Arijit Banerjee, Junguk Cho, Eric Eide, Jonathon Duerig, Binh Nguyen, Robert Ricci, Jacobus Van der Merwe, Kirk Webb and Gary Wong *Flux Research Group, University of Utah*

Editors: Sharad Agarwal and Marco Gruteser

PhantomNet:

Research Infrastructure for Mobile Networking, Cloud Computing and Software-Defined Networking

Photo, bigstockphoto.com

The PhantomNet facility allows experimenters to combine mobile networking, cloud computing and software-defined networking in a single environment. It is an *end-to-end testbed*, meaning that it supports experiments not just with mobile end-user devices but also with a cellular core network that can be configured and extended with new technologies. This article introduces PhantomNet and presents a road map for its future development. The current PhantomNet prototype is available now at no cost to researchers and educational users.

THE NEED FOR A MOBILE NETWORKING TESTBED

Three technologies are transforming the Internet as we have known it. The first is mobile networking, driven by the ubiquity of cellular networking and devices. The second is cloud computing, driven by the economics of server-side computing and the need to deploy large, elastic services. The third is software-defined networking (SDN), which makes it possible for a network manager to rapidly provision, configure, and reprogram physical network infrastructure to meet a wide variety of requirements. How do these technologies relate to each other? How can they be combined to better meet the needs of today's increasingly mobile population of Internet users? What new services and applications will this enable? How can these technologies be used to realize evolvable network architectures? Together with the broader networking research community, we recognize that current mobile network architectures are at an inflection point where the concerted effort of the research community is needed to unlock the full potential of mobile networks, cloud computing and SDN.

To unlock this potential, educators and researchers need *testbeds:* facilities where the next generation of technologists can receive hands-on training on state-of-the-art equipment and where new ideas can be implemented, deployed and rigorously evaluated under realistic but controlled conditions.

To address this need we are creating the PhantomNet testbed. Located at the University of Utah, but accessible to researchers and educators everywhere via the Internet, PhantomNet is an *end-to-end mobility testbed*. By this we mean that it supports experimentation not only with mobile devices, such as smartphones and tablets, but also with cellular data networks that support these devices. Moreover, because it allows experimenters to control the core services and topology of a cellular network, it allows experimenters to blend mobility with other technologies–especially cloud computing and SDN.

An experimenter can instantiate a small but complete cellular network within PhantomNet, including end-user devices, cellular base stations, compute nodes for hosting cellular core network devices and a network that ties the devices together. Balancing the trade-off between scale and realism, and to cater to the differing needs of researchers, PhantomNet contains both actual, physical devices, as well as emulated versions thereof. The experimenter has complete control over the allocated devices and can configure them as needed. Moreover, multiple experimenters can use PhantomNet at the same time: each experimenter receives a separate experiment or "slice" of the testbed resources, and PhantomNet ensures that concurrent experiments will not affect each other.

To support both cutting-edge research and education, we are creating PhantomNet with four key goals in mind. These are mobility realism, end-to-end control, flexibility and repeatability.

Mobility realism. A key challenge with mobile network testbeds is creating realism in the movement of mobile devices. Current approaches include trace-driven robotic mobility, where devices are mounted on robots that move according to prerecorded traces [12, 9]; vehicle-mounted mobility, where devices are attached to buses or other vehicles [14]; and volunteer-driven mobility, where human couriers carry mobile devices from place to place [3, 5]. While PhantomNet does not currently attempt to provide full mobile channel emulation, a fully programmable attenuator array is provided to testbed users to mediate the radio access network (RAN). This facility is adequate for simulating various mobile environments under completely automatic and deterministic control and is capable of introducing controlled path loss specified by the user (e.g., as predicted by the widely used COST 231 RF propagation model [6]).

End-to-end control. A mobile testbed is most useful when experimenters can control and change all aspects of their mobile networks. In particular, one should be able to change "core network" aspects that deal with functions such as routing, forwarding, mobility management, authorization and authentication. Research into future mobile network architectures (e.g., [11, 13]) requires the ability to modify most or all mobile-network functions. PhantomNet provides this kind of control at multiple levels, through network configuration and software extensions.

Flexibility. It should be relatively simple for a beginning researcher or student to use basic testbed functionality. At the same time, it must be possible for a more advanced user to fundamentally change functionality as required. These requirements can easily be in tension. For example, if a testbed relies on a commercial mobile provider (e.g., [3]), then network functionality is "taken care of," but the ability to change that network functionality is very limited. On the other hand, if a testbed only provides "bare metal" functionality, then the knowledge and effort needed to create even a basic mobile network can be too much for a novice user. PHANTOMNET resolves this tension by providing both preconfigured experiments for beginning users and deep programmability for experts.

Repeatability. As a scientific instrument, a testbed should allow for repeatability of experiments. This goal is sometimes in direct tension with realism. For example, because volunteer-driven mobility involves a person carrying an actual device from place to place, it may provide the greatest realism for an experiment – but it provides no inherent repeatability. Conversely, simulated wireless conditions can provide high repeatability, but the simulation may not correspond to realistic conditions. PHANTOMNET addresses the challenge of repeatability in two ways. First, for simulating the conditions of a mobile network, it uses a programmable attenuation array mentioned previously. Thus, mobility experiments in PHANTOMNET use real wireless signals over real devices, and the program that simulates the motion of those devices is repeatable.

Second, because PHANTOMNET is based on the Emulab testbed-management software [15], it inherits the features of that software that promote repeatable experimentation. This includes the ability to capture experiment setups – e.g., complete disk images—and the isolation of experiments that are running on the testbed at the same time.

Below, we first provide a brief overview of the PHANTOMNET components and functionality. We then consider examples of the teaching and research enabled by PHANTOMNET, and conclude with the current status of our infrastructure and our future plans.

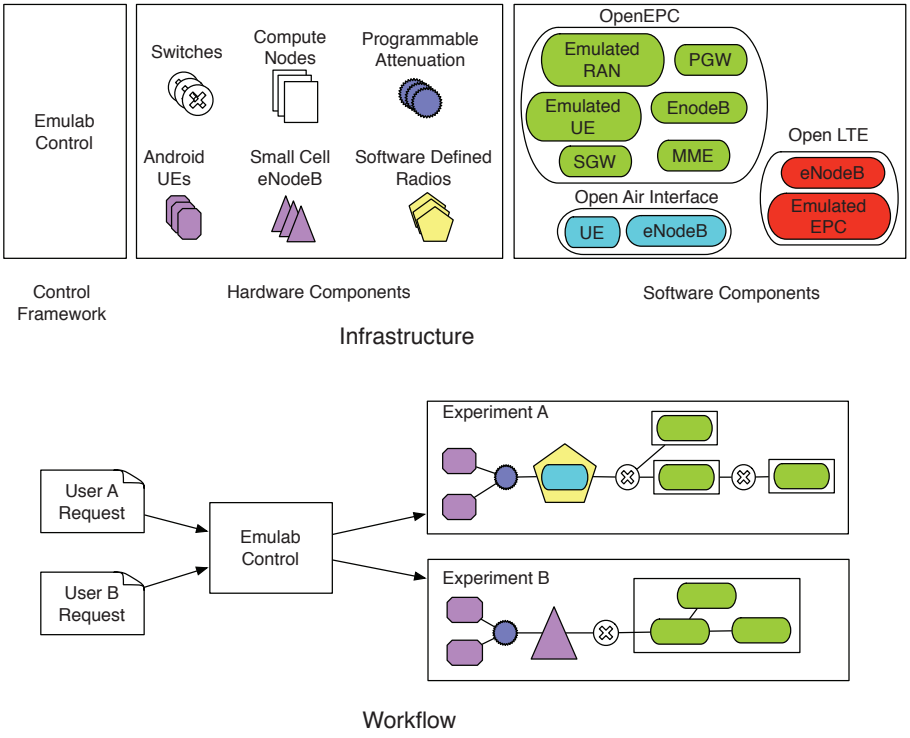

FIGURE 1. PHANTOMNET infrastructure and workflow.

PHANTOMNET OVERVIEW

Figure 1 depicts the PHANTOMNET infrastructure and workflow. As shown in the top part of the figure, PHANTOMNET is composed of three top-level components: a control framework, a set of hardware components and a set of software components. PHANTOMNET utilizes and builds on the Emulab control framework [15].

Hardware. PHANTOMNET hardware components include traditional network testbed resources, such as compute nodes connected by switches. In addition, PHANTOMNET provides a number of hardware resources that are of interest to researchers working with mobile platforms. PHANTOMNET gives access to off-the-shelf mobile handsets, or UEs (user equipment) using mobile networking nomenclature, in the form of Android handsets. These handsets are paired with compute nodes, allowing experimenters with Android Debug Bridge (ADB) access to the devices. PHANTOMNET also provides off-the-shelf small cell base stations, or eNodeBs according to mobile networking terminology. We use small cell eNodeBs from ip.access, operating on LTE (long term evolution) spectrum bands compatible with our Android handsets.

PHANTOMNET also provides access to software defined radio (SDR) hardware. The SDR devices are attached as peripherals to dedicated compute nodes. When combined with the appropriate software (see below), these devices can act as base stations (eNodeBs). We have SDR hardware in the form of Ettus Research USRP B210 and Nuand bladeRF radios. These devices perform tuning, amplification and ADC/DAC in hardware, and then communicate baseband samples over USB 3.0 links to the host nodes, allowing a great deal of flexibility in higher-level signal processing. Each of the SDR devices offered in PHANTOMNET cover (at least) the entire UHF spectrum, and provide at least 28 MHz of full-duplex RF bandwidth.

This mix of resource types allows experimenters to target different areas of interest from higher-level protocol interactions down to wireless signal manipulation. To facilitate clean, repeatable experimentation, wireless devices are connected through a programmable

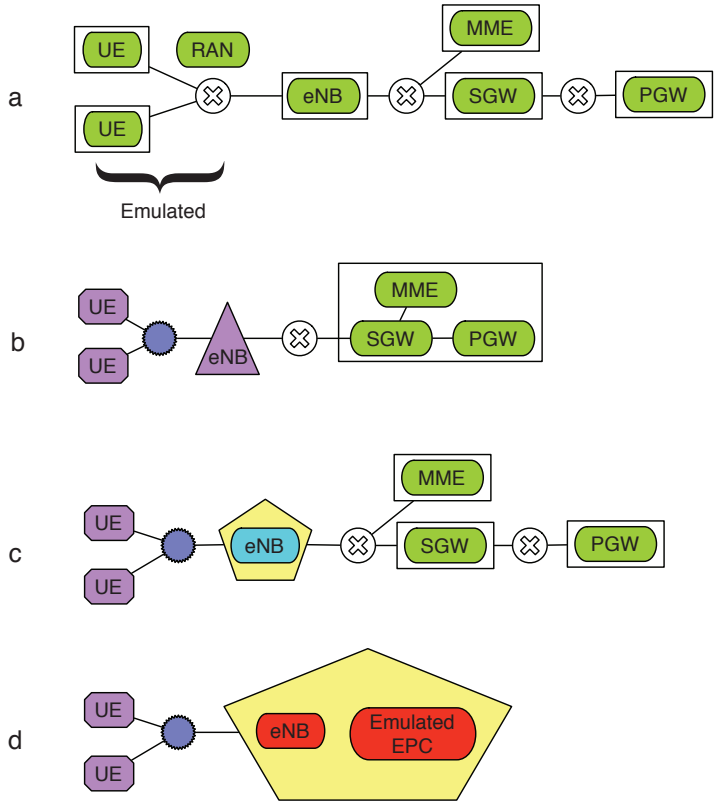

FIGURE 2. Example of PHANTOMNET experiments.

attenuator matrix. This matrix allows users to mix and match end-user equipment (UE devices) flexibly with access point nodes (off-the-shelf and software-defined varieties). Relative signal strength between devices can be programmatically adjusted with this setup.

Software. Software resources available in PhantomNet include: an evolved packet core suite called OpenEPC [1], and radio access network (RAN) over SDR implementations from Open Air Interface (OAI) [10] and OpenLTE [2]. OpenEPC is discussed in the next paragraph. OAI includes SDR-based user equipment (UE) and access point (eNodeB) implementations and an emerging 3GPP LTE Evolved Packet Core (EPC) implementation. OpenLTE is similar, but is built on top of GNU Radio and does not tread much beyond the radio access layers of LTE. PhantomNet allows users to request these resources and configure them for particular purposes.

A centerpiece in PhantomNet's set of software offerings is OpenEPC. This Evolved Packet Core (EPC) software [1], developed by Fraunhofer FOCUS (www.fokus.fraunhofer.de), includes much of the functionality codified by the 3GPP LTE version 12 specification. This includes services for handling end-user device (UE) attachment, handover (mobility), policy and charging, etc. Because of license restrictions, the OpenEPC functionality in PhantomNet is available only in binary form. This functionality is, however, fully composable and configurable. (OAI and OpenLTE are open source projects with full source code access.)

Workflow. The bottom part of Figure 1 shows a simplified depiction of the PhantomNet workflow. Users of the testbed submit a description of their desired resources and topology as an NS script (based loosely on the ns-2 simulator's scripting language). This user-supplied resource and topology specifications are mapped to available resources by Emulab's constraint solver, and then the actual resources are allocated and configured as requested. Outside of individual experiment resource management, PhantomNet also relies on the Emulab framework to manage user accounts, project membership, disk images and storage space.

At the tail end of the provisioning process, PhantomNet hooks in to the Emulab setup process to perform tasks that are specific to its unique resources. For example, on compute nodes running OpenEPC services, PhantomNetcode binds and modifies configurations for the specified core mobile network "role" (e.g., S-GW), and also handles address management for the created experiment. The node-side setup code also allows for user-provided hooks, which are run before and/or after the rest of the PhantomNet setup pieces. Users specify setup directives via the NS file, which are opaquely passed through by Emulab to the PhantomNet setup code.

Experiments. The series of diagrams in Figure 2 illustrate a number of different mobile networking setups that PhantomNet can support. These are only example configurations. I.e., given the flexibility of the PhantomNet environment, various combinations and configurations are possible.

Figure 2 (a) shows an evolved packet core (EPC) setup where all components are provided by OpenEPC. In this case, the end user equipment (UEs) and the access point (eNodeB) radio access network (RAN) components are emulated. The topology is simple here; users can add more emulated UEs, eNodeB access points and other components as desired. As in a standard EPC deployment, once a UE attaches, its data traffic will flow across the eNodeB, through the S-GW, on to the P-GW, and finally out to the Internet.

Figure 2 (b) shows a mobility setup with two off-the-shelf Android devices and an ip.access small cell in place of the respective emulated components in diagram (a). As shown in the figure, these (real) RAN components are connected through the programmable attenuator. The rest of the setup provides the same functionality as in diagram (a), however, in this case it is instantiated as virtual machine (VM) instances on a single physical compute node.

Figure 2 (c) shows a version of the setup

in diagram (b) where the eNodeB access point has been replaced by one realized with Open Air Interface software running on a compute node hosting Software Defined Radio hardware (the pentagon in the diagram represents the host/radio combination). RAN communication occurs via the USRP or bladeRF SDR hardware, while higher layer protocols are processed on the host. The UEs are still commodity Android devices and the rest of the EPC is realized with OpenEPC.

Finally, Figure 2 (d) illustrates an alternative mobile configuration where OpenLTE runs on a compute node and provides RAN services via SDR. The same OpenLTE node implements just enough emulated EPC functionality to allow unmodified, standard UE devices to attach. UE data traffic egresses directly from the Emulated EPC node. Alternative mobile core technologies under investigation, e.g., SoftCell [8], could be combined with such an edge topology.

Experiment specification. We have mentioned that experimental configurations are specified through NS files via the PhantomNet front-end. To illustrate the simplicity of this approach, the following snippet demonstrates how a user can request a node running OpenEPC with a combined S-GW/MME EPC role, and another acting as PDN-GW:

```
...
# Add node with combined S-GW/MME role.
epcnode sgw "sgw-mme-sgsn"
addtolan net_b $sgw
addtolan net_d $sgw
# Add node with PDN-GW role.
epcnode pgw "pgw"
addtolan net_a $pgw
addtolan net_b $pgw
...
```

As shown in this extract, only a small number of directives are required to declare the compute node, set its role, and connect it to the (wired) topology. The "net a," "net b," etc., names in these directives are user-friendly labels we have used to declare the various LANs (these align with the labels used in the OpenEPC documentation). The PhantomNetweb user interface renders a graphical view and resource report from the NS specification that a user can inspect to verify his or her topology.

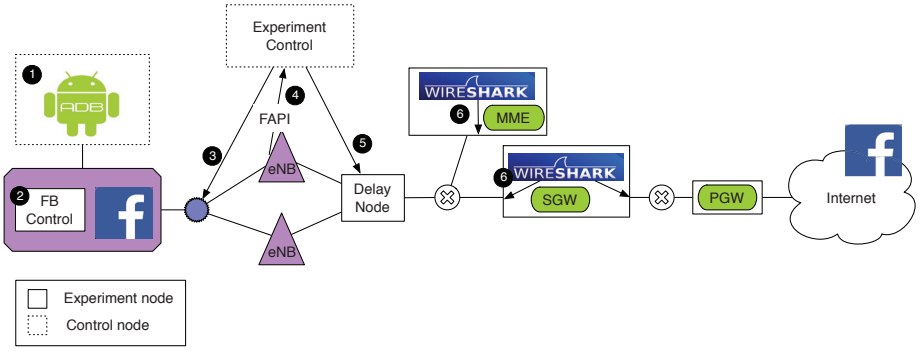

FIGURE 3. Exploring application/network interaction.

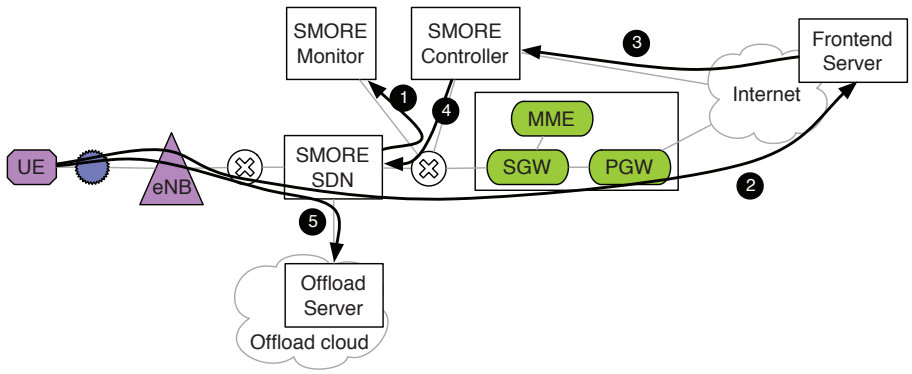

FIGURE 4. SMORE architecture.

USING PHANTOMNET

PhantomNet offers significant flexibility depending on the teaching or research goals of the user. For example, setups such as those shown in Figure 2 (a) and (b), might be used for teaching or research related to standard LTE/EPC functionality. Since core mobile network functions (EPC functions) can be run as virtual machine instances, as shown in Figure 2 (b), the platform might also be used to explore network function virtualization (NFV). Setups like Figure 2 (c) and (d) might be used by researchers who want to modify low level eNodeB mechanisms, e.g., exploring eNodeB scheduling algorithms, or, in the case of Figure 2 (d), who are interested in alternative mobile core functionality, e.g., not using standard EPC core functions.

Below we consider two additional use cases in a bit more detail. We note that variants of these use cases are available as self-help tutorials and/or lab assignments from the PhantomNet portal [7] and we are constantly adding to this collection.

Application and network interaction. Figure 3 shows a fairly generic experimental setup that might be used to explore the interaction between apps (e.g., Facebook) running on a mobile device and the mobile network. From left to right, the setup consist of an Android device, connected to two small cell eNodeBs via a programmable attenuator array. The eNodeBs are connected to the core mobile network (MME, SGW, PGW) via a delay node (available as a standard component in Emulab) and finally to the (real) Internet. Note that the figure shows both nodes that are part of the experiment (with solid lines) as well as shared infrastructure hosts (with dotted lines), which experimenters can access out-of-band to control channels and interact with their experiment. For example, each Android device is connected to a compute node via USB to enable Android Debug

Bridge control from the host (#1). To ensure repeatability of the experiment, the user might deploy an orchestration program (e.g., "FB control", #2) to control the application under study. If the user is interested in understanding the behavior of the app during a mobile handover, the relative attenuation between the Android device and the two eNodeB devices can be manipulated to cause a handover (#3). If the user is interested in the behavior of the radio access network (RAN), the femtocell API (FAPI) interface on the eNodeB can be enable to realize this monitoring (#4). If the user wants to experiment with the impact of delay in the core network, the delay can be manipulated via the delay node (#5). If the user wants to understand the impact of core network protocol interaction on application behavior – e.g., when the device goes into idle state, network resources are released and requires a subsequent paging request to be reestablished – the user can monitor the control plane interaction on the core elements using Wireshark (#6).

Mobile network architecture. Our second use case is a mobile offloading architecture, called SMORE, we have developed and prototyped on PhantomNet[4]. Figure 4 depicts the SMORE architecture in the context of PhantomNet. The purpose of the SMORE architecture is to allow offloading of *selected* traffic to an in-mobile-core-network cloud platform. For example, offloading might be performed for delay sensitive applications. In SMORE this is achieved, without requiring protocol changes to the mobile core network, by deploying an SDN framework in the core mobile network, between the eNodeBs and the standard core network elements (SGW, PGW, MME). Figure 4 shows a simplified workflow for this architecture in PhantomNet context. A mobile device (UE) is assumed to attach to the mobile network as per normal. The SMORE SDN and SMORE Monitor elements capture and store relevant information about this attach procedure, i.e., tunnel identifiers, by monitoring the normal control plane interaction (#1). In this scenario, it is assumed that the user would access a "front-end server" in the Internet via the normal data path (#2). The front-end server then explicitly requests that traffic associated with its service be offloaded to an offload cloud instance (#3). For example, the front-end server might be associated with a game matchmaking service, and request offloading of gaming traffic to a specific gaming engine in the offload cloud. Based on this request and stored information about the UE in question, the SMORE controller will issue an offloading request to the SMORE SDN framework (#4). The SMORE SDN framework will then perform the necessary decapsulation and encapsulation functions to ensure that only the traffic associated with the offloaded service be directed to and from the offload server in the cloud (#5).

LOOKING AHEAD

The PhantomNet testbed is open for business for academic users [7]. Variants of the use cases described here are available from the PhantomNet portal as self-help tutorials or example lab assignments, and we are continuously adding to that set. Because of hardware constraints, some of the functionality described above, notably access to physical RAN equipment (Android devices, small-cell eNodeBs, and SDR devices) is currently not available through the PhantomNet automated framework. (On request, we are making it available through a semi-automated process.) However, we are actively building out the PhantomNet infrastructure and we expect general availability of this functionality in the near future. We are actively looking for educators and researchers to use PhantomNet, so please contact us if you are interested.

ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation under Grant No. 1305384. ■

REFERENCES

[1] OpenEPC. http://www.openepc.com/.
[2] OpenLTE. http://openlte.sourceforge.net/.
[3] Rishi Baldawa, Geoffrey Challen, Murat Demirbas, Jay Inamdar, Taeyeon K, Steven Y. Ko, Tevfik Kosar, Lokesh Mandvekar, and Chunming Qiao. Phonelab: A large-scale participatory smartphone testbed. Poster at MobiCom, September 2011. Whitepaper: http://sensorlab.cs.dartmouth.edu/NSFPervasiveComputingAtScale/pdf/1569392923.pdf
[4] Junguk Cho, Binh Nguyen, Arijit Banerjee, Robert Ricci, Jacobus Van der Merwe, and Kirk Webb. SMORE: Software-defined networking mobile offloading architecture. In *Proceedings of the 4th Workshop on All Things Cellular*, pages 21 – 26, August 2014.
[5] Eduardo Cuervo, Peter Gilbert, Bi Wu, and Landon P. Cox. Crowdlab: An architecture for volunteer mobile testbeds. In *Proceedings of the 3rd International Conference on Communication Systems & Networks*, January 2011.
[6] Eraldo Damosso and Luis Correia, editors. *Digital Mobile Radio Towards Future Generation Systems*. European Co-operation in the field of Scientific and Technical Research, 1996.
[7] Flux Research Group. PhantomNet – Mobility Testbed. https://www.phantomnet.org/.
[8] Xin Jin, Li Erran Li, Laurent Vanbever, and Jennifer Rexford. Softcell: Scalable and flexible cellular core network architecture. In *Proceedings of the 9th ACM Conference on Emerging Networking Experiments and Technologies*, pages 163 – 174, December 2013.
[9] D. Johnson, T. Stack, R. Fish, D. M. Flickinger, L. Stoller, R. Ricci, and J. Lepreau. Mobile Emulab: A robotic wireless and sensor network testbed. In *Proceedings of the 25th IEEE International Conference on Computer Communications*, April 2006.
[10] MobileCommunicationsDepartmentat EURECOM.OpenAirInterface. http://www.openairinterface.org/.
[11] D. Raychaudhuri. MobilityFirst: A robust and trustworthy mobility-centric architecture for the future inter- net. PIMRC Keynote, 2011. http://mobilityfirst.winlab.rutgers.edu/documents/PIMRC_ keynote_MF_911.pdf.
[12] D. Raychaudhuri, M. Ott, and I. Seskar. Orbit radio grid tested for evaluation of next-generation wireless network protocols. In *Proceedings of the 1st International Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities*, February 2005.
[13] Ivan Seskar, Kiran Nagaraja, Sam Nelson, and Dipankar Raychaudhuri. MobilityFirst future internet architecture project. In *Proceedings of the 7th Asian Internet Engineering Conference*, December 2011.
[14] Hamed Soroush, Nilanjan Banerjee, Aruna Balasubramanian, Mark D. Corner, Brian Neil Levine, and Brian Lynn. Dome: a diverse outdoor mobile testbed. In *Proceedings of the 1st ACM International Workshop on Hot Topics of Planet-Scale Mobility Measurements*, June 2009.
[15] Brian White, Jay Lepreau, Leigh Stoller, Robert Ricci, Shashi Guruprasad, Mac Newbold, Mike Hibler, Chad Barb, and Abhijeet Joglekar. An Integrated Experimental Environment for Distributed Systems and Networks. In *Proceedings of the 5th Symposium on Operating Systems Design and Implementation*, December 2002.