} - \sum_{i=1}^{n} P_{s_i}, \qquad P_{sl} = \prod_{i=1}^{n} (1 - \tau_i). \quad (5)$$

Fig. 1 compares the throughput predictions from this model to NS2 simulations. The parameters here are the standard 802.11g values, sending 1000 byte packets. It can be seen that the throughput predictions from the model and the results obtained from NS2 are close. This demonstrates that the model accurately captures the effect of adjusting P_{ack}.

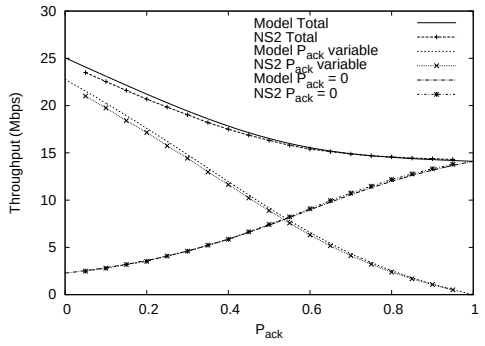

Fig. 1. NS2/model comparison for a WLAN with N_1 nodes with $P_{ack} = 0$ and N_2 with nodes with P_{ack} variable. In the figure $(N_1, N_2) = (1, 10)$ and we see similar accuracy for other values. The aggregate throughput of each group of nodes and the network's total throughput are shown.

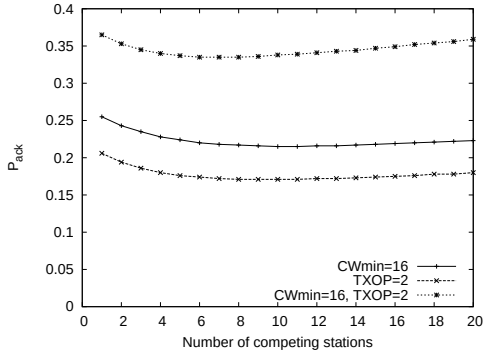

Fig. 2. P_{ack} value required to correct for one misbehaving node versus number of nodes in the WLAN.

III. Policing Nodes

The minimum contention window size, CWmin, and the TXOP burst size are configurable on most modern WiFi (802.11e-compliant) cards. For example, consider three easily implemented cheating strategies/configuration problems: (i) reducing CWmin by a half (approximately doubling the share of throughput compared to well-behaved nodes), (ii) setting TXOP to 2 packets (doubling throughput) and (iii) combining this TXOP setting with halving CWmin (giving approximately a four-fold increase). Fig. 2 shows the value of P_{ack} required to restore fair operation for each of these cheating strategies. We see that the P_{ack} value is not sensitive to the number of other nodes using the network. We now consider a measurement-based policing algorithm for automatically determining the appropriate value for P_{ack}, using feedback based on current network conditions and node behaviour.

Feedback Algorithm

The objective of the policing algorithm is to select an appropriate P_{ack}. As the behaviour of a node can change over time, we use an online algorithm to respond to observed behaviour. We want the algorithm to not only enforce a fair throughput allocation, but also optionally apply a penalty related to the degree of misbehaviour. We aim to make the scheme agnostic to the exact details of the misbehaviour.

Algorithm: On step k the AP updates P_{ack} for node i using

$$P^{k+1}_{ack,i} = \left[P^k_{ack,i} + \alpha \left(\frac{S^k_{c,i}}{S^k_f} - (1 - \gamma P^k_{ack,i}) \right) \right]_{0,1-\epsilon}, \quad (6)$$

where $[x]_{a,b} = \max(a, \min(b, x))$. Here, $S^k_{c,i}$ is the throughput of client node i and S^k_f is the maximum throughput of a well-behaved node. The design parameters are $\alpha > 0$, which controls the rate of adaptation, $\gamma \in [0, 1]$ which controls the size of penalty and $\epsilon \in (0, 1)$ which limits the maximum penalty. The AP is in a good position to estimate S_f, as it is commonly saturated and is subject to the same competition for transmission opportunities as clients.

Analysis: We first show what action the algorithm takes when it converges and then give conditions for convergence. It is useful to consider $A^k_{c,i} := S^k_{c,i}/(1 - P^k_{ack,i})$, which is the attempt rate for transmissions that are successful or discarded by the AP.

Lemma 1: If $P^k_{ack,i}$ converges to a value in $(0, 1 - \epsilon)$ then $S^k_{c,i} \to S^k_f$ (for $\gamma = 0$) or $A^k_{c,i} \to S^k_f$ (for $\gamma = 1$); if $P^k_{ack,i} \to 0$ then the limits above hold or $S_{c,i} \leq S_f$ at $P_{ack,i} = 0$; if $P^k_{ack,i} \to 1 - \epsilon$ then the limits above hold or $S_{c,i} \geq (1 - \gamma(1 - \epsilon))S_f$ at $P_{ack,i} = 1 - \epsilon$.

Proof: Use Eq. 6 and note that $P^k_{ack,i}$ converges implies that the difference between successive terms goes to zero. ■

We conclude that if the algorithm converges then it will equalise throughout/attempt rate or go to an extremal value of $P_{ack,i}$ where the node is insufficiently or too aggressive. Now we consider conditions for convergence. We focus on $\gamma = 1$, but similar results hold for other γ values. We begin by considering well-behaved nodes, where using [9] we can bound how nodes backoff as $P_{ack,i}$ increases.

Lemma 2: If $A_{c,i} \leq S_f(1 - cP_{ack,i})$ for some $c > 0$ then $P_{ack,i}$ converges to 0.

Proof: Using Eq. 6 we can show that $P^{k+1}_{ack,i} \leq (1 - \alpha c\epsilon)P^{k+1}_{ack,i}$ or $P^{k+1}_{ack,i} = 0$, which is a fixed point. ■

Thus the algorithm does not drop packets of well-behaved nodes. We now turn to misbehaving nodes. Consider nodes where $A_{c,i} \geq S_f$, as might be the case if backoff was disabled.

Lemma 3: If $A_{c,i} \geq S_f$ for all $P_{ack,i} \in [0, 1 - \epsilon]$ then $A^k_{c,i} \to S^k_f$ or $P_{ack,i} \to 1 - \epsilon$.

Proof: If $A_{c,i} \geq S_f$ then $P^k_{ack,i}$ is nondecreasing and bounded above, and so we may apply Lemma 1. ■

Disabling contention window backoff is a particularly serious form of misbehaviour as it can lead to network congestion collapse. The algorithm selects $P_{ack,i} = 1 - \epsilon$, providing an implicit way to detect such serious misbehaviour and additional policing action, such as disassociating the node, might then be taken.

Now consider a situation where a node seeks to gain throughput but, for a sufficiently high P_{ack}, its attempt rate can be reduced to that of a well-behaved node. The following lemma gives a simple set of sufficient conditions, but these conditions are not necessary and convergence can be shown in other situations.

Lemma 4: Let $A_{c,i}/S_f$ be continuous and strictly decreasing function of $P_{ack,i}$ with $A_{c,i}/S_f > 1$ when $P_{ack,i} = 0$ and $A_{c,i}/S_f < 1$ when $P_{ack,i} = 1 - \epsilon$. Then $P^k_{ack,i}$ converges to a ball around a fixed point of Eq. 6. Moreover, if $A_{c,i}/S_f$ is Lipschitz with a sufficiently small constant, then $P^k_{ack,i}$ will converge.

Proof: Our conditions ensure there will be a unique fixed point, P. Using the Lyapunov function $V(k) = (P^k_{ack,i} - P)^2$

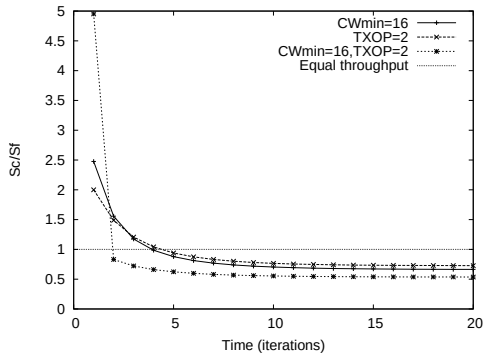

Fig. 3. Ratio of throughput of misbehaving node with that of a fair node, WLAN with 2 nodes, $\alpha = 0.1$, $\gamma = 1$.

we can show that

$$V(k+1) \leq V(k) - 2\delta(P^k_{ack,i} - P) + \delta^2, \qquad (7)$$

where $\delta = \alpha(1 - A^k_{c,i}/S^k_f)(1 - P^k_{ack,i})$. Note that $\delta(P^k_{ack,i} - P)$ is positive, so $P^k_{ack,i}$ converges to a ball around P. If we have the Lipschitz condition $\alpha|1 - A_{c,i}/S_f| < 2|P_{ack,i} - P|$ then $V(k)$ decreases when $P_{ack,i} \neq P$ and so $P^k_{ack,i}$ converges. ■

In our tests of various TXOP and CWmin parameters we find we can approximate the Lipschitz condition by ensuring that $\alpha|1 - A_{c,i}/S_f| < 2$ when $P_{ack,i} = 0$. For example, when $\gamma = 1$ and $\alpha = 0.1$ we find the scheme is stable when $A_{c,i}/S_f < 21$, which encompasses many practical situations.

IV. Results

A key benefit of this algorithm is that it is agnostic of the cheating strategy. Revisiting the strategies from fig. 2, fig. 3 illustrates the operation of the algorithm for $\alpha = 0.1$, $\gamma = 1$ and $\epsilon = 0.001$. The algorithm reduces the throughput of the misbehaving node to less than that of a well-behaved node, as $\gamma = 1$. We see that the resulting throughput is lower for the combined TXOP/CWmin strategy, illustrating that a larger penalty has been applied to a higher degree of misbehaviour.

Fig. 4 shows results for a node which initially misbehaves by halving its CWmin, but which modifies its behaviour at $t = 30$ to use the correct CWmin value. Results for WLANs with 1 and 10 fair nodes are shown. Initially, the algorithm penalises the cheating node, but restores P_{ack} to 0 after the node becomes well behaved. As $\gamma = 1$ the algorithm aims to equalise the attempt rate for fair nodes and cheating nodes. Thus the equilibrium throughput of the fair nodes in the presence of the cheating node is the same regardless of whether the cheating node has CWmin is 16 or 32.

Fig. 5 shows the effect of a node with CWmax set to CWmin, effectively disabling back-off. As described above, the attempt rates cannot be equalised, so the misbehaving node's throughput is driven to zero as $P_{ack} \to 1$. This is a useful feature of the algorithm as we do not have to explicitly detect if a node is backing off. If $P_{ack} = 1$, or above some suitable threshold, the AP can disassociate the offending node.

Also shown in fig. 5 is the impact of the node enabling back-off, at $t = 30$. It can be seen that the policing algorithm responds by decreasing P_{ack} to 0 and the node's throughput returns to its fair share.

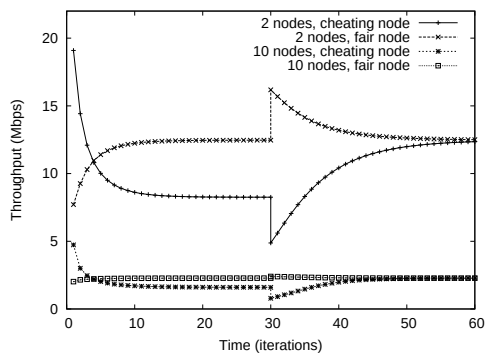

Fig. 4. One node misbehaves by halving CWmin(=16) restores Cwmin(=32) at $t = 30$. WLANs with 1 and 10 fair nodes, $\alpha = 0.1$, $\gamma = 1$.

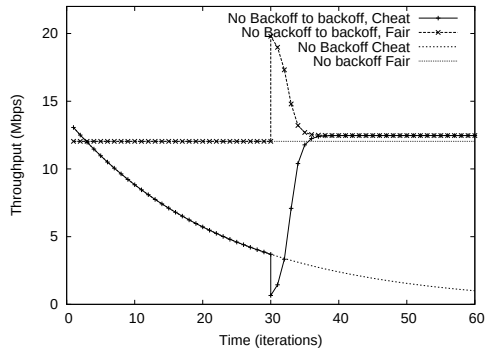

Fig. 5. One node with CWmin=CWmax, impact of back-off versus no back-off. WLAN with 1 misbehaving and 1 fair node, $\alpha = 0.1$, $\gamma = 1$.

V. Conclusions

Dropping ACKs to correct for misconfiguration provides an access-point-only mechanism to penalise nodes, without relying on either cooperation or changes to the standard behaviour of the nodes. We propose an online policing algorithm which has the desirable properties that (1) nodes can be incentivised to behave fairly and (2) specific forms of misbehaviour do not have to be identified. Further, misbehaviour which prevents a node backing off, potentially impacting the stability of the network, is implicitly detected.

References

[1] M. Raya, I. Aad, J.-P. Hubaux, and A. E. Fawal, "DOMINO: Detecting MAC layer greedy behavior in IEEE 802.11 hotspots," *IEEE Trans. Mob. Comput.*, vol. 5, pp. 1691–1705, December 2006.

[2] A. A. Cárdenas, S. Radosavac, and J. S. Baras, "Detection and prevention of MAC layer misbehavior in ad hoc networks," in *ACM workshop on Security of ad hoc and sensor networks*, 2004, pp. 17–22.

[3] L. Vollero and G. Iannello, "Frame dropping: A QoS mechanism for multimedia communications in WiFi hot spots," in *International Conference on Parallel Processing Workshops*, August 2004, pp. 54–59.

[4] L. Vollero, A. Banchs, and G. Iannello, "ACKS: a technique to reduce the impact of legacy stations in 802.11e EDCA WLANs," *IEEE Comm Lett.*, vol. 9, no. 4, pp. 346–348, April 2005.

[5] T. Murase, Y. Hirano, S. Shioda, and S. Sakata, "Proposal on wireless LAN MAC frame control in receive opportunity for flow QoS," in *IEEE Communication Quality and Reliability Workshop*, April 2008, pp. 1–6.

[6] H. Kameda, E. Altman, C. Touati, and A. Legrand, "Nash equilibrium based fairness," in *GameNets*, May 2009, pp. 533–539.

[7] A. Kumar, E. Altman, D. Miorandi, and M. Goya, "New insights from a fixed point analysis of single cell IEEE 802.11 WLANs," in *INFOCOM, Miami, USA*, vol. 3, 2005, pp. 1550–1561.

[8] Q. Ni, T. Li, T. Turletti, and Y. Xiao, "Saturation throughput analysis of error-prone 802.11 wireless networks," *J. Wireless Mobile Comput.*, vol. 5, pp. 945–956, December 2005.

[9] G. Bianchi, "Performance analysis of IEEE 802.11 distributed coordination function," *IEEE J. Sel. Areas Commun.*, vol. 18, no. 3, pp. 535–547, March 2000.