EmPOWER: A Testbed for Network Function Virtualization Research and Experimentation

Roberto Riggio[†], Tinku Rasheed[†], and Fabrizio Granelli[‡]
[†]CREATE-NET, Trento, Italy; *Email: name.surname@create-net.org*
[‡]University of Trento, Trento, Italy; *Email: granelli@unitn.it*

***Abstract*—Software Defined Networking (SDN) and Network Function Virtualization (NFV) are making their way into the research agenda of all the major players in the networking domain. Parallely, testbeds and experimental facilities are widely regarded as the fundamental step–stone to future "clean slate" networking. However, designing and building experimental facilities can hardly be considered a trivial step for either researchers and practitioners. Scale, flexibility, and ease of use are just some of the challenges faced by a testbed designer. These considerations are at the base of efforts such as GENI in USA, AKARI in Japan, FEDERICA, NOVI and OFELIA in Europe which provide federated and open facilities for the Future Internet research agenda. Albeit the importance of such facilities is unquestioned, today there is still a dearth of testbed exploiting SDN and NFV concepts in the wireless networking domain. In this paper we present *EmPOWER* an experimental testbed which aims at filling this gap by offering an open platform on top of which novel concepts can be tested at scale. The *EmPOWER* testbed is composed by 30 nodes and is currently used by both undergraduate and graduate students at the University of Trento and by the research staff at CREATE-NET.**

***Index Terms*—Software Defined Networking, Network Function Virtualization, WiFi, Testbeds, Open–source**

I. Introduction

SDN and NFV are two of the most promising concepts that are set to bring innovation in the ossified networking landscape. Current SDN efforts start from the consideration that, by providing full visibility of the network from a logically–centralized controller, it is possible to simplify network control and management tasks [1]. Nevertheless, the so called "north–bound" API exposed by today's controllers is still very primitive hindering the development of modular and flexible network applications: As a matter of fact, if with OpenFlow a practical and concrete forwarding abstraction has been found, considerable efforts are still required toward the definition of new programming models. In this regard, several SDN proponents argue in favor of high level declarative languages in order to specify the desired behavior of the network leaving to the underlying Network Operating System, or NOS, its actual implementation. Such vision is summarized by the seminal speech by Scott Shenker: "The Future of Networking, and the Past of Protocols". High–level languages [2], such as: Frenetic [3], Pyretic [4], Procera [1], The Flow Management Language [5], and Nettle [6], aim exactly at providing such level of abstraction.

SDN and NFV are in the research agenda of all the major projects and initiative in the broad Future Internet domain. Examples are GENI in USA, AKARI in Japan, FEDERICA, NOVI and OFELIA in Europe. Several open facilities such as Norbit (NICTA), w-iLab.t (iMinds), NITOS (UTH), Netmode (NTUA), SmartSantander (UC), and FuSeCo (FOKUS) already focus on wireless technologies. Some of these testbeds (Norbit, NITOS, Netmode) focus on research and experimentation on the WiFi domain allowing experimenters to gain full access of a variable number of open WiFi Access Points (APs) where custom software can be installed. Other facilities, such as w-iLab.t and SmartSantander, aim at providing support for experimentation in the IoT domain. Finally, FuSeCo delivers a research facility, integrating OpenIMS and a 3GPP Evolved Packet Core prototype platform. However, albeit the relevance of such facilities is beyond doubt, at the moment there is a dearth of fully virtualized experimental facilities exploiting NFV concepts in the Wireless Networking domain in general and for WiFi-based networks in particular. Moreover, the current OpenFlow ecosystem in term of controller, slicing platforms, and software/hardware switches, provides little support for the WiFi domain.

In this paper we present *EmPOWER* a novel open experimental testbed which aims at filling the gap in the experimental facilities offering for SDN&NFV research and experimentation. The *EmPOWER* testbed is composed by 30 nodes and is currently used by both undergraduate and graduate students at the University of Trento and by the research staff at CREATE-NET. Experiments can take full control of a slice of the network which is kept isolated (at a logical level) from the other slices. Traffic can come from either users that decide to opt–in a certain experiment or by mirroring the traffic of a production. Moreover, the experimenter can monitor in real–time and with the desired resolution the actual energy consumption at either device or slice level using the Energino open energy consumption monitoring and management toolkit [7], [8].

The remainder of this paper is structured as follows. The basic requirements that drove *EmPOWER*'s design are discussed in Sec. II. Section III presents the *EmPOWER* architecture. A particular use case focusing on energy efficiency is presented in Sec.IV. Finally, we draw the conclusions highlighting limitations and future work in Sec. V.

II. Requirements

Implementing an effective SDN platform supporting advanced virtualization concepts and running on top of commodity WiFi devices raises several challenges. In this section we survey three of such challenges dealing with the conceptual slicing and programming model to be supported, with the collection of the actual state of the network, and finally with actual sharing of the facility resources among experimenters.

A. Slicing and programming model

The platform shall support high–level programming primitives with regard to the control of the network. Such primitive shall be powerful enough to relieve the experimenter from the implementation details specific to the WiFi standard, i.e. association/deassociation mechanisms, control frame exchange and status management. The feature is deemed of capital importance if the policies devised by the experimenters are to be ported to other wireless environments such as LTE–A. Moreover, the platform shall not impose additional requirements on the WiFi clients. This means that, unless the experimenter is planning to deploy custom WiFi clients, the platform shall not mandate for either software and/or hardware modification to the WiFi clients. Finally, the slicing mechanism shall put the experimenter in control of a portion of the network (AP and switches). An opt–in mechanism shall be available for users that want to use a certain slice for their traffic.

B. Querying the status of the network

The platform shall provide the experimenter with a rich set of primitive to query the status of the network. Such primitives shall be as much general as possible in order to support a broad spectrum of use cases. OpenFlow switches already allow collecting statistics related to ports and flows in the network in terms of number and size of the packets. A wireless deployment shall support statistics related to the wireless medium including, for example, RSSI, frame loss ratio, per–MCS (Modulation Coding Scheme) statistics, etc. Given the momentum generated by current research activities on energy efficient networking, the platform shall provide the experimenter with real–time energy consumption information with high temporal precision and small granularity. Such information shall be provided both at the device and at the sliver level, i.e. the platform shall report both the energy consumption of an AP as well as the energy consumption of a specific slice.

C. Federation Architecture

The instantiation of a virtual networks on top of the platform should be performed through either a web–based control framework or an equivalent command line interface which should allow an easy reservation of network resources, either nodes or links. Ideally, a user should be allowed to select the APs and the switches he/she intends to use during the experiment and then the system should instantiate the network

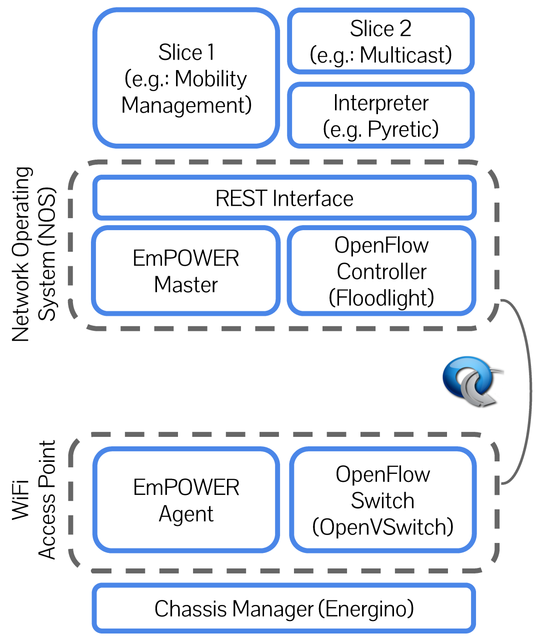
Slice 1
(e.g.: Mobility Management)
Slice 2
(e.g.: Multicast)
Interpreter
(e.g. Pyretic)
Network Operating System (NOS)
REST Interface
EmPOWER Master
OpenFlow Controller (Floodlight)
WiFi Access Point
EmPOWER Agent
OpenFlow Switch (OpenVSwitch)
Chassis Manager (Energino)

Fig. 1: The *EmPOWER* system architecture.

in a transparent way to the user. Suitable traffic generation and collection tools shall also be available.

III. The *EmPOWER* platform

A. System Architecture

The system architecture, sketched in Fig. 1, consists of a single Master and multiple Agents running on each AP. The Master, implemented on top of an OpenFlow controller, has a global view of the network in terms of clients, flows, and infrastructure. The Agents allow multiple clients to be treated as a set of logically isolated clients connected to different ports of a switch. Network application run on top of the controller and can exploit either the embedded Floodlight REST interface or an intermediate interpreter, e.g. Pyretic [4]. Each network application effectively runs in an isolated slice controlling all or just a subset of the available APs.

The *EmPOWER* testbed is built around open and freely available toolkits: OpenVSwitch and the Click Modular Router for the datapath; Floodlight as the controller and Arduino as the power manager. Network applications, i.e. slices can either exploit the Floodlight REST interface or can be built on top of other SDN frameworks. The Pyretic interpreter is also being integrated in the framework in order to allow experimenters to take advantage of this composable programming language.

The *EmPOWER* framework builds on a light virtual AP (LVAP) abstraction [9] which decouples association/authentication from the physical connection between clients and AP. With LVAPs every client that tries to associate to the WLAN receives a unique BSSID, i.e. every client is given the illusion of having a dedicated AP. Similarly, each physical AP hosts an LVAP for each connected client.

Therefore, migrating an LVAP between two physical APs, effectively results in client handover without requiring any re-association and re-authentication. The agent running within each APs is implemented using the Click Modular Router [10].

Finally, Energino[1] is an Arduino add–on, which allows measuring the energy consumption of a device. The measurement circuit is composed of a voltage sensor (based on a voltage divider), and a current sensor (based on the Hall effect). The powering off is done using a mechanical relay. The maximum sampling rate for measurements is about 10.000 samples/s. Voltage and current measurements are periodically sent to the Energy Manager for statistical purposes. Fluctuations in the values read from the analog inputs are filtered out by continuously polling the voltage and the current sensors between update periods and by dispatching the average values. For example, if the sampling period is set to 1s, both the voltage and the current readings will be the average of ≈ 5000 samples. Finally, Energino acts also as chassis manager allowing the testbed administrator to power on/off any node in the network using an HTTP RESTful interface.

While, Energino allows the experimenter to measure the overall power consumption of the AP, it does not provide any info about the actual impact in terms of energy consumption of a slice on the testbed. In order to address this challenge we extended to acts as virtual power meter allowing the experimenter to gain insight into the energy efficiency of his/her network application. In order to do so, a set of power consumption models already developed by the proposers [11], [7] have been embedded into the control framework allowing it to isolate the actual contribution of each slice to the overall consumption of each AP. Energy consumption model take as input measurable network statistics, such as packet transmitted/received over a certain interface, CPU usage, etc. Such statistics are fed to a centralized entity which is in charge of estimating the energy consumption on a per–slice basis. Results are then made available to the experimenter over the controller north–bound interface.

The system exploits a "logically centralized" architecture in order to provide experimenters and network applications developers with a set of powerful programming abstractions to control the behavior of the network. Applications can, for example, register events associated with the actual network conditions and receive updates when such conditions change, e.g. a client moving away from an AP and closer to another. Such primitives can be used to devise and implement novel resource allocation and/or mobility management schemes without having to deal with all the WiFi–dependent implementation details, such as directly handling the IEEE 802.11 state machine or devising workarounds to the limitations of the IEEE 802.11 standard that do not allow the infrastructure to control clients' handovers. Moreover, the availability of a real–time energy monitoring platform will provide experimenters with empirical evidence about the energy consumption performances of their solutions.

[1]Online resources available at: http://www.energino-project.org/

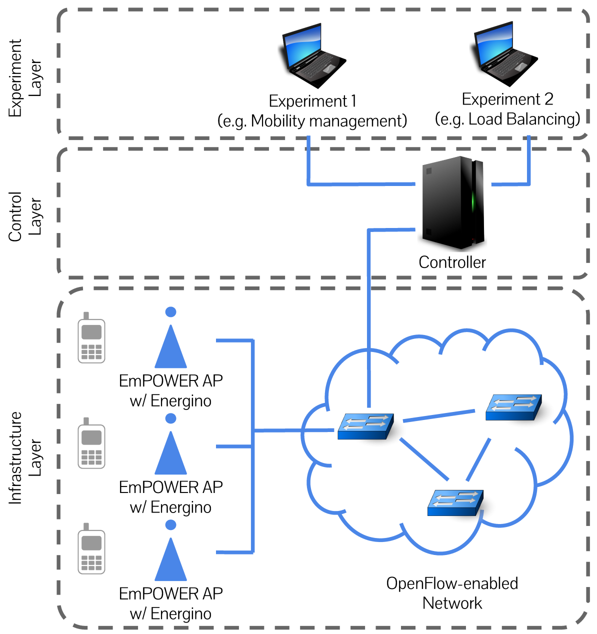

Fig. 2: The *EmPOWER* network architecture..

B. Network Architecture

The *EmPOWER* testbed architecture is sketched in Fig. 2. Each programmable Access Point is equipped with two Ethernet ports. One of them is connected to the control and management network. This allows experimenters collect network statistics and to perform administrative tasks without affecting the actual user traffic that flows trough the second Ethernet interfaces. VLANs are used at the switch in order to keep control and data traffic separated. The *EmPOWER* testbed is currently equipped with the following devices:

- 30 programmable APs based on PCEngines ALIX 2D2 (500MHz x86 CPU, 256MB of RAM) platform and equipped with two Mikrotik R52Hn IEEE 802.11 interfaces (a/b/g/n). The AP exploits OpenWRT 12.09 as operating system. Each AP runs and instance of OpenVSwitch version 1.9 together with an instance of the Click Modular Router.
- 30 Energino power meters. Each Energino is monitoring the power consumption of the AP it is attached to with a sampling period as low as 100 usec and with a resolution of 10mW. Statistics are exported in a format compatible with IoT platforms such as Xively. A REST interface for integration with additional monitoring and management systems is available. Energino acts also as "chassis manager" allowing the testbed manager to power on/off APs remotely.
- 2 Pronto 3295 switches supporting the OpenFlow version 1.0 protocol with 48 Fast Ethernet interfaces.

Each node is equipped with two Ethernet ports. One of them is connected to the control network allowing the controller to collect statistics without affecting the experiment. The second interface is connected to the OpenFlow switch and is used for running the actual experiment's traffic. Finally, another network collects the energy consumption statistics generated by the Energino devices. It is worth noticing that, unlike other WiFi testbeds, *EmPOWER* does not allow the experimenter to upload a custom OS on each AP but rather provides a set of APIs trough which the experimenter can control the behavior of the AP from a centralized controller.

The server runs the latest available software for Floodlight and FlowVisor. It is worth stressing that in the *EmPOWER* architecture new services and algorithms are deployed in the form of Network Applications on top of the Floodlight controller and exploiting its native REST interface. Additional interpreters, such as Pyrethic, are being ported to the platform. Each application is logically isolated from the others and has complete control over its slice, however physical level parameters such as the operating frequency for the hotspot are not. Nevertheless the application can control parameters such as Modulation and Coding Scheme and Transmission Power on a per–frame basis (if required by the experiment).

IV. Energy Programmable WiFi Networks

In this section we shall describe in details a particular use case that make full use of the features made available from the *EmPOWER* testbed starting from the SDN framework for WiFi to the Energino energy monitoring and management toolkit.

A recent report from CEET (Center for Energy Efficient Telecommunications, University of Sydney) stated that by 2015 the wireless access infrastructure will account for 90% of the entire energy footprint of the Wireless Cloud domain, which includes also datacenters and distribution networks [12]. WiFi hotspots are increasingly deployed to relieve cellular networks from the burden generated by data-hungry mobile applications. Such deployments generally cater for the worst case scenario, which leads to a sub-optimal usage of resources when little or no traffic is present. Real improvements in this context can only be delivered with true programmability of network functionalities which in time will allow better resource management, seamless handover between different technologies, and always best connected services. Recently, energy efficiency has also emerged as one of the evaluation metric for new networking solutions.

Hence, it is necessary to gracefully adjust the network to the current demand, improving both energy consumption and traffic pollution. Using the *EmPOWER* testbed, a researcher can test novel energy aware mobility management schemes over a realistic infrastructure. In this regard the *EmPOWER* testbed provides support both in the sensing part allowing the experimenter to subscribe a series of event such as RSSI of a client at one or more APs, energy consumption at device and network level (per–slice). The facility provides the

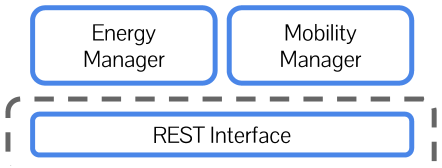

Fig. 3: The *EmPOWER* system architecture particularized for energy programmable WiFi networks use case.

experimenter with set of APIs to both query the actual state of the network and to implements handover policies.

This use case, which has already been demonstrated by the authors in [8], aims at demonstrating that *EmPOWER* can exploited to implement real–time energy consumption monitoring and management solutions aiming at reducing the actual energy consumption of WiFi infrastructures. The argument here is that, in WiFi networks,the extent of energy savings is limited by the actual client distribution (i.e., even if a single client device is attached to an AP, then the AP needs to stay on). This limitation can be traced back to the IEEE 802.11 standard that places all the (re)association initiation to the clients. However, the *EmPOWER* testbed allows the controller to dynamically handover WiFi clients between APs and to selectively shutdown the part of the network that is not deemed necessary.

Figure 3 sketches this use case implementation as two separated network applications running on top of *EmPOWER*. The system exploits a joint mobility and energy management solution. In particular the Energy Manager is responsible of energy management in the network. The decisions that lead to client handovers are handled by the Mobility Manager.

The reference network model for this use case is sketched in Fig. 4. APs are partitioned into clusters with a single Master (represented in blue) and multiple Slaves (represented in either gray or red). Masters are manually chosen at deployment time to provide full coverage and must remain always active. Slaves are deployed for providing additional capacity, and can be selectively turned on/off by the Energy Manager.

In this experiment, APs can support multiple operating modes. Possible events and corresponding transitions between modes are implemented as a finite state machine (FSM) by the Energy Manager. For this use case, we focus on two main operating modes. In the *Online* mode, an AP and all its wireless interfaces are on. In the *Offline* mode, the entire AP is turned off and only the Energino is powered. It is worth noticing that, due to the shared nature of the infrastructure, experimenter are not allowed to actually turn APs on/off. Nevertheless APs can be put in a *virtual* power down mode where no traffic is sent to the slice controller and where the power meter reports a null energy consumption.

We define $W_n \in \mathbb{N}^+$ as the number of clients that must be present in the AP n's cluster so that the AP must remain active. Based on the FSM, a *Slave* AP n belonging to a cluster

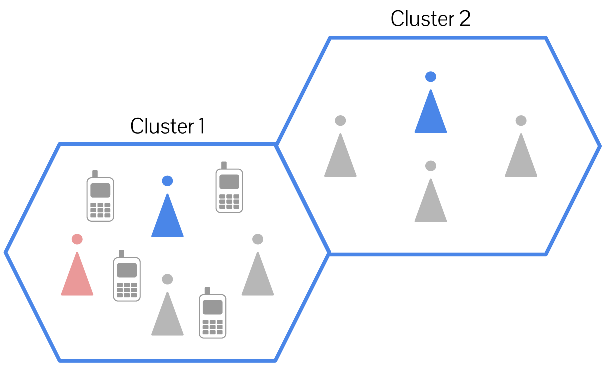

Fig. 4: Reference network model for the Energy Programmable WiFi Network use case. A minimum set of APs (Masters, in blue) providing full coverage must remain always on, while the remaining APs (Slaves, in red or gray) are at disposal of the Energy Manager.

with less than W_n clients and that has been inactive for at least T_{idle} seconds is transitioned to the *Offline* state. Here, inactive means that no LVAPs is hosted by the AP, i.e. no client is connected to the AP. If there are more than W_n clients in the cluster and if the AP has been offline for at least $T_{offline}$ seconds then the AP is brought back to *Online* mode. Notice that, W_n is statically defined for each AP at deployment time and that $W_n = 0$ only for *Master* APs.

This FSM provides a simple example, but it can be extended to support other operating modes according to the APs' capabilities, e.g., single or dual band, support for HT–rates. For example, if an AP has two interfaces, one can be tuned on the 2.4 GHz band and the other tuned on the 5 GHz band. Different operating modes can be created by turning on and off different the interfaces depending on, for instance, the existence of clients supporting the 5GHz band in the cluster.

Clients joining the network are handed over by the Mobility Manager to the AP that provides the best performance in terms of Signal–to–noise ratio (SNR). However, in order to trade–off performance with energy consumption, the Mobility Manager is allowed to handover clients to APs with lower SNR if their W_n is smaller. The rationale is that, by consolidating clients around APs with a small W_n, the Energy Manager will be allowed to turn off APs with bigger W_n. More precisely, if $S(n)$ is the SNR between the client and the AP n, N is the number of clients in the cluster, and $0 \leq \delta \leq 1$ is a tuning parameter specifying how much performance degradation are we willing to accept w.r.t. the best SNR $\hat{S}$, we define the optimal AP $\hat{n}$ as follows:

$$\hat{n} = \underset{n \in \psi}{argmin}(W_n), \quad \psi = \{n \in V | W_n \leq N, \; S(n) \geq \delta \cdot \hat{S}\}$$

where we assumed that V is the set of $|V|$ APs in a cluster (including the *Master* AP. Notice that, since $W_n = 0$ only for *Master* APs, the Mobility Manager will always try to handover clients to a cluster's *Master* AP if its SNR is acceptable.

Notice that albeit simplistic in nature, this use case shows the potential of the *EmPOWER* framework as practical platform for research and experimentation in the Wireless SDN domain. The actual implementation of the described energy management solution consists in $\approx$ 120 lines of java code and could be simplified even further by introducing more sophisticate domain–specific programming languages.

V. Conclusions and Future Work

The paper articulated the design of a novel open testbed aimed at offering an experimental facility for furthering SDN&NFV research and experimentation. Implementing an effective SDN platform supporting advanced virtualization concepts and running on top of commodity WiFi devices is a significant challenge which we address in this paper with the *EmPOWER* experimental testbed. The *EmPOWER* framework builds on top an SDN framework for WiFi networks combining OpenFlow with an open energy consumption monitoring and management toolkit. The facility is currently being extended to include programmable wireless base stations (LTE eNodeBs) to deploy and test heterogeneous scenarios and to experiment with programmable cellular networks.

References

[1] A. Voellmy, H. Kim, and N. Feamster, "Procera: a language for high-level reactive network control," in *Proceedings of the first workshop on Hot topics in software defined networks*. ACM, 2012, pp. 43–48.

[2] N. Foster, A. Guha, M. Reitblatt, A. Story, M. J. Freedman, N. P. Katta, C. Monsanto, J. Reich, J. Rexford, C. Schlesinger *et al.*, "Languages for software-defined networks," *Communications Magazine, IEEE*, vol. 51, no. 2, pp. 128–134, 2013.

[3] N. Foster, R. Harrison, M. J. Freedman, C. Monsanto, J. Rexford, A. Story, and D. Walker, "Frenetic: A network programming language," *ACM SIGPLAN Notices*, vol. 46, no. 9, pp. 279–291, 2011.

[4] C. Monsanto, J. Reich, N. Foster, J. Rexford, and D. Walker, "Composing software defined networks," in *Proc. of USENIX NSDI*, 2013.

[5] T. L. Hinrichs, N. S. Gude, M. Casado, J. C. Mitchell, and S. Shenker, "Practical declarative network management," in *Proceedings of the 1st ACM workshop on Research on enterprise networking*. ACM, 2009, pp. 1–10.

[6] A. Voellmy and P. Hudak, "Nettle: Taking the sting out of programming network routers," in *Practical Aspects of Declarative Languages*. Springer, 2011, pp. 235–249.

[7] K. M. Gomez and Roberto Riggio and Tinku Rasheed and Daniele Miorandi and Fabrizio Granelli, "Energino: An hardware and software solution for energy consumption monitoring," in *WinMee*, 2012.

[8] Roberto Riggio and Cigdem Sengul and Lalith Suresh and Julius Schulz–Zander and Anja Feldmann, "Thor: Energy programmable wifi networks," in *Proc. of IEEE INFOCOM*, 2013.

[9] L. Suresh et al., "Towards programmable enterprise wlans with odin," in *Proc. of ACM HotSDN*, 2012.

[10] E. Kohler, R. Morris, B. Chen, J. Jannotti, and M. F. Kaashoek, "The click modular router," *ACM Transactions on Computer Systems (TOCS)*, vol. 18, no. 3, pp. 263–297, 2000.

[11] Roberto Riggio and Douglas J.Leith, "A measurement-based model of energy consumption in femtocells," in *IEEE Wireless Days*, 2012.

[12] CEET, "The Power of Wireless Cloud: An analysis of the energy consumption of wireless cloud," Centre for Energy-Efficient Telecommunications, Tech. Rep., April 2013.