A DTN WIRELESS SENSOR NETWORK FOR WILDLIFE HABITAT MONITORING

A. Tovar, T. Friesen*, K. Ferens*, and B. McLeod
University of Manitoba, Winnipeg, Manitoba, Canada (tovar|ferens|mcleod@ee.umanitoba.ca)
*TRLabs, Winnipeg, Manitoba, Canada (friesen|kferens@win.trlabs.ca)

ABSTRACT

This paper reports on a preliminary study of applying a wireless sensor network using DTN (Delay Tolerant Network) to sample the current status of a certain wildlife mammal with the name of White Tail Deer (Odocoileus virginianus) in the WMU Area 47 Ontario, Canada North of Parry Sound district. System architecture is proposed to obtain the highest monitoring range achievable with the least amount of sensors. With this method we can obtain the status of white tail deer, and detect other kinds of animals as well. The current method for surveying the white tail deer needs to be reviewed in order to obtain a most frequent status report of this important animal. By using DTN we expect to obtain disruption support capabilities in our sensor network. Furthermore, we simulated the wildlife habitat monitoring system on the PlanetLab environment. Our future work will develop sensors, deploy them in the field, and create a DTN network for the monitoring of wildlife habitat.

Index Terms— Delay and disruption tolerant networking, (DTN), PlanetLab, network simulation, sensor network, wildlife habitat monitoring.

1. INTRODUCTION

The habitat monitoring of Odocoileus virginianus and the terrain where they live will allow the people of Ontario Canada to respond in case of a large decline or rise of the population, and that can have a great impact on the ecosystem (This has been well documented in part of Ontario where hunting is prohibited and predators are scarce, like Long Point National Wildlife Area, Pinery Provincial Park, Point Pelee National and Park and Rondeau Provincial Park). Also it will allow Ontario's citizens to open or close the hunting season based of the data acquired. "Deer are a valuable resource for Ontario's citizens, providing both viewing and hunting opportunities, along with the associated economic benefits." [1]

There are many survey techniques for wildlife counting; the current method used is called "Pellet Group Count". It consists of two parts. First, a helicopter is used to obtain a $1km^2$ topographical map to stratify the area. In addition, a number of sampled plots are selected based on the expected density of deer per stratum. The sampled plot is an equilateral triangle of 1km per side. The second part deploys a two person crew ground survey, which begins from each start point obtained from the sampled plot, and then proceeds to walk all three sides of the triangle plot looking for dead deer. Pellet group counts are conducted at evenly spaced intervals. There are five pellet group subplots per side of the triangle, each 40 m long and 2 m wide. The entire area of each subplot is searched, and all deer pellet groups observed are counted. At the end of the survey, the number of pellet groups counted (and the total area searched for pellets) is summed for each stratum. The total is divided by the mean daily defecation rate for deer, and by the number of days since the fall of leafs, corrected for the estimated number of dead deer, and then extrapolated out to the total area of the stratum to obtain the population estimate for that stratum. The population estimates for each stratum are added to obtain a population estimate for the total area [1]. The limitations of this method are: first, it needs 2 professional pilots and 2 trained observers to stratify WMU area 47, taking around one week of flight in a helicopter to obtain good quality stratification. Aircraft costs make it difficult to conduct this survey on regular basis. Second adequate numbers of trained observers must be available for the two or three week period in the spring when the ground survey is conducted. It is necessary to have access to get close to the plot start point. Weather conditions will affect pellet group observation ability and flight stratification. Therefore, this technique takes too much time and human and economic resources for its success. The regular use of this technique is not possible for economic reasons and for weather factor. The first part of the process is usually done before spring (late winter); the weather conditions have to be good. The second part usually takes place in late spring. Weather conditions and road access affect the ground survey.

This paper proposes to use a wireless DTN sensor network. The DTN obtains the required sensor data. Subsequently, a data mule (helicopter or automobile) is used to recover the data from the principal node in the network via DTN Intermittent schedule contact from a node on the site (capable of storing data, i.e., the custodian). Each network will be located in key areas where the density of white tailed deer has previously been marked as high. After all the sensor network areas have been selected, the data mule will trace a route to recover the data of each principal node. With this method it will be possible to obtain the sensor data on a regular basis; sending a helicopter or automobile to recover the data passing through the traced route, and then an analyst will be able to access the data from a personal computer in home or office when the data reaches the gateway. Using this system will lead to less time to acquire the sampled data than current survey method: the acquired information is more accurate because it is based on current deer number, also the data can be used for terrain recognition and monitoring of other kinds of species (animal or plant). In addition, with the passage of time this method will lead to less use of economic resources. A wireless sensor network habitat monitoring has been done before [2] but in this paper we use DTN as transmission protocol instead of TCP; this will add disruption and delay support to the wireless sensor network; it will also secure transmission reception in any kind of environment.

One of the difficulties with developing a DTN application is that properly testing the application with a large number of nodes can require a great deal of resources that may not be available. To setup and configure a volume of nodes that is reflective of the intended application can be time and money intensive, and is generally not possible in the early stages of development, which benefit from the ability to rapidly prototype and test initial configurations.

For most projects, researchers will develop one-of small scale models of the environment they wish to test, such as in [3] and [4]. This is useful when the variable under test is the performance of the networks or algorithms under certain conditions, rather than the application itself. In still other cases, researchers made mathematical extrapolations from existing datasets, as in [5]. Again, while this is useful for testing routing algorithms or estimating the effectiveness of an approach for successful communication, the application itself remains untested. For true development, it is generally desirable to perform most extensive testing with the designed application software itself. In many cases, the details of the simulation environments were unfortunately not extensively discussed [6].

The most extensive simulation environment was created for SPINDLE [7]. In the SPINDLE Final report, the researchers attempt to evaluate DTNs for the US military. Their testbed system was dubbed Multi-VM Infrastructure for Networked Applications and Systems, or MINAS. Basically, MINAS consists of multiple virtual machines, or VMs, running on a single host computer. This virtualization is achieved using User Mode Linux, or UML. This allows for a complete and fully segregated operating environment for each simulated node in the network. Connections between nodes are managed through the Network Simulator 2, ns2. The ns2 is an easily extensible, open source network simulator popular in academic research [8]. It allows the developer complete control over the network environment. He or she can easily control latency, bandwidth and availability of nodes and links through a handful of scripts and configuration files included with MINAS. In the SPINDLE report, the researchers identify several advantages to this approach. As the simulation is run on a single host machine, each developer can have his or her own simulated network to play with; developers need not worry about their experiments or tests conflicting with the activities of other researchers. MINAS also uses full virtualized environments, giving the developers a great deal of power and control as to what software is running on what node. The tools used, Linux and ns2, are well-known and well-documented, enabling a short learning curve. However, the approach used in MINAS has some limitations. Most notably is that the level of scalability depends on the capability of the hardware available. Each node simulated is a full OS, so running multiple nodes can quickly consume the host's resources, such as RAM and CPU time. Even on a high-end quad-core machine, the researchers only ran about 20 experiments at a time, though the researchers identified several steps that could be taken to improve scalability. Ultimately, if the tests require a significantly larger number of nodes, a single host would be insufficient, and at this time, MINAS seems to provide no mechanism for having a simulated network bridge multiple host machines. The advantage that each developer has his or her own testing environment can also be a downside, as then this limits the ability for developers to share experiments and environment setups. The developers of MINAS note with the current approach of using User Mode Linux, the VMs and the emulator itself run as processes under the control of the host OS. Due to potential idiosyncrasies in scheduling, this breaks the guarantees that the virtualized OS's will get enough CPU time, etc so as to act as real-time systems. Also, under MINAS, a relatively old version of the Linux kernel is used for the host (2.6.9, released 2004, as opposed to current, 2.6.30), limiting the environment's appeal and availability of up-to-date tools.

PlanetLab [9] is a shared testbed network that allows for some limited real-world testing, with tools that allow for emulation of further real-world variables. The size of the PlanetLab network (over 1000 hosts!) allows for testing efforts to scale up to very large sizes, while management tools provided for PlanetLab allow relatively easy maintenance of the growing network. The PlanetLab Network is an open, distributed platform for research and development. While the individual nodes are maintained by those who donate them, the network itself is developed and maintained by a project at Princeton University [9]. Research labs and educational institutions interested in using PlanetLab can join the consortium by donating computers and using them to host one or more nodes to the network. Interested researchers can then create a "slice," which is a collection of virtual machines that run on the hosts scattered across the PlanetLab network. Hosts are added to the slice by a variety of tools provided by PlanetLab, at which point a virtual machine is automatically created on the host, and access is granted to the user. PlanetLab contains over 1000 hosts at nearly 500 different sites. Every continent except Antarctica is represented on PlanetLab, offering an incredible diversity of host location. Most connections between hosts occur over the public Internet, but when possible, high-speed interconnects between hosts is used, such as the academic networks CANet or Abilene. Once the virtual machines are setup, they run an identical variant of Fedora Core 8, with a few minor modifications to accommodate the virtualized environment. However, the OS runs as close to "bare-metal" as possible, keeping the environment "real-world" while limiting any headaches due to differences in configuration. A number of utilities exist to manage the nodes of a slice on PlanetLab. The PlanetLab project maintains a list of available software on its website. Because our setup was relatively small, node management was done using parallel ssh (pssh) [10] and basic shell scripts. Parallel ssh allows commands to be executed on multiple nodes simultaneously via the ssh protocol. The suite also includes a utility for copying files to multiple nodes, based on secure copy, or scp. The nodes themselves were manually selected through the PlanetLab slice management tool on the PlanetLab website. For larger deployments, some of the utilities and APIs made available by the Planet-Lab project should be used to ease maintenance and initialization of the environment.

2. WILDLIFE MANAGEMENT UNIT 47

The white tail deer (Odocoileus virginianus) [11] are found throughout southern and central Ontario, as well as the southern portions of northwestern and north-central Ontario; however, higher concentration can be found in the WMU 47 area, which is located in Southeastern Ontario in the Parry Sound district. This unit has an area of approximately 4,000 km^2.

2.1 REQUIREMENTS

For the proposed DTN based wireless sensor network for wildlife habitat monitoring to be successful, there are functional and quality requirements which must be met.

The wireless sensor networks should be strategically located to enable automobile data gathering by nearest town personnel, instead of a helicopter. Furthermore, the sensor nodes will be statically located within the sensed region. (Other systems may place nodes on animals for tracking purposes, which would make the network mobile). However, some of the nodes may not be accessible by car so the helicopter is still needed; nevertheless, by locating some nodes easily reached by car it may decrease the flight time of the helicopter saving some valuable economic resources. A constraint here is that the sensors should be distributed in areas where the population of the white tail deer is high to increase the statistical significance.

By choosing a radio technology to cover a large area, we will be able to cover the whole Unit 47 with fewer nodes; therefore, the initial economic resource expended will decrease greatly; also, maintenance cost will decrease.

Long periods of autonomous functionality without human aid will decrease the time and resources expended on the maintenance of the nodes.

Each node should be able to handle the habitat environment without losing performance in order to have long periods of autonomous operation. The weather conditions in the area range from −38˚C in winter to +36˚C in summer. Rain and snow are also common in their respective season.

In some cases the sensors and the principal node may have intermittent interruptions (due to environmental hostility); thus, the wireless transmission should be able to support this kind of unexpected disruptions without losing data.

By designing the principal node in the wireless sensor network to send the information using a relatively high bit rate to the data mule, a helicopter or car may be able collect each node information faster reducing data pick up time.

Sensor data archiving is essential in a DTN wireless network, given that DTN uses store and forward transmission, so each sensor and principal node should have a persistent flash memory to store data.

Time synchronization is extremely important in each wireless sensor network, because we are using scheduled contact to activate the system for transmission and reception. Therefore, each sensor and the principal node must be synchronized.

3. SYSTEM ARCHITECTURE

We now describe the system architecture and offer some solutions to the requirements above. The choice of a camera is a tradeoff between cost, range, and image size. To maximize range and minimize the number of sensors, we chose a 4MP camera, with no audio capability, and small lens (50mm). Other sensors like photo resistors, IC temperature sensor, and humidity sensor could be useful for gathering more environmental data. A 360˚ rotational capability is needed to scan the area. Also included is a solar panel battery charger, 1GB micro SDHC memory card, and RFM TR1100 [12] wireless technology (see Figure#1 for more details). By using a 16 dbi antenna we can obtain a range of 370 m, placing antenna as high as possible to get near line of sight. The principal node is a micro server running Linux, with sensor measurement capabilities. The micro server processor can be similar to the Stargate device [13], 400 MHz PXA255 XScale processor, 64MB of RAM and 8GB micro SHDC. To receive the information from the sensors the RFM TR1100 device may be used with the 16 dbi antenna. For the transmission to the data mule a CF 802.11b/g wireless card can be used. Because the principal node has higher power consumption, solar panel cell should be used for battery charging. As for the data mule there is no power restriction, and no sensor is needed; so, it can be a computer server with high range wireless reception range, with a high bit transfer rate to reduce information recollection time from the data mule. The data archiving requirement is solved by installing the micro SDHC card on the sensors and principal node. By using CF 802.11b/g wireless card in the principal node, we solved the fast transmission rate to the data mule requirement since the CF 802.11b/g support a transmission rate of 56 Mbps. Furthermore, the hardware operation range can support the extreme condition presented in the area, so the harsh environment support requirement is solved.

Each wireless sensor network will have four devices, three sensors node and the principal node (that also contains sensor capabilities) organized in a rhombus like shape, the separation between each sensor and the principal node will be approximately 300m. If each sensor camera is able to take clear picture of the environment at a 2 km distance using hyper focus techniques [14], [15], then the area covered by each network is around 14 km^2, so deploying around 100 networks should be enough to cover the important regions of the Wildlife Management Unit 47. Furthermore, by distributing the 100 networks in strategic position around WMU 47, we address the wireless sensor network area distribution requirement. By employing DTN as transmission protocol, the data transmission between the sensors and the principal node will be able to support interruption without losing connectivity; this is very useful in situations where environmental hostility like thunderstorm are constantly cutting the connection. The time of interruption supported is completely dependent on the bundle protocol specifications, and the type of convergence layer used. By using the bundle protocol and UDP as convergence layer, a transmission may be able to support interruption without losing the data. If the acknowledgment option in the bundle protocol is enabled, the transmitter device will wait for the return receipt before closing the connection. If the return receipt is not received, the bundle protocol will not delete the information, and it's up to the application to resend the file. One way to address the sensor network longevity, which is dependent on energy consumption, is to provide an external power source like solar energy with rechargeable Li-poly batteries. Another way would be to make a tradeoff between performance and energy consumption. In this project we make use of solar panel energy to recharge the Li-poly battery (3600mAh approximately 3.5V). A good solar cell of around 15cm by 15cm is able to produce a maximum of 3.3W (price 12$) [16], and by using the rule of thumb we may be able to draw 3.3Wx20%x24 = 15.84Wh = 4525mAh in a day enough to charge the battery. Now that we have energy charge in a day, we need to know the energy consumption of each device. If we follow the specification of the "Canon PowerShot A2000 IS" it will expend approximately 10 mAh for a fully processed image if the sensor takes around 80 pictures in a day it will expend 800 mAh daily only taking the pictures (other systems use motion detection hardware to activate the sampling, but this would require additional resources). The Stargate processor with the CF 802.11b/g WiFi card will consume around 250 mAh at idle [13]. If we maintain the principal node in suspended mode until it's needed and use the RFM TR1100 device instead of the CF 802.11b/g to communicate between the principal node and the sensors, the system will consume less power since the RFM TR1100 only draws 12 mA in active mode [17]. Fig. 1 shows the typical energy consumption of the devices. The energy consumed by the other sensors are low compared to the camera.

Device	Sleep M	Tx(transmission)	Rx(Reception)	Speed	Range
RFMT R1100	0.7μA	12mA	8mA	1Mbps	200m
CF 802.11b/g	25μA	300mA	250mA	56Mbps	150m
Device	**Sleep Mode**			**Active (100%)**	
Sensor Microprocessor	3nA			11mA	
PXA255 system	~40mA (with memory in self refresh mode) and system waiting for RTC signal			400mA	

Figure 1: Power requirements.

4. SIMULATION ENVIRONMENT

The entire simulation was performed on PlanetLab hosts and one external host to access data mule data. We used the latest DTN software reference implementation (mercurial change set 3442:490af2a13378) released by the DTN research group [18]. Deployment was simplified, since the binaries and utilities (DTN reference implementation and our own application) were compiled and configured on a single node, and then copied over to each additional node that was to run the software. On larger networks, we recommend using a tool such as CoDeploy [19], to facilitate deployment of software, and to minimize strain on a local web server. The reference DTN implementation contains several tools for testing connectivity between nodes. For instance, the tool *dtnping* performs a similar role to tradition ping, but over DTN connections. Also, when run in the foreground, the dtn daemon can be used to report useful debugging information and the state of the bundles on the node. The utilities *dtnsend* and *dtnrecv*, also included in the reference implementation, were also used to send and receive files via DTN links. In addition to these tools, we created a few simple DTN applications to supplement testing. Our own applications had the additional benefit of showing that the configured nodes were capable of compiling, linking, and running a new DTN application, if necessary. Using these tools, we were able to successfully communicate over our DTN network in a variety of use-cases.

As shown in Fig. 2, the simulation used the machines called “echo”, “plonk” and “plink,” which represented sensors in the wireless network. The planetlab2 machine represented the principal node; Alice represented the data mule; and “End user” was a personal computer that accessed Alice to obtain sensor data. The data recollection operated as follows: each sensor enabled machine (echo, plonk, plink, planetlab2) acquired measurement data (pictures + other information) by day and saved this data to a hard disk. Each node waited for the schedule time to begin transmission to the principal node (planetlab2 machine). The principal node received the information from each sensor machine, and waited for the data mule (Alice machine) to pass by. When the data mule passed by, the principal node transmitted the collected information from each sensor to the data mule computer server. Later the data mule uploaded its data to a personal computer. For simplicity, we assumed a picture file size of 350 KB, so each sensor will transmit around 28 MB of data by day to the principal node. Using the RFM TR1100 it would take around 4 minutes to transmit all data. The RFM TR1100 transmission power is quite small, so even if it would take 90 minutes for the upload, this would still be within the energy budget.

The sensor software (for echo, plonk and plink machines) was designed (using the C language) as follows:

- Exit sleep mode.
- Checking Battery Status: If battery low do not collect data and enter sleep mode until next day.
- Collect sensor data and create a status report.
- Enter Sleep mode until schedule time for transmission.
- Wait for return receipt signal for each file sent: if return receipt signal timeout then do not drop data; otherwise delete the file sent. If transmission was not successful the data will not be erased and the system will keep the files until next schedule time for transmission.
- If transmission successful then drop the data.
- Close the connection.
  Enter sleep mode until next data recollection.

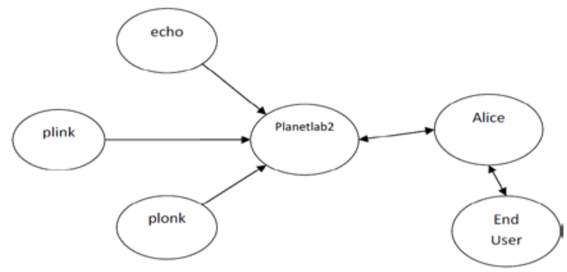

Figure 2: Simulation on PlanetLab.

5. ANALYSIS OF RESULTS

The simulation measured (1) the processing time when the system is in the active mode, (2) disruption support capabilities, and (3) information reliability. Several tests showed that the software developed for the sensors, principal node and data mule took less than 50 seconds of processing time to run the program without counting transmission time. As a result the energy budget taken by the DTN protocol software is not significant compared with energy taken by the sensors and the transmission energy that depend on the radio transceiver hardware. For disruption capabilities support for the proposed method we disconnected a sensor machine (echo, plonk, plink, planetlab2) from the network when the transmission was in progress; we noticed that the information being transmitted by the time of the interruption was not lost, and the information was safely kept in the transmitter HD. This is because the bundle protocol of the transmitter included the wait for custodian receipt signal, and unless this signal is received the transmitter will not delete the data. Furthermore, we disconnected a host when the custodian receipt signal was being sent. In this case we found the transmitter did not delete the file, even though the file was successfully transmitted, but the receipt bundle was lost, as expected. This may cause redundancy in some of the information received by the principal node and the data mule. As for information reliability in the normal operation test (no disruption of any kind) the information obtained in the end user computer was very precise; there were no inconsistencies between the information on the sensor and the information received by the end user computer. In case of a disruption, when some bits of the transmission were modified, the end user received erroneous information, unless a convergence layer like TCP or LTP were used. Either way, depending on the kind of data being received by the end user a margin of error of some bits may not be important, especially because some picture data are irrelevant.

The PlanetLab approach offers greater flexibility than MINAS when it comes to sharing environments between developers. If developers need segregated environments, separate slices and setups can be created for each user. Alternatively, multiple developers can deploy and access a single, common, testbed if the situation calls for it. Probably the most significant advantage to using PlanetLab is the level of scalability available. As mentioned, PlanetLab consists of over 1000 host machines, each capable of running a separate DTN environment. This is contrasted to the 20 or so VMs available under a MINAS setup, and even such a simple configuration required top-level hardware. Furthermore, PlanetLab is designed so that each node is guaranteed to have an identical development environment, facilitating configuration and setup. PlanetLab's scalability can be further extended by including extra VMs on top of the existing host, increasing potential DTN nodes to over 10,000 simultaneously, depending on available hardware. However, dissimilarities in deployed hardware mean that not every host is capable of running a large number of VMs. The use of the Netem tool allows developers to simulate a great deal of network environments, including high delay, high latency and/or variable bandwidth conditions. Unfortunately, PlanetLab prohibits modification of the Linux firewall rules; however, it is possible to overcome this limitation by the use of user space tools. One drawback to using a distributed test environment run by volunteers is that the hosts involved in the experiment may not be under the control of the developer. One or more hosts may occasionally go down for maintenance with little to no warning. For this reason, we recommend that a handful of "hot backup" hosts are maintained, ready to be brought into the network on short notice.

6. CONCLUSIONS

As we know the habitat monitoring of any kind of species or environments represent a special kind of network called sensor networks that consist of spatially distributed autonomous devices used for this kind of task. In this paper we proposed a sensor network using DTN as transmission protocol for the monitoring of the white tail deer in the province of Ontario with the purposed of obtaining a cheaper and reliable sensor network with disruption support for this special kind of task. In this paper we introduce the first DTN wireless sensor network simulation implemented in PlanetLab environment. We also bring in an alternative method for the white tail deer monitoring that has some good advantages, like autonomous monitoring and disruption capabilities. In this paper we prove that using DTN as transmission protocol in a wireless sensor network habitat monitoring is in fact helpful to solved problems like disruptions and delays in the transmission. Therefore, using DTN will add more advantages in the wireless sensor network without inserting any inconvenience.

The habitat monitoring of the white tail deer in the WMU 47 using DTN wireless sensor network can be further improved specially requirements like network infrastructure and sensor range capabilities that impact the quantities of networks that would need to be deploy in the WMU 47. Improving this requirement will decrease the quantity of networks needed to cover Area WMU 47 consequently reducing the economic resource needed to its implementation. Also, using tools like a dummy-net (free BSD) to evaluate more efficiently the transmission protocol and further prove the advantages of using a DTN wireless sensor network. For future work, we plan to apply error detection and correction to mitigate bit errors or loss of data.

7. REFERENCES

[1] Konze, Karl and McLaren, Margaret. Wildlife Monitoring Ministry of Natural Resources. Northeast Programs and Inventory Techniques for Ontario. Ontario Science and Technology. Technical Manual TM-009. 1998.

[2] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, J. Anderson "Wireless Sensor Networks for Habitat Monitoring." [Online]. Available: http://www.polastre.com/papers/wsna02.pdf, [Accessed: January 8, 2010].

[3] J. LeBrun, C.-N. Chuah, and D. Ghosal, aAIKnowledge Based Opportunistic Forwarding in Vehicular Wireless Ad Hoc Networks, a Ai in Proc. of IEEE Vehicular Technology Conference (VTC) Spring, pp. 1aA$5, 2005.

[4] P. Juang, H. Oki, Y. Wang, M. Martonosi, L.aA$S. Peh, D. Rubenstein, aAIEnergy-Efinlcient Computing for Wildlife Tracking: Design Tradeofs and Early Experiences with ZebraNet", in Proc. ASPLOSX, Oct. 2002, San Jose, CA.

[5] A. Chaintreau, P. Hui, J. Crowcroft, C. Diot, R. Gass, J. Scott, aAIImpact of Human Mobility on the Design of Opportunistic Forwarding Algorithms", in Proc. IEEE INFOCOM aAZ06, Barcelona, Spain, April 2006.

[6] Y. Wang and H. Wu. Yu Wang and Hongyi Wu. DFT-MSN: The Delay Fault Tolerant Mobile Sensor Network for Pervasive Information Gathering. In Proceedings of IEEE INFOCOMaAZ06, Barcelona, Spain, April 23-29, 2006.

[7] R. Krishnan, et al. "SPINDLE Final Report" BBN Technologies LLC., Nov. 2006.

[8] The Network Simulator 2 -ns-2. [Online]. Available: http://www.isi.edu/nsnam/ns/. [Accessed June 16, 2009].

[9] Planetlab: An open platform for developing, deploying and accessing planetary scale services [Online]. Available: http://www.planet-lab.org. [Accessed on Feb 17, 2009].

[10] Parallel SSH (pssh). [Online]. Available: www.theether.org/pssh/. [Accessed on June 16, 2009].

[11] Dewey, T. and Animal Diversity Web Staff. 2003. "Odocoileus virginianus". [Online]. Available: http://animaldiversity.ummz.umich.edu/site/accounts/information/Odocoileus virginianus.html. [Accessed Feb 17, 2009].

[12] RFM TR1100 Datasheet RF Monolithics, Inc.

[13] Nilanjan Banerjee Jacob Sorber Mark D. Corner Sami Rollins Deepak Ganesan "Triage: Balancing Energy and Quality of Service in a Microserver"

[14] "Hyperfocal distance guide." [Online]. Available: http://www.dofmaster.com/hyperfocal.html [Accessed on Mar 15, 2009].

[15] Wikipedia Foundation Inc. Depth of Field. [Online]. Available: http://en.wikipedia.org/wiki/Depth_of_field. [Accessed on Mar 15, 2009].

[16] Comercial Solar Cell. [Online]. Available: http://www.siliconsolar.com/ commercial-solar-cells.html. [Accessed on Apr 15, 2009].

[17] Kevin Fall Intel Research "Delay-Tolerant Networking for Extreme Environments".

[18] Delay Tolerant Networking Research Group. [Online]. Available: www.dtnrg.org. [Accessed June 16, 2009].

[19] CoDeploy: A Scalable Deployment Service for PlanetLab. [Online]. Available: http://codeen.cs.princeton.edu/codeploy. [Accessed June 16, 2009].