Overlays on Wireless Mesh Networks: Implementation and Cross-Layer Searching

Gang Ding[1], John Vicente[2, 3], Sanjay Rungta[3], Dilip Krishnaswamy[4], Winson Chan[3], Kai Miao[3]

[1] School of Electrical and Computer Engineering, Purdue University
[2] Department of Electrical Engineering, Columbia University
[3] Intel Information Technology Research, Intel Corporation
[4] Intel Mobility Group, Intel Corporation
dingg@purdue.edu, {john.vicente, sanjay.rungta, dilip.krishnaswamy, winson.c.chan, kai.miao}@intel.com

Abstract. After years of research on ad hoc networks, practical wireless mesh networks are moving towards mainstream industry deployment. As wireless mesh networks become more ubiquitous, how to enable distributed applications and services is a challenging research topic. A new network architecture called OverMesh is recently proposed, in which computational overlays provide the facility to deploy distributed services across mobile mesh nodes. In this paper, we present the first implementation of the OverMesh architecture. The overlays are built over an IEEE 802.11s wireless mesh network pre-standard prototype. The platform enables development and deployment of concurrent distributed experiments on wireless mesh networks. Based on this platform, we further introduce a cross-layer searching algorithm, which combines traditional overlay searching with ad hoc network routing so that a physically short searching route is facilitated. Both experimentation and simulation results are presented.

Keywords: Overlay, virtualization, wireless mesh network, cross-layer.

1 Introduction

There has been a long history of research on mobile ad hoc networks in which peer nodes relay packets for other nodes while no fixed infrastructure exists to control or manage the network [2]. While wireless mesh networks have been actively studied [3], there has been increasing interest from the industry for its deployment use. A wireless mesh network is a form of wireless ad hoc network where some mesh nodes are stationary, thereby reducing the power constraint concerns. Standardization of wireless mesh networks for different settings have already started, for example, the IEEE 802.11s mesh network for wireless local area network (WLAN) [4], the IEEE 802.15.5 mesh for wireless personal area network (WPAN) [5], the ZigBee mesh network for low rate WPAN or sensor networks [6], and the mesh mode for IEEE 802.16 wireless metropolitan area network (WiMAX) [7]. Moreover, the next generation wireless network is expected to be a hybrid of all these mesh networks, which can be further integrated to the cellular networks and the Internet. When this vision becomes true in the near future, it is anticipated that a growing number of services and applications will run on it. While current research activities on wireless mesh networks are mostly focused on the physical, data link, and network layers; in this paper, however, we take one step further towards enabling distributed services and applications on the wireless mesh networks.

An obvious indication of the success of the Internet is the wide deployment of services and applications, including client-server applications such as the World Wide Web (WWW), email, and ftp, and more recently, the overlay networks and peer-to-peer applications. When we study future services and applications in wireless mesh networks, the overlay network would be preferred because it shares similar features with underlying wireless ad hoc networks, such as no centralized server, dynamic network topology, and localized operation. This motivates our interest to investigate an architecture of overlays on wireless mesh networks, called OverMesh [1]. However, the challenges to integrate them were not straightforward because the overlay network is a virtual network running in the application layer and the underlying network is transparent to the nodes in the overlay, while nodes in a wireless mesh network should participate in the actual routing in network or link layer. In addition, the mesh nodes in wireless networks are limited by bandwidth, computation capacity, and interferences, which is not a big concern in overlay networks on the Internet because of the high speed cable and dedicated routers. Therefore, how to enable efficient overlay services and applications on the resource constrained wireless mesh networks is a challenging research proposition. We believe that the cross-layer approach should be employed because it can facilitate information exchange between the overlay in the application layer and the network and lower layers. Therefore, we propose a cross-layer overlay searching algorithm which takes advantage of the broadcast based routing in the underlying wireless mesh network to quickly find the shortest physical searching route. The cross-layer searching protocol is realized and compared with the distributed hash table (DHT) [10] based overlay searching algorithms.

The rest of the paper is organized as follows. Section 2 briefly introduces the OverMesh architecture. Our implementation experience along with the details of the current OverMesh platform are described in Section 3. The cross-layer searching protocol is presented in Section 4. Section 5 gives the experimentation and simulation results. Related work is reviewed in Section 6. Section 7 concludes the paper and proposes some future directions.

2 OverMesh Networks

We have proposed a generic overlay architecture called OverMesh [1]. We view it as a parallel edge/access internetworking strategy positioned for novel use scenarios including, for example, residential or local community networks, office networks, home networks, and first-response networks. The followings are collectively the differentiating properties of OverMesh against traditional networking, peer-to-peer computing systems, or ad hoc networking systems:

Infrastructure-free. We position an aggressive convergence strategy for node computation, network processing, and data storage. Any physical node may be capable of supporting alternative properties of interconnection and source and sink functions;

Network virtualization. Facilitated by distributed virtual machines, the overlays would enable a computational model for provisioning and managing network structure and resources, distributed network services and applications;

Emergent control and manageability. To achieve the level of robustness and resilience seen in today's internetworking systems under a decentralized networking system, we argue that it is necessary that some services exhibit behaviors typically seen in emergent biological systems; we call for alternative learning and statistical inference techniques to off-load human-dependency on operational management;

Cooperative and adaptive end-to-end control. To support a horizontal and vertical system orientation to scale and adapt wireless communications, we believe the end-to-end principle and in-network control must converge. We call for tighter layer integration and cross-layer control and management.

The conceptual OverMesh architecture can be applied to a variety of wireless mesh networks. At its current stage, we chose to realize it on one of the mesh networks that are being actively standardized – the IEEE 802.11s WLAN mesh network [4]. The PlanetLab [9] architecture was customized and integrated with the WLAN mesh network to manage the distributed virtual machine based overlays. Overlays and virtualization have been proven to facilitate deployment of large distributed services and peer-to-peer applications on the Internet, but when applying them to the resource constrained mobile wireless environments, we pursed investigation of the following new issues:

Testbed for distributed applications on wireless networks. The testbed should provide an open platform for research on real wireless mesh networks. Researchers can develop and deploy their own distributed experiments and test the performance on the whole mesh network. Each experiment spans multiple mesh nodes. The virtualized overlay enables concurrent but isolated experiments on the same physical testbed. The details of realizing such a testbed is the focus of next section.

Efficient management of resource-limited wireless mesh networks. The traditional overlay for wired networks tries to make the underlying network transparent to the users. This may not be desirable for wireless networks. Given limited bandwidth, computation capacity, and dynamic topology in wireless networks, the resource management and control should consider the underlying network condition. Theoretical approach to the resource management in wireless ad hoc networks has been extensively studied. But the current wireless ad hoc networks still lack a generic platform to enable these results. The OverMesh architecture provides a feasible way to support distributed resource management and control: each node contributes one of its virtual machines to form a resource management overlay. All virtual machines in this overlay work together and try to balance the network load in a fully distributed fashion. The difference of this overlay from other overlays is that it can collect information from the network and physical layer or send control signals down to lower layers. This shows the importance of information exchange between different layers. In particular, this paper proposes a cross-layer searching algorithm in Section 4. It combines the overlay searching and the ad hoc routing so that a searching request from the overlay can be quickly replied with the least use of resources in the lower wireless network.

Develop distributed services for future wireless mesh networks. In addition to the large number of services already available in the wired network, there will be many novel services and applications specific to the next generation wireless networks, for example, distributed network measurement and monitoring, localization of mobile nodes, data aggregation and mining. This paper is not focused on a particular application, but we have implemented the open platform and will demonstrate how to develop and deploy a service on it. Results from extensive experimentations and simulations are reported in Section 5. We believe that the implemented OverMesh platform will provide a unique testbed for developing a wide variety of novel services and applications on wireless mesh networks.

3 Implementation

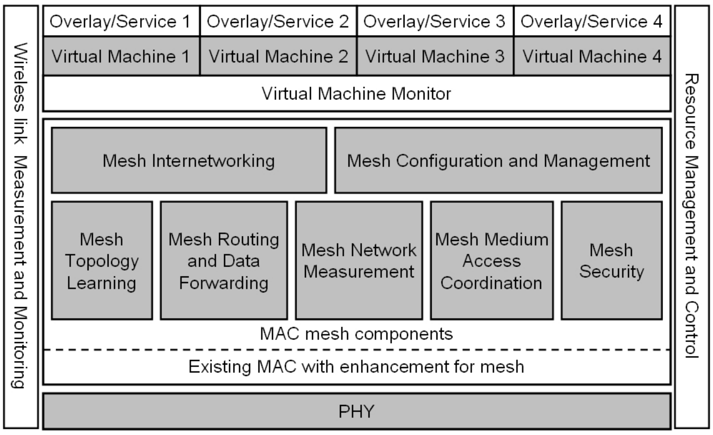


Fig. 1. Mesh node implementation.

The prototype of wireless mesh network is based on the draft of IEEE 802.11s working group, called Simple Efficient Extensible (SEE) mesh. A mesh node provides 802.11 conformant MAC and PHY interfaces. Some mesh nodes serve as access points providing additional basic service set to support communications with simple mobile wireless clients. A mesh portal is a mesh node that specifically serves as an entry or exit point for packets in the network and routes packets into or out of the mesh network from other parts of a distribution service or non-802.11 networks. The implementation of 802.11s pre-standard is illustrated in Figure 1. It supports metric-based multi-hop routing and data forwarding [11] at link layer, neighbor discovery, link quality measurement, media access coordination with quality of service support, security, and a user interface for mesh network configuration and management. To realize these mesh components, the firmware of the wireless network interface card is updated, new kernel modules and user interface applications are developed. The mesh prototype supports both Linux and Windows. However, all experiments introduced in this paper are conducted on Linux machines with Linux kernel 2.6.5+.

We employ the PlanetLab architecture to facilitate overlay maintenance and virtualization on the OverMesh platform. PlanetLab is a large distributed testbed for overlay networks on the Internet. It has evolved rapidly over the past three years and as of this writing, it encompasses 670 servers hosted at 326 sites and spanning 35 countries. PlanetLab employs Linux VServer [12] to virtualize servers. A central goal of PlanetLab is to support distributed virtualization – allocating a widely distributed set of virtual machines for a user community or application. Research groups are able to build their own virtualized overlay (called *slice* in PlanetLab) in which they can experiment with a variety of planetary-scale services, including file sharing and network-embedded storage, content distribution networks, routing and multicast overlays, scalable object location, anomaly detection mechanisms, and network measurement tools. Researchers continue to expand PlanetLab and investigate new research directions, such as introducing clusters to PlanetLab, using other machine virtualization technologies [23-27], federating PlanetLab, and private PlanetLab [34-36].

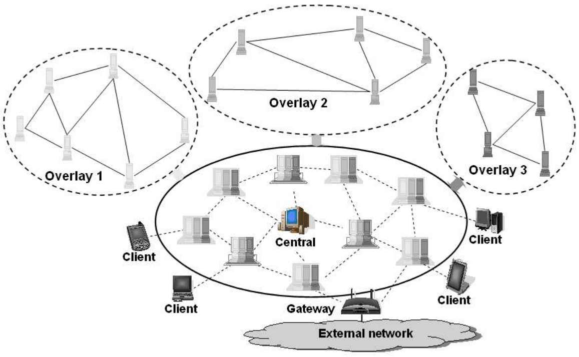


Fig. 2. OverMesh distributed system.

As illustrated in Figure 2, the current OverMesh prototype includes the following four components:

Central is responsible for overlay management. Any new node joining the system needs to contact the Central in order to be authorized. Whenever a node is selected to participate in an overlay, its information is stored in the Central. But the Central itself does not need to participate in any distributed services provided by overlays, nor does it need to help with multi-hop routing in the wireless mesh network.

Gateways connect the mesh network to external networks, such as the Internet or other wireless networks.

Mesh nodes are the physical nodes participating in the multi-hop wireless communication and the overlay services, as represented by the computers in the solid circle in Figure 2. Each mesh node is partitioned to multiple separated virtual machines at the operating system level. Each virtual machine may participate in an overlay (dashed circle in Figure 2) that involves virtual machines from multiple mesh nodes. All virtual machines in an overlay form a virtual network where nodes collaborate with each other in order to provide a distributed service for any of the mesh nodes and clients in the system requiring access to the service. We assume that mesh nodes are mostly stationary; however, some may be mobile albeit maintaining a certain degree of operational stability and connectivity to the OverMesh system.

Clients connect to the nearest mesh node through wired or wireless links. They can participate in the multi-hop wireless networking and data forwarding if they are mobile and communicating through wireless links, but they do not contribute to providing overlay services. Thus they are not managed by the Central.

While PlanetLab targets large distributed overlay networks supported by dedicated servers in the Internet, we focus on realizing the virtualized overlay on personal computers in a wireless mesh network. By participating in the research activity on private PlanetLab [34], firstly we reengineered the existing PlanetLab service architecture to operate in a local intranet environment. We then simplified and customized the PlanetLab codes to operate on a private WLAN mesh network. Figure 1 shows the current system stack of a mesh node. In addition to the mesh networking components residing in link layer, the virtual machines are supported by a virtual machine monitor. It sits below the traditional network layer while the existing distributed services in PlanetLab can operate on the platform. However, to make more efficient use of limited resources in the underlying wireless mesh networks, several cross-layer functions are included. They provide the status of underlying networks to the upper layers for resource management and control purposes. The major steps to setup an OverMesh platform are described below.

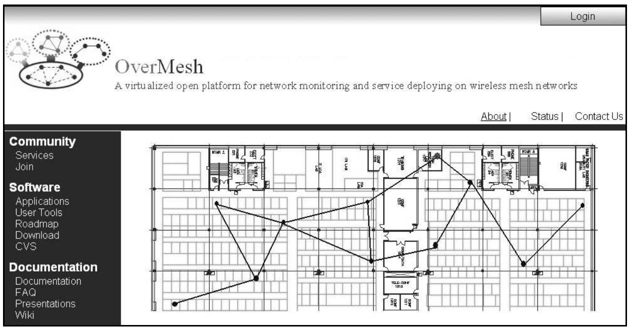


Fig. 3. Website at Central.

1. Setup the Central. Although a server is usually used to host the Central in a private PlanetLab, we use a laptop as a Central in order to facilitate its wireless communication with other mesh nodes. It has an Intel Pro Wireless 2200 card [13] supporting IEEE 802.11b and g. It is connected to the external Internet through a cable so that it can download packages from external server. It also acts as a gateway so that all mesh nodes can also be connected to the Internet. The Central maintains the authorization, installation, and status of all mesh nodes. For this purpose, it hosts a web site for managing the mesh nodes, distributed virtual machines, and overlays. A snapshot of the homepage is displayed in Figure 3. It illustrates a possible mesh network topology in an office building with many cubicles and meeting rooms. The dots represent the Central and mesh nodes. They are all communicating through multi-hop wireless links.

2. Install a mesh node. We use laptops as mesh nodes. A new mesh node is booted from a CD which temporarily installs a light-weight Linux kernel in the memory and enables the wireless mesh networking. The CD contains a specific key file generated by the Central for each new node. When the new mesh node is booted from the CD, it first contacts the Central through wireless links to verify the key. If it is authorized by the Central, the new mesh node will continue to download a latest installation kernel from the Central and install it. In this manner, the installation kernel can always be upgraded by the Central. The mesh networking support was added to both the boot CD and the installation Linux kernel in the Central so that any new mesh node can always communicate with the Central through wireless links.

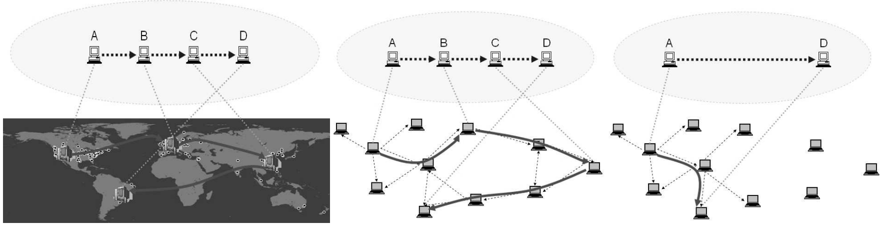
3. Setup the clients. A client only needs to install the mesh network software so that it can communicate to any node in the system through a nearby mesh node or client.

4. Deploy a service. When users want to deploy a new service on the mesh network, they should first request the administrator to add a new overlay through the website at the Central. Then a user can login to the website to manage this overlay. Since the Central has both wired and wireless connections, the user can login from either the external Internet or a mesh node or client in the mesh network. The users will be able to add or delete any mesh node in their own overlay. Every selected mesh node will be automatically notified and configure itself to add a new virtual machine to participate in the overlay. Finally, users can login to every participating virtual machine and install their particular service and application software without interaction with any other services supported by the same physical machine. Most existing services running on the public PlanetLab can be easily ported to this platform for wireless mesh networks.

4 Cross-Layer Searching

When realizing the OverMesh platform, we noticed that the existing overlay networks designed for the Internet do not work as efficiently on wireless networks as expected. One could experience significant delays and high packet loss rate. This is mainly due to the separation between the upper overlays and the lower network, MAC, and physical layers. How to provide fast and efficient services on the resource constrained wireless mesh networks is a challenging research topic. Current work on overlays usually does not consider the underlying network conditions. On the other hand, current research on wireless mesh networks is mainly focused on the network and lower layers. Our approach to deal with this issue is cross-layer adaptation. Cross-layer design for mobile wireless networks is becoming increasingly important [16, 17]. In general, it can be used in network monitoring and network resource management and control. Specifically, we study a cross-layer overlay searching algorithm in this paper.

A searching overlay, such as OpenDHT [14, 15] deployed in PlanetLab, provides a common lookup service to various applications such as information queries, distributed file storage and sharing. A (key, value) pair is stored in a randomly selected node in the searching overlay. However, it can be found by any node in the overlay using a searching algorithm. The most efficient overlay searching algorithms, such as Chord, Pastry, Tapestry, and CAN, are based on the DHT [10]. Each virtual node maintains a small overlay routing table. For Chord, Pastry and Tapestry, the routing table size is $O(\log n)$ and the hop count of a searching route is $O(\log n)$, where n is the network size. For CAN, the routing table size is $O(d)$ and the hop count of a searching path is $O(dn^{1/d})$, where d is a constant. However, these overlay searching algorithms only achieve the short searching route in terms of the number of virtual hops in the overlay, while they attempt to make the underlying network transparent to the nodes in the overlay. Figure 4a illustrates the overlay searching on the Internet. We assume that a (key, value) pair is stored in node D, while node A only knows the key and sends out a query for the corresponding value. The actual searching path of the query involves two loops. The outer loop is the searching in the overlay, the short virtual path A→B→C→D in the overlay can be found by some DHT based overlay searching algorithm. The inner loop is the real network routing. For example, in order to send the query from A to B, it should be routed in the physical network. But the overlay neighbors A and B may be physically far away from each other. In a wired network, the overhead in the underlying network is not a big concern due to the high speed cable and dedicated routers. However, for the multi-hop wireless mesh network, in order to find a physical route for each overlay hop, the source node has to broadcast a route request to all its neighbors repeatedly until the destination is reached, as demonstrated by Figure 4b. This makes the previous two-loop overlay searching algorithm on wired networks inefficient on wireless mesh networks.



(a) (b) (c)

Fig. 4. Searching route on wired and wireless networks. (a) Overlay searching on the Internet. (b) Overlay searching on a wireless mesh network. (c) Cross-layer searching on a wireless mesh network.

Given factors such as the limited bandwidth available, sharing of the medium, power constraint, mobility, dynamic topologies, and varying link conditions in wireless mesh networks, the searching algorithm should consider the network condition in lower layers in order to quickly find the value corresponding to the requested key by using minimal network resources. At the physical layer, due to the broadcast nature of wireless channel, every transmission from a node can be heard by all its neighbors. At the network layer, the most widely used ad hoc routing protocols, such as AODV and DSR, are all based on broadcast. Whenever a source node wants to find a route to a destination node, the source node broadcasts a route request. If any node receiving the request is the destination node or knows a route to the destination, it sends a route reply back to the source; otherwise it will rebroadcast the route request. The complexity of a broadcast at the network and lower layers is O(n). We know that the complexity of searching algorithms in the overlay is lower-bounded by log$n$ achieved by DHT-based overlay searching algorithms. When we apply the overlay searching algorithm on the wireless mesh networks, it will need O(log(n)) virtual hops in the overlay, while each virtual hop requires O(n) physical hops to find the real route. So the complexity we can currently achieve is lower-bounded by O($n$log$n$).

The proposed cross-layer searching algorithm achieves the complexity of O(n) by taking advantage of the network layer broadcast to route the overlay searching request. When a node knows a key k and wants to find its corresponding value, it first maps k to a virtual network address *vAddress* using some known hash function. The hash function is employed to make *vAddress* unique and randomly distributed in the network address space. A magic mark may be embedded to *vAddress* so that other nodes can easily recognize that this is not a real network address. The source node then broadcasts a network route request for *vAddress*. Any node in the searching overlay will check received route request and compare the requested virtual address against all the keys it has. If the same key is found, the corresponding value will be sent back to the source node as a network route reply. Basically, this new algorithm combines the two searching processes in the overlay and the network layer into one single searching process. Some techniques can be further employed to enhance the algorithm. For example, when the value is sent back to the source node in the network route reply, an intermediate node relaying it can cache the (key, value) pair so that it can also respond when the same searching request comes next time. In addition to storing (key, value) pairs, a node can also save the corresponding *vAddress* for each pair and sort them by *vAddress*, so it can search the list and respond to a route request faster. Figure 5 details the cross-layer information exchanges between the overlay and the network layer in both the source and destination nodes.

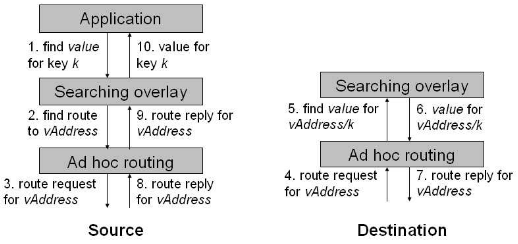

Application
1. find value for key k
10. value for key k
Searching overlay
2. find route to vAddress
9. route reply for vAddress
Ad hoc routing
3. route request for vAddress
8. route reply for vAddress
Source
Searching overlay
5. find value for vAddress/k
6. value for vAddress/k
Ad hoc routing
4. route request for vAddress
7. route reply for vAddress
Destination

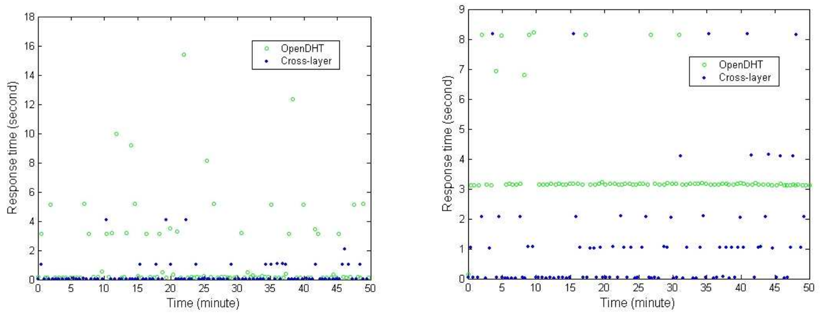
Fig. 5. Cross-layer searching algorithm.

Figure 4c illustrates the advantage of using the proposed cross-layer searching algorithm. The physically shortest route can be quickly found. Experimentation and simulation results will be presented in the next section.

5 Results

Extensive experiments have been conducted on the implemented OverMesh platform. They demonstrate that the network-wide performance is affected by network topology, data traffic load, mobility, and even human activities. Due to the space limitation, we will only present the experimentation results related to the searching algorithm.

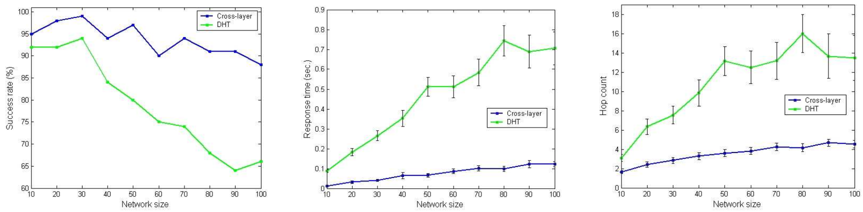
Figure 6 compares the cross-layer overlay searching (dot) and the OpenDHT overlay searching (circle) when 100 searching requests are sent. Two types of network topology are tested. In the mesh topology, a node can communicate with any of its neighbors, while in the linear topology, a node is forced to only communicate to one of its neighbors. It is evident that the cross-layer searching renders less response time. In average, the response time of cross-layer searching is 1.15 seconds, while the response time of OpenDHT searching is 3.55 seconds.



(a) Mesh network topology (b) Linear network topology

Fig. 6. Experimentation results for searching.

To compare the cross-layer searching and DHT searching in more scenarios, we further did simulations based on an event-driven simulator for mobile wireless networks that we recently developed [33]. This is an open source simulator which supports physical layer broadcast, Signal-Noise-Ratio (SNR) based packet capture, random way-point mobility model, CSMA based MAC layer protocol, and link quality based ad hoc network routing. It enables us to study larger mobile wireless mesh networks and many different network topologies. Figure 7 gives the simulation results when there are up to 100 nodes. Each result shown in the figure represents the average of 100 runs on randomly generated network topologies, with confidence interval of 95%. The cross-layer searching requires much less hops for each request than the DHT based overlay searching. This is mainly due to the mismatch between the overlay topology and the physical network topology. The larger number of physical hops in the DHT overlay searching introduces a lot more network routing control packets and therefore more packet collisions and losses. This will also make the response time much larger than that of the cross-layer searching.



(a) Success ratio (b) Response time (c) Hop count

Fig. 7. Simulation result for searching with different network size.

More simulations have been done when the number of concurrent searching requests is up to 10, when the maximum moving speed of each node increases from 1 m/s to 10 m/s, and when the packet loss zone increases from 0.1 db to 1 db. All these simulation results demonstrate that the cross-layer overlay searching significantly outperforms the DHT based overlay searching on the wireless mesh networks in terms of higher success ratio, less response time, and less physical hop count.

6 Conclusion and Future Work

This paper proposes a platform that realizes the conceptual architecture of overlays on wireless mesh networks, i.e. OverMesh. It provides an open and scalable testbed for developing and deploying distributed services and applications on wireless mesh networks. The current prototype is based on IEEE 802.11s pre-standard and private PlanetLab. A cross-layer overlay searching algorithm is proposed to enable fast searching on the resource limited wireless mesh networks.

We are currently pursuing several research directions based on the OverMesh architecture and the implemented platform. For example, extending the prototype to different scale wireless mesh networks or the multiple radio wireless networks; applying Xen virtual machine [26] and hardware virtualization technology [27]; investigating more cross-layer approaches to optimize resource usage and improve network efficiency in the wireless environment. We are also working on improving the current implementation. For example, customizing the boot CD in order to make the installation process easier; introducing Windows machines to the system; providing an open source toolkit so that other researchers can easily build their own system and conduct various research activities on wireless mesh networks.

References

1. Vicente, J., Rungta, S., Ding, G., Krishnaswamy, D., Chan, W., Miao, K.: OverMesh: Network Centric Computing. under submission to IEEE Communications Magazine (2006)
2. Chlamtac, I., Conti, M., Liu, J. J. -N.: Mobile Ad Hoc Networking: Imperatives and Challenges. Ad Hoc Networks 1 (2003) 13–64

3. Akyildiz, I. F., Wang, X., Wang, W.: Wireless Mesh Networks: A Survey. Computer Networks Journal 47 (2005) 445–487
4. IEEE 802.11s WLAN ESS Mesh Network working group
5. IEEE 802.15 WPAN working group, task group 5
6. http://www.zigbee.org
7. http://www.wimaxforum.org
8. Figueiredo, R., Dinda, P. A., Fortes, J.: Resource Virtualization Renaissance. IEEE Computer 38 (2005) 28–31
9. http://www.planet-lab.org
10. Balakrishnan, H., Kaashoek, M. F., Karger, D., Morris, R., Stoica, I.: Looking Up Data in P2P Systems. Communications of the ACM 46 (2003) 43–48
11. De Couto, D. S. J., Aguayo, D., Bicket, J., Morris, R.: A High-Throughput Path Metric for Multi-Hop Wireless Routing. ACM MobiCom (2003)
12. http://linux-vserver.org
13. http://ipw2200.sourceforge.net
14. Rhea, S., *et al.*: OpenDHT: A Public DHT Service and Its Uses. ACM SigComm (2005)
15. http://www.bamboo-dht.org
16. Shakkottai, S., Rappaport, T. S., Karlsson, P. C.: Cross-Layer Design for Wireless Networks. IEEE Communications Magazine 41(2003) 74–80
17. Cross-Layer Protocol Engineering for Wireless Mobile Networks. IEEE Communications Magazine 44 (2006)
18. Andersen, D. G., Balakrishnan, H., Kaashoek, M. F., Morris, R.: Resilient Overlay Networks. ACM SOSP (2001)
19. Eriksson, H.: Mbone: The Multicast Backbone. Communications of the ACM 37(1994) 54–60
20. Day, M., Cain, B., Tomlinson, B., Rzewski, P.: A Model for Content Internetworking. Internet RFC 3466
21. Xu, Z., Tang, C., Zhang, Z.: Building Topology-Aware Overlays using Global Soft-State. ICDCS (2003)
22. Nakao, A., Peterson, L., Bavier, A.: A Routing Underlay for Overlay Networks. ACM SigComm (2003)
23. http://www.vmware.com
24. http://denali.cs.washington.edu
25. http://user-mode-linux.sourceforge.net
26. http://www.xensource.com
27. Ublig, R., *et al.*: Intel Virtualization Technology. IEEE Computer 38(2005) 48–56
28. Sirer, E. G., Grimm, R., Bershad, B. N., Gregory, A. J., McDirmid, S.: Distributed Virtual Machines: A System Architecture for Network Computing. SIGOPS European Workshop (1998)
29. http://virtuoso.cs.northwestern.edu
30. http://www.emulab.net
31. Chandra, R., Bahl, P., Bahl, P.: MultiNet: Connecting to Multiple IEEE 802.11 Networks Using a Single Wireless Card. IEEE InfoCom (2004)
32. http://research.microsoft.com/netres/projects/virtualwifi
33. http://wireless-matlab.sourceforge.net
34. http://www.planet-lab.org/php/information.php?meeting_id=8
35. EverLab, http://www.cs.huji.ac.il/labs/danss/p2p/private-planetlab/index.html
36. OneLab, http://lsirwww.epfl.ch/PlanetLabEverywhere/slides/Lausanne.ppt