Thor: Energy Programmable WiFi Networks

Roberto Riggio†, Cigdem Sengul‡, Lalith Suresh§, Julius Schulz–Zander§, and Anja Feldmann§
†CREATE-NET, Trento, Italy; *Email: roberto.riggio@create-net.org*
‡Oxford Brookes University, London, UK; *Email: csengul@brookes.ac.uk*
§Telekom Innovation Laboratories, Berlin, Germany; *Email: name.surname@telekom.de*

***Abstract*—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 over–provisioning of resources eventually resulting in a significant energy waste when little or no traffic is present. In this demo, we present an integrated energy and mobility management solution for WiFi networks. The demo combines *Energino*, an open real–time energy consumption monitoring toolkit with *Odin* a software defined networking framework for WLANs. This integration enables energy programmable WiFi networks, allowing to cluster clients around access points in an energy-efficient and performant manner.**

I. Introduction

Current wireless networks account for energy efficiency only on end–user side, where battery life is one of the most important considerations. Not much attention has been paid to the energy consumption of wireless infrastructures as they are typically attached to the power grid. This picture is currently changing due to the massive adoption of data–hungry mobile devices, such as smartphones and tablets, which is mandating the deployment of dense WiFi hotspots, to offload data traffic from cellular networks and also, provide users a more energy-efficient alternative. However, these networks do not always serve peak demand [1]. Hence, it is necessary to gracefully adjust the network to the current demand, improving both energy consumption and traffic pollution [2].

In our earlier work, we showed that energy consumption of WiFi infrastructure networks could be reduced using a real–time energy consumption monitoring and control solution [3], [4], *Energino*, which is used to turn off Access Points (AP) with no clients attached. However, 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. In this demo, we address this issue by a joint mobility and energy management solution, which combines *Energino* with *Odin* [5], a Software Defined Networking (SDN) Framework for enterprise WLANs. *Odin* builds on a light virtual AP abstraction, which allows the infrastructure to move clients to different APs, allowing *Energino* to selectively turn off parts of the network that are currently not needed. To the best of our knowledge, there are no similar solutions available from either academia or commercial vendors.

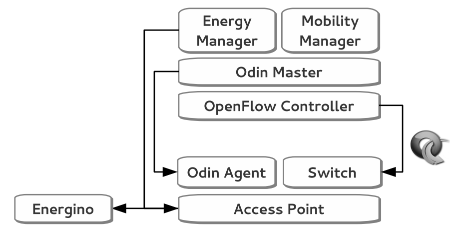

Fig. 1: System architecture. The Odin Master speaks OpenFlow to the switches and uses a custom protocol to communicate with the Odin Agents. The *Energy Manager* uses a custom protocol to communicate with the APs and with *Energino*.

II. System Architecture

This section details the system's main components: the *Energino* devices, the *Odin* SDN framework, and the *Energy & Mobility Managers*. The architecture is sketched in Fig. 1. The *Odin* framework allows deploying new services as *Network Apps*. The *Energy Manager* is responsible of energy management in the network. The decisions that lead to client handovers are handled by the *Mobility Manager*.

A. Energino

Energino is an Arduino add–on, which allows measuring the energy consumption of a wireless device, as well as powering it off. 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 $10kHz$. 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.

B. Odin

Odin's architecture 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.

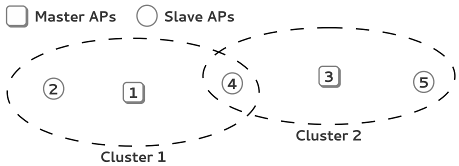

Fig. 2: Network architecture. A minimum set of APs (Masters) providing full coverage must remain always on, while the remaining APs (Slaves) are at disposal of the *Energy Manager*.

A central primitive of *Odin* is the Light Virtual Access Point (LVAP). LVAPs decouple 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.

C. Joint Mobility and Energy Management

Our reference network model is sketched in Fig. 2. APs are partitioned into clusters with a single *Master* (nodes 1 and 3) and multiple *Slaves* (nodes 2, 4, and 5). *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*. *Slaves* can belong to multiple clusters, e.g., node 4 in Fig. 2, due to overlaps in coverage between clusters.

Operating Modes. In our design 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 demo, 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.

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 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.

Client handover. 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.

III. Demo Scenario

The demo setup will consist of five APs and five *Energino* connected via a switch to a laptop acting as network controller. The APs are embedded boards running OpenWRT and equipped with two Ubiquiti SR71-A 802.11n interfaces (Atheros chipset). Each AP runs the *Odin Agent* while the laptop runs the *Odin Master* and the *Energy* and *Mobility Managers*. We assume that all APs form a single cluster and that the single *Master* can provide sufficient coverage. The demo will display the following aspects of our system:

- We will have our own 5 clients consisting of tablets and smartphones. Attendees will also be able to access the Internet trough our network using their own devices.
- Using a web–based dashboard, we will display for each AP: its operating mode, its instantaneous power consumption and the connected clients.
- Starting from a configuration where only the *Master* AP is on, we will demonstrate how the system selectively activates additional APs as the number of clients increases and how these clients as shuffled around the network.
- Starting from a configuration where all APs are active, we will demonstrate how, as clients leave the network, the remaining ones are handed over to the *Master* AP.
- During the demo, a wireless client will run a video stream demonstraing *Odin*'s transparent handover capabilities.

References

[1] M. Afanasyev et al., "Usage patterns in an urban wifi network," *IEEE/ACM Trans. Netw.*, vol. 18, no. 5, pp. 1359–1372, Oct. 2010.

[2] J. Manweiler and R. Roy Choudhury, "Avoiding the rush hours: Wifi energy management via traffic isolation," in *MobiSys '11*, pp. 253–266.

[3] R. Riggio et al., "Energino: energy saving tips for your wireless network," in *Proc. of ACM SIGCOMM*, 2012.

[4] K. M. Gomez et al., "Energino: An hardware and software solution for energy consumption monitoring," in *Proc. of WinMee*, 2012.

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