By John Champa,* K8OCL

HSMM

Communicating Voice, Video, and Data with Amateur Radio

The Hinternet and VPN Projects

This column is guest authored by Dave Stubbs, VA3BHF, HSMM Virtual Private Network Project Leader. He can be contacted at: <va3bhf@rac.ca>.

One of the stated objectives of the ARRL HSMM Working Group is the creation of the "Hinternet," an amateur-radio-run network that has capabilities similar to those of the internet and can operate as an alternative to it. This aim harks back to the original ideas of the creators of ampr.net in the early to mid '70s The general guiding principle in this effort is the pervasive use of radio as the "physical layer" of the network. This is, after all, an amateur *radio* pursuit.

The current work of the HSMM Working Group has some very distinct implications when considered against the goal of spanning the globe. As an example, the most common network technology under evaluation, 802.11b, is practically useful inside a quarter mile unless using high towers and directional antennas, which extend the range to less than 15 miles, or maybe up to 30 miles with very carefully applied antenna work and a bit of luck.

The HSMM OFDM modem currently under development by the Working Group has the potential to be used in several of the existing amateur bands. One of these is UHF, where the OFDM modem, if it is anything like regular FM

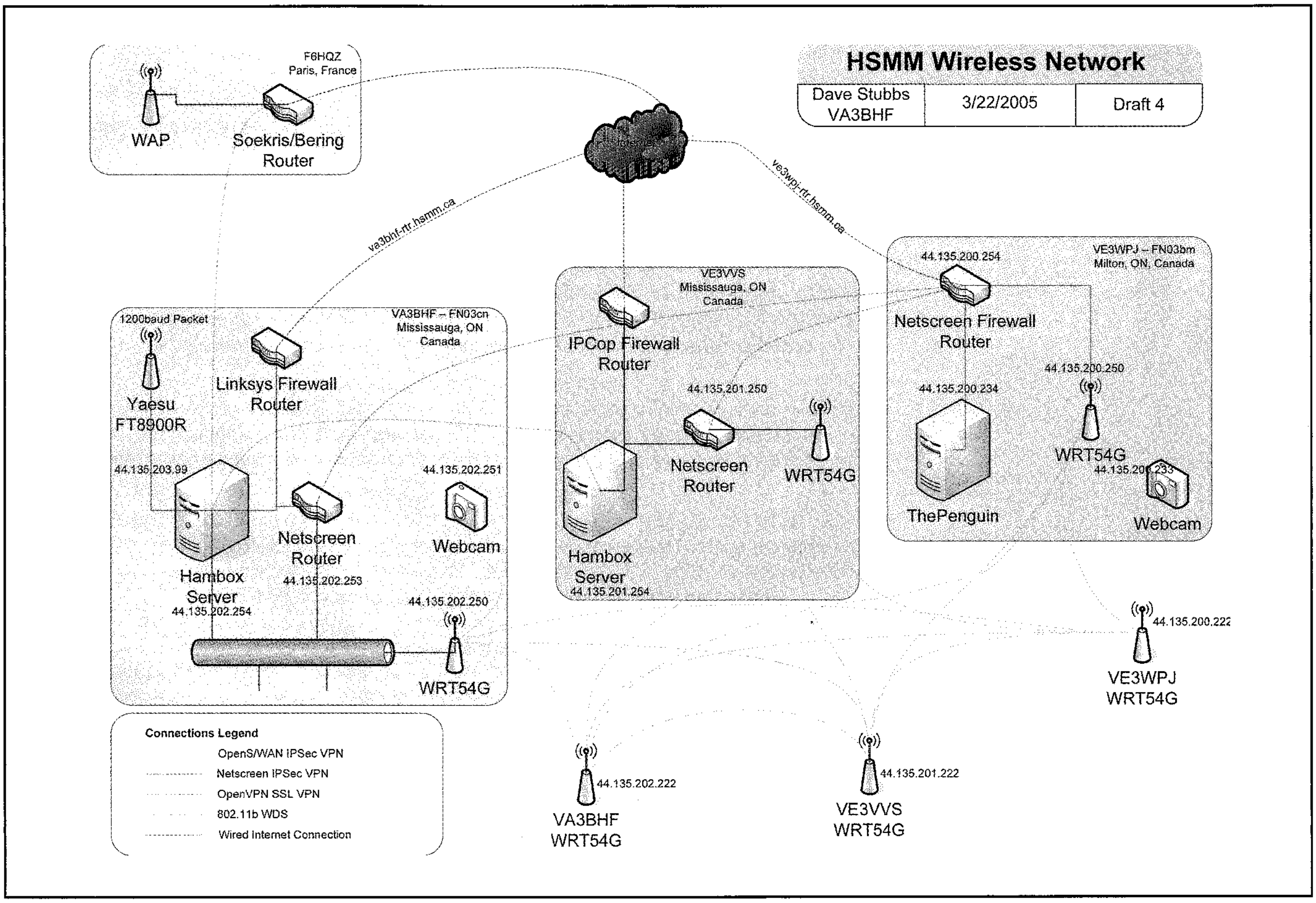

Figure 1. The HSMM Virtual Private Network test system.

*Chairman of the ARRL Technology Task Force on High Speed Multimedia (HSMM) Radio Networking; Moon Wolf Spring, 2491 Itsell Road, Howell, MI 48843-6458
e-mail: <k8ocl@arrl.net>

Definitions and Terms

AH – Authenticated Header
ARRL – American Radio Relay League
HA – High Availability
HSMM – High Speed Multi Media
MMC – Microsoft Management Console
NAT – Network Address Translation
NOC – Network Operations Center
OFDM – Orthogonal Frequency Division Multiplexing
OSPF – Open Shortest Path First
PPTP – Point-to-Point Tunneling Protocol
RIP – Route Information Protocol
RSA – An encription algorithm that was invented in 1978 by Ron Rivest, Adi Shamir, and Leonard Adelman. The "RSA" acronym comes from the first letter of the names of the inventors.
SLA – Service Level Agreement
SSL – Secure Sockets Layer, a protocol that Netscape developed for transmitting private documents via the Internet.
VPN – Virtual Private Network
WG – Working Group

voice, will likely be able to cover somewhere up to 100 miles. There may be flash improvements from tropo-ducting and other similar phenomena, but always-up network reliability can't be based on transient effects like that. Another exciting possibility is the use of the OFDM modem on the 6-meter "Magic Band." However, it should be remembered that 6 meters, while good at times, is quite decidedly dead at other times.

The bottom line is that these methods currently being pursued are locally reliable, but over long distances they are spotty at best. To practically extend a network around the world, some other type of link is required—at least until a good long-distance RF method arrives that provides fat bandwidth around the planet, for example an AMSAT Phase III or Phase IV satellite. That link is secure, easy-to-use tunnels through the internet. Such connectivity would tie our various experimental efforts into one big, planet-spanning network, which we could begin to *use* for things such as message passing, digital group chats, voice/video conferencing, emergency communications, and other possible uses as we come up with them.

Guidelines for the Hinternet

Simple—This is for use by radio amateurs. Only some are computer/network geniuses. Wide use will require usability by more than the comp/net people.

Affordable—Should use cheap (but good) hardware, or "already have it" hardware. Should, where possible, leverage free software.

Dynamic—Should be able to adapt to a constantly changing network, with parts of it going online and offline. Realistically we are not carrier-grade NOCs. We are hobbyists. We won't have SLAs on our networks or HA setups that are always there.

Interoperable—Where possible, we should support most (if not all) of radio amateurs' favorite platforms. Practically, this would be Windows®, MacOS®, and Linux. If done correctly (i.e., simple bootable CDs or USB keys), this requirement virtually disappears, and the connecting system becomes just another "rig."

Secure—We should be able to leverage quality crypto to keep the tunnel part of our network scrambled and protected while traversing the public internet, and it should be done easily.

Interesting—While we should investigate ways to make this a simple "rig" or "appliance" to the non-network-guru ham, there should also be allowance for the tweaker and experimenter to deal with this infrastructure at his/her level. If it's boring, it will be gone soon.

Scope of Research

The following are some of the platforms and VPN technologies under active investigation. First, the operating systems:

Windows—Preferentially working on Windows 2000 and XP. Win9X is so embarrassingly obsolete it gives me hives. Windows is being evaluated, not so much for use as a core part of the network, but more as a way to get many more hams linked into it from their personal shack computers. It is expected that the Windows support will become increasingly important as we begin to "use" the Hinternet.

Linux—Debian Linux is the platform of choice, because it is commonly agreed to be the "ham friendly Linux." This includes, of course, useful debian extensions, such as Knoppix and Morphix, as applicability permits. Linux has an industrial strength network implementation, amateur radio support right in the kernel, and a veritable wealth of tunneling technologies available for use. Many of these work well when communicating with Windows machines. Linux is a main contender for the "core" part of the Hinternet.

MacOS—Preferentially working on MacOSX because it is a nice, proper, reliable, robust, stable, UNIX-based operating system with a properly accessible networking stack. The Mac, like Windows, is being evaluated as an end-point for the network. No one will be expected to donate their shiny Dual G5s to be core routers.

Embedded—In seeking to come up with a rapidly deployable, easy-to-use "rig," it is important not to rule out well-made boot-and-run solutions such as m0n0wall, which runs on a light version of BSD.

Maybe more important than the operating systems that run our network is the actual VPN tunnel technology that gets our packets around the planet. The current commercial focus on network security is driving the creation of many different VPN technologies. We only benefit from this as it provides us with a rich assortment of tools from which to choose. The technologies being evaluated are as follows:

IPSec—The ultra-secure, but very complicated crypto technology that spans almost every possible platform available. It comes native in all versions of Windows after and including Windows 2000, and shows up as the penSWAN/FreeSWAN/SuperFreeSWAN package *or* the IPSec tools package on Linux, and as KAME on BSD and MacOSX. In addition, it is built into many different routers, such as Netscreen, Cisco, LinkSys, SMC, and Nortel. It generally requires several late nights and lots of coffee to get IPSec working properly on any platform, including Windows, and it tends to give firewalls and NAT-gateways fits. Based on almost a decade of love-hate with IPSec, I categorically vote that it be relegated to the "interesting" zone, reserved for the tweakers among us. It should be noted, as an aside, that IPSec is very modular, and the AH portion of it would probably be the most technically correct method of securing amateur wireless communications without scrambling the content. However, that would still require the aforementioned late nights and coffee, and a very capable key distribution infrastructure to support it. It is also probably outside the scope of this project.

OpenVPN—The current favorite *du jour.* This is a completely free, SSL-encrypted tunnel program that runs on all of the above platforms. It relies on the very solid OpenSSH software for its crypto, and is so simple from a network point of view that it slips through firewalls, NAT-gateways, and even multiple cascaded secure corporate web proxies with ease. Its configuration takes the form of a text file (yuck for Windows users), but the file is so small that it can be understood after about 5 minutes of reading. Highly recommended.

PPTP—The drop-dead-easy tunneling technology built into all versions of Windows, as far back as Win95. Since it interoperates with Linux quite well, it has lots of promise as a quick end-point-connection technology. Concerns about security and hack-ability make its core use a bit risky. However, with a light-but-solid grasp of networking and routing basics, it is possible to set up truly massive meshes of VPNs crossing the world with this protocol—and it works quite well. Cautiously recommended.

Currently Running

As can be seen in the accompanying diagram, our test system is becoming quite extensive. What follows is a description of the setup as it exists today, with a few notes about relevant lessons learned from examining these solutions.

OpenSWAN—The VPN between Mississauga and France runs on this Linux-based implementation of IPSec.

Progress Report on the VHF OFDM Modem

By John B. Stephenson, KD6OZH

1. Introduction

The purpose of this project is to test wide-bandwidth digital transmission for applications such as image transmission in the Amateur Service on its VHF allocations. I am doing development and testing, under the auspices of the ARRL HSMM Working Group. If this system proves successful, the ARRL may petition the FCC for use of up to 200 kHz bandwidth in the VHF amateur band. The current petition for "regulation by bandwidth" requests 100-kHz maximum bandwidths on the VHF bands.

Current regulations limit bandwidths to 20 kHz on VHF amateur bands, so testing is authorized under an FCC STA (Special Temporary Authority) that is effective until September 10, 2006. It authorizes emissions with a bandwidth up to 200 kHz in the band 50.3–50.8 MHz. This frequency range is consistent with both ARRL national voluntary band plans and applicable local band plans in the Fresno, California area, where testing will be done. Specifically, in those band plans the segment 50.3–50.6 MHz is designated for "all modes" and the segment 50.6–50.8 MHz is designated for "nonvoice communications." The STA authorizes 1.5 KW peak, a 200-kHz maximum bandwidth, and 384 kbps maximum data rate.

2. Description

The modem being developed is a modification of the UHF offset frequency division multiplexed (OFDM) modem tested on the 70-cm amateur band. OFDM splits a high-rate data stream into multiple parallel low-rate streams transmitted on multiple subcarriers. As the signal propagates, it is reflected and refracted, giving rise to multiple echoes that corrupt data. OFDM slows the rate so those echoes are confined to small gaps between transmitted symbols. The technology is similar to that used on HF bands, but the data rate of each subcarrier is higher, as the maximum path length is considerably shorter.

This modem consists of software written in Verilog, assembly and C language that runs on a DCP-1 card containing a Xilinx XC3S400 field-programmable gate array (FPGA) and Oki Semiconductor ML67Q5003 microcontroller. One DCP-1 attaches to the intermediate frequency (IF) output of a modified ATR-2000 receiver at one end of the link and another attaches to small printed circuit board, with a quadrature modulator IC (Atmel U2790) and the power amplifier from the ATR-2000 at the other end of the link. The transmitter is in a fixed location (Lat: 36d 46m 30s N, Lon: 119d 46m 22s W) and can generate 150 W PEP into an 8-dBi gain vertical antenna (Diamond Antenna DP-GH62). The receiver is mobile and uses a Hamstick antenna mounted on the roof of an SUV.

The DCP-1 and ATR-2000 are documented in articles published in *QEX*. The UHF OFDM modem specification is on the ARRL website (http://www.arrl.org/hsmm/), and a new version,covering VHF operation, of the document will be published as part of this testing. Documentation of the modem hardware and source code will be made available to all radio amateurs after testing.

Three data formats will be tested. These are designed for 50-, 100-, and 200-kHz channels. Initial tests will use 750-Hz subcarrier spacing with an inter-symbol guard interval of 1/8 symbol period. This combination was selected to support relatively small channels and still have a guard interval that allows rejection of inter-symbol interference in any conceivable operating environment. Field testing of ATSC (digital) television systems showed a maximum delay spread of 90 μs and the resulting guard interval for the VHF OFDM modem is 166 μs. The resulting system should be able operate over LOS or NLOS paths in most terrain with directional or omnidirectional antennas. The signal formats use a central pilot carrier and upper and lower sidebands with 24, 48, or 96 data subcarriers plus 2 trellis-terminating subcarriers each. The subcarriers use differential 8-ary phase shift keyed (PSK) modulation with rate 2/3 convolutional coding. The resulting encoded data rates are 96, 192, and 384 kbps with user data rates of 64, 128, and 256 kbps.

3. Progress

Verilog code has been generated for the VHF OFDM modem and 95% of it has been successfully simulated. Simulation has taken longer than expected due to problems associated with IP (intellectual property) upgrades in the Xilinx development software. Modifications to the original UHF modem included:

A. Reducing the size of the finite impulse response (FIR) digital low-pass filters to allow three programmable filter stages instead of two.

B. Increasing filter coefficient precision from 18 to 24 bits.

C. Writing a new encoder and Viterbi decoder to convert from the Ungerboeck trellis-coded modulation (TCM) using a four-state encoder in the UHF modem to a bit-independent coded modulation (BICM) with eight states.

D. Modifying the fast Fourier transform (FFT) implementation to support 17-bit data and reduce memory requirements. The FFT multiplexes and demultiplexes the subcarriers.

E. Modifications in multiple modules to allow different numbers of subcarriers.

F. Converting interfaces between circuits using different clock rates from using block RAM (random access memory) in the FPGA to distributed RAM in the FPGA to free block RAM for additional filtering.

The change from TCM to BICM was made to increase performance over fading paths. The TCM scheme was originally designed for additive white Gaussian noise (AWGN) channels as encountered with telephone modems. This will be incorporated into the UHF modem specification. The filtering changes were required to support higher decimation and interpolation rates and provide a better shape factor to limit occupied bandwidth.

Once all the code has been simulated, it will be loaded onto a DCP-1 and the output checked with a spectrum analyzer. Of particular interest is the level of high-order intermodulation distortion (IMD), or splatter, that will be generated in the power amplifier. A signal generator and arbitrary function generator is also available for receiver testing.

The final test will be one-way over-the-air with measurement of bit error rates, allowable SNR (signal to noise ratio), and coverage area. Testing will also be done on the 70-cm band, either before or after the STA expires, to compare coverage areas and amplifier IMD.

It's actually a nice example of different versions of a piece of software working okay together. The Mississauga end of this tunnel runs OpenSWAN, while the France end is running SuperFreeSWAN, which is a slightly older version. The nicest thing that can be said about this tunnel is that it exemplifies well the general truth about IPSec and Linux: Once you make it through the pain and anguish of getting it all set up, it "just works." You can forget about it. Leave it there for years and years, and when you come back to it and actually need it, it's still running.

The chore of setting this up served to highlight some of the more annoying and esoteric "features" of IPSec, however. Because the Mississauga side of this tunnel is on a dynamic address, the whole "easy" part of OpenSWAN setup, which uses pre-shared keys, would not work. It was necessary to learn the more complicated RSA-key-exchange method of authentication, which allowed a dynamic-to-static tunnel to work fine. To make this setup work with Windows IPSec, it will be necessary to learn the even-more-painfully-complicated X.509 key exchange authentication method. This is still on the to-do list, but motivation in this direction is low, because it requires extensive manual configuration (death by MMC) on the Windows side as well, making it unlikely that this would achieve wide use among hams.

One major downside that has come up from IPSec testing, however, is the very different practical reality of policy-based routing. OpenSWAN tunnels do not show up on the computer as regular network interfaces, so normal routing doesn't work on them. This limits the use of dynamic routing protocols on this type of tunnel, and requires a lot of manual configuration. This should be considered a major tick against the use of IPSec.

Netscreen IPSec—some of the VPNs between the various involved parties in Mississauga have been created using Netscreen router appliances. This is mainly because we were interested in these from our day jobs and some of us had these devices sitting around. It is unlikely that these will make it into mainstream HSMM use, but they may provide a nice excuse to try Linux-to-Netscreen VPN testing. For the purposes of this project, however, it is little more than a distraction.

OpenVPN—after a little work on OpenVPN for other reasons, which proved very practical and useful, it was decided to try using this protocol for one of the HSMM VPNs previously provided by Netscreen routers. This tunnel was basically set up within 5 minutes, and full OSPF routing was completed within an afternoon. Further tests of Windows-to-Linux OpenVPN tunnels have been made and have turned out to be just as easy. Using the TAP interface, it is possible to run full OSPF and RIP routing on OpenVPN tunnels (and even mobile-mesh as well), which makes them *very* useful for extending a distributed network and having the routing "just work." For this reason, OpenVPN has enthusiastic support from this project and is recommended as a "first choice" for core network connections.

PPTP—Several of the Linux servers involved in this network have been set up with the *pop-top* server, which allows Windows clients to connect via PPTP VPNs. Tests so far have shown this to be quite reliable, although some manual configuration needs to be done on the Windows VPN advanced settings, which allow passwords to be sent in the clear. There is a possibility to mitigate this with some additional software on the Linux side, which is also on the list for future testing. For now, though, the tunnels run reliably. It would be worthwhile to do some future testing on full PPTP network-to-network routing in addition to just single clients. The pervasive use of Windows among hams makes this endeavor worthwhile.

Future Efforts

A lot of work is being done, and a lot of fun is being had at the same time. However, there is still a long way to go. While we are definitely fulfilling some of the previously mentioned guidelines (*interoperable, interesting,* at least), there is still a lot of work to be done on others (*simple,* especially). The main items follow, in approximate order of priority, from highest to lowest.

Simple BootCD Router—The main Linux system being used for this project is a powerful server, with a Gigabyte of RAM, a powerful CPU, mirrored hard disks, fault tolerant power supply, and a carefully hand-configured array of software on it, doing many things beyond the scope of this project. One of the chief goals of this project is to boil that down into a simple, boot-and-leave-it system that runs from a bootable CD on a reasonable group of readily available laptops. Once attained, there are plans to extend this to bootable USB keys and other embedded systems other than laptops.

PPTP Enhancements on Linux for Secure Authentication—An effort to add components to Linux to fully support the exchange of encrypted passwords as the tunnels get set up. This is useful for Windows clients as well as Windows Server VPNs, so it has a medium-to-high priority.

Full Router Kit Spec—This effort is an extension of the above Simple BootCD item, which specifies an easy-to-build kit using something like a Soekris embedded system board, with a ready-to-boot CF card for it.

OpenVPN in LinkSys WRT54G—Taking the idea further is the possibility of putting a major squeeze on the software requirements for the RMAN-VPN router and loading it all into the brilliantly amazing yet very cheap LinkSys WRT54G router, which is already specified in the S.H.A.R.K kit spec.

OpenVPN Windows-Linux Fully Routed—Since OpenVPN runs so well on Windows, it is worthwhile to investigate it as a full alternative to the Windows PPTP tunneling capability, especially considering its robustness, and the nimble way in which it works through firewalls, gateways, and proxies.

PPTP Windows-Linux Fully Routed—An effort to test the use of Windows Server VPN routing connected with Linux, as an option to deploy a major portion of the network on Windows, should the need arise (a sudden influx of funding dollars from Microsoft Research, for instance).

IPSec X.509 Authentication—A low-priority effort to extend knowledge of IPSec and have it fully working between Windows and Linux.

Requests from the HSMM Working Group

In short, the main request from this project is for people to get involved. As development of an easy-to-deploy kit progresses, it would be great to have some more nodes connected to this distributed network, and even get to the point where our regular conference calls are held on the network instead.

It would be great to get to the point where we do not consider a node really "deployed" unless it is connected into this distributed network.

Summary

This project, for a network-engineer-turned-ham, is a lot of fun, and also a great challenge. It is hoped that the technologies under experimentation in this effort can be put to good use as a simple method to hook together our various HSMM experimental networks, and by bringing them together, achieve a critical mass where "the whole is greater than the sum of the parts," giving us a great tool to use in communicating and supporting our emergency response services.

The ultimate goal of this project, of course, is to become no longer needed, as we blanket our planet with long-distance RF links, making VPNs no longer necessary. However, until then…

If you have any questions or suggestions about using VPNs on the Hinternet, or if you perhaps would like to get involved in the HSMM VPN Project, please feel free to contact me: Dave Stubbs, VA3BHF, HSMM VPN Project Leader, at <va3bhf@rac.ca>.