DD-WRT and RS-232 Communication with Linksys WRT54GL Router (March 2009)

David Tietz, *Student, IEEE*

***Abstract*—OpenWRT is a resource available for wireless router devices to be customized to allow for many different uses of a router. The idea behind this project is to install DD-WRT on a Linksys router that will allow for the use of an added RS-232 port to allow direct communication to any other device on a given network. Then DD-WRT will also be modified allow the creation of a mesh and possibily a secure connection, such as through a VPN tunnel into another network. All of these functions will allow for a secure reliable connection regardless of its location in an unsecured environment.**

***Index Terms*— Linksys router with Linux installation, OpenVPN, DD-WRT, RS-232 Port.**

I. Introduction

THIS proposed project is designed to give a better understanding of the possible uses of a DD-WRT interfaced into a Linksys router and the possible applications and modifications that can be made to allow for a broad range of uses. Such uses include simple RS-232 communication, wireless mesh networks, and secure communication, such as a VPN tunnel.

Firstly, a WRT54GL router will be acquired that will have its factory firmware overwritten with a clean install of DD-WRT in one of its many different distributions. From there, the DD-WRT software will be modified to allow for the interface of an RS-232 port. The port will enable the user to create hardware that can directly interface with the router for any sort of embedded system type communication. One instance would be the designing of a paging device that can be sent from any user on the network to this router and the message will be displayed on the device. This would require a program that interfaces with the hardware installed onto the router. The software would need to be installed on a given machine and would provide a sort of messaging type service. Anything typed into the message window would then be displayed on an LCD display installed to the RS-232 port mounted on the DD-WRT Linksys router.

Manuscript received January 29,2009. This work was supported in part by the University of Florida, the Electrical Engineering Department at the University of Florida, and Dr. Latchman's EEL5718 Computer Communication Course

David Tietz is with the Electrical Engineering Department, University of Florida, Gainesville, Florida.(e-mail: dtietz@ufl.edu).

Secondly, in order for the router to perform across firewalls and other roadblocks, it is desired that the DD-WRT router be modified to accept a VPN protocol to connect to a secure server allowing for a secure connection. This tunnel would help to ensure a secure connection regardless of firewalls on either end of the system. This would allow for a more versatile method of communication and would open more uses that can be utilized with this device.

Thirdly, the DD-WRT software would need to be modified to allow for a type of routing protocol that would enable the Linksys router to hop between wireless connections. This would be useful in a mobile situation or in a wireless mesh network. One of the very many benefits of the DD-WRT software is it allows the router to be a 'node' of a much larger network of wireless devices. These devices can communicate between each other and perform "hopping" that allows for communication over very large distances without ever needing to be physically connected to a network. While this sounds like a good idea in practice, one of the problems that arise is finding the best route to take across this large network. The DD-WRT software would need to be modified to allow for computations to take place to find the best possible path for communication. Such things include signal strength from one router to the next, as well as the largest possible amount of bandwidth that each route may offer. The DD-WRT software would help to perform both of these calculations and would need to do refresh often, in the case that the mobile nodes move or become unavailable.

As far as testing the communication between mesh networks with this router, we will only have very limited access to equipment that may be needed to create this type of network. If it comes down to it, testing and collecting data for the wireless mesh network may need to be simulated in some other sort of software such as Matlab or possibly even NS-2. NS-2 is designed specifically for simulating networks, even wireless ones. However, one drawback to the NS-2 interface is that it would be very hard to "randomize" and therefore create a real world type scenario when testing the mesh network. Either way, we will see which direction the project ends up going.

II. FirmWare

Apparently there are many different type of firmware available. Open-WRT, and DD-WRT are the two different distributions that were covered in this document. Both of them provide an easy to use web GUI to access, and both provide some sort linux terminal access (through putty using ssh or just a terminal client). Originally this project was geared towards the Open-WRT projects, however, after some research and trials, it was found that DD-WRT actually seemed to have more built-in support for Open-VPN.

Besides the different types of distributions, there are actually subsets of each distribution:

1. Micro – The smallest of the distribution, is basically just a functional system, but there is no user-friendly interface to interact with. This would be used in a case where space is limited on the router.
2. Bin – This is the standard image, slightly limited in the capabilities of the router and will function basically the same as the default Linksys firmware but more effectively.
3. PPTP -- This is the full blown version of the firmware distribution. It requires the most amount of storage space and has all the features currently available to the software distribution. This is what I installed in both Open-WRT and DD-WRT.

It can be noted that in trying to test the equipment used in this project, it was decided to use other router hardware available to try to communicate with this router. The other routers, while WRT-54G routers, were not designed to have this type of linux distribution flashed to them. In this case, in order for the firmware to work, a smaller "Micro" version as discussed was flashed to the routers. Unfortunately at this time, it is unclear as to whether the routers will be allowed to run OpenVPN software because of their limited space. In order for them to have OpenVPN installed, it may be necessary, and possible to add external memory to the router. Apparently, it is a fairly inexpensive and easy to do process. However, that is outside the scope of this project.

III. Firmware Installation

To begin the project, a Linksys WRT-54GL router was purchases from Amazon.com for $57. The L stands for Linux, because some of the firmware for these router boxes are written in Linux so that open source software can be uploaded to the router box, such as OpenWRT. At first it was understood that flashing the firmware to a Linux distribution would be quite complicated, so tutorials were used to step through the process. The WRT-54GL was flashed fairly easily, however, and then beginning to configure the router was not quite as simple. The web GUI that comes installed with the firmware is pretty straightforward. However, to do more complicated configurations, putty was used in telnet to configure the router. I am pretty new to this whole Linux world. I have a basic idea of Linux, but the idea of Linux running on something other than a PC still confuses me. I have had a little experience using putty and telenet. Putty seems to be the easiest way to access the backbone of the firmware. However, the Linux commands used to access the backbone are still new territory for me.

IV. Wireless Bridge

After messing around with the firmware for a while, the first goal of mine that I thought was fairly simple was creating a repeater bridge. One thing I have noticed about networking, especially wireless networking, is that one second nothing works, and then the next, everything works. This seemed to be the case here. I spent what seemed like hours just trying to get the WRT-54GL to act as a client and connect to another secured wireless connection. It just suddenly started to work; a combination of the correct configuration and the right about of rebooting seemed to do the trick. It is pretty cool now that the router can wirelessly connect to another secure router and rebroadcast the wireless signal as well as create its own local network through the Ethernet ports in the back. The reason behind this possibility is that the Broadcom microcontroller built into most Linksys routers apparently has two wireless controllers. It is unclear as to why Linksys does not allow this compatibility in its own firmware distribution. Maybe they did this so that Linksys' parent company, Cisco, can charge a much greater amount for a piece of hardware capable of this.

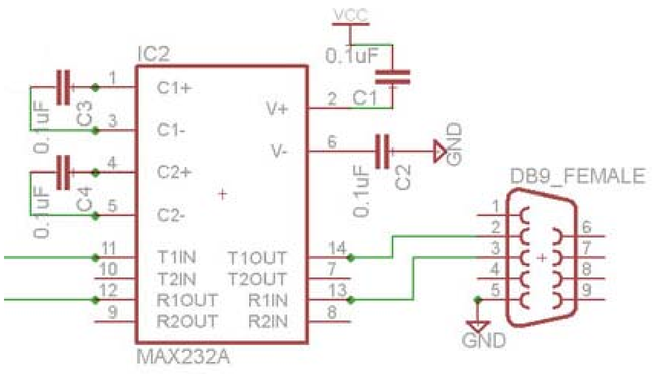
Another possibility I would like to explore in this project is the question of whether the wireless link between two routers is in fact stronger (and hence provides fast data rates, because the SNR would be lower) than the link between a router and a laptop or PC. If this were the case, it would be evident that it would be better (and cost about the same) to have a router to router to lan connection than have a wireless connection because some PC do not have direct access to a lan connection and therefore need to have a wireless card installed.

V. OpenVPN

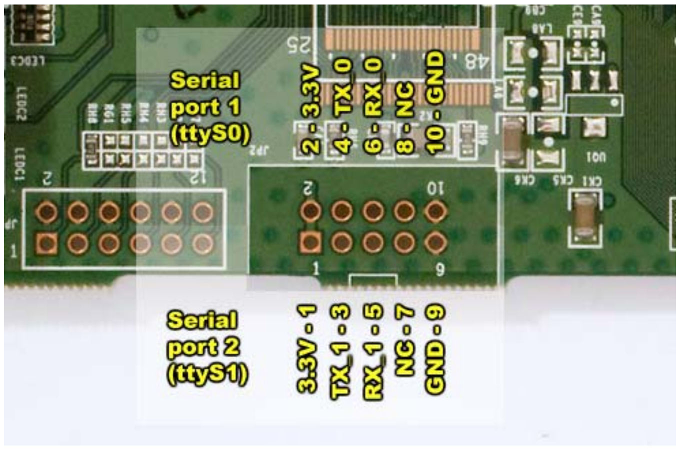
Currently, OpenVPN has posed problems. I have successfully installed OpenVPN using the tutorial documented; however, I have yet to get the router to function as an OpenVPN server. Instead I have successfully gotten the WRT-54GL to connect to a OpenVPN server running on a desktop computer. While this is somewhat of a step forward, it is not the overall goal here. The goal of this project was to implement this router as a client or a server so that it can connect anywhere remotely and connect two remote networks together.

VI. RS-232 Interface

The RS-232 Interface should be pretty straightforward but has not yet been tested. This requires additional hardware that I will need to build onto a simple PCB board. The layout is as follows:



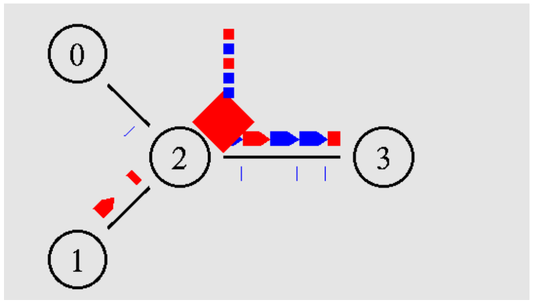
This PCB then needs to be connected to the Interface on the router and the RS-232 on a computer to complete the communication loop. The two leads coming out of the MAX232 chip will be the leads going to one of the serial ports on the WRT-54GL PCB. The DB-9 connection of this PCB will then go to the PC of a computer to display any data that might be programmed to run through this system. Shown below, the connections needed for the connection on the WRT-54GL.



VII. NS-2 Software

After much toiling, the NS-2 software was successfully installed on Ubuntu Linux. Several attempts were made just to get NS-2 to run inside an emulator such as Cygwin for windows, but to no success.

After successfully getting NS-2 to run in Ubuntu, a simulation was needed to try it out. A simple simulation was created simulating 4 nodes. In the figure below, Node 0 is sending via TCP and is therefore expecting and acknowledgement back, which is shown in below in the response from Node 3, which is the destination. Node 1 is sending via UDP and not expecting an acknowledgement and just keeps sending the data that it needs without any response. Since all packets are being routed through Node 2, the figure shows with the large red square, periodic packet loss because of overuse of the node.



VIII. Conclusion

A lot of the testing that needs to be done is awaiting the completion of much of the hardware that is in the works, but is yet to be completed. Once the RS-232 port is in place, and the Open-VPN connections are fully functional, then testing can begin; i.e. such things as optimal data paths through multiple router connections, etc.

After this project is complete, the researcher will have a better understanding of routing protocols in using OpenWRT. The user will also become more familiar with the type of algorithms that must be run when computing the best possible routes for a system to run through. The researcher will also become more familiar with the type of software that must be written in order to enable simple communication over such a network.

Appendix

Appendixes, if needed, appear before the acknowledgment.

Acknowledgment

David Tietz thanks the University of Florida, the Electrical Engineering Department at the University of Florida, as well as Dr. Latchman for the guidance, support and providing the resources necessary to complete this research.

References

[1] OpenWRT: Wireless Freedom (2009, January) Available: http://openwrt.org/
[2] The Consolidated Hacking Guide for the Linksys WRT54GL (2009, January) Available: http://www.linuxelectrons.com/features/howto/consolidated-hacking-guide -linksys-wrt54gl
[3] OpenVPN: The Open Source VPN (2009, January) Available: http://openvpn.net/
[4] Wifi Robot: A Robot that implements a WRT54GL router for its control system. (2009, January) Available: http://www.jbprojects.net/projects/wifirobot/

[5] Tutorial on how to upload linux firmware to router: http://wiki.openwrt.org/OpenWrtDocs/Installing
[6] WRT-54GL Used as repeater example: http://wiki.openwrt.org/Repeater
[7] WRT-54G Used as OpenVPN service: http://forum.openwrt.org/viewtopic.php?id=12979

David Tietz (NYAM'09) is a student at the University of Florida, where he is a candidate for a Masters of Science in Electrical Engineering. He recently graduated in May of 2008 at the University of Florida, Gainesville with a degree in Electrical Engineering. His primary interest in research is in embedded systems work such as microprocessor and FPGA integration into real world designs. Tietz will graduate in the Spring of 2010, in which he hopes to obtain a rewarding career in embedded systems in the Central Florida Area.