Decentralized Handoffs in Heterogeneous Networks: A Kernel Module Approach

S.V.Srikanth
C-DAC
India

Saratchandra Babu
C-DAC
India

Dileep K.P
C-DAC
India

Trilok Inakota
C-DAC
India

ABSTRACT
Our proposed solution is a Linux kernel module approach to vertical handoff for heterogeneous networks. We develop a kernel module that accomplishes the handoff process between various network interfaces supporting IP. This addresses seamless continuity, handoff decision & interface selection problems faced during the vertical handoff. This module is integrated into the Linux kernel and works for all the multimedia networking applications running on Linux systems. The handoff can be achieved at the client devices and there is no need of any centralized handoff agents or servers required. Linux being the fastest growing operating system for both wired & wireless devices and the flexibility it offers for the developer makes it the most suitable for the implementation of this solution. This paper reviews how the proposed module running at the kernel level of Linux operating system maintains seamless continuity in heterogeneity.

Keywords
Heterogeneous Networks, vertical handoff, Linux Kernel Module, WLAN, multimedia, seamless session continuity

1. INTRODUCTION
Today, mobile networks are expected to achieve a high degree of inter-networking so that the mobile users can truly experience ubiquitous access through switching between heterogeneous networks. We mean that the device neither requires user intervention nor does it disrupt the existing session during the interface switching process. IP based networking is the key to turn this vision into a reality. Future clientele devices tend to be equipped with multiple network interfaces. So, there is a need to switch between different network interfaces to achieve seamless mobility [7]. Mobile devices that cater to one type of access technology have the ability to roam among networks of the same type are referred to as horizontal handoff. Seamless access to services between various access technologies such as WLAN, WCDMA, Cellular networks, WiMAX etc, irrespective of the radio technology being used is referred as vertical handoff [4].

On most of the devices today wired and wireless LAN cards have become basic components, apart from supporting other radio access technologies such as Bluetooth, GPRS etc [7]. If these devices are to connect to various networks and perform a continuous data transmission whatever location they roam to, while switching from one network to another, technique of vertical handoff is very much needed.

2. PROBLEM
There are several key issues that are associated with the implementation of vertical handoff 1) Seamless Continuity 2) Handoff Decision & 3) Interface Selection. First, we have to ensure that the device makes a switch between hardware interfaces to connect to the best available network. In our case, whenever a connection to Ethernet is available, we connect to it and switch to the WLAN or cellular when there is mobility. Next, the switching between the networks has to be transparent to the user. The mobile user should not have to configure anything for the switch to occur [11]. In the homogeneous network handoff mechanism, the network infrastructure makes it possible for devices to move without disrupting the existing connection. Similarly, heterogeneous network switching should also ensure that the ongoing session is maintained. We have to make sure that the switching is fast enough so that the user connection is kept intact at the TCP/UDP layer. Most of the existing approaches for vertical handoff are centralized [3]. The need of the hour is to develop a decentralized approach for vertical handoff, which is significantly faster than currently followed approaches.

3. ASSUMPTIONS
We restrict our experiments to have two different networks interface cards i.e. Ethernet and WLAN. It is also assumed that the proposed module architecture also suits different network interface cards supporting IP. We use a 2.6.32 version of the Linux Kernel that is suitable for running on a Laptop or an embedded device.

4. RELATED WORK
There are various approaches to achieve seamless mobility in heterogeneous networks. Some of the approaches are generalized as follows.

One solution is to introduce handoff servers to realized vertical handoff. In USHA [8] system, all mobile hosts connect to the Internet with the help of a handoff server, which is equipped with multiple network interfaces with heterogeneous physical properties. Hence, mobile hosts can communicate with the handoff server in various physical connections. The IP tunneling technique (IP encapsulation) is used in USHA with the handoff server functioning as one end and the mobile host as the other. Upper layer communications are bounded to a virtual interface - the tunnel interface, instead of physical interfaces. All data packets are transmitted through this IP tunnel. When the handoff event occurs, the underlying physical connection of the virtual tunnel is automatically switched to the new physical interface.

Now a days mobile devices support several interfaces although the protocol stack used in each interface tends to be interface specific at the lower layers. This limits the ability of a device to switch back and forth between networks as need and opportunity dictates. As a step towards providing a more flexible handover infrastructure, this project addresses the issue of integrating heterogeneous, mobile ad-hoc networks that use different MAC layer protocols. The goal is to provide an end-to-end communication abstraction that hides heterogeneity [2].

Currently, there is a significant trend toward using not only one wireless network, but to utilize all carriers available on a mobile device. In addition, for real-time applications like voice, it is desired to make a seamless change of network without user interaction. To keep track of this evolution, the use of heterogeneous networks has gained focus. Instead of only using one type of network, it is desired to utilize all carriers available on a device, and hence choose the one best suited. Furthermore it is proposed an application-layer handover scheme for session continuity in heterogeneous networks. It is a centralized approach consisting of SIP servers [3,5].

5. PROPOSED ARCHITECTURE

When a user is in roaming state and still wishes to maintain all active sessions, an automatic handoff mechanism is needed to change the communication flow and the physical interface in use. Moreover, the handoff mechanism should be user unaware and decentralized, so that the handoff process can be handled smoothly and quickly [5].

Figure 1 provides a high-level view of the Linux network stack. The upper layer is the application, which defines the users/clients of the network stack. The next layer is the transport layer, responsible for process-to-process communication. Below this is the network layer, which is responsible for routing the packets to their destinations. At the bottom of the stack is the link layer. The link layer refers to the device drivers providing access to the various physical layer mediums, such as Ethernet devices or WLAN devices etc [9]. Now, depending on the active device drivers the link layer chooses the network card through which packets have to be transmitted. This card is bound till the session is completed. The problem we face here is that during the vertical handoff there needs to be a switching between different NIC cards depending on the availability of the active cards. Currently the Linux network stack doesn't have a support for vertical handoff [4].

Our proposed solution is development of Linux kernel module to achieve vertical handoff for seamless continuity over heterogeneous networks. We develop a kernel module that integrates into the Linux kernel and accomplishes the handoff process between various network interfaces supporting IP.

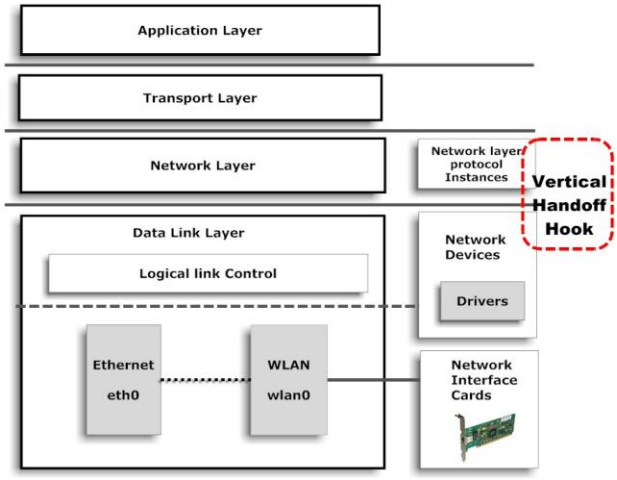

Fig 1: Proposed Architecture

This runs at the kernel level of the Linux operating system between the network layer and the datalink layer. The proposed module known as vertical handoff hook, always polls for the interfaces and checks for the interfaces that are alive. It also updates the network layer of the TCP/IP stack to maintain the seamlessness alive between the mobile station and the server. This architecture is well suited for all the multimedia networking applications running.

6. IMPLEMENTATION

In this section we give a clear picture of Kernel Network Structure and our proposed vertical handoff hook.

6.1 Kernel Network Structure

The Linux Kernel Network Structure maintains a socket buffer for packet management. This buffer consists of two data classes Packet data and Manage data. Every network device registered contains a structure named as *net_device* and all the *net_device* structures are mapped by socket buffer.

If the network device was configured for the IP, then *ip_ptr* points to a structure of the type *in_device*, which manages information and configuration parameters of the relevant IP instance and this is the data maintained at network layer.

When the first packet of a socket is ready and there is no route present, then the function named *ip_route_output ()* chooses a route from route table named *fib_table* by using helper functions. A *fib_table* structure forms the base structure for a routing table. Linux kernel supports two configuration settings for managing routing tables. If the kernel is configured with CONFIG_IP_MULTIPLE_TABLE macro, it follows rule based routing; otherwise it maintains two routing tables i.e. local table

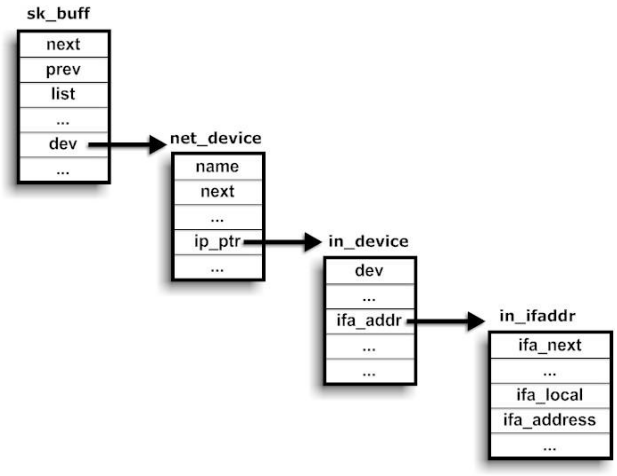

Fig 2: Socket buffer Management

and main table. Once the device gets deactivated, a driver specific function *stop ()* is called and then, the kernel releases or flushes routing information related to the specified registered device.

6.2 Vertical Handoff Hook

Our proposed vertical handoff hook consists of two modules namely route and device module.

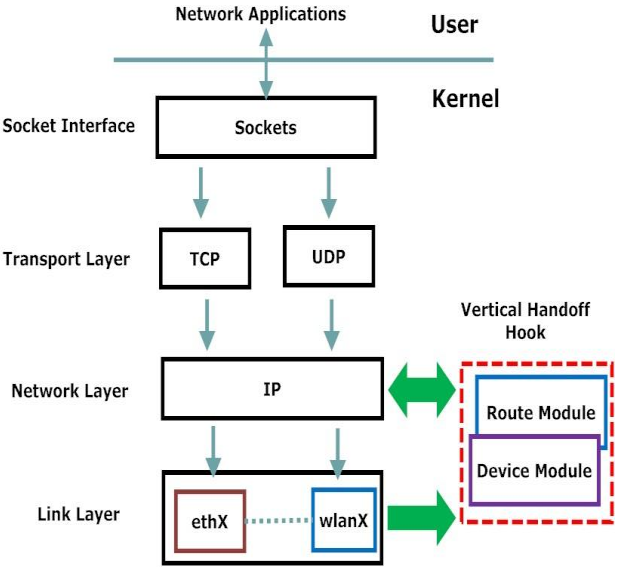

Fig 3: Vertical Handoff Hook

6.2.1 Route Module

As shown, (see figure 3) route module lies at the network layer of Linux network stack. This module plays a major role in handling vertical handoff. It mainly updates the routing table continuously by using RT netlink functions.

This module mainly checks for the first network device registered and its configured IP from the route table i.e. *fib_table*. Then, it immediately reads this configured IP and writes the IP to all the network devices registered. After configuring all the devices with the same IP, it then looks for any gateway route address configured. It, then route adds all the devices with this gateway address. This is done to keep the session continuity alive, as we know that a single IP is bound for a session. When the first network device becomes deactivated, it instantly checks for the next network device registered to continue the session. As the route module has already assigned the same IP to all the network devices, the session will be intact even though there is a change in the network interface.

6.2.2 Device Module

This module gets the status information of the registered network devices from the structure *net_device*. Based on the status information (active or dormant), this device module invokes the route module for the route table updation and change of network device, for keeping the session intact.

7. EXPERIMENTAL SETUP

This section details the complete experimental setup and the hardware used. The basic experimental setup consists of a Laptop running Linux Kernel version 2.6.32 [21] supporting two interfaces Ethernet (eth0) & WLAN (wlan0), Linksys WRT54G2 Accesspoint, Server running Ubuntu Linux 9.04 [22] and an ACTi Camera [16].

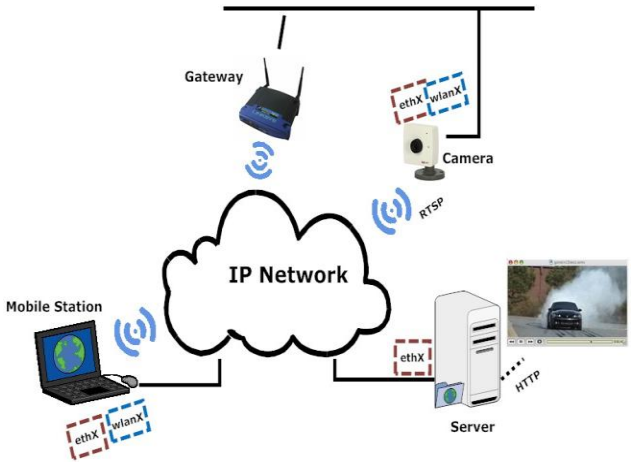

Fig 4: Experimental Setup

As shown, (see figure 4) the whole setup is connected to LAN. The initial step is to validate and configure the network cards by using ifconfig. Once the configuration is done, then the next step is check the communication between the mobile station and the server for both the interfaces i.e. Ethernet & WLAN. When both the interfaces are alive, Linux kernel chooses one of the interfaces to transmit packets to the server. But when the chosen interface is plugged out, Linux kernel doesn't shift to the other

interface for communication with the server. It immediately stops the session and waits for the chosen interface to get plugged in. For example if ethX is chosen as interface to send packets and is plugged out, Linux kernel doesn't have the mechanism to shift WLAN X to continue communication.

To solve this problem, we developed and implemented a vertical handoff kernel hook. This module completely runs at the kernel space of the Linux operation system. When this module is inserted, it immediately checks the first registered device. For example, if ethX is chosen as the first interface, it takes the IP address of ethX and assigns the same IP address to all the registered network devices shown in *net_device*. This allows the session to be continued even if there is a change in the network interface.

8. RESULTS

Experiments were conducted for centralized approaches using SIP [3] and decentralized approaches using daemon [4] & kernel module. The scope of the results is limited to kernel module and a comparison chart of all the approaches is depicted at the end.

We have tested the functionality of our kernel module named vertical handoff hook for all the 3 forms of media i.e. data, audio & video. Open source Wireshark tool [20] is used to validate the switching between the interfaces. The MAC ID of eth0 & wlan0 of mobile station are 00:24:bc: 43:1d:0c and 00:26:5c:f5: 39:89 respectively.

8.1 For Data:

We used Secured Copy (SCP), Secured Shell (SSH) and our in house network applications to check the functionality of our proposed vertical handoff hook.

8.1.1 Scenario 1: Switching from eth0 to wlan0

In this scenario, initially our mobile station will be connected to both eth0 and wlan0. Now, when we start our network application, packets flow from eth0 interface. Now, we insert our proposed module, it takes the IP configured for eth0 and writes the same IP to wlan0.

Once, when the eth0 interface is plugged off, immediately it switches to wlan0 to keep the session alive. The experimental results show that the switching between eth0 and wlan0 is around ~0.0013 sec.

8.1.2 Scenario 2: Switching from wlan0 to eth0

This is in continuation to the above, the communication between mobile station and server happens with wlan0 interface. But, once when the eth0 interface is up and wlan0 interface is down, immediately the network interface is shifted to eth0 making the session alive. The experimental results show that the switching from wlan0 to eth0 is around ~0.0034 sec

8.2 For Multimedia:

We also tested our developed kernel module on both HTTP and RTCP based applications

8.2.1 HTTP Application:

To demonstrate the HTTP application we used VLC player [23] as a medium to stream recorded video.

Server: *vlc -vvv /home/Movies --sout '#standard{access=http,mux=ogg,dst=192.168.51.16:1234}*

Client: *vlc http://192.168.51.16:1234*

Initially, streaming was done on eth0 network interface.

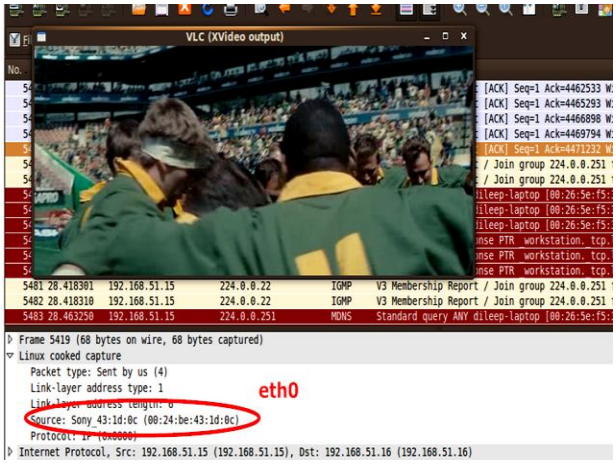

Fig 5: Communication with eth0 (HTTP)

As shown (see figure 6), when the Ethernet cable was plugged out, it took around ~0.0071 seconds to shift to wlan0 and continue the session.

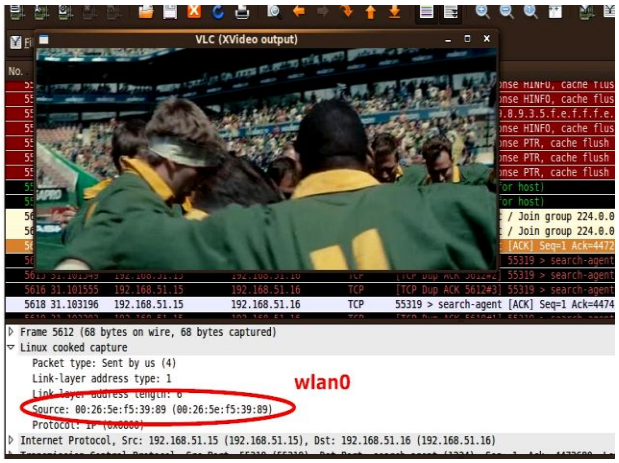

Fig 6: Switching from eth0 to wlan0 (HTTP)

Similarly, we tested the case of switching from wlan0 to eth0 running the HTTP application and results are depicted in Table 1.

8.2.2 RTCP Application:

To demonstrate this application we used an ACTi camera and a VLC player.

Client: *vlc rtsp://user:user123@192.168.51.92:554/axis-media/media.amp*

Once the camera is started, our mobile station receives packets from the wlan0 interface.

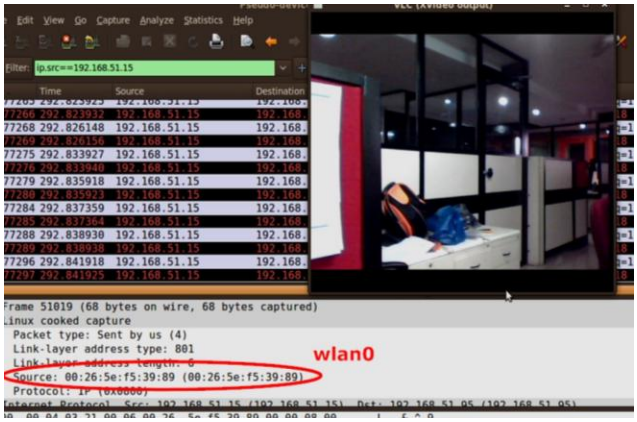

Fig 7: Communication with wlan0 (RTCP)

Now when we disable wlan0 interface, it instantly shifts to eth0 interface within a span of around ~0.023 sec

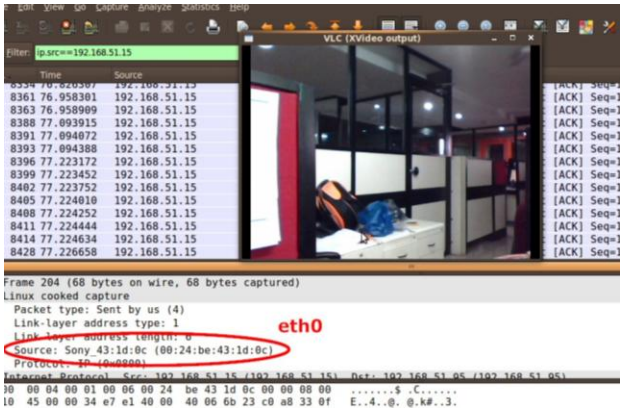

Fig 8: Switching from wlan0 to eth0 (RTCP)

Similarly, we tested the case of switching from wlan0 to eth0 running the RTCP application and the results are shown in Table 1.

Table 1. shows the complete average time taken for vertical handoff between both the interfaces for data, Multimedia (HTTP) & Multimedia (RTCP) for all the approaches experimented.

Table 1. Vertical Handoff Average Time

Approach	Medium	Vertical Handoff	Average Time
Centralized Approach (using SIP)	Data	eth0 to wlan0	~2.0 sec
		wlan0 to eth0	~3.0 sec
	Multimedia (HTTP)	eth0 to wlan0	~3.0 sec
		wlan0 to eth0	~4.0 sec
	Multimedia (RTCP)	eth0 to wlan0	~3.0 sec
		wlan0 to eth0	~4.0 sec
Decentralized Approach (using Daemon)	Data	eth0 to wlan0	~0.203 sec
		wlan0 to eth0	~0.047 sec
	Multimedia (HTTP)	eth0 to wlan0	~3.0 sec
		wlan0 to eth0	~1.0 sec
	Multimedia (RTCP)	eth0 to wlan0	~3.0 sec
		wlan0 to eth0	~1.0 sec
Decentralized Approach (using Kernel Module)	Data	eth0 to wlan0	~0.0013 sec
		wlan0 to eth0	~0.0034 sec
	Multimedia (HTTP)	eth0 to wlan0	~0.0071 sec
		wlan0 to eth0	~0.012 sec
	Multimedia (RTCP)	eth0 to wlan0	~0.015 sec
		wlan0 to eth0	~0.023 sec

9. CONCLUSION & FUTURE WORK

The proposed decentralized vertical handoff kernel module was implemented and tested for multimedia networking applications successfully on Linux Operating System. It runs directly at the client devices and there is no need of any handoff agents or servers. This module is completely implemented at the kernel level of the Linux operating system. This architecture works for any network interface supporting IP. Currently, the work is limited to Linux based operating system. In future, our focus will be to design and develop kernel module for mobile operating systems such as Android, Symbian to achieve seamless continuity in heterogeneity.

10. ACKNOWLEDGMENTS

We thank Centre for Development of Advanced Computing (C-DAC) for giving us an opportunity to do R&D in the area of heterogeneous networks and providing us with required infrastructure.

11. REFERENCES

[1] Selim İckin (2010): Implementation Of Measurement Module For Seamless Vertical Handover, Blekinge Institute of Technology

[2] Patrick Stuedi and Gustavo Alonso (2008): Transparent Heterogeneous Mobile Ad Hoc Networks, Swiss Federal Institute of Technology (ETHZ)

[3] Mohit Malhotra, Pramod P. J and S. V Srikanth (2011): Integration of IMS and 802.21 in a Heterogeneous Environment: An empirical analysis, IEEE ICDC

[4] S.V. Srikanth, Dr. Sarat Chandra Babu, Dileep K Panjala, I Trilok (2011): Seamless Multimedia Communication Over Heterogeneous Networks: A Linux Daemon Approach, Indian Journal of Computer Science and Engineering (IJCSE)

[5] Hakon Eyde Kjuus (2007): Session Continuity in Heterogeneous Networks: A SIP-based Proactive Handover Scheme, University of Oslo

[6] Nikolaou, N.A., Vaxevanakis, K.G., Maniatis, S.I., and Venieris, I..S., Zervos, N.A. (2002): Wireless Convergence Architecture: A Case Study Using GSM and Wireless LAN. Mobile Networks and Applications, pp. 259–267

[7] Wei-Cheng Xiao, Shih-Hsuan Tang, Ling-Jyh Chen, and Cheng-Fu Chou, (2007): A Novel Seamless Vertical Handoff Solution, IEEE Consumer Electronics

[8] L. J. Chen, Tony Sun, and Mario Gerla. (2005): USHA: A Practical Vertical Handoff Solution. MSAN

[9] M. Tim Jones (2007): Anatomy of the Linux networking stack from sockets to device drivers, Emulex Corp.

[10] Ashwin Kumar Chimata (2005): Path Of A Packet In The Linux Kernel Stack, University of Kansas

[11] Mahesh K. Subramanian (2005): A Kernel-based solution to seamless mobility in wireless networks http://www.technology.asu.edu/files/documents/.../Mahesh FinalReport_V6.pdf

[12] J. Ylitalo et al. (2003): Dynamic Network Interface Selection in Multihomed Hosts, Proceedings of the 36th Hawaii International Conference on System Sciences (HICSS'03), pp. 315-324.

[13] Seok Joo Koh, Sang Wook Kim (2005): mSCTP for Vertical Handover Between Heterogeneous Networks. Human.Society@Internet, pp.28-36

[14] M. Williams (2006): Linux ethernet bonding driver HOWTO

[15] Maria Rosa Frias Gonzalez (2009): Analysis of Interworking Functions for Heterogeneous Networks, Thesis, Aachen

[16] ACTi Camera, Available from *http://www.acti.*

[17] Stefan Aust, Jong-Ok Kim, Peter Davis, Akira Yamaguchi, Sadao Obana (2006): Evaluation of Linux Bonding Features, Communication Technology, ICCT '06

[18] L. J. Chen, Tony Sun, Benny Chen, Venkatesh Rajendran, and Mario Gerla (2004): A Smart Decision Model for Vertical Handoff, ANWIRE

[19] Srikant Sharma, Inho Baek, Yuvrajsinh Dodia, and Tzi-cker Chiueh. OmniCon (2004): A Mobile IP-based Vertical Handoff System for Wireless LAN and GPRS Links. IWNDA

[20] WireShark monitoring tool, Available from www.wireshark.org

[21] Linux Kernel Archives, Available from http://www.kernel.org/

[22] Ubuntu 9.04 Linux, Available from http://www.ubuntu.com/

[23] VLC player, Available from http://www.videolan.org/vlc/