

Evaluating the impact of RTS-CTS in OLPC's XO's Mesh Networks

Leonardo Hideki, Raphael Martins, Arthur Guerrante, Ricardo Carrano, Luiz Magalhães

Departamento de Eng. de Telecomunicações – Universidade Federal Fluminense (UFF)
Rua Passo da Pátria, 156 – CEP 24220-121 – Niterói – RJ – Brazil

E-mail: {hideki,raphael,arthur,carrano,schara}@midia.com.uff.br

Abstract – RTS-CTS is a mechanism used to solve the hidden node and exposed node problems in a wireless environment. However, it has been shown by analysis and simulations that in general in a multiple hop network RTS-CTS is not always beneficial, and can, in fact, hurt throughput. In this paper, we measure the performance of a layer-2 mesh network implementation with and without the RTS-CTS mechanism. The test results shown in this paper were obtained through experiments performed using OLPC's XO's laptops. The mesh network plays a vital part in OLPC's project, both by allowing easy collaboration within a school with no further infrastructure and also by potentially being a tool for digital inclusion by extending Internet connectivity to students' homes by multi-hop paths. The main contribution of this paper is the use of actual hardware for experiments, since the majority of the research done is based on simulations, which may not account for all effects seen on the experiments.

I. Introduction

Mesh Networks is a generic name used to refer to multi-hop ad-hoc wireless networks. Usually, these networks are implemented using layer three routing protocols. However, IEEE is working on a standard, 802.11s, which implements a layer 2 mesh network. OLPC's XO's [13] - the one-hundred-dollar laptop - implement their mesh network according to IEEE 802.11s draft.

One of the key points of IEEE 802.11 standard is how to control multiple accesses. To accomplish this goal, the standard presents two mechanisms: CSMA (Carrier Sense Multiple Access) and CSMA/CA which uses RTS - CTS (Request to send - Clear to send) to avoid collision on data frames. CSMA/CA differs from CSMA/CD as it avoids collision instead of just detecting them. [1][2] It is well-known that the use of CSMA can suffer from the hidden node and exposed node problems, which are responsible for serious network performance issues.

In order to minimize these problems, it is also possible to enable the RTS-CTS mechanism. The RTS-CTS mechanism was developed to solve the hidden node and exposed node problems in infrastructure wireless networks. It would be possible to expect that, even for an ad-hoc network, performance improvements would be seen. However, as shown in

related work and by the results obtained the opposite holds.

Midia.com Laboratory, which belongs to UFF's Telecommunications Engineering Department, is performing experiments with OLPC's XO's as a part of the RUCA project [3]. In this paper, we present the results of the tests performed in multi-hop environments with and without RTS-CTS.

In section II, we present the hidden node problem and how the RTS-CTS mechanism solves it. Positive and negative aspects of this mechanism are presented together with related work. In section III, the main characteristics of IEEE 802.11s draft mesh networks are shown. In section IV, the methodology and description of the tests performed are presented. The results are shown in section IV, and future work and conclusions in section V.

II. Hidden Node Problem and Related Work

Suppose node B is transmitting data to node C and that node A cannot listen to node B's transmission. So, node A may start its transmission to node C, while node C is receiving data from B. A collision will occur in node C. This problem is known as the hidden node problem (Figure 1).

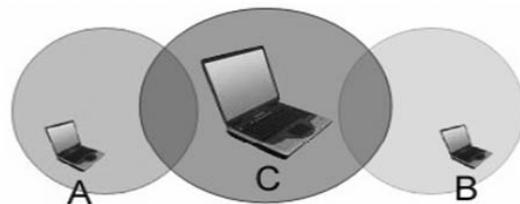


Figure 1 - Hidden node problem

RTS-CTS mechanism is used to avoid this scenario. When the sender wants to transmit, it has to send a RTS frame to the destination first, in order to reserve the medium. Every node inside this sender transmission range is able to listen to this packet. The RTS contains an estimated time of the total transmission (transmission time plus ack arrival time). If the destination accepts this transmission, it sends a CTS frame back to the sender, containing the total transmission time that should be enough to attend the value specified in the RTS frame.

Any other node that receives the CTS should not transmit until the time set in the frame is up. It is

Ethernet adapter and forward traffic between the wireless mesh and the wired network. The second method uses scripts developed by Cozybit (www.cozybit.com) that take advantage of the XO's virtual interfaces, associating eth0 to an access point and forwarding traffic between msh0 and eth0. In this case, both interfaces should operate on the same channel as there is a single PHY in use. Presently, both options – the adapter or the MPP scripts, will need extra software and configuration.

One last issue concerning MPPs is that stations should be able to find them and to choose between them if there is more than one.

B. XO radio subsystem design

The XO's radio subsystem is completely independent of the rest of the computer, although resides in the same board¹. It is composed of a Marvell 88W8388 chip, an onboard ARM 9 processor (plus ROM and RAM) and an 802.11g interface. XO's two rotating bunny ear antennas provide diversity and are quite effective if compared to the usual concealed antennas of commercial laptops.

The radio system is connected to the main cpu (an AMD Geode processor) by a Universal Serial Bus. This brings important implications to throughput, since all the IP traffic will be transferred through the USB, from the cpu to the radio and vice versa. Because of this architecture, the maximum throughput we could register in our tests, was 13.9 Mbps, using the iperf tool [12] to generate an UDP flow. On the other hand, this allows for operation of the mesh even with the main processor in sleep mode.

C. Routing

Currently, 802.11s' mandatory routing protocol is the Hybrid Wireless Mesh Protocol, or HWMP [9], which uses elements of Ad hoc On-Demand Distance Vector (AODV [10]), and also concepts of tree-based routing. The draft also allows the use of the RA-OLSR, which is based on OLSR [11].

AODV is an IP routing protocol, which exchanges routing messages via UDP datagrams. In contrast, HWMP is a layer two protocol. As we will demonstrate next, the choice of layer two for the mesh implementation brings some advantages.

Because of XO's architecture (described above), all layer two processing is handled by the radio subsystem, relying on the main CPU (AMD Geode GX-500) only for TCP/IP processing. The main CPU and the radio are connected via Universal Serial Bus (USB), which imposes a limit on performance, not

because of USB's speed, which is quite adequate, but because of constraints on the subsystem that manages USB.

One of the design goals of the XO is enabling a node to forward frames and routing information even when the main processor is turned off. And this is possible not only because the radio and layer two processing is detached from the main CPU but also because the layer two network subsystem needs no more than 0.5 watts to operate. This way another important premise – low power consumption – is not violated.

The OLPC mesh implementation is based on version 0.1 of the 802.11s draft and its routing protocol is a simplified version of the HWMP. Currently, XOs implement only on-demand route discovery and no proactive routing mechanism, i.e. no tree-based routing.

XO's mechanism for finding another XO's given its MAC address is also based on HWMP. If a XO (S) needs to discover a path to another XO (D) it broadcasts a RREQ frame (Route Request Mesh Management Frame). Every node who receives an RREQ and is not the destination node will broadcast it again. The request will eventually reach its destination D, who will respond with an RREP (Route Reply Mesh Management Frame). The RREP will be forwarded back to S through a sequence of unicast transmissions. This is possible only because every node that broadcasted the original RREQ has learned a reverse path to S.

In a similar way, when forwarding the RREP back to S, intermediary nodes will learn a "forward route" to D. When S finally receives this RREP he can transmit frames to D because they will be forwarded through the forward path.

When S in its turn needs to send frames to D the process is the same, i.e. S sends an RREQ frame, and the above cycle is repeated. In this mechanism the forward paths from S to D and the one from D to S may be different.

It is worth noting that broadcasted frames are not acknowledged in 802.11 which means that lost RREQs will not be retransmitted by the sender. We observed that XOs send out many copies of the RREPs and they do so by varying the transmission rate and associating different metrics to each one of the successive requests. For requests broadcasted in 54Mbps, for instance, the metric will be lower (better) than the consecutive try, in 36 Mbps, and so on.

Frames transmitted at lower rates have higher probability to succeed but their associated metric is higher (worse). So, if a choice exists, the protocol tends to select the higher throughput path. However, the choice for higher performance links must take the number of hops into account. In terms of airtime, energy savings and aggregated cpu cycles, one slow hop can be more effective than many fast hops.

If during the forwarding mechanism a frame can not be delivered, the sending node must transmit a

¹ In some prototypes it is a daughter board, because components have different pin spacing than normal PC parts, as they were made for cellular telephones, and isolating on a daughter board makes production easier and also minimizes electrical noise.

RERR (Route Error Frames) back the path, thus enabling predecessors to mark the route as lost. Also, because routing information is supposed to be soft state, the XOs will periodically “forget” the route and restart the path discovery cycle. At present this refresh time is set to 10 seconds and presents another protocol tuning point to be investigated.

By now, the only mechanism a XO has to find its neighbors is the RREQ/RREP and this is also used to discover and select a Mesh Portal Point. Whenever a station wants to find a MPP – for instance, if it has internet traffic to send – it sends a RREQ to a special address (C0:27:C0:27:C0:27). Each Mesh Portal Point present will answer to that request (sending a RREP). If the STA receives more than one answer, it will select the MPP with the lower cost path.

IV. Methodology and Description of the Tests

In an effort to evaluate the influence of RTS-CTS mechanism in multi-hop ad-hoc wireless networks we used OLPC’s XOs.

We used three methods to compare the throughput in each scenario (with RTS-CTS activated, and with RTS-CTS not activated): iperf-udp, iperf-tcp and scp. Iperf is a measurement tool developed by the University of Illinois. It measures throughput, TCP or UDP, between two nodes. Packet loss data was also obtained via iperf. SCP is a secure copy application that runs over TCP.

In order to be sure that the frames were being delivered the way we needed we employed the blinding table (BT) feature. BT allows us to set which MAC addresses should be ignored by the XOs. Or, alternatively, which MAC addresses should be accepted by the XOs, ignoring all the unspecified ones. This feature is very useful to force multiple hops in a chain of nodes topology, for example, as in Figure 3.

The first step of the tests was to make sure that the XOs were implementing the RTS-CTS mechanism properly. To accomplish this, we used a sniffer to capture the traffic we generated with three-XO-chain, prior to the actual testing. We set the RTS threshold to 999 bytes and also configured the blinding table then we used ping from one side of the chain to the other and no RTS-CTS frame was captured by the sniffer. As the ping packet is smaller than the RTS threshold we set, this behavior was expected. After that, we ran iperf from one side of the chain to the other. This time, we could see RTS-CTS frames being captured by the sniffer, again, as expected. This way, we could be sure that the RTS-CTS mechanism was implemented correctly by the XOs. Notice that in this scenario, we just wanted to verify its functionality, not its performance. So, the fact that the laptops were side by side didn’t have any effect on the results.



Figure 3 - Blinding table operation – the nodes in the edge must forward their frames through the central node

In order to evaluate the impact of RTS-CTS to the network performance, we moved the laptops far away from each other, so that the interference among them could be minimized. We placed each laptop in a different floor of the five-floor University building. Along with this procedure, the blinding table was configured to force the multiple hop communication. For one particular test, we reduced the transmission power of the XOs, so that the interference among them could be at the lowest level we could get.

V. Results

As we have already described in section IV, we ran these tests with iperf generating TCP and UDP traffic and also with the nodes downloading a file through SCP. However, throughput measurements are more representative when we evaluate UDP traffic instead of TCP. TCP has a lot of inherent features like slow start and flow control that inhibits network throughput. This is the reason why we will present our UDP results in more detail, even though we got the same behavior running the tests with TCP or SCP (which also runs over TCP).

The first test used iperf, and it generated UDP packets with the default size of 1460 bytes, which is smaller than the default RTS-CTS threshold value (of 2347 bytes). In other words, the traffic generated should not enable the RTS-CTS mechanism. In Figure 4, we present the minimum, maximum and average results, along with an error bar capturing the standard deviations for the test.

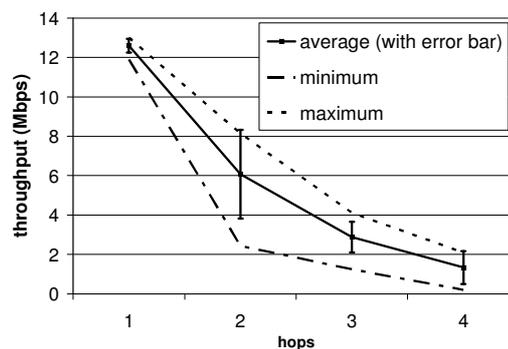


Figure 4- UDP Throughput vs Hops without RTS-CTS

For the second test we set the RTS threshold set to 999 bytes, thus generating UDP traffic with packets larger than the threshold, i.e. RTS-CTS mechanism was enabled in this scenario. The results are plotted in Figure 5.

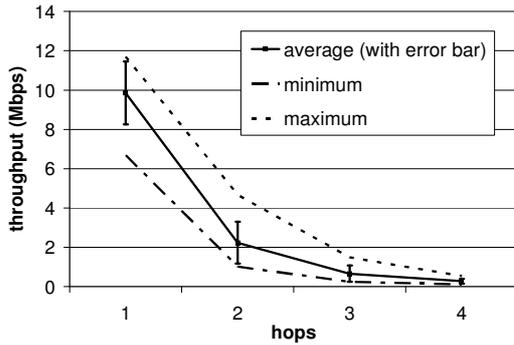


Figure 5 - UDP Throughput vs. Hops with RTS-CTS

The comparison between the scenario with and without RTS-CTS is shown in Figure 6. As can be seen, in the topology described in Section IV, the use of RTS-CTS makes the network performance worse. We have a 10% loss for the first hop, 30% for the second, 64% for the third and 74% for the fourth hop.

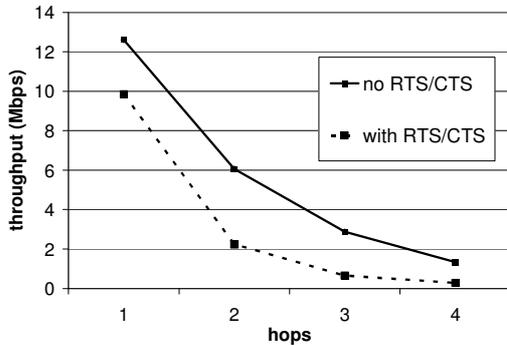


Figure 6 - UDP throughput with/without RTS-CTS comparison

As we mentioned before, the same tests were performed with iperf generating TCP traffic and also with the nodes downloading a file through scp and the same behavior was observed – RTS-CTS enabling not only did not help, it actually presented worse results. Figures 7 (for SCP) and 8 (for iperf TCP) demonstrate that.

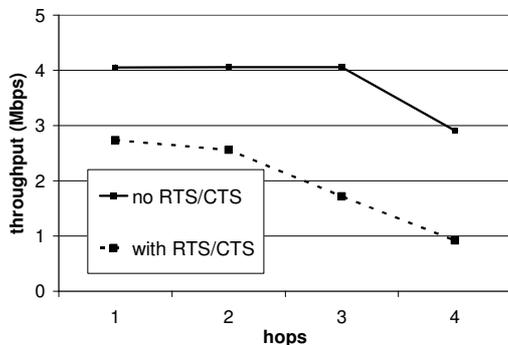


Figure 7 - Comparison for a SCP file transfer

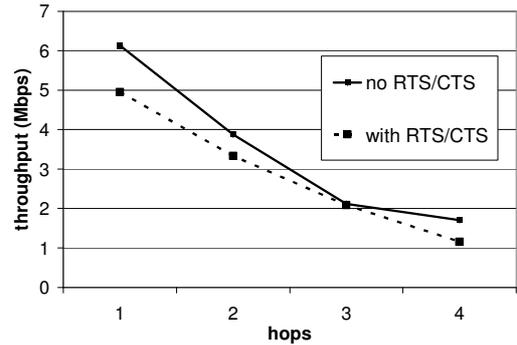


Figure 8 - Comparison for an Iperf TCP series of tests

Finally, we also reduced the XOs transmission power and repeated the tests, so that the interference among them could be minimum. Even in this scenario, the same behavior was observed and, for the sake of conciseness we omitted these results.

VI. Future Work and Conclusions

There are many reasonable explanations of why the use of RTS-CTS is malefic to the network performance. In the four-hop topology we suggested, the main fact that can be responsible for this behavior is the interference range that is proved to be much larger than the transmission range. The overhead issue becomes more critical when dealing with small packets. Also, the RTS-CTS mechanism prevents the nodes from transmitting under such circumstances that CSMA/CA would allow.

The results of our tests along with all the previous works done about the RTS-CTS effectiveness show that, in a real network, the use of this mechanism, most of the times, makes the network performance even worse. We couldn't find one single scenario where the use of RTS-CTS improves the network performance.

The main contribution of this paper is to present practical results of a subject that many people have studied theoretically, through analytical analysis and simulations.

On another phase of the RUCA project, we also tested the maximum transmission range between two OLPC's XOs. One of the results of this test is presented on Figure 9. In this test we generated a flow of small iperf-udp packets (50 bytes) and that accounts for the relatively low throughput obtained.

Notice that until 400 meters away from each other, the XOs' throughput was around 218.5 kbps, which is very acceptable considering that the maximum throughput for packets this size would be less than 550 Kbps. Packet loss was approximately zero. At 450m, the throughput went down to 69.42kbps, which is still acceptable. However, if we consider the same performance loss due to the RTS-CTS mechanism we had in section V, we would have much less throughput.

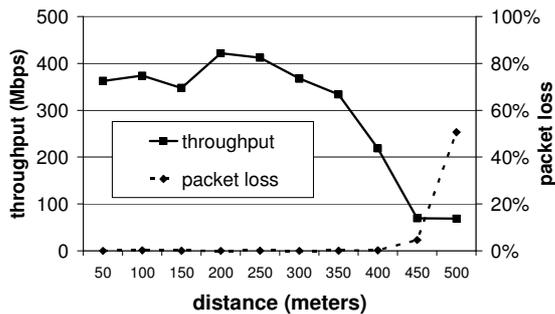


Figure 9- Maximum transmission range

We couldn't perform multi-hop tests along with the maximum transmission range because it is very hard to find a place to deploy this test bed. Our test place should be at least 1.5 kilometers long, plain, free of electromagnetic interferences, especially in the 2.4GHz band. From now on, we'll put all our effort to find a place that follows the conditions mentioned above, so that we'll be able to perform multi-hop tests along with the maximum transmission range of the XOs. We believe that the interference among the XOs will be reduced, but so will be the received transmission power. This way, we'll have to find the perfect balance of the signal to noise ratio if we want to figure out the XOs best performance scenario.

For digital inclusion purposes, the decision of using RTS-CTS can be the difference of making one more child to access to the Internet, for example. When you consider countries like Brazil, with high statistics of poor children, each child reached by a digital inclusion should be taken into account. This is why a simple decision of enabling or disabling RTS-CTS should be taken very carefully.

The same way we carefully tested RTS-CTS, and will perform more tests in the future, all tunable parameters of the IEEE 802.11 are being investigated, so the best solution, or at least the one that will cover most scenarios and result in good throughput can be implemented.

VII. References

- [1] Peterson, Larry L., Davie, Bruce S., *Computer Networks – A system approach*, 3rd edition, Elsevier, 2004.
- [2] Kurose, James F., Ross, Keith W., *Computer Networks – A top-down approach featuring the Internet*, 1st edition, Pearson Addison Wesley, 2003.
- [3] Carrano, Ricardo C., Bletsas, Michail, Magalhães, Luiz C. S., *Mesh Networks for digital inclusion – Testing OLPC's XOs Mesh Implementation*, 8^o Fórum Internacional de Software Livre (FISL), Porto Alegre, 2007.
- [4] Yomo, Hiroyuki; Chakraborty, Shyam S.; Prasad, Ramjee, *IEEE 802.11 WLAN with Packet Combining*, Technical Report, HUT, Helsinki, Finland, January, 2004.
- [5] Xu, Kaixin, Gerla, Mario, Bae, Sang, *How effective is the IEEE 802.11 RTS/CTS Handshake in Ad Hoc networks?* Technical Report, University of California, Los Angeles, USA, November 2002.
- [6] Sobrinho, João L., de Haan, Roland, Brázio, José Manuel, *Why RTS-CTS is not your ideal wireless LAN multiple access protocol*, Technical Report. IST, Lisboa, Portugal, March 2005.
- [7] Xu, Kaixin, Gerla, Mario, *Effectiveness of RTS/CTS Handshake in IEEE 802.11 based Ad Hoc Networks*, Technical Report, UCLA, Los Angeles, USA, July 2003 .
- [8] Tsai, Tzu-Chieh, Tu, Chien-Ming, *Improving IEEE 802.11 RTS/CTS Handshake in Wireless Ad Hoc Networks Considering Large Interference Range*, Technical Report, NCU, Chengchi, Taiwan, July 2004.
- [9] Michael Bahr *Proposed Routing for IEEE 802.11s WLAN Mesh Networks* Siemens Corporate Technology, Information & Communications Otto-Hahn-Ring 6 81730 München, Germany bahr@siemens.com, August 2006.
- [10] Perkins, C. E., Belding-Royer, E. M., and Das, S. R. *Ad hoc On-Demand Distance Vector (AODV) Routing*. IETF Experimental RFC 3561, July 2003.
- [11] T. Clausen, and P. Jacquet (Editors). *RFC 3626 - Optimized Link State Routing Protocol (OLSR)*. October 2003
- [12] NLANR/DAST : *Iperf 1.7.0 - The TCP/UDP Bandwidth Measurement Tool* – <http://dast.nlanr.net/Projects/Iperf/>
- [13] OLPC – One Laptop per Child – <http://www.laptop.org>