

Implementation Experiences of Bandwidth Guarantee on a Wireless LAN

Srikant Sharma Kartik Gopalan Ningning Zhu
Gang Peng Pradipta De Tzi-cker Chiueh

Rether Networks Inc.
99 Mark Tree Road, Suite 301, Centereach, NY 11720
rni@rether.com 631-467-4381

ABSTRACT

Rether¹ was originally developed to support guaranteed Quality of Service (QoS) for shared Ethernet LANs. With the growing popularity of wireless LANs, we modified the Rether protocol to provide QoS guarantee on wireless networks. In this paper, we present the design and implementation of the Wireless Rether protocol for 802.11 networks. We also describe our experiences with wireless LAN hardware. Wireless Rether supports QoS for TCP and UDP traffic in both upstream and downstream directions. The protocol can seamlessly interoperate with any priority-based QoS mechanisms (such as Diffserv²) on the wired networks that connect the wireless access network to the rest of the Internet. QoS requirements of real-time applications are specified as a simple configurable policy table. Legacy networking applications can benefit from QoS guarantees provided by Wireless Rether without requiring any modifications.

1. INTRODUCTION

With the growing popularity and acceptance of 802.11 wireless LANs, it is essential to look beyond connectivity and security issues and focus on Quality of Service (QoS) support for advanced multimedia applications, such as one-way video playback or two-way audio/video communication. Typically wireless LANs serve as the access networks to the wired infrastructure for mobile terminals, as shown in Figure 1. Mobile hosts are connected to the rest of the network through Layer-2 base stations (or access points) and Layer-3 edge routers. Mobile IP³ takes these access networks one step further and provides roaming support for mobile terminals. Roaming allows mobile terminals to migrate between different co-operating 802.11 wireless network segments seamlessly without loss of connectivity.

The wireless connectivity and the ability to roam across multiple networks make wireless 802.11 LANs an ideal choice for transient and dynamic networks. Networks are dynamic in the sense that the constituent nodes in the network keep changing as terminals roam from one network to another. Ideal locations that require wireless networking infrastructure include public gathering places like conference venues, airports, university campus etc. where people are always on the move.

Future applications of wireless LAN-based access networks would include audio/video conferencing and media streaming, both of which are sensitive to packet latency and effective bandwidth characteristics of the

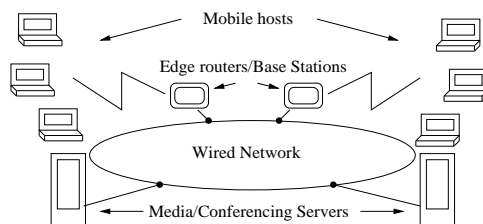


Figure 1. *Wired/Wireless network architecture. The wireless LANs lie at the edge of the wired infrastructure. Typically media playback servers reside on the wired network and stream multimedia data to mobile hosts. On the other hand, audio/video conferencing applications require special conferencing servers to relay traffic among mobile hosts.*

underlying network. On the wired network, IETF's Integrated Services⁴ and Differentiated Services^{2, 5} architectures are available to support guaranteed QoS and traffic prioritization respectively above the link layer. IEEE's 802.1p is a layer 2 traffic prioritization standard for switched Ethernet environments. However, no commercially available QoS solution exists for wireless LANs, particularly 802.11 networks. This research aims at developing a bandwidth guarantee mechanism for 802.11-compatible wireless LANs.

Since a wireless LAN is a shared medium, collisions arise and subsequently some form of random back-off mechanism is triggered when multiple nodes attempt to send data simultaneously. In order to guarantee deterministic network access and eventually support QoS for different network nodes on a wireless LAN, some special media control is required that can avoid collisions in the first place. The basic idea of the original Rether¹ protocol is to repeatedly circulate a software token among the Ethernet nodes within a fixed-length cycle, and to permit only the network node that currently holds the token to transmit data on the wired link. The amount of data a node can send when it is the token holder depends on its bandwidth requirement. This way there is no collision by construction and every wired LAN node can reserve a different amount of the shared link bandwidth. A unique feature of Rether is that it is designed to be a software-based protocol that can run on existing Ethernet hardware, thus allowing users to leverage their existing network hardware investment.

Although, Rether appears to be immediately applicable to *wireless* LANs, there are major technological differences between Ethernet and 802.11 networks that require significant re-thinking of the details of Rether. This paper describes the design and implementation of a variant of Rether that is specifically tailored to 802.11 wireless networks.

2. RELATED WORK

Quality of Service research on wireless LANs is mainly driven by attempts to make wireless LANs inter-operate with telephony networks. To carry voice traffic, 802.11 networks support distributed and point coordination functions (DCF and PCF). PCF is a connection oriented capability. It involves a central co-ordination by a *point coordinator* (PC) which initiates and controls *contention free periods*. PCs poll terminals with time bound traffic by sending poll tokens. The distributed coordination function is similar to CSMA/CA during the contention period. The actual use of these protocols are not common and these have been mostly analyzed for performance in simulated environments.^{6, 7} IEEE is proposing new 802.11e standard aimed at improving QoS on 802.11a and 802.11b networks.

HomeRF aims at providing wireless networking for home appliances. The latest standard HomeRF 2.0 targets quality of service by providing different types of services using separate time-slots. HomeRF supports reservation slots of 100Kbps, which are useful only for voice data but not for video. Although HomeRF networks are considered to be slower than 802.11, support for QoS enables IP telephony in home networking.

Other works in QoS for wireless networks include mostly algorithmic approaches like Packet Fair Queuing⁸ and simulation based approaches like comparisons of PCF and DCF.⁶ An extensive treatment of programmable mobile networks is done as a part of Mobicore⁹ which requires programmable Mac protocols.

3. DESIGN ISSUES AND SYSTEM ARCHITECTURE OF WIRELESS RETHER

In this section we discuss important design issues in the development of the Wireless Rether protocol, the solutions we chose and their rationale.

3.1. Hardware vs Software Implementation

Wireless Rether protocol can either be integrated with the firmware on the wireless LAN card or as a software layer above the network hardware's device driver. The advantage of the hardware approach is that it can significantly reduce the overhead involved in passing the token between hosts. The disadvantages of the hardware approach are that the solution becomes tied to individual vendor's hardware, the cost of hardware implementation is significantly higher, and there is less flexibility in configuring QoS policies.

On the other hand, a software-based approach has the advantages of being able to work with any vendor hardware, being cost-effective and providing much more flexibility in configuring QoS policies and in refining

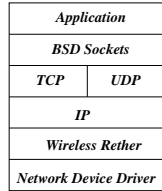


Figure 2. *Wireless Rether protocol layer. Wireless Rether is implemented as a layer between the link-layer hardware’s device driver and the IP protocol layer. With this layering structure, Wireless Rether is able to exercise QoS-related control over all outgoing network connections.*

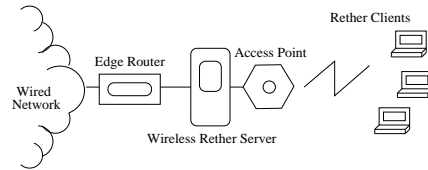


Figure 3. *Wireless Rether Architecture. Each edge network is augmented with a Wireless Rether Server (WRS), which acts as a bridge between the Layer-2 access point and the Layer-3 edge router. All wireless LAN hosts are equipped with Wireless Rether client software.*

implementation details of the QoS mechanism. Our previous experiences with Rether¹⁰ showed that with careful protocol and software design, the token passing overhead can be kept to a minimum. Hence we decided to adopt the software approach shown in Figure 2.

3.2. Peer-to-peer vs. Centralized Token Passing

A central design decision in Wireless Rether is how the token is passed from one host to the next. One alternative is to maintain a logical ring among the wireless hosts and to implement a *peer-to-peer token passing* protocol. The host holding the token transmits the token to the next logical neighbor in the ring and transmits an ACK to the previous logical neighbor in the ring. With such a distributed token maintenance protocol there is no single point of failure in the network and hosts can join and leave the the ring at any time. This was the scheme used in *wired Rether*¹ and it worked well in the context of shared Ethernet segments, where all hosts can communicate with each other directly. In the context of wireless LAN, since mobile hosts can move out of each other’s transmission range, direct token passing between wireless hosts is impractical and infeasible. However, since all hosts are assumed to be reachable from the access point, the access point is in a better position to relay the token.

As a result, we chose to implement a *centralized token-passing protocol* in which a central server, called *Wireless Rether Server* (WRS), is placed right between the access point and the wired network and is responsible for granting the token to wireless LAN hosts, called *Wireless Rether Clients* (WRC). The WRS grants tokens to all the WRCs in a weighted round robin fashion. The weight corresponds to the duration for which a WRC can hold the token and thus transmit data over the wireless channel, i.e., the approved bandwidth reservation for the WRC. The sum total of all the weights is smaller than the *cycle time*, which is the average token revisit time for each WRC. A portion of the token cycle time is dedicated to best-effort or non-real-time traffic.

This centralized architecture, shown in Figure 3, has several advantages. First, it is the WRS and not the token that maintains the QoS-related state. Therefore token loss is not fatal to the proper functioning of the Wireless Rether protocol. Secondly, since the WRS can intercept all traffic entering and leaving the wireless LAN, it can snoop on the wireless channel to determine the end of a packet that a WRC sends and reduces the token passing overhead by eliminating the need of ACKs from WRCs. A potential problem with Wireless Rether is that it now introduces a single point of failure in the WRS. This is not a major concern to us because eventually we expect the WRS to be merged into the access point. Overall, the centralized token passing protocol is much simpler, lightweight, and efficient than its distributed counterpart.

Combining the WRS and the access point would have lead to a cleaner hardware set-up. However, this integration is not possible currently since commercial off-the-shelf access points do not expose any programming interface to add third-party code. In retrospect, this limitation may be a blessing in disguise, because the separate-WRS architecture requires the resulting Wireless Rether implementation to be independent of and thus able to inter-operate with 802.11 access points from multiple vendors.

3.3. Work-Conserving vs. Non-Work-Conserving

Within a Wireless Rether cycle, the token first visits those network nodes that have made bandwidth reservation (called real-time or RT nodes), after all the RT nodes have been visited the token visits the network nodes that potentially have best-effort traffic to send (called non-real-time or NRT nodes). Note that *every* wireless LAN node is an NRT node. If the token cannot visit all NRT nodes within a cycle, it continues to visit the NRT nodes in the next cycle starting from where it left off in the previous cycle. Wireless Rether supports a non-work-conserving service model because even when there is no NRT traffic, the token is still passed among NRT nodes until the current cycle ends, i.e. RT nodes can never send data at a higher rate than their reservation even when other RT or NRT nodes have less data to send. A non-work-conserving model reduces the extent of data burst and thus decreases the delay jitters that applications experience. In addition, this model fits well with the goal of minimizing performance impacts on NRT or best-effort traffic.

3.4. To ACK or Not to ACK

The Wireless Rether protocol requires *mandatory ACK*, i.e. a WRC holding the token would send an acknowledgment message to the WRS at the end of its data transmission. The WRS would transmit the token to the next WRC only upon receipt of the acknowledgment from the previous WRC. The acknowledgment serves two purposes. First, it informs the WRS when the token-holding WRC's data transmission is complete and the channel is free for the next WRC. Second, it informs the WRS that the WRC is still alive and participating in the protocol. Unfortunately, a token-ACK message pair incurs considerable transmission overhead over the wireless media - around 2-ms round-trip delay - which is clearly inefficient. Specifically there is a large gap between the air time for the token-ACK message and the token-ACK round-trip delay which suggests that the wireless channel may be idle during most of the round-trip delay period.* Given a 33-msec cycle time, this means that Wireless Rether can only support up to 16 WRCs without even transporting any payload.

To use the wireless channel more efficiently, we considered an *optional ACK* design, in which the WRS sends a token to the next WRC at the end of the time slot allocated to the current WRC, without stalling for an ack from the current WRC. Here we assumed that the WRS is able to predict the end of a WRC's data transmission transaction. The WRS does not rely on ACKs for distributing tokens, except when a WRC is done with its data transmission and there is plenty of time left in its allocated time slot. The WRS will send out a token immediately to the next WRC upon receipt of such an ACK. An advantage of this design is that the token-ACK round-trip delay would no longer be on the critical path, and therefore the token cycle can be more efficiently utilized. However we found that the data transmission time over the wireless network can not be predicted reliably and as a result collisions were induced. In addition to collision, access point buffering delays accumulated over the time. This eventually led to multiple tokens circulating simultaneously in the system, which in turn introduced more collision. The token-ack protocol could not work in a reliable manner because of this unpredictability of wireless channel usage. Due to these problems we chose to stick with the mandatory ack design whose advantage of deterministic nature far outweighs the inefficiency due to token-ACK exchanges.

3.5. Intralan Traffic between Mobile Nodes

It is possible that two communicating nodes may belong to the same wireless LAN. In *ad hoc* mode the packet transmission is peer-to-peer and there is no additional overhead. But in *infrastructure* mode the source first hands all packets over to the access point and then the access point relays the packets to the destination node. Since the access point has to retransmit packets over the same wireless medium the transmission cannot be cut-through but has to be store-and-forward. Thus in infrastructure mode each Intralan packet effectively occupies two wireless packet transmission slots. This requires special consideration when making bandwidth reservation. The bandwidth requirement in this case is doubled. The reservation mechanism needs to check if the destination host is an Intralan host and if so it needs to double the reservation requested.

*The header of an 802.11b packet is physically transmitted at 1 Mbit/sec to maintain backward compatibility with 802.11, even though the packet body can be transmitted at 11 Mbits/sec. Therefore small messages are extremely inefficient on 802.11b networks

Since the number of hosts in a wireless LAN is limited and the WRS has the complete knowledge of participating hosts, periodically the list of mobile hosts can be transmitted to all the WRCs to update their local copies. The alternative approach which we chose to implement is to route all packets through the WRS and shift the responsibility of reserving additional bandwidth over to WRS. The advantage of the second approach is in saving on the broadcast messages to exchange the list which would otherwise consume the precious bandwidth. Also, when bandwidth reservation is high, data forwarding through the access point may collide with the next set of data from the sender. This can result in degradation of performance and affect the QoS guarantees.

3.6. Bandwidth Reservation Mechanism

As mentioned earlier, the QoS mechanism of Wireless Rether needs to work seamlessly with the QoS mechanism in the wired network in order to achieve end-to-end QoS for applications. One of the first considerations is whether or not to support end-to-end application level signaling such as RSVP.⁴ Such a signaling mechanism would be ideal from the perspective of new QoS-aware applications that can be written from scratch to make use of signaling. However, this is impractical for two reasons. First, end-to-end per-connection signaling requires that intermediate routers maintain state about each connection. This tends to hurt scalability and is one of the main reasons that RSVP did not succeed. The second reason is that many legacy applications exist which require QoS guarantees, but cannot be rewritten to make use of end-to-end signaling.

One of the primary goals of developing the Wireless Rether protocol was to support QoS for legacy and third party applications without modifying them. Since such applications cannot perform explicit reservation, there is no direct way of determining their bandwidth requirements. The solution is to use an indirect way of determining the bandwidth requirements. The end-points for such network applications are network hosts and ports. A commonly used convention is port based reservation mapping. In such a scheme the network port number used determines the bandwidth requirements. The mapping between the port numbers and the bandwidth is specified via system policy specifications. A further refinement is to use the network addresses in conjunction with wild cards as well. The reservation mechanism in Wireless Rether uses a *quintuple specification* of `{SrcAddress/Mask, DestAddress/Mask, SrcPortRange, DestPortRange, Bandwidth}`

For example, a policy specification like `{192.168.1.0/24, 192.168.2.6/32, *, 80-80, 1000}` would specify a reservation of 1Kbits/sec for Web traffic between the subnet 192.168.1.x and HTTP server 192.168.2.6. If a matching policy specification is found for a new packet stream, the reservation request for the stream is sent to the WRS. The Wireless Rether server performs admission control checks to admit or reject the reservation request for the new stream. If the request is admitted, at run-time the network packets of the corresponding stream are queued in a special real-time (RT) packet queue and are dispatched by the network scheduler according to the QoS specifications. If the request is rejected, the stream is treated as a best effort stream.

3.7. Support for TCP

Wireless Rether protocol is designed to reserve bandwidth for UDP, TCP, and ICMP traffic types. TCP traffic is inherently bidirectional in nature due to presence of TCP ACKs and hence requires special consideration. In order to guarantee bandwidth reservation for TCP data traffic in one direction, it is necessary to reserve bandwidth for TCP ACKs in the reverse direction since absence of ACKs would trigger TCP's congestion control mechanism and cause the sender to reduce transmission rate. Thus the perceived bandwidth for TCP connection would be less than the available bandwidth, leading to a wastage of reserved bandwidth.

Wireless Rether can transparently detect TCP streams and perform reverse bandwidth reservation for TCP ACKs. WRC module snoops on each outgoing packet. If the packet happens to be a TCP ACK, it consults the policy table to determine if the data traffic for the same TCP stream in the reverse direction has an associated reservation in the policy table. If this is the case, the WRC module makes reservation for TCP ACKs with a bandwidth reservations which is a certain percentages of that in the reverse direction. This percentages is a configurable parameter and is currently set to 10%. This value is determined after observing the bandwidth required for non-piggy-backed TCP acknowledgments and is believed to be a conservative estimate.

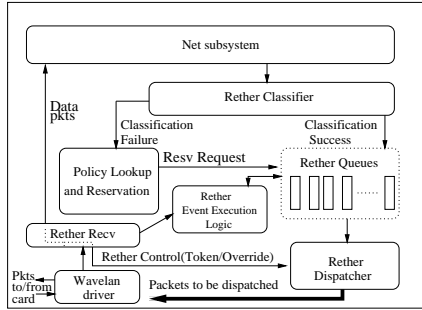


Figure 4. WRC Software Architecture: Rether Classifier intercepts all the outgoing packets and queues them in appropriate Rether Queues. Policy Lookup mechanism is responsible for appropriate processing of reservation requests. The event execution logic takes appropriate actions based on control messages from WRS. Rether Dispatcher is responsible for dispatching the packets as per the QoS requirements.

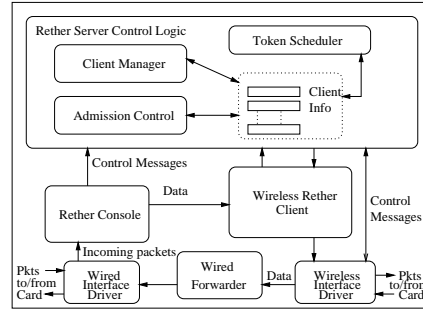


Figure 5. WRS Software Architecture: Each WRS has a built in WRC module to deal with the downstream traffic. The upstream traffic is forwarded to the wired network in a prioritized manner. Rether Console deals with the management messages from remote machines. Server control logic is responsible for maintaining client related data structures, admission control and channel access scheduling.

3.8. Buffer Overrun

It is quite possible that the data transmission rate for an application may exceed its reserved bandwidth. It is also possible that the available link bandwidth for a particular node may be less than the expected link bandwidth because of poor radio characteristics. In both cases the packet dispatching rate will be less than the generation rate by the applications. Such packet overrun will cause an accumulation of packets in the WRC buffers. There are two options to deal with this *run over* effect. The Rether module can choose to discard excess packets after a certain limit or it can choose to queue these packets in best-effort queues. Both approaches have advantages and disadvantages. If the radio characteristics of a channel are poor then utilizing the best-effort bandwidth to compensate for perceived bandwidth loss of real time streams seems to be a justifiable approach. If the packets are discarded then the degradation in the quality of application because of data loss may not be acceptable. On the other hand if the application is aggressive and starts generating packets in excess of reservation then redirection of excess packets to the best-effort queue may have an undesirable effect on the legitimate best-effort traffic. While dealing with aggressive applications, discarding packets seems to be an appropriate approach. On close observation it can be deduced that an optimum approach would be to somehow avoid the data loss without penalizing the existing best-effort traffic. If we assume that the degradation in the radio characteristics are transient and the mobile host will move to a better radio region within a reasonable period time we can solve this problem by providing longer per-connection buffers to queue the packets without discarding them. In Wireless Rether we decided to adopt the approach of providing larger buffers which could absorb transient radio characteristics problems.

3.9. Software Architecture

The Wireless Rether architecture is a client-server architecture where the *Wireless Rether Server* (WRS) has a global view of the bandwidth usage and the *Wireless Rether Clients* (WRCs) act according to the directives from the WRS. The overall setup of Wireless Rether is depicted in Figure 3. The WRS and WRCs communicate with each other by exchanging *Rether Control* messages. These include messages like *beacons* for liveness and presence detection, *registration* requests and replies, *reservation and deregistration* requests and replies, *token and ack* messages, etc.

3.9.1. Wireless Rether Client

In a Wireless Rether network, all nodes which transmit data on the wireless channel act as the *clients* for the wireless channel. In order to provide proper QoS guarantees it is imperative that *all* the clients should be

enabled with Rether software layer. Every WRC is granted exclusive channel access in a weighted round robin fashion in the form of Rether tokens on a regular basis. The internal architecture of WRC is shown in Figure 4.

Packet Queuing, Scheduling and Policy Lookup

Each outgoing packet in a client is classified by Wireless Rether as an RT or best-effort packet. Each RT packet is queued in a corresponding RT queue that is created by Wireless Rether when the first packet belonging to the RT stream is encountered by the Rether Classifier. Upon creation of every new RT queue a lookup on the Policy Table is performed to generate and send a reservation request for this queue. The best-effort packets are classified into three categories, namely, *control* for Rether protocol messages, *special* NRT for urgent messages without reservation requirements, and NRT for routine best-effort traffic.

Whenever a token visits a client, Wireless Rether's packet scheduler can dispatch packets from appropriate queues using either *time-based* or *byte-based* policy. Time-based policy dispatches packets based on share of token cycle time of the client whereas byte-based policy dispatches packets based on number of bytes reserved by a client during each cycle. Dispatching packets based on time share gives a fairly good control over the duration of transmission but results in a bursty dispatch of packets when the wireless bandwidth is better than expected. Dispatching packets based on number of bytes to be transmitted during each cycle avoids burstiness but results in long token holding time if the link bandwidth drops below the expected level. To overcome these problems we implemented a *hybrid* packet scheduler in which the limiting parameters are the reserved time share and the data share per cycle as well. In this approach the packets are dispatched till either the time share expires or the required amount of data is transmitted. This scheme properly utilizes bandwidth in case of conservative reservations and also deals gracefully with the variable link bandwidth conditions.

3.9.2. Wireless Rether Server

Wireless Rether Server (WRS) is the central node of a Wireless Rether network that has the overall control of the entire network. The WRS is responsible for distributing the wireless link bandwidth among the wireless nodes by providing them channel access tokens. Since all the downstream traffic is transmitted by the access point, which can not run the Rether protocol, the WRS also takes the role of a proxy WRC for the access point. The internal architecture of WRS is shown in Figure 5.

Token Scheduler and Admission Control

The primary responsibility of WRS is to provide access tokens to the WRCs in a timely manner for their real-time traffic and in a fair manner for their best-effort traffic. It is quite possible that the transmission from some client may exceed its allocated time share because of poor radio characteristics. Usually such time share extensions are absorbed by the time share reserved for the best-effort traffic. But when the channel is heavily loaded this may result in an extension of the cycle length. In order to provide QoS guarantees the WRS needs to maintain the average cycle length close to the specified cycle length. The token scheduler employs a cycle length compensation technique wherein it keeps track of recent cycle lengths and dynamically adjusts the cycle length by growing and shrinking best-effort time slots in future cycles. In Rether we adopted a simple static admission control policy for new reservations where new reservation requests are approved only if the overall reservation does not exceed the allowed fixed bandwidth share for real-time streams.

3.9.3. Prototype Setup

The current Rether prototype is implemented on the Linux platform (kernel version 2.2.16, user environment Redhat 7.0 and Redhat 6.2) The WRS is a Pentium II - 400MHz machine with 128MB of RAM, and EEPRO100 Ethernet cards. The WRCs are a mix of IBM compatible desktops (Pentium III 650 MHz) and portables (Pentium III 600 MHz) with 64MB RAM and PCMCIA Lucent Orinoco cards as wireless interfaces. The desktops use PCI-PCMCIA adaptors to interface with the Orinoco cards. The access point used is Lucent AP-1000 with Orinoco PCMCIA cards. The wired interface of access point operates at 100Mbps and the wireless interface is configured to run at 11Mbps.

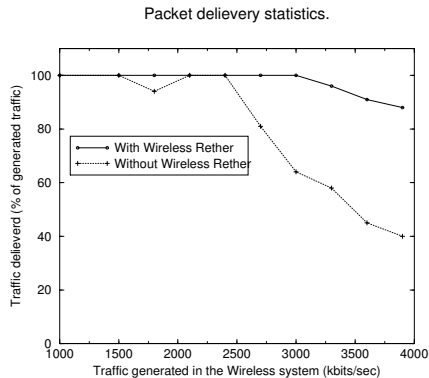


Figure 6. Comparison of packet loss with and without Wireless Rether. Even with just three senders transmitting, packet loss upto 60% is observed without Wireless Rether. This can be limited to a mere 10% by enabling Wireless Rether while providing bandwidth guarantees.

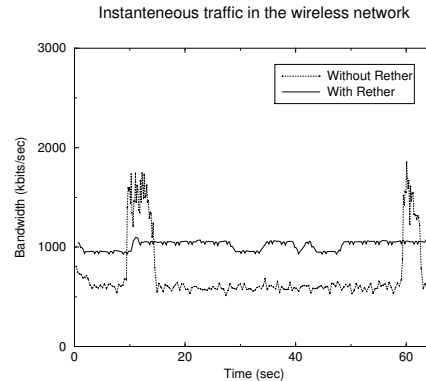


Figure 7. When there is no QoS in the network the transmissions tend to be bursty in nature. This bursty nature can be reduced by enabling QoS.

4. LESSONS AND OPTIMIZATIONS

4.1. Performance Comparison

Although the peak link bandwidth of the 802.11b hardware is 11 Mbps, the sustained bandwidth that is attainable on the wireless network is around 6.5 Mbps when a single sender transmits 1500-byte packets without collision. When there are multiple senders, collisions result in packet loss and hence reduction in the overall throughput. Our experimental setup consisted of three hosts transmitting simultaneously - two of them upstream and one downstream. Figure 6 shows a comparison of packet loss observed in the presence and absence of Wireless Rether. Without Wireless Rether, packet loss starts to appear even at a total load of 2.5 Mbps, beyond which the loss increases drastically to 60% when the load is around 4 Mbps. In contrast, with Wireless Rether, the packet drop rate stays within 10% of the total system load and the primary reasons of packet loss are the buffer overflow at the WRC end occurring because of reduced throughput at higher bandwidth and the packet drop at the access point because of load.

Another advantage of Wireless Rether is that it reduces the bursty nature of the traffic. Figure 7 compares the throughput variation of a wireless network in the presence and absence of the Wireless Rether. In both cases three senders attempt to transmit data at a rate of 1.1 Mbps. We observe that without Wireless Rether the traffic tends to be bursty and actual throughput is much lower than the required rate of 1.1 Mbps. In the presence of Wireless Rether, the required throughput is maintained while reducing burstiness. The figure shows one of the three streams. It can be seen that the deviation in the bandwidth guarantee for a reserved stream is minimal and is within 5% of the reservation.

4.2. Extended Cycle Time

Even when the channel is guaranteed to be collision free, the actual time for transmission over the wireless channel is not deterministic. It is observed that the transmission time for a constant amount of data varies from instance to instance. The direct effect of this irregular transmission time is an *extended cycle length*. Rether deals with this cycle length extension in a graceful manner by carrying out cycle length compensation. Since the average wireless channel transmission time over several cycles is predictable and is dependent on the available link bandwidth, the long term transmission behavior over the wireless channel can be controlled. The cycle length compensation technique works by dynamically growing and shrinking the cycle length so that the deviation from the requested cycle length stays minimal without affecting the QoS guarantees of real-time streams.

Compared to the transmission time, the time to queue data on wireless LAN card is surprisingly stable. Also the packet queuing time with the card is small compared to even the average channel usage time. This

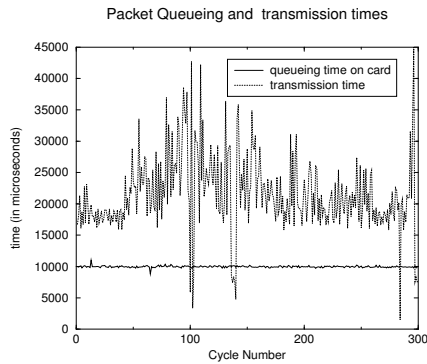


Figure 8. Observed packet transmission times per cycle and the queuing time for the packets. The packet transmission time is not constant for the same amount of data across successive cycles. The queuing time is very stable and considerably less than the transmission time. In this case the sender node is transmitting data at rate 3 Mbps and the packet size is 1400 bytes.

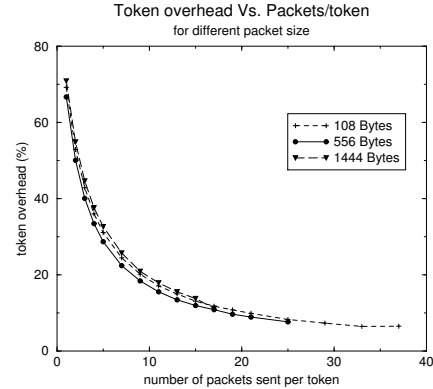


Figure 9. Rether overhead in terms of % lost time. Maximum overhead is around 70% when only one data packet is transmitted per token-ack exchange. The overhead reduces to 7% to 11% when there is only one token-ack exchange in the entire cycle.

observation can be attributed to the fact that usually data queuing is carried out using DMA and if the card has multiple buffers then consecutive packets can be queued independent of the transmission status of the previous packet. The channel usage characteristics and the queuing time behaviors are shown in Figure 8. The channel transmission time is measured by snooping on the wireless network using a machine that is set in promiscuous mode. The queuing time is measured by taking timestamps in Rether kernel before and after the queuing operation. This mismatch in queuing time and transmission time has an immediate impact on the Rether scheduler. Since the Rether scheduler is a hybrid scheduler where the transmission is stopped when either the required amount of data is transmitted or the allocated channel time is completely utilized, the knowledge of the exact duration of time spent during transmission becomes pertinent. Since the average transmission time is more important than the instantaneous rate, the knowledge of channel usage can be accurately obtained from WRS itself. The WRS can provide the WRC with this information while sending the next token. The WRC can correlate the channel usage time and the amount of data transmitted over the past few cycles and estimate the current available link bandwidth. Using this information the amount of data that can be transmitted over the next cycles can be determined.

4.3. Token Passing Overhead

Every Rether transmission is limited by a token and an acknowledgment for the token. For small bandwidth reservations the data to be transmitted per cycle can be as low as a single packet. In this case Rether needs to pay a heavy performance penalty of two small packets in each token visit. Figure 9 shows that maximum overhead for all packet sizes, when highest possible number of clients are active, is around 70% when only one data packet is transmitted per token-ack exchange, and is around 7% to 11% when there is only one token-ack exchange in the entire cycle. Figure 11 shows that the maximum number token-ack exchanges in a cycle, and hence number of WRCs, is around 12 or 13 for small payload packets and around 5 or 6 for large payload packets with a cycle time of 33 ms.

4.4. Peculiarities of Wireless LAN

4.4.1. Access Point delays

The access point is a link layer bridge between wireless and wired networks. Packet forwarding delay on the access point between the wired and wireless networks is not symmetric.[†] For upstream transmissions, the delay

[†]This is an observed behavior with Lucent AP1000 access point. Other access points may behave differently.

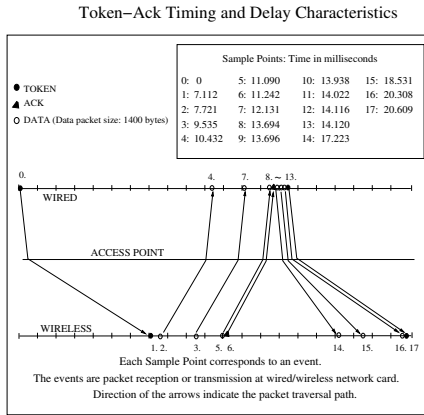


Figure 10. Detailed timing observation of a Token-Ack transactions and delay between wired and wireless networks forwarding.

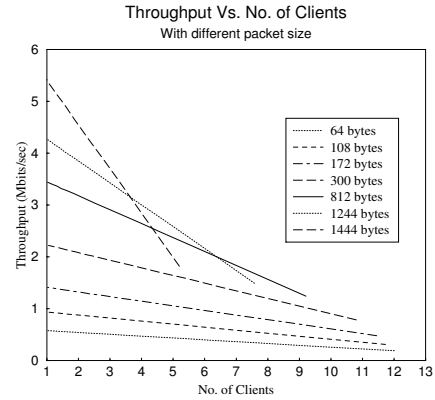


Figure 11. Maximum throughput possible in Wireless Rether for different packet sizes with different number of clients.

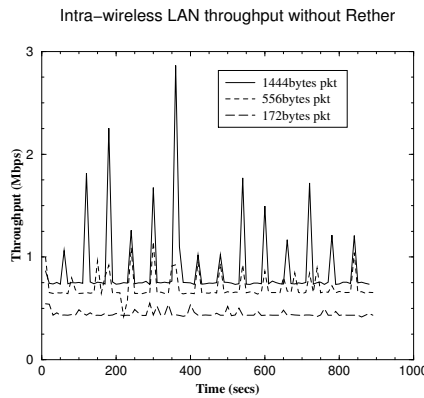


Figure 12. Channel usage in a wireless LAN without Rether for Intralan traffic. The traffic from a sender collides with the traffic forwarded by the access point resulting in performance degradation.

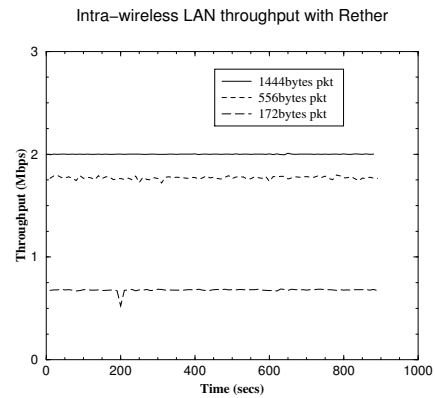


Figure 13. Channel usage in a wireless LAN with Rether for Intralan traffic. All traffic is diverted through WRS and hence there is no self-collision resulting in better performance. The small kinks in the graph occur due to occasional token loss.

introduced for each packet is of the order of 700 to 800 microseconds. Whereas for downstream packets this delay is of the order of milliseconds. This is *probably* because the upstream packets are forwarded in a cut-through fashion and downstream packets are buffered first and then forwarded over the wireless channel. Another effect of this *probable* buffering is that the delay for the first downstream packet after a burst of upstream traffic is considerably larger than other delays. The delays for such packets exceed other delays by a factor of two to three times. Usually the first downstream packet after a bursty upstream traffic happens to be the token for some other client. The delay in forwarding token impacts the Rether performance since the channel is unutilized during the period the token is buffered at the access point. These delay characteristics are shown in Figure 10.

4.4.2. Intralan traffic

Intralan traffic in infrastructure wireless networks requires special care. Since the access point acts as a repeater for all the Intralan packets, this results in self-collisions among received and relayed packets which leads to packet loss and degradation in data rate. The channel usage for Intralan traffic is shown in Figure 12. If the wireless LAN is Rether enabled and the traffic is diverted through the WRS then there is no self-collision and hence the QoS guarantees for the traffic can be maintained. The resulting performance is shown in Figure 13. It is also possible to control the self-collisions by transmitting the data from WRC in a staggered fashion. The

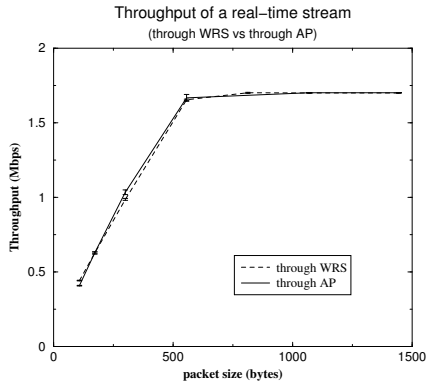


Figure 14. A throughput comparison of Intralan traffic through access point and WRS at 33ms. There is no performance penalty for diverting the traffic through WRS. The vertical lines in the graph indicate the deviation from the normal throughput.

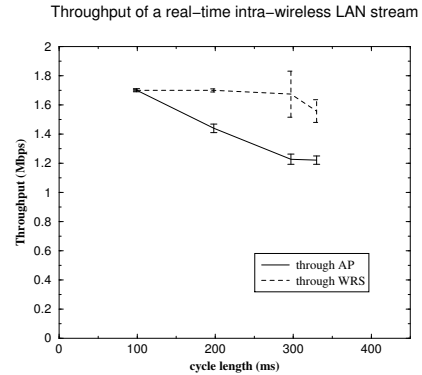


Figure 15. Comparison of Intralan traffic through access point and through WRS for different cycle lengths. At higher cycle lengths, the performance through AP drops whereas the performance through WRS is unaffected to a large extent. The vertical bars in the graph indicate deviation from the average throughput.

staggering is automatically introduced by Rether scheduler by limiting the data transmission during a cycle. Thus at smaller cycle length it is not necessary to divert Intralan traffic through the WRS. A performance comparison of throughput for cycle length of 33 ms when diverting data through the WRS and relaying through the access point is shown in Figure 14. But at larger cycle length the stagger size is increased and Intralan data relaying through access point is no longer efficient. A comparison of throughput with varying cycle lengths when relaying packets through access point and diverting through WRS is shown in Figure 15.

4.5. Scalability Study

The number of connections or mobile clients that Rether can support increases with the cycle length due to the reduction in the Token-Ack overhead. The assumed base cycle length for Rether is 33 milliseconds. Rether can provide QoS guarantees for at most 3 clients at reservation levels of 1 Mbps each when the cycle length is 33 ms. The number of supported clients can be increased to 4 by increasing the cycle length to 132 ms. Similarly for a reservation of 300 Kbps each, the maximum number of clients supported is 6 at 33 ms and can be increased to 8 at 132 ms. For 16 Kbps reservations, the maximum number of clients is 11 at 33 ms and it increases upto 25 for 132 ms. Figure 16 shows the relationship between packet size and network throughput. This number can be increased further by increasing the cycle length but increase in cycle length also results in a burstier traffic.

Figure 17 shows how the maximum possible bandwidth guarantee for a single client varies with the packet size. For smaller packet sizes the token-ACK overhead dominates. As the packet size increases the token-ACK overhead decreases since the total amount of traffic increases with the packet size. The maximum bandwidth possible with 64 byte packets is around 1 Mbps. It increases to 2 Mbps with a packet size of 256 bytes and is maximum of 5 Mbps at around 1400 bytes packet size.

5. CONCLUSION

In this paper we presented the design and implementation of Wireless Rether protocol that provides QoS guarantees on 802.11 networks. Wireless Rether grew from attempts to adapt the original Rether¹ protocol to wireless LANs. It adopts a centralized token passing architecture and supports a non-work-conserving service model that reduces the extent of data bursts and decreases delay jitters. Sequential token-ACK message exchanges between WRS and WRCs ensure that the wireless channel is kept collision free, thus increasing the effective throughput of the channel. Wireless Rether supports port-based reservation for real-time traffic streams. For real-time TCP streams, it transparently performs reverse bandwidth reservation for TCP ACKs. It uses a hybrid packet scheduler at WRCs that limits the transmission rate of real-time streams based on

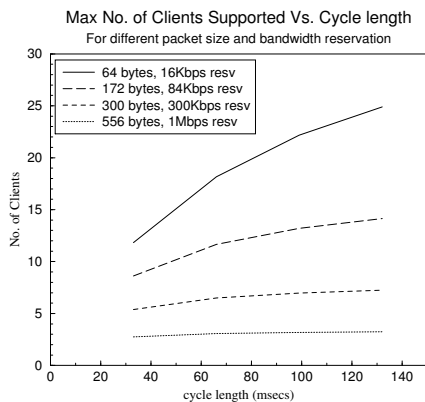


Figure 16. Scalability of Rether. The maximum supported clients increase with increasing cycle length. The base cycle length is 33 ms.

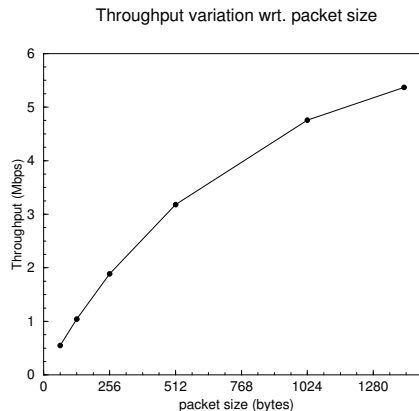


Figure 17. Bandwidth throughput variation with packet size. As the packet size increases the maximum possible bandwidth also increases.

time-share as well as byte-share per cycle. Our extensive performance measurements validate the ability of Wireless Rether to provide bandwidth guarantees over wireless LANs.

In future, we plan to build an extensive support for mobile IP in Wireless Rether so as to provide smooth handoff mechanism. We also plan to support reservation guarantee mechanism for multicast traffic. One the major potential of Wireless Rether lies in its utility value as wireless traffic manager. By regulating the traffic in downstream direction the WRS can be converted into a traffic manager.

REFERENCES

1. T. Chiueh and C. Venkatramani, "Supporting Real-Time Traffic on Ethernet," in *IEEE Real-Time Symposium*, 1994.
2. W. Feng, K. Kandlur, D. Saha, and K. Shin, "Adaptive packet marking for providing differentiated services on the Internet," in *Intl. Conf. on Network Protocols*, Oct 1998.
3. C. Perkins, ed., "IP Mobility Support." RFC 2002, Oct 1996.
4. L. Zhang, et. al., "RSVP: A New Resource Reservation Protocol." *IEEE Network Magazine*, Sep 1993.
5. P. Almquist, "Type of Service in the Internet Protocol Suite." RFC 1349, Jul 1992.
6. A. Kospel, J. Ebert, and A. Wolisz, "A Performance Comparison of Point and Distributed Coordination Function of an IEEE 802.11 WLAN in the Presence of Real-Time Requirements," in *Seventh Int'l Workshop on Mobile Multimedia Communications*, Oct 2000.
7. S. Khurana, A. Kahol, S. K. S. Gupta, and P. K. Srimani, "Performance evaluation of distributed coordination function for IEEE 802.11 LAN protocol in the presence of mobile and hidden terminals," in *Seventh Int'l Symp. on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, Oct 1999.
8. T.S. Eugene Ng, I. Stoica, and H. Zhang, "Packet fair queuing algorithms for wireless networks with location-dependent errors," in *IEEE Infocom*, Mar 1998.
9. O. Angin, A. T. Campbell, M. E. Kounavis, and R. R. Liao, "The Mobiware Toolkit: Programmable Support for Adaptive Mobile Networking." *IEEE Personal Commun. Mag*, Aug 1998.
10. P. Pradhan and T. cker Chiueh, "Real-Time Performance Guarantees over Wired/Wireless LANs," in *IEEE Conference on Real-Time Applications and Systems*, Jun 1998.