IOSR Journal of Electronics and Communication Engineering (IOSR-JECE)
ISSN: 2278-2834, ISBN: 2278-8735. Volume 4, Issue 1 (Nov-Dec. 2012), PP 20-25
www.iosrjournals.org

Virtual Wi-Fi for single hopping

Mr. Mohan Singh[1], Mr. Ranjeet Kumar[2]
[1] *(Dept. of Electronics and Communication, Satya College of Engineering and Technology, Palwal, India)*
[2] *(Dept. of Electronics and Communication, Made easy Institute, Noida, India)*

Abstract: *Single hopping is implemented by using software approach over IEEE802.11WLAN card using orthogonality concept and protocol. Most feature of single hopping is to achieve virtual Wi-Fi network. Design parameters of this approach are time delay and energy consumption .There are many scenarios where a wireless device connect to virtual Wi-Fi network. In this paper we improve the energy consumption and reduce the switching delay over popular IEEE 802.11 WLAN and present the performance of virtual Wi-Fi for single hop ad hoc network in terms of delay and energy consumption. Finally we approached multihopping in an ad hoc network using channelization with node synchronization.*
Keywords: *Wi-Fi, ad hoc network, IEEE 802.11.*

I. Introduction

Virtual Wi-Fi helps a user connect to multiple IEEE 802.11 networks with one card . It works by exposing multiple virtual adapters, one for each wireless network to which connectivity is desired, virtual Wi-Fi uses a network hopping scheme to switch the wireless card across the desired wireless networks. Switching between networks is transparent to the applications, such that the user feels he is connected to multiple wireless networks simultaneously. Virtual Wi-Fi is implemented as an intermediate driver and a user level service in window xp. Virtual Wi-Fi interacts with the card device driver at the lower end ,and network protocol at the upper end .the buffering protocol is implemented in the kernel and the switching logic is implemented as a user level device .There are many papers that articulate the benefits of virtualization [5],[9],[18].Dr. Ranveer Chandra said in his paper [1] that virtual Wi-Fi nodes can save up to 50% of energy consumed over nodes with two cards , while providing similar functionality . In this paper we also quantify the delay versus energy trade off for switching nodes over performance sensitive applications.

Virtual Wi-Fi is virtualization architecture for wireless LAN for a single ho adhoc network .Virtual Wi-Fi allows a user to simultaneously connect his machine to multiple wireless networks. This new functionality introduced by virtual Wi-Fi enables many new applications which are not possible earlier using a WLAN card. For example-

- With virtual Wi-Fi we can connect to a guest's machine or play games over an adhoc network; while surfing the web via an infrastructure network.
- We can use virtual Wi-Fi to connect our adhoc network which may contain many nodes to the internet using only one node
- Virtual Wi-Fi can helps make our home infrastructure network elastic by extending its access to nodes that are out of range of our home Wi-Fi Access point.

This virtual Wi-Fi is used for singlehop ad hoc network. For multihop nodes of adhoc network synchronization is problem and can we approach by sloted seeded channel hopping (SSCH).

Algorithm:

The main objective of this algorithm is to introduce and reduce the switching delay so that power consumption of the device can be reduced and this algorithm uses the following variables.

1. Active period of the network *i, ActPi*
2. Switching time for the network *i, STi.*
3. Switching cycle for all networks, SC.
4. Elapsed time inside Act*Pi*, ETActPi.

When a virtual Wi-Fi node switches to a network j where it has not yet form a adhoc network with other node. It stays at least twice of switching cycle to hear announcements from other nodes in j. There is a leader of an ad hoc network with largest MAC address .every node knows the switching cycle, which is the same for all the nodes and all nodes are to be synchronized with leader node.

Estate diagram:

The working of virtual Wi-Fi is illustrated using a state diagram in Figure 3 comprising eight states. It is assumed that the wireless card running virtual Wi-Fi is connected to a maximum of *n* networks, *{N*1*, N2, Nn }.*

Let *numNets* denote the number of simultaneous networks to which the card is associated at a particular instant, and *Ti* denote the activity period, Act*Pi*, i.e. the time a card stays in a network *Ni*.Given these notations the states in Figure 1 are explained as follows:

• START: Cards start in this state when they are either not connected to a wireless network, or are not using virtual Wi-Fi to connect to them. They might be connected to at most one wireless network.

• INIT *Nj*: Wireless cards enter this state when they want to join a new wireless network *Nj*. They can enter this state either from the 'START' state, or the 'ACTIVE *Ni*' state described later. On entering the 'INIT *Nj'* state, the card sets up a virtual adapter for network *Nj* if it does not exist. After creating the virtual adapter, it synchronizes with other nodes if *Nj* is an ad hoc network. Nodes also set up the data structures for buffering and maintaining other information in this state.

• ACTIVE *Ni*: The wireless card is connected to wireless network *Ni* in this state. Packets sent over this network are sent based on the scheme described in subsections V-B and V-D, and the packets sent over other networks *N k* _= *Ni* are buffered to be sent later.

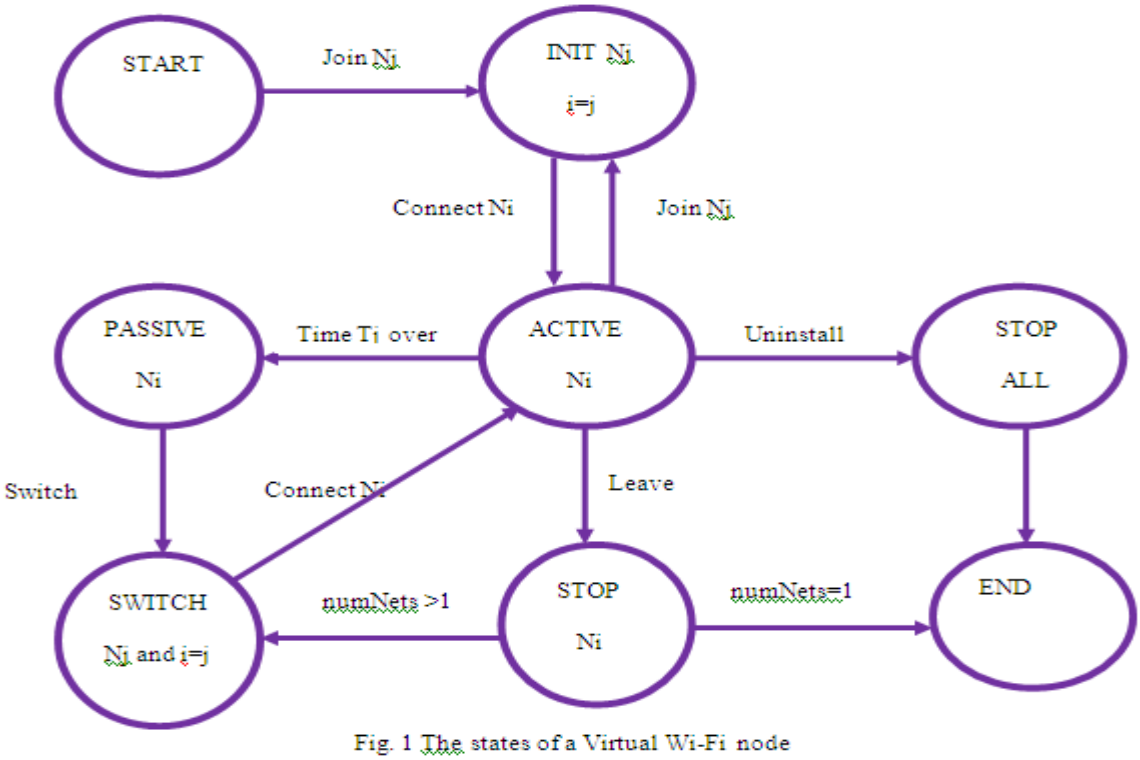

Fig. 1 The states of a Virtual Wi-Fi node

• PASSIVE *Ni*: After the wireless card has spent a time *Ti* in network *Ni* it moves to the 'PASSIVE *Ni*' state. The network stack corresponding to *Ni* is deactivated. Consequently, all packets sent over *Ni* are buffered till the corresponding stack is activated later in state 'SWITCH *Nj*'. The state corresponding to *Ni* is stored, and is used to switch back to this network later on.

• SWITCH *Nj:* Wireless cards enter this state when the current network *Ni* is either removed in state 'STOP *Ni*', or the time *Ti* in it has expired. In both these cases, cards use the switching strategy to determine the next wireless network, *Nj* , to connect to, and the time, *Tj* , the card should stay on it. The card then connects to network *Nj*, activates the corresponding network stack, and sends all the packets that were buffered on it. Then the card sets *i* = *j* and moves to state 'ACTIVE *Ni*'.

• STOP *Ni*: Cards have the option of leaving a network. They enter this state if they want to leave a network *Ni*. All the packets for this network are canceled and the virtual adapter for this network is destroyed. If Virtual Wi-Fi is still used to connect to more than one wireless network, i.e. *numNets* > 1, the card goes to the 'SWITCH *Nj*' state, and connects to the next network. Otherwise, the card goes to 'END' state.

• STOP ALL: Cards get to this state when Virtual Wi-Fi is to be uninstalled from the network interface. Virtual Wi-Fi cancels the
packets buffered for all the *numNets* networks, and destroys the corresponding virtual adapters. Then the card moves to the 'END' state.

• END: The wireless card is connected to at most one network in this state. Virtual Wi-Fi is uninstalled from the network interface, and traditional techniques are used to connect to the singular wireless network.

II. Approach For Implementation

Switching delay:

Good performance of Virtual Wi-Fi depends on a short delay when switching across networks. However, legacy IEEE 802.11b cards perform the entire association procedure every time they switch to a network. We carried out a detailed analysis of the time to associate to an IEEE 802.11 network eplaced a laptop in the vicinity of the two Virtual Wi-Fi machines and installed an IEEE 802.11 wireless LAN packet analyzer called AiroPeek [6] on it. AiroPeek supports higher level network protocols such as TCP/IP and fully decodes IEEE 802.11a and IEEE 802.11b WLAN protocols. It is used for analyzing wireless network performance with accurate identification of signal strength, channel and data rates. The 802.11 messages sent when switching to an ad hoc network are illustrated in Figure 2. Figure 3 shows the steps on switching to an infrastructure network. Of particular interest is the significant overhead when switching to a network

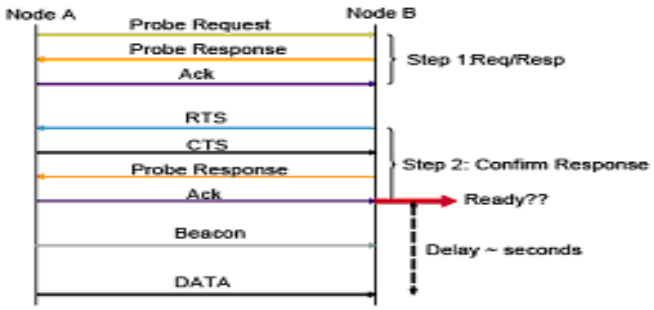

Fig.2 Message exchange when switching to an ad hoc network on the wireless adapter

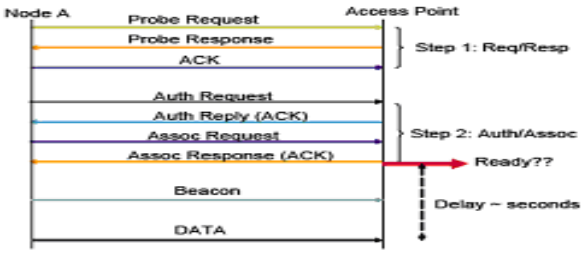

Fig.3 Message exchange, when switching to an infrastructure network on the wireless adapter

Looking at Figures 2 and 3, we see a significant overhead when switching from one network to another. In fact, a more baffling characteristic is the amount of delay. The delay is an astronomical delay of 3.9 seconds was observed from the time the card started associating to an ad hoc network, after switching from an infrastructure network, to the time it started sending data.

Minor modification results in significant improvement in the switching overhead as shown in Table I. Using the above optimization, we were able to reduce the switching delay from 2.8 seconds to 300 ms when switching from an ad hoc network to an infrastructure network and from 3.9 seconds to 170 ms when switching from an infrastructure network to an ad hoc network. *These numbers are further reduced to as low as 30 ms and 25 ms respectively, when NextGen 802.11 cards are used* [12]

Table-1

The Delays On Switching Between Is And Ah Networks For Ieee 802.11 Cards With And Without The Optimization Of Trapping Media Connect And Disconnect Messages.

Switching form	Unoptimized legacy	Optimized Legacy	Optimized Next Generation 802.11
IS to AH	3.8 sec.	170 ms	25 ms
Ah to IS	2.8 sec.	300 ms	30 ms

Interaction with Zero Configuration:

Wireless Zero Configuration (WZC) is a service in Windows XP that forces connectivity to the first available wireless network in a user's list of 'Preferred Networks'. This feature of WZC interferes with the correct functionality of virtual Wi-Fi, since there is not a single 'Preferred Network' in virtual Wi-Fi, but multiple wireless networks to which connectivity is desired. Therefore, we have to stop WZC for MultiNet to

function correctly. WZC implements the client 802.1X [11] protocol for wireless network authentication, and we are currently modifying the WZC code to not force network connectivity, and run virtual Wi-Fi along with it, to provide the best features of both WZC and virtual Wi-Fi. Alternatively, it is possible to treat the different virtual miniports as different wireless adapters to the user, and then allow WZC to have a preferred network for every virtual adapter.

Addressing:

Virtual Wi-Fi requires network dependent IP addresses for the different virtual adapters exposed by it. However, this turned out to be a difficult constraint. In particular, an ad hoc network works in Windows by assigning the node an autonet address. i.e.169.254.x.y address. In Windows XP, a node is assigned an autonet address only if it is unable to get a proper DHCP address. This scheme works fine if the network card stays in the ad hoc node for some time. However, in our implementation, the network card periodically switches between networks. If one of the networks is an infrastructure network, or a network with a DHCP server, the DHCP request gets through and the ad hoc node gets an IP address outside the range of autonet addresses. We fixed this problem by manually allocating IP addresses for the virtual miniports.

III. Results

Figure 4 shows the time taken to simultaneously transfer this file over Virtual Wi-Fi using different switching strategies for legacy cards. We evaluated 3 different Fixed Priority switching schemes. In the '50%IS 50%AH' strategy the node stays on each network for 500ms. In the '75%IS 25%AH' scheme it stays on the infrastructure network for 750ms and on the ad hoc network for 250ms, and in the '25%IS 75%AH' scheme the node stays on the infrastructure network for 250 ms and the ad hoc network for 750ms. For the Adaptive Traffic algorithm we used a window of 3 switching cycles to estimate the Activity Periods. In this case the window is 3*1.8 = 5.4 seconds since a switching cycle is 500+300+1000 = 1800 ms.

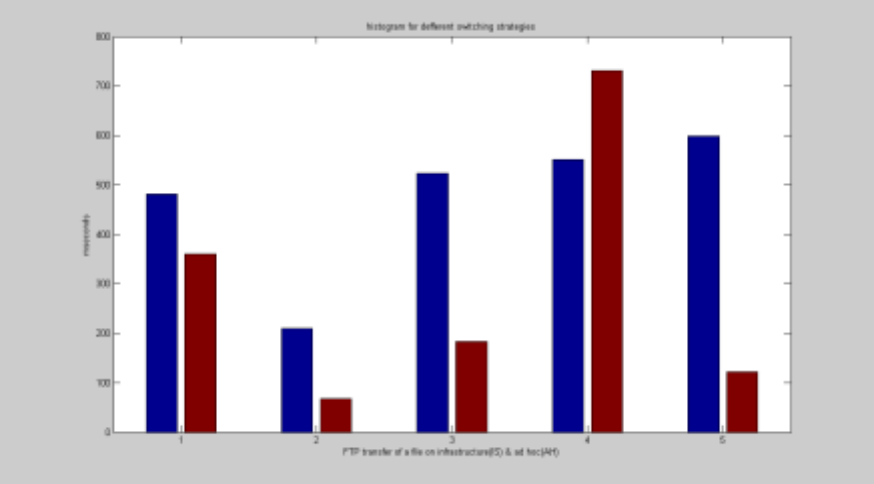

Fig. 4 Amount of time taken to complete the FTP transfer of a file on an ad hoc and infrastructure network for different switching strategies.

TCP performance:

we show the performance degradation for a two network Virtual Wi-Fi in Figure 5.Packets were sent, using ntttcp, over the infrastructure network from the Virtual Wi-Fi node to another node in the network. Ntttcp, which is a port of ttcp [17] to Windows, works by establishing a TCP session between two nodes and sending the packets at the maximum rate. The activity period for both the networks was fixed at 500 ms. We present results for three scenarios in Figure 5.'No Virtual Wi-Fi' corresponds to the case when the sender and receiver are connected to just one network, 'Virtual Wi-Fi No Buffer' is when the sender is connected to two networks using Virtual Wi-Fi while the receiver does not buffer packets, and 'Virtual Wi-Fi Buffer's the scenario when the receiver buffers packets for the Virtual Wi-Fi node. The results show that the performance drops by a factor of four when using Virtual Wi-Fi with buffering and drops further when the receiver does not buffer packets. Without buffering the throughput of the system goes down to a seventh of the maximum achievable throughput. Although performance drops significantly, Virtual Wi-Fi is still usable with a throughput of around 500 Kbps.

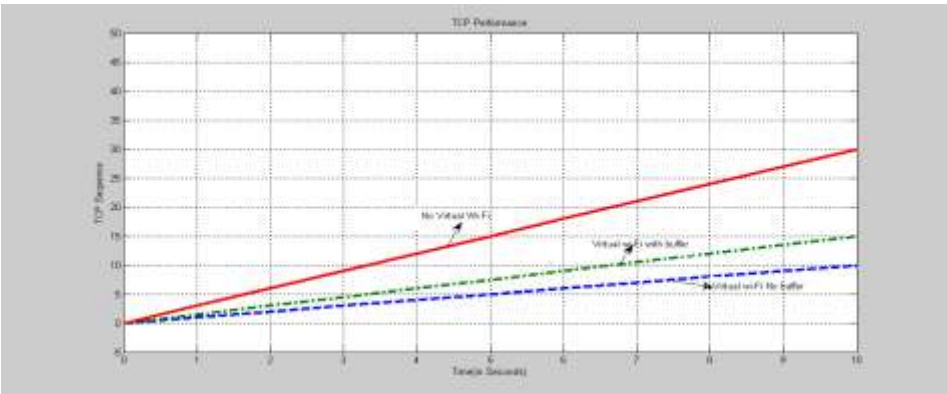

Fig.5. TCP Performance in mixed deployment settings: with and without Virtual Wi-Fi buffering.

As shown in Table 2, the average throughput of Virtual Wi-Fi in the infrastructure mode is 4.35 Mbps compared to 5.8 Mbps in the two radio case. The average throughput in the ad hoc network is 1.1 Mbps in Virtual Wi-Fi and 4.4 Mbps when using two radios. Switching results in lesser throughput across individual networks

Table -2
The Average Throughput In The Ad Hoc And Infrastructure Networks Using Both Strategies Of Virtual Wi-Fi And Two Radios

Network	Two radio	Virtual Wi-Fi
Ad Hoc	4.4 Mbps	1.1 Mbps
Infrastructure	5.8 Mbps	4.35 Mbps

Power save mode:

In 802.11 WLAN there is a power save mode (PSM), using this mode we can reduce the amount of energy more in Virtual Wi-Fi .Figure 6 shows the energy uses by two radios and virtual Wi-Fi with and without PSM. Table 3 shows the average packet delay over the network with PSM.

Table-3

Scheme	Average Delay (in sec.)	Improved average delay (in sec.)
Two radio	o.oo1	0.001
Virtual Wi-Fi	0.157	0.157
Two radio with PS	0.156	0.156
Virtual Wi-Fi with PS	0.167	0.169

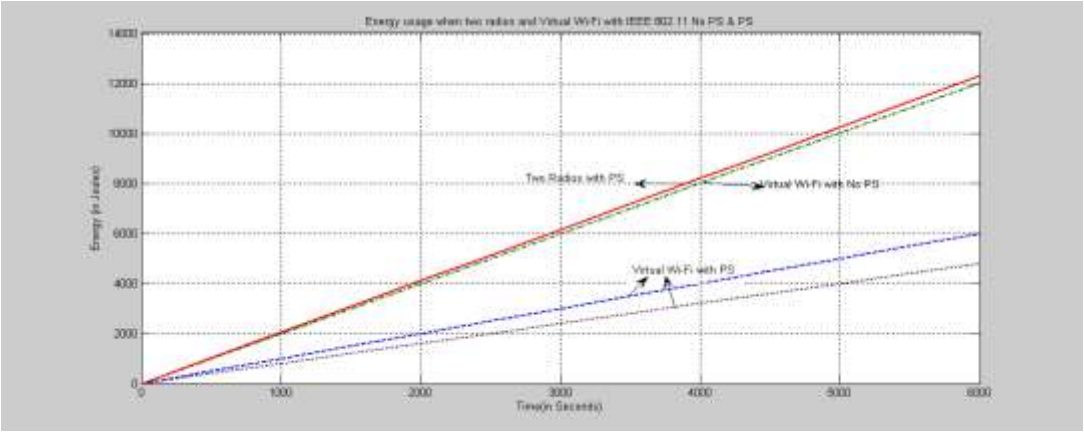

Fig.6. Energy usage when using Virtual Wi-Fi and two radios with IEEE 802.11Power Saving

IV. Conclusion

We described a new virtualization architecture called Virtual Wi-Fi, which allows a user to connect to multiple wireless networks by virtualizing the WLAN card. Several compelling real life scenarios are described that motivate the need for Virtual Wi-Fi. We analyze switching algorithm in terms of delay and energy consumption.

V. Future Work

Virtual Wi-Fi is implemented for single hopping ad hoc network because there is no problem for node synchronization but this problem exists in multihopping. Our approach is to implement this in multihopping using sloted seeded channel hopping (SSCH) with the help of channelization of orthogonalization

References

[1] Ranveer Chandra, Christ of Fetzer and Karin Hogstedt. Proceedings of Informatics, 1st International Conference on Ad-hoc Networks and Wireless, Toronto, Vol. 16, pp 1-16, September 20-22, 2002. P. Barford and M. Crovella.

[2] ATA Flash Memory Cards. http://www.magicram.com/flshcrd.htm.

[3] The ethereal network analyzer. http://www.ethereal.com/.

[4] Relatek. http://www.realtech.com.tw/

[5] VMware: Enterprise-Class Virtualization Software. http://www.vmware.com/.

[6] WildPackets Airopeek. http://www.wildpackets.com/products/airopeek.

[7] Paul Barford and Mark Crovella. Generating Representative Web Workloads for Network and Server Performance Evaluation. In *ACM SIGMETRICS 1998*, pages 151–160, July 1998.

[8] Josh Broch, David A. Maltz, and David B. Johnson. Supporting Hierarchy and Heterogeneous Interfaces in Multi-Hop Wireless Ad Hoc Networks. In *Workshop on Mobile Computing held in conjunction with the International Symposium on Parallel Architectures*, June 1999.

[9] E. Bugnion, S. Devine, and M. Rosenblum. Disco: Running Commodity Operating Systems on Scalable Multiprocessors. In *Sixteenth ACM Symposium on Operating System Principles*, October 1997.

[10] Martin Heusse, Franck Rousseau, Gilles Berger-Sabbatel, and Andrzej Duda. Performance Anomaly of 802.11b. In *IEEE INFOCOM*, 2003.

[11] IEEE. IEEE 802.1x-2001 IEEE Standards for Local and Metropolitan Area Networks: Port-Based Network Access Control. 1999.

[12] IEEE802.11b/D3.0. Wireless LAN Medium Access Control (MAC) and Physical (PHY) Layer Specification: High Speed Physical Layer Extensions in the 2.4 GHz Band. 1999.

[13] Ching Law, Amar K. Mehta, and Kai-Yeung Siu. A New Bluetooth Scatternet Formation Protocol. In *To appear in ACM Mobile Networks and Applications Journal*, 2002.

[14] Ching Law and Kai-Yeung Siu. A Bluetooth Scatternet Formation Algorithm. In *IEEE Symposium on Ad Hoc Wireless Networks 2001*, November 2001.

[15] A. Nasipuri and S. R. Das. Multichannel CSMA with Signal Power-Based Channel Selection for Multihop Wireless Networks. In *IEEE Vehicular Technology Conference (VTC)*, September 2000.

[16] Eugene Shih, Paramvir Bahl, and Mike Sinclair. Wake On Wireless: An Event Driven Energy Saving Strategy for Battery Operated Devices. In *ACM MobiCom 2002*, September 2002.

[17] R. Stine. FYI on a Network Management Tool: Catalog Tools for Monitoring and Debugging TCP/IP Internets and Interconnected Devices. In *IETF RFC 1147*, April 1990.

[18] Andrew Whitaker, Marianne Shaw, and Steven D. Gribble. Scale and Performance in the Denali Isolation Kernel. In *Fifth Symposium on Operating Systems Design and Implementation*, December 2002.