# Wireless Network Performance Evaluation through Emulation: WiMAX Case Study

Razvan Beuran*, Muhammad Imran Tariq†, Shinsuke Miwa* and Yoichi Shinoda†
*National Institute of Information and Communications Technology,
Hokuriku StarBED Technology Center
Nomi, Ishikawa, Japan
Email: {razvan, danna}@nict.go.jp
†Japan Advanced Institute of Science and Technology
Nomi, Ishikawa, Japan
Email: {imran, shinoda}@jaist.ac.jp

***Abstract*—In this paper we present a framework for evaluating wireless network performance through emulation. The framework uses a hybrid design in which the PHY layer is represented by a probabilistic model, whereas the MAC layer is executed as a functional module. Actual network applications and protocols are employed in order to create realistic test conditions.**

**We use WiMAX as a case study to show the feasibility of our emulation approach. A standard-based model for the WiMAX PHY was developed, and it was integrated with the WiMAX MAC implementation in ns-3 to make WiMAX performance evaluation through emulation possible. Several experimental results, both in static and mobile scenarios, demonstrate the validity of the framework and indicate some of its potential uses.**

***Keywords*—*Performance evaluation, wireless networks, network emulation, WiMAX, ns-3.***

## I. Introduction

Network emulation [1] is a powerful technique for performance evaluation using real network applications and protocols, and it is particularly useful for scenarios involving wireless networks and mobility, which are difficult to reproduce and control accurately in real environments. Moreover, the use of real applications and protocols provides an increased realism compared to pure network simulation experiments.

While our initial work focused on the performance evaluation of the network applications and protocols themselves (Layer 3 and above) [2], we believe that there are many situations in which the protocol to be evaluated is at a lower layer, such as Layer 2 (MAC layer). Using real applications and protocols provides a realistic workload for these experiments.

In this paper we present our approach for making possible MAC layer protocol evaluation through network emulation. To demonstrate the feasibility of this approach, we implemented the corresponding experiment framework using WiMAX as a case study. Nevertheless, our approach is applicable to other wireless network technologies; for instance LTE support could be added with minimal effort using the technique described in this paper. Moreover, a similar extension for Wi-Fi networks is ongoing, with the target of studying IEEE 802.11s (Layer 2 mesh network) protocol improvements.

Our framework is built on top of the large-scale network testbed StarBED [3], and extends the wireless network emulation toolset QOMET [4], which are both developed and managed by the Hokuriku StarBED Technology Center of the National Institute of Information and Communications Technology (NICT) in Ishikawa, Japan.

The main contributions of this paper are as follows:

- An IEEE 802.16e standard-based mobile WiMAX PHY model that was implemented in QOMET;
- The integration of the above PHY model with an ns-3 based WiMAX MAC module that was previously developed on StarBED [5], thus providing full WiMAX emulation functionality;
- A series of experimental results that validate the correct operation of the network emulation framework, and that demonstrate its practicality.

The remainder of this paper is organized as follows. In Section II we present an overview of the network emulation framework that we designed and implemented. Descriptions of the main components of the framework, namely the WiMAX PHY layer model and MAC layer module follow in Sections III and IV, respectively. We then present a series of validation experiments in Section V. The paper ends with related work, conclusions and references.

## II. Framework Overview

Our work on wireless network emulation started with the assumption that the components under test will be real network applications and protocols running on top of the emulated wireless networks; examples include robot motion planning algorithms [6], or routing metrics for wireless mesh networks [7]. This task is achieved in the most efficient manner by modeling both lower network layers, PHY and MAC, in a probabilistic manner [4] as shown in Figure 1(a).

However, there are situations when researchers want to focus on MAC layer protocols, while still using real network applications and higher-layer protocols for performance evaluation. This requires a different experiment approach, as follows: while the PHY layer can still be modeled in a statistical manner, the MAC layer under test must be implemented and executed on top of the PHY layer as shown in Figure 1(b).

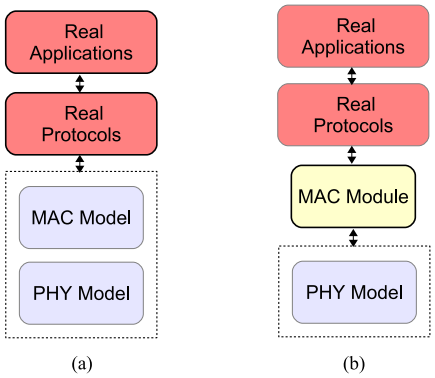


Fig. 1. Emulation experiment approaches: (a) Both PHY and MAC layers are modeled; (b) Only the PHY layer is modeled, whereas the MAC layer implementation is executed as such.

WiMAX represents a good case study for demonstrating the usefulness of the latter approach because several aspects of the WiMAX MAC layer are still being actively researched. For instance, aspects such as scheduling mechanisms or rate adaptation algorithms are not explicitly specified in the IEEE 802.16d/e standards, and various alternatives for these mechanisms are under investigation.

### A. Framework Infrastructure

The infrastructure for our framework is represented by the large-scale network experiment environment called StarBED, located at the Hokuriku StarBED Technology Center of NICT in Ishikawa, Japan [3].

StarBED makes available for experiments more than 1400 PCs and the interconnecting network equipment. The large number of experiment hosts, and the versatile network architecture of StarBED make it possible to conduct a wide range of network experiments on this testbed. The StarBED experiment and control networks are separated, so as to isolate experiment traffic from the management one.

SpringOS is the fundamental experiment-support software tool for StarBED [3]. SpringOS makes it possible to perform complex experiments with a large number of hosts by assisting users in the following tasks:

- *Experiment preparation:* Configure the experiment hosts and network so that they are ready to use;
- *Experiment execution:* Effectively carry out the experiment by performing the necessary commands on the experiment hosts in the required order.

### B. Wireless Network Emulation

The wireless network emulation capabilities discussed in this paper are based on the extension of QOMET (Quality Observation and Mobility Experiment Tools) [4]. QOMET is a set of tools for reproducing the communication conditions between experiment nodes. QOMET relies on the experiment management mechanisms of StarBED for its distributed execution to achieve the emulation of the overall network on the testbed. The main QOMET features are:

- *Wireless network technologies:* Support for standards such as IEEE 802.11a/b/g, IEEE 802.15.4, etc;
- *Topography:* The topography of the emulated environment (streets and buildings), can be defined in 2D/3D;
- *Node mobility:* Several models can be used to describe the trajectory of the emulated nodes: random way, behavioral model, etc.

The core functionality of QOMET is provided through the libraries called `deltaQ` and `wireconf`. The `deltaQ` library is used to compute the communication conditions between wireless nodes given a user-defined scenario. The library includes the implementation of models for wireless network technologies, propagation, and mobility. The user-defined scenario specifies the properties of the wireless nodes (position, network technology parameters, mobility pattern), and of the environment in which they are placed (attenuation, shadowing, street and building structures, and so on). These properties are used to create a "virtual world" that corresponds to the emulated scenario, in which the wireless nodes move and communicate with each other.

The `wireconf` library recreates during live experiments the communication conditions computed by `deltaQ` by applying the corresponding network degradation (packet loss, delay and bandwidth limitation) to the experiment traffic. This is effectively achieved on the testbed by means of the `ipfw3` link emulation module [8].

## III. WiMAX PHY Model

The function of the WiMAX PHY model that we implemented in the QOMET `deltaQ` library is to make it possible to calculate the PHY layer conditions that are to be reproduced through emulation. The parameters that describe these conditions are: (i) bandwidth; (ii) frame error rate; (iii) delay. The models used for the first two parameters will be described next. Delay in WiMAX results implicitly from the operation of the MAC layer, therefore at PHY layer it is not necessary to introduce any additional delay; hence nothing needs to be done for this parameter in the PHY model.

### A. Bandwidth

The model that we implemented in order to compute the available bandwidth in WiMAX at PHY layer is based on the IEEE 802.16e standard [9] and the discussion in [10]. The algorithm is the following:

1) Initialize the model parameters (sampling factor, FFT size, signaling overhead, number of downlink and uplink symbols, number of subcarriers) depending on the modulation selected and other relevant user settings;
2) Compute the basic parameters of the model (sampling frequency, sample time, subcarrier spacing, symbol and guard time, total number of symbols, number of downlink and uplink slots);
3) Compute the bandwidth-related parameters (e.g., bytes per slot) and the resulting data rate depending on other PHY settings, such as MIMO configuration, repetition factor, etc.

We emphasize the fact that bandwidth in this context refers to the bandwidth available at PHY layer for the WiMAX nodes, and that will be enforced by `wireconf` during the live experiment. Aspects such as how this bandwidth is shared by multiple nodes are handled at MAC layer by the ns-3 based MAC module that will be described in Section IV, therefore they do not need to be dealt with in the PHY model.

The above model has a low computation complexity, and it only needs to be reevaluated if the configured WiMAX modulation changes. Thus, for real-time operation it is preferable to the ns-3 model, which does certain calculations on a per packet basis, since it doesn't use the concept of available bandwidth. Moreover, our model can still be used with WiMAX MAC implementations other than the ns-3 one if they are available.

### B. Frame error rate

In order to determine the frame error rate corresponding to a certain WiMAX modulation and its associated PHY parameters, the receive sensitivity threshold for that modulation is necessary. The value of this threshold could be obtained from the specifications of particular WiMAX devices, but we opted here for a more generic approach. Thus, we calculate the receive sensitivity threshold, $S$, based on the corresponding recommended equation in the IEEE 802.16e [9] standard:

$$\begin{aligned} S = & -114 + SNR_{Rx} - 10\log_{10}(R) + \\ & + 10\log_{10}(F_S \cdot N_{Used}/N_{FFT}) + Imp_{Loss} + NF, \end{aligned} \quad (1)$$

where $SNR_{Rx}$ is a modulation-dependent Signal-to-Noise Ratio constant, $R$ is the repetition factor, $F_S$ is the sampling frequency, $N_{Used}$ is the number of used downlink subcarriers, and $N_{FFT}$ is the size of the FFT. For the additional constants $Imp_{Loss}$, the implementation loss, and $NF$, the noise factor, we use the default values specified in the 802.16e standard, namely 5 and 8, respectively.

As done previously in QOMET [4], the threshold $S$ computed above is then used to calculate the bit error rate, $BER$, according to an exponential dependency:

$$BER = BER_S \cdot e^{\gamma(S+\delta-P_r)}, \quad (2)$$

where $BER_S$ is the bit error rate at the sensitivity threshold (equal to $10^{-6}$ according to the 802.16e standard), $\gamma$ and $\delta$ are calibration constants, and $P_r$ is the received power strength in dBm. Through fitting with respect to the corresponding detailed analytical models for error rate, the values for $\gamma$ and $\delta$ were determined to be equal to 1.7 and 0.0 for AWGN fading, and 0.23 and 32.73 for Rayleigh fading, respectively (as average for all modulations). Note that we limit the $BER$ given by the previous equation to the value 1.0, since it represents a probability.

Finally, the frame error rate, $FER$, which will be enforced during emulation by `wireconf`, is calculated from $BER$ as:

$$FER = 1 - (1 - BER)^{8 \cdot F_{size}}, \quad (3)$$

where $F_{size}$ is the frame size expressed in bytes.

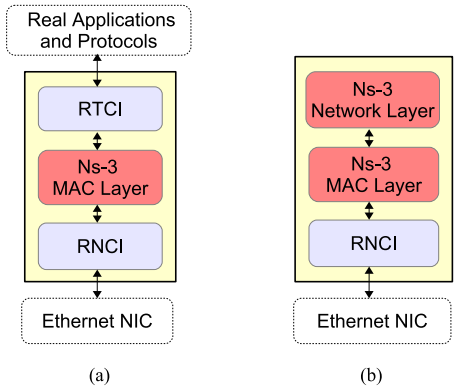


Fig. 2. Ns-3 based WiMAX MAC modules: (a) Subscriber Station; (b) Base Station.

## IV. WiMAX MAC Module

For some wireless network technologies, software implementations of the MAC layer are readily available as open source. This is the case for instance for open80211s [11], an open-source implementation of the IEEE 802.11s wireless mesh standard. In other cases (e.g., WiMAX or LTE) such implementations are not yet publicly available.

Developing a MAC layer module from scratch is time consuming, and we believe that a promising alternative for fast prototyping is to reuse existing simulation code such as that in ns-3. In [5] we presented in detail our approach for making possible ns-3 based emulation of WiMAX networks, and we summarize here the main points.

In Figure 2 we show the conceptual structure of the ns-3 based WiMAX MAC modules that we developed. In the case of the Subscriber Station (SS), two supplementary modules were required in addition to the ns-3 WiMAX MAC code. One was the Real Traffic Communication Interface (RTCI), which allows passing real application and protocol traffic between the OS and the WiMAX module. The other one was the Real Network Communication Interface (RNCI), which allows passing traffic between the WiMAX module and the Ethernet NIC so that it can reach other nodes, thus enabling distributed execution. In the case of the Base Station (BS) only the RNCI module is required, because the BS doesn't need to communicate with actual applications/protocols, hence it can use the default network layer (Layer 3) in ns-3.

### A. Real Traffic Communication Interface (RTCI)

The RTCI component leverages the TapBridge NetDevice class in ns-3 to provide real traffic communication between the actual host PC and the ns-3 instance, so as to connect the host PC network stack with the ns-3 MAC layer. In particular, we configured the UseBridge mode of the TapBridge device, thus effectively extending a host OS bridge into ns-3.

To accomplish this, one needs to create a virtual device and add it to a predefined bridge. Ns-3 will then create a tap device and connect it to the same bridge, so that the host OS treats the tap device as a real network device connected to that bridge. Ns-3 then uses inter-process communication to connect the tap device to the equivalent bridge present in the simulator

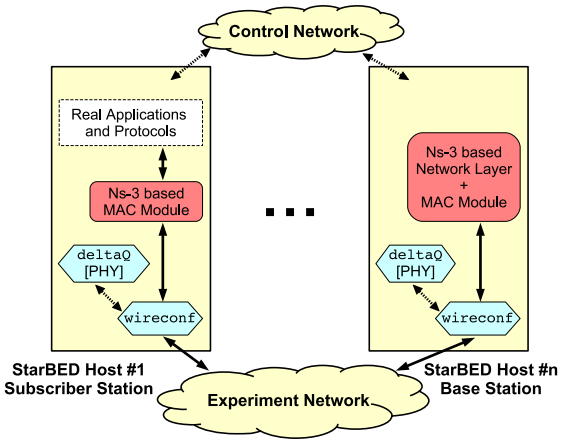


Fig. 3. Overall system architecture of the experiment framework.

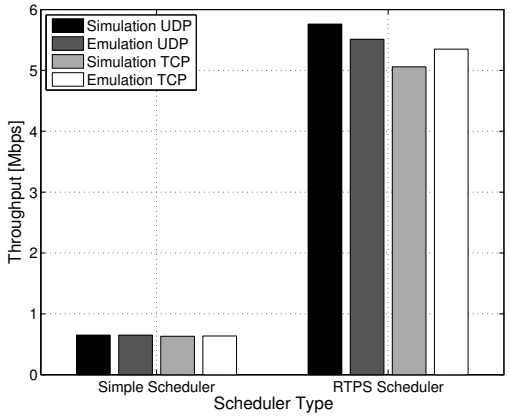


Fig. 4. Throughput results for static experiments with 1 BS and 2 SSs; we used both UDP and TCP traffic and two different schedulers.

instance, which is in our case part of the WiMAX MAC layer implementation.

### B. Real Network Communication Interface (RNCI)

By default, in ns-3 all the scenario nodes will be executed on the same machine. However, for emulation, which requires real-time execution, this can quickly become a bottleneck. The RNCI component is essential for making it possible to execute different WiMAX nodes on different hosts PCs. Such a distributed execution environment dramatically improves experiment scalability, since the available computing resources can be used in a more flexible manner.

For this purpose, RNCI makes it possible to send and receive the traffic of the ns-3 MAC layer over the experiment network of StarBED. The design of RNCI is inspired by the EmuNetDevice class in ns-3, and it has been adapted to the case of WiMAX. The RNCI of the sender connects to a real network interface and encapsulates the WiMAX frames into Ethernet frames, which are then sent over the testbed network, decapsulated by the RNCI of the receiver, and passed on to the corresponding ns-3 instance.

### C. System Architecture

The overall architecture of the hybrid system that we implemented is shown in Figure 3. Each StarBED host can work either as a WiMAX base station or as a subscriber station. SSs will run the real applications and protocols used during the evaluation, the ns-3 based MAC module, as well as the QOMET emulation components: `deltaQ` (that includes the WiMAX PHY implementation) and `wireconf`. The BS will only run the ns-3 based MAC module together with the ns-3 network layer, and the QOMET components. Experiment traffic produced by the nodes is communicated over the StarBED experiment network, whereas the management is done via the control network.

## V. Experimental Results

The experiments presented here focus mainly on demonstrating that the emulation architecture used in our framework produces similar results when compared to ns-3 simulation (as expected given that the ns-3 MAC implementation is integrated into the framework), while having the advantage of using real network applications/protocols on the end nodes.

The simulation experiments were done using ns-3.18. The emulation experiments were carried out on StarBED, with each emulated node running on a different physical host with dual-core Pentium 4, 3.2 GHz CPU and 8 GB RAM.

### A. Static Experiments

We tested first a static scenario in which 1 BS and 2 SSs are in the vicinity of each other, and the WiMAX modulation used for communication was set to the default one, `qam16_12`. In the case of simulation, traffic was generated using the UdpClient and UdpServer ns-3 applications for UDP experiments, and BulkSend and PacketSink for TCP ones. For emulation we used iperf v2.0.5 both for UDP and TCP transfers. The experiment duration was 300 s. In both cases the experiment traffic was classified as real-time traffic by the WiMAX classifier, and we used either the default ns-3 WiMAX scheduler (named “Simple Scheduler”) or the Real Time Polling Service (RTPS) scheduler.

As it can be seen in Figure 4, for the Simple Scheduler there are essentially no differences between simulation and emulation, and for the RTPS scheduler the differences are not larger than about 5%. We attribute these differences partially to the effects of the real network communication in the case of emulation, and more importantly to the differences between the applications used to generate traffic: ns-3 models for simulation, and iperf for emulation. Thus we conclude that there is a good general agreement between simulation and emulation results.

Experiments done with ping in the same circumstances (results not shown here due to space limitations) show that, as expected, there is a fixed increase in the end-to-end delay when using emulation by about 10 ms. This is due to the fact that application traffic has to pass through several virtual and real network interfaces from sender to receiver, as follows: RTCI and RNCI of the sender SS, RNCI of the BS, then RNCI and RTCI of the receiver SS. We will study possible OS optimizations that would allow us to decrease these OS level delays. Note that, although the experiment traffic is forwarded

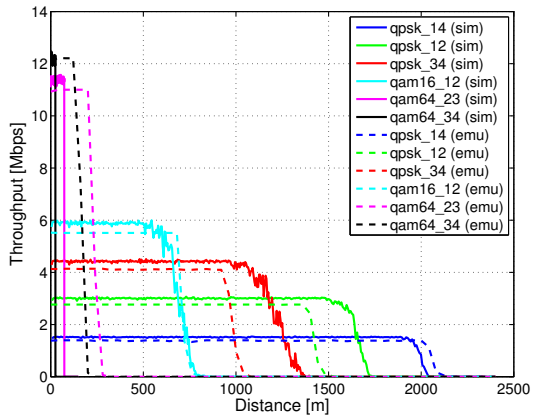


Fig. 5. Throughput results for mobility experiments with one SS moving away from the BS and the other SS; we used UDP traffic and various modulations.

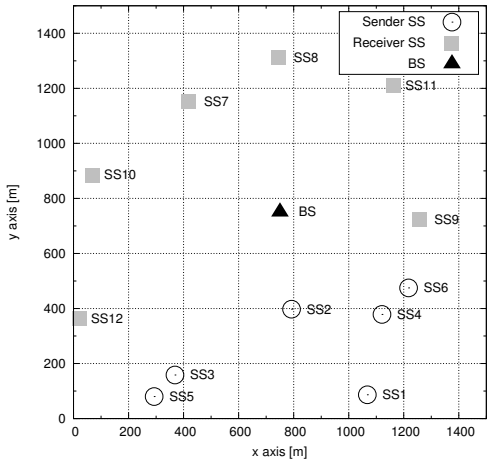


Fig. 6. Placement of the WiMAX nodes in an experiment with 12 SSs.

over the real testbed network, this delay is only of a few hundreds of microseconds, hence it does not significantly affect the end-to-end delay.

### B. Mobility Experiments

A second series of experiments investigated a mobility scenario in which one of the SSs moves away from the BS and the other SS with constant speed. Various WiMAX modulations available in ns-3 were used, and throughput was calculated using UDP traffic sent by the mobile SS in the same conditions as those described above. The scheduler used was RTPS Scheduler.

The throughput results are plotted in Figure 5 versus the distance between the mobile SS and the BS. Simulation results are shown with continuous line and emulation results with dashed line. For all modulations the maximum throughput values are close to each other, and for several of them (`qam16_12` and `qpsk_14`) the behavior is similar even as the mobile SS moves farther and farther away from the BS.

Given that the capacity-based PHY model implemented in QOMET is different from the one in ns-3, a certain difference in the communication range, as noticed for some modulations (e.g., `qpsk_34` and `qpsk_12`), was expected. Hence, we conclude that for mobility experiments too there is a generally good agreement between simulation and emulation results for all modulations, although differences in communication ranges are observed for some of them.

### C. Scalability Experiments

Finally we did several scalability tests to see how our framework performs in scenarios with a larger number of nodes. We used again UDP traffic and the RTPS scheduler.

After testing various scenarios, we noticed that with the used testbed PCs the maximum number of subscriber stations we can have in an experiment is 12. With a larger number of SSs, the CPU utilization of the BS exceeds 80% and the system becomes unstable. One solution is to move at least the BS, which represents the bottleneck of the system since it handles the traffic of all the SSs, to a more powerful machine. Various optimization techniques could also potentially be applied.

In Figure 6 we show the placement of the nodes in an experiment with 12 SSs that are placed randomly in a 1500 x 1500 m area, with the BS located in the center of the area (denoted by a triangle). The first six SSs, SS1 to SS6, behave as UDP traffic senders (represented by circles), whereas the last six ones, SS7 to SS12, behave as receivers (represented by squares). The offered load was 500 Kbps since larger values resulted in segmentation faults in ns-3; we are investigating the actual cause of this issue. We estimate that the maximum possible throughput is around 800 Kbps per sender, given the total number of senders and the modulation used.

In Figure 7 we compare the throughput achieved in simulation and emulation in the above scenario. We notice that the amount of traffic received shows similar trends in simulation and emulation, albeit with a lower performance for some nodes in the case of emulation. Note that the subscriber station pairs that have lower throughput (SS5-SS11 and SS6-SS12) are those affected by poor communication conditions due to the distance with respect to the BS. Poor communication is precisely the case when the differences between the PHY models used in simulation and emulation become evident, as emphasized already in Section V-B. Therefore we deem these differences, of about 20% at most, to be acceptable for our hybrid framework.

## VI. Related Work

Several other projects use emulation in order to evaluate WiMAX network performance. In the case of the NCTUns-based hybrid system presented in [12], a real WiMAX network is interfaced with the NCTUns simulator. However, the simulator itself only reproduces the backbone network and wired end nodes, hence the communication conditions in the WiMAX network are not under user's control.

The work related to the WEBS WiMAX emulation testbed [13] actually focuses on the multimedia components of the testbed, whereas the WiMAX emulation itself is done by using the already existing emulation capabilities of the QualNet simulator, namely the IPNE module. The authors only emulate

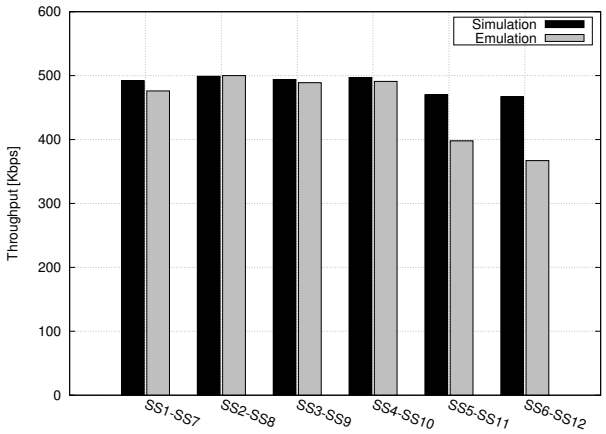


Fig. 7. Throughput results for scalability experiments with 12 SSs; we used UDP traffic and 6 sender-receiver pairs.

a network with one BS and one SS, and no analysis is done about scalability, which we assume to be low given our own experience with IPNE.

Other testbeds use real WiMAX devices, such as the WiMAX Extension to Isolated Research Data networks (WEIRD) [14]. WEIRD is an architecture for outdoor testbeds that was used for several deployments in various locations. The focus of the architecture is however on the applications running over the WiMAX network, and experiment conditions cannot be controlled.

The GENI project provides a WiMAX virtualized experimental testbed [15] in which real WiMAX devices are connected to each other via a programmable attenuator, with a total of 1 BS and 8 SSs. While offering realism and control to users, the testbed is limited by its physical scale. Moreover, aspects such as mobility are not directly addressed.

## VII. Conclusion

We have presented an emulation framework that uses a hybrid approach in order to allow the evaluation of MAC protocols in realistic conditions. This is achieved by running MAC protocol implementations over a wired network that reproduces PHY level network conditions. The traffic is generated by using real network applications and protocols (Layer 3 and above).

To demonstrate the potential of this approach we have used WiMAX as a case study, by modeling the PHY layer of IEEE 802.16e and running on top of it the ns-3 WiMAX MAC layer implementation. Appropriate interface modules were created to make possible sending and receiving real traffic to and from this implementation.

A series of static experiments showed that the emulation framework has good accuracy in terms of throughput compared to simulation, both for UDP and TCP, with differences not exceeding 5% in all the tested cases. In mobile scenarios too comparable results in terms of throughput were obtained for all modulations, altough communication range differences were observed for some of them. Our scalability experiments showed that the CPU is a bottleneck for the BS when exceeding a total number of 12 SSs, but the measured throughput had similar trends in simulation and emulation. We believe that this issue can be solved by utilizing more powerful hosts for emulation, and we are currently in the process of evaluating this solution.

The approach that we used when designing our WiMAX emulation framework can be used for ns-3 based LTE emulation with only minor changes, and we plan to add such support in the future in order to enable experiments with this wireless network technology as well. A similar approach is applicable for the 802.11s mesh network standard, and we are collaborating with an ongoing project focusing on this topic that is being carried out at the Japan Advanced Institute of Science and Technology in Ishikawa, Japan.

## References

[1] R. Beuran, *Introduction to Network Emulation*. Pan Stanford Publishing, 2012.

[2] R. Beuran, L. T. Nguyen, T. Miyachi, J. Nakata, K. Chinen, Y. Tan, and Y. Shinoda, "QOMB: A Wireless Network Emulation Testbed," in *Proceedings of the IEEE Global Communications Conference (GLOBECOM 2009)*, 2009.

[3] T. Miyachi, K. Chinen, and Y. Shinoda, "StarBED and SpringOS: Large-scale General Purpose Network Testbed and Supporting Software," in *Proceedings of the Intl. Conf. on Performance Evaluation Methodologies and Tools (Valuetools 2006)*, 2006.

[4] R. Beuran, J. Nakata, T. Okada, L. T. Nguyen, Y. Tan, and Y. Shinoda, "A Multi-purpose Wireless Network Emulator: QOMET," in *Proceedings of the 22nd IEEE International Conference on Advanced Information Networking and Applications (AINA 2008) Workshops, FINA 2008 symposium*, 2008, pp. 223–228.

[5] M. I. Tariq, R. Beuran, S. Miwa, and Y. Shinoda, "Ns-3 Based WiMAX Emulation System," in *Proceedings of the 6th Asia Pacific Conference on Wireless and Mobile 2014 (APWiMob 2014)*, 2014.

[6] R. Beuran, J. Nakata, T. Okada, Y. Tan, and Y. Shinoda, "Real-time Emulation of Networked Robot Systems," in *Proceedings of the 1st International Conference on Simulation Tools and Techniques for Communications, Networks and Systems (SIMUTools 2008)*, 2008.

[7] L. T. Nguyen, R. Beuran, and Y. Shinoda, "An Interference and Load Aware Routing Metric for Wireless Mesh Networks," *International Journal of Ad Hoc and Ubiquitous Computing (IJAHUC)*, vol. 7, no. 1, pp. 25–37, 2011.

[8] M. Carbone and L. Rizzo, "Dummynet Revisited," *ACM SIGCOMM Computer Communication Review*, vol. 40, no. 2, pp. 12–20, 2010.

[9] IEEE 802.16e-2005 Standard, *Mobile Broadband Wireless Access System*. IEEE, 2005.

[10] C. So-In, R. Jain, and A. K. Tamimi, "Capacity Evaluation for IEEE 802.16e Mobile WiMAX," *Journal of Computer Systems, Networks and Communications*, 2010.

[11] Open80211s, "Open-source implementation of the IEEE 802.11s wireless mesh standard," http://open80211s.org/open80211s/index.html.

[12] S. Y. Wang, C. Lin, P. H. Koo, Y. M. Haung, J. S. Y. Taleb, Z. Zhang, and J. Hou, "NCTUns Emulation Testbed for Evaluating Real-Life Applications over WiMAX Networks," in *Proceeding of the 21st IEEE International Symposium on Personal Indoor and Mobile Radio Communications (PIMRC)*, 2010, pp. 2030–2034.

[13] R. K. Kalle, M. Raj, K. Balakrishnan, and D. Das, "WEBS: WiMAX Emulation Testbed to Benchmark Streaming Multimedia QoS," in *Proceedings of International Conference on Internet Multimedia Services Architecture and Applications (IMSAA)*, 2009, pp. 1–6.

[14] S. Mignanti, G. Tamea, I. Marchetti, M. Castellanoa, A. Cimmino, F. Andreotti, M. Spada, P. M. Neves, G. Landi, P. Simoes, and K. Pentikousis, "WEIRD testbeds with fixed and mobile WiMAX technology for user applications, telemedicine and monitoring of impervious areas," in *Proceedings of the 4th International Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities (TRIDENTCOM)*, 2008.

[15] GENI, "Global Environment for Network Innovation," http://www.geni.net/.