Mobile Data Offloading: An Experimental Evaluation

Matteo Pozza
University of Padua, Italy
matteo.pozza
@studenti.unipd.it

Claudio Enrico Palazzi
University of Padua, Italy
cpalazzi
@math.unipd.it

Armir Bujari
University of Padua, Italy
abujari
@math.unipd.it

ABSTRACT

Mobile data traffic is becoming a key issue, causing traffic overload problems for classic resource providers. Tackling this issue, different solutions have been proposed, communed by the goal of alleviating the burden on the infrastructure while guaranteeing service availability. In this context, swarming protocols present the advantage of addressing the problem at its source, e.g., by engaging a lower number of direct connections with the network hosted server(s), while the requesters share these same resources among them. However, these solutions are generally verified only through simulation studies. While simulations consist in a necessary first step, they can be far from the actual outcome of employing real protocols and devices. To this end, we discuss BlueFall, an application platform that allows a rapid development and testing of swarming protocols using off-the-shelf smartphones and PCs. Pursuing our goal, we provide some evidence on two swarming schemes we have implemented in our testbed, profiling different access technologies.

Categories and Subject Descriptors

C.4 [**Computer Systems Organization**]: Performance of systems—*measurement techniques*

Keywords

Experiment; Mobile; Bluetooth; Wi-Fi Direct

1. INTRODUCTION

The everywhere, anytime network access model has boosted the demand for data traffic and this trend is expected to grow even further in the near future [1, 2, 3]. This ever increasing demand poses a major burden on network providers, causing traffic overload problems resulting in a degraded network service provisioning [4]. In-network solutions to the issue have been proposed, spanning from the deployment of specialized infrastructure for data offload to intelligent software solutions [5, 6, 7]. However, evidence shows this is not sufficient and additional means are required.

MobiArch'15, September 07 2015, Paris, France
 ISBN 978-1-4503-3695-6/15/09...$15.00
DOI: http://dx.doi.org/10.1145/2795381.2795394.

We consider a general scenario with a network hosted server and a set of clients requesting a certain content from it. In this context, swarming protocols could tackle the issue at its source by shifting the burden from the network towards the client(s). This is achieved through client-side cooperation, e.g., by engaging fewer direct data connection with the network hosted server(s) while the clients share the resource progress among them. Although swarming protocols are not a radical new idea, many of the proposed solutions have been verified only through simulation studies [8, 9, 10], which consist in a necessary first step but far from the real outcome of employing real devices.

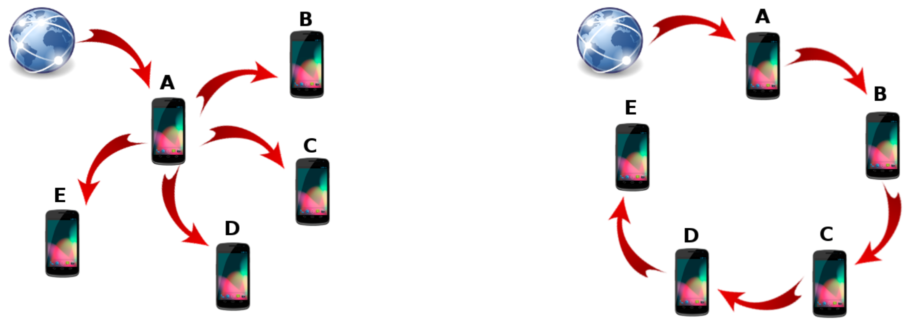
Filling this gap, we discuss BlueFall, a desktop and mobile application platform that allows for rapid development and testing of swarming protocols with real devices and protocols.

Its two main components are (i) a sharing scheme (the swarming protocol) coordinating the clients and (ii) the capability of exploiting a wireless communication technology such as Bluetooth and Wi-Fi Direct. We present and discuss results achieved through our testbed considering different configuration settings, sharing scheme and wireless access technology.

The remainder of this paper is organized as follows: in Section 2, we describe the general idea behind BlueFall and its *modus operandi*. Next, in Section 3, we discuss the application platform along with design and implementation issues that arose and the solution adopted to address them. Following, in Section 4, we present the testbed configuration along with the test trials involved in validating it. Section 5 analyzes the trials' outcome.

2. CONSIDERED SCENARIO

The general idea considered here is to have a single client downloading a certain content directly from the server, while applying a swarming approach to spread it among the other clients. This reduces the server and network burden and limits the need for Internet connectivity among interested clients. BlueFall allows to test classic swarming approaches, considering both a *star* sharing scheme (Figure 1a) and a *waterfall* sharing scheme (Figure 1b). Through the former scheme, the only client retrieving the resource from the server (A) also acts as resource provider for all the other clients (B, C, D and E). Since a single client becomes the new server for the other ones, this solution is also referred to as a *centralized* sharing scheme. In the second approach the



(a) Star (b) Waterfall

Figure 1: Classic swarming approaches

only client downloading the content from the server (A) also acts as a provider for only one other client (B), which does the same in turn (for C), and so on till the end of the "chain" (node E). Since each client acts as a resource provider for a different client, this solution is also referred to as a *distributed* sharing scheme. Clearly, the two sharing schemes here considered are just two representative examples of the many topologies we may create with our solution, including all sort for mesh configurations [11]. Our considered clients share the content through state of the art device-to-device connectivity. To this end, our BlueFall testbed supports Bluetooth and Wi-Fi Direct. Therefore, our testbed can evaluate the performance of the combination of swarming protocol-wireless technology.

3. BLUEFALL

A preliminary mobile version of the BlueFall framework has been previously introduced and discussed in [12]. In addition, in this work, we present several extensions to the framework including but not limited to a desktop variant. Moreover, we perform and discuss a comparison analysis of both variants employing an enriched set of real measurements.

The main aim of BlueFall is measuring the performance of networks based on swarming protocols; to do so, it handles two distinct flows of data, the *incoming* flow and the *outgoing* one. The first one consists in the data flow going from a certain source (i.e., the server or another client) to the considered device, while the second one is the data flow departing from the considered device and reaching one or more destinations. Several measurements could be performed on each of these flows, including the instantaneous data rate, the average data rate, the standard deviation of the data rate and the total transfer time.

These measurements could be both shown on screen and stored into XLS and XML log files.

3.1 Design Considerations

The use of a swarming protocol implies that a client should simultaneously execute three different main activities:

- receiving the data from the source or from a forwarder;
- progressively storing the data;
- forwarding the data to another client.

Clearly, the last client involved in a waterfall sharing scheme does not forward the content to anyone. Since the activities deal with the same data, a common-buffer structure is desirable. At the same time, the activities should be as decoupled as possible to collect unbiased measurements of incoming and outgoing flows. These considerations lead to the employment of the producer-consumer model; the flow taking care of the data reception can be seen as the producer, while the data forwarding and storage flow can be considered as consumers. In fact, the producer gains bytes of the content shared from the source and deposits them in the common buffer, while the consumers read the bytes in the buffer and store/forward them.

One of the most interesting aspects of BlueFall regards the combination between the waterfall scheme and Wi-Fi Direct. In a typical Wi-Fi Direct network, a device creates a Wi-Fi Direct group to which other devices can connect and through which all communications have to transit [13]. The key point is that each device joining the Wi-Fi Direct group knows only the creator of the group (the *group owner*), but not the other members. To be more precise, a device joining a group is able to gain only the IP address of the group owner, while the group owner can obtain the IP addresses of each one of the devices that have joined the group. Clearly, this may impede communication between two devices in a waterfall scheme. Even creating a Wi-Fi Direct group for each couple of devices, so as to create a multihop chain, is not a solution since the Android implementation of the Wi-Fi Direct's specifications does not allow to a single device to be engaged in more than one group. To overcome this problem, an IP-redistribution phase has been introduced; in this phase, the group owner takes care of communicating to each device the IP address of the device that will be the source of its incoming flow of data. Three steps are required:

- each device communicates with the group owner that stores an order number along with the corresponding IP address;
- once the table is completed, the group owner sends to each device the IP address of the device that has the immediately preceding order number, which can hence be contacted to establish a connection;
- once all the devices have obtained the target IP address, the waterfall transmission can be built up.

The cost in time of these steps is proportional to the number of devices involved in the network, but since the exchanged control data is a small amount (integers and short strings) it cannot be considered onerous.

3.2 Implementation Details

BlueFall has been developed for both mobile and desktop platforms. The two versions are quite similar since the key aspects (threads separation, producer-consumer model, etc.) are common. The mobile version is an application which runs on Android 4.0 (API level 14) or later versions. Indeed, previous versions of Android do not support the Wi-Fi Direct protocol; however, the Android Developer wiki reports that this includes more than 90% of active Android devices. Another aspect, is the lack of native support in the Android framework for XLS files; to solve this, we adopted an external library[1].

The desktop version is a C/C++ application developed on Linux Ubuntu 14.04 LTS. The library used to deal with Bluetooth communication is *Bluez* and is available in the official Ubuntu repository as *libbluetooth-dev*; it is a powerful C library which allows not only the communication but also the management of Bluetooth devices. The main inconvenience of the library is its poor documentation.

To employ Wi-Fi Direct, BlueFall uses the *wpa_supplicant*, a backend software for communication control that supports the main wireless encryption standards. The software needs to be specifically tailored to support Wi-Fi Direct via specific drivers[2]. This software package allows establishing secure P2P connections only among couples of devices, hence a way of looking to the set of devices as a cohesive group is needed. To this end, we rely on two simple software packages: *udhcpd* and *udhcpc*. The first one is executed by the group owner, which acts as DHCP server, while the second one is executed by each device that has joined the network.

One of the goals of BlueFall is to compare the performance of different versions of the same protocol: for example, it could be interesting to test the improvements of Bluetooth version 4.0 compared with the version 3.0 given the same network configuration. While in a mobile scenario we are forced to change our smartphones with other phones supporting the latest version of the protocol, in the desktop scenario we only need to change the peripherals. On the other hand, if we want to test the behaviour of a configuration adding mobility, the smartphone version seems more practical.

4. USING BLUEFALL TO GATHER DATA

To test BlueFall, we have considered all the combinations among the following configuration parameters:

- Total number of devices involved: {3, 5}.
- Sharing schemes: {star, waterfall}.
- Wireless technologies: {Bluetooth v3.0, Wi-Fi Direct}.
- Platforms: {mobile, desktop}.

[1]Java Excel API - A Java API providing access to Excel spreadsheets: `http://jexcelapi.sourceforge.net/`

[2]The guide to build wpa_supplicant in order to support Wi-Fi Direct: `http://wireless.kernel.org/en/developers/p2p/howto`.

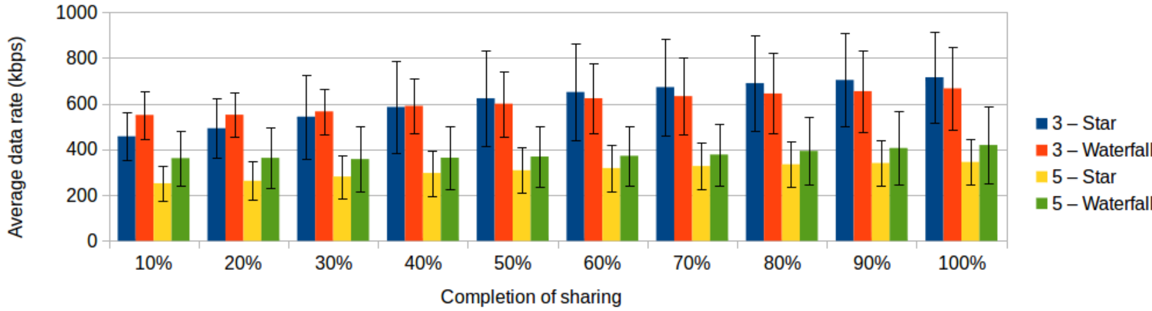
For each of the 16 configurations we have run (and averaged the outcome of) 20 tests where a *.deb* file of about 10 MB had to be downloaded from a server and then distributed on all devices. Each test involves 10 samples gathered at regular intervals during the download of the content by each device. So the first sample is taken when 10% of the resource has been received, the second one when 20% of the resource has been received, and so on. Each sample includes:

- the instantaneous data rate;
- the total average data rate;
- the standard deviation of the data rate;
- the time elapsed from the beginning of the sharing.

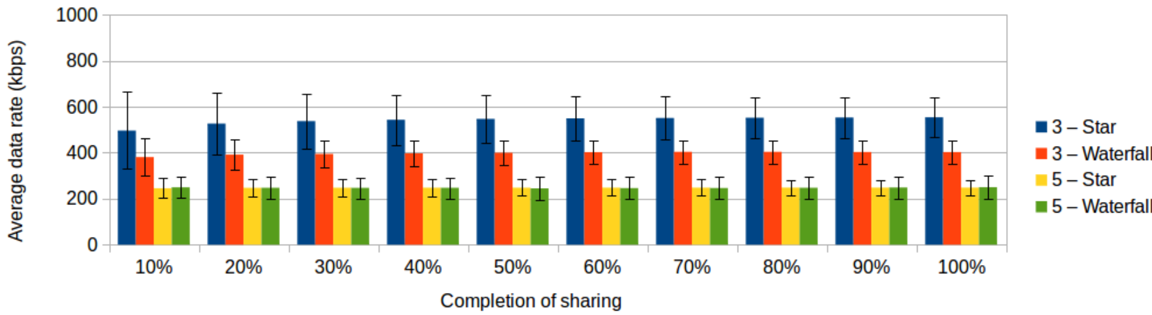
The mobile version of BlueFall has been tested with Samsung Galaxy Nexus GT-i9250 devices. Instead, the desktop version has been run on a mixed set of desktops and laptops, each equipped with a *(i)* Trust Bluetooth 3.0 USB adapter and a *(ii)* Netgear Wireless-N 300 USB adapter.

We have tested representative energetic profiles corresponding to different energy consumption sources. To distinguish among them, we provide a description and an acronym, where the first letter identifies the considered node (A for node A; X for intermediate nodes B, C, and D; Z for the last node receiving the file), the second one represents Bluetooth (B) or Wi-Fi Direct (D), the third one if present discriminates among star (S) and waterfall (F) topologies, and the last one is a number that if present represents the number of nodes in the configuration (3 or 5).

- **Profile ABS3.** Device A in Figure 1a, using regular Wi-Fi to download the file from the server and Bluetooth to transmit it to B and C, whereas D and E are not part of the experiment.
- **Profile ADS3.** Device A in Figure 1a, using regular Wi-Fi to download the file from the server and Wi-Fi Direct to transmit it to B and C, whereas D and E are not part of the experiment.
- **Profile ABS5.** Device A in Figure 1a, using regular Wi-Fi to download the file from the server and Bluetooth to transmit it to B, C, D and E.
- **Profile ADS5.** Device A in Figure 1a, using regular Wi-Fi to download the file from the server and Wi-Fi Direct to transmit it to B, C, D and E.
- **Profile ABF.** Device A in Figure 1b, using regular Wi-Fi to download the file and Bluetooth to transmit it to B.
- **Profile ADF.** Device A in Figure 1b, using regular Wi-Fi to download the file from the server and Wi-Fi Direct to transmit it to B.
- **Profile XBF.** Devices B, C and D in Figure 1b, using Bluetooth to receive and transmit the file.
- **Profile XDF.** Devices B, C and D in Figure 1b, using Wi-Fi Direct to receive and transmit the file.
- **Profile ZB.** Devices B, C, D and E of Figure 1a and device E of Figure 1b, which are only receiving the file via Bluetooth.



(a) Mobile



(b) Desktop

Figure 2: Average data rate trend employing Bluetooth communication

- **Profile ZD.** Devices B, C, D and E of Figure 1a and device E of Figure 1b, which are only receiving the file via Wi-Fi Direct.

To measure the battery consumption we used *PowerMonitor* [14], a tool for precise measurements on energy consumption. This tool has to be used in substitution of the battery of a device to compute the amount of energy the application is requesting.

In order to better understand the measured energy consumption, a **baseline profile** value is needed. Our baseline configuration consists in the following elements:

- the brightness of the screen has been set to the maximum value;
- the background of the device has been set with a custom defined image[3];
- all the communication devices (Wi-Fi, 3G/HSDPA, Bluetooth, NFC) have been switched off;
- no application is active in background;
- the energy consumption is considered as the power consumption after 10 minutes of inactivity.

Both the average power consumption and the expected battery life have been collected and compared with the results obtained from the analysis of the energetic profiles.

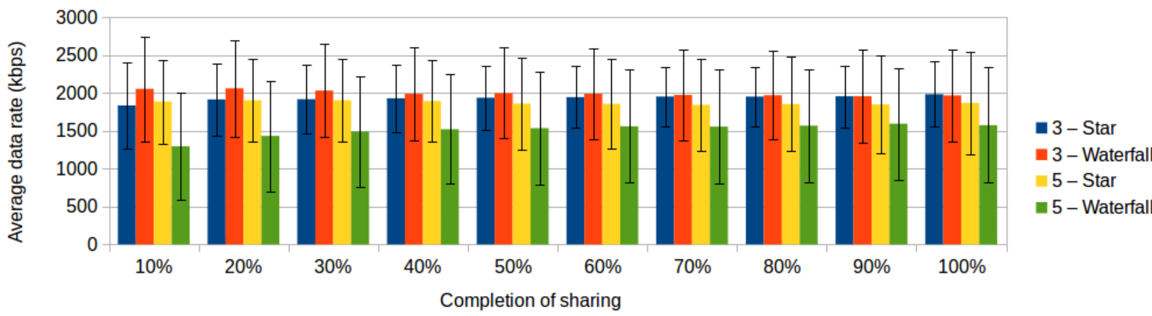
[3]Solid color background, Hue: 0, Saturation: 5, Value: 33, Red: 85, Green: 81, Blue: 81, HTML code: 555151.

5. RESULTS

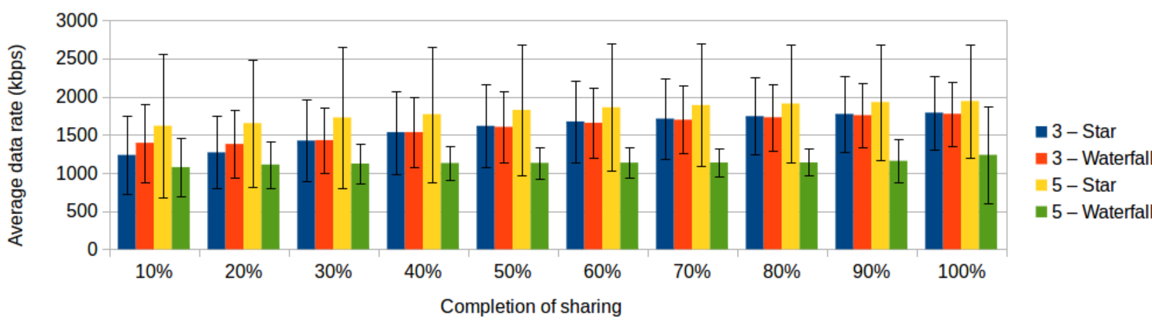
We discuss here data rate and time measurements regarding incoming flows at the receivers. Clearly, when the last node in the waterfall topology has received the whole file then the file has been spread to all nodes. We present the results in the charts dividing them by transmission technology, Bluetooth and Wi-Fi Direct.

To this end, Figure 2 shows the results achieved using Bluetooth technology; in particular, Figure 2a concerns mobile tests, while Figure 2b is related to the desktop ones. In both cases we consider both Star and Waterfall topologies with 3 or 5 nodes. Average values and standard deviation bars are reported in the charts. Considering the configurations involving three devices, the star topology records higher data rate values than the waterfall one, especially when employing desktops. Instead, with 5 devices the two schemes behave similarly, even if the waterfall scheme seems slightly better in the mobile configuration. The enhancement gained using waterfall sharing scheme reflects the fact that the star sharing scheme seems quite inadequate when the number of clients starts to increase as it would recreate the same bottleneck problem that made us discussing about offloading the burden of distributing a content from a single server to many clients.

It is interesting to notice that although our effort to utilize the same (or very similar) communication technology, mobile Bluetooth seems to perform better than its desktop counterpart does. Furthermore, mobile configurations keep increasing their data rate values while desktop ones tend to have a very stable behaviour. This diversity has to be kept in mind when designing a new testbed as the outcome may



(a) Mobile



(b) Desktop

Figure 3: Average data rate trend employing Wi-Fi Direct communication

present unexpected differences just by employing different devices and Bluetooth interfaces, although belonging to the same standard version (in our experiments Bluetooth v3.0 was used).

Results obtained testing Wi-Fi Direct are reported in Figure 3. Even in this case we have better performance in the mobile environment than in the desktop one, but the trends are significantly different from Bluetooth results. In fact, while mobile configurations do not show any significant improvement during the sharing, desktop configurations record a slightly growing average data rate for all considered configurations but the 5-waterfall one.

It is very interesting to point out that the star topology is able to not lose data rate when passing from a total of 3 nodes to 5 nodes. This is because Wi-Fi Direct relies on 802.11 MAC, which uses CSMA/CA to regulate accesses to the medium. With a star sharing scheme, it is easier to avoid interferences among devices' transmissions as there is only a single device (node A in Figure 1a) transmitting data to all the others. Instead, in a waterfall configuration each device tries to communicate with others forwarding data and this could lead to frequent collisions and hence retransmissions. Even worse, this represents the favourable condition for the channel capture phenomenon [15], which could attenuate significantly the pipelining implied by the waterfall sharing scheme.

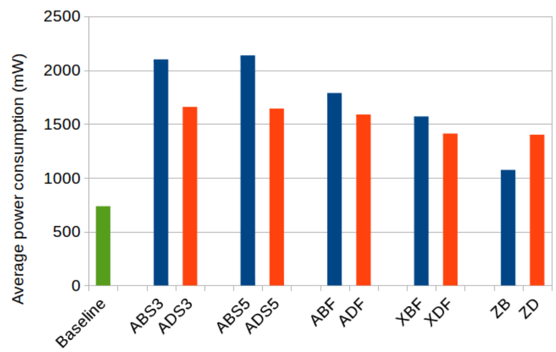
Bluetooth uses time division multiplexing in order to avoid interferences, so each device knows exactly when it has to transmit, and this stands also when the network is a *scatternet*, which matches our employment of the waterfall sharing scheme. This means that in a waterfall configuration there can be actual pipelining, whereas in a star configuration the client acting as the server simply iterates over the connected clients and sends data in a precise order.

The difference in performance between the mobile and desktop version are to some extent contrary to popular belief. This disagreement between the two platforms is to be attributed to the USB peripheral equipment (ver. 1.0) in the desktops available to us which represent a bottleneck for the wireless adapter(s). However, the relative performance trend between the sharing schemes and wireless technologies should not be affected by this.

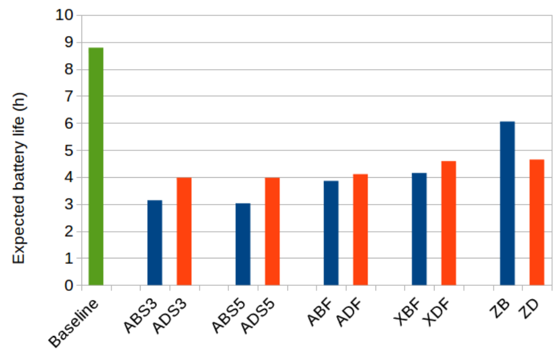
The charts in Figure 4 show the energy measurements. The leftmost column stands for the baseline profile, while the blue and the orange columns match the profiles using Bluetooth and Wi-Fi Direct, respectively. The descriptions of the energetic profiles can be found in Section 4.

In Figure 4a we can note that Bluetooth profiles (blue columns) have very dis-homogeneous values. Clearly, the energy consumption decreases when considering nodes with less destinations to serve. The small difference between profile ABF and profile XBF can be imputed to the download of the resource from the Internet, while the divergence between profile XBF and profile ZB corresponds to the change from receiving and sending data to only receiving them.

As with Bluetooth, even with Wi-Fi Direct it is possible to detect the small power consumption due to the content download from the Internet by just looking at the difference between the values of profile ADF and profile XDF. Instead, the difference between power consumption in both receiving and sending data and in only receiving data (profile XDF and profile ZD, respectively) is almost undetectable. Unexpectedly, Wi-Fi Direct profiles show lower battery consumptions than Bluetooth ones in almost all comparisons



(a) Average power consumption



(b) Expected battery lifetime

Figure 4: Energetic profile result for Bluetooth (in blue) and Wi-Fi Direct (red) communication technologies

except between profile ZB and profile ZD. This is due to the fact that Bluetooth is a technology designed to be a cable replacement and conceived for short and relatively rare transmissions of a few bytes, while Wi-Fi Direct inherits the Wi-Fi standard properties devoted to big and relatively fast data transmissions. Therefore, BlueFall uses Bluetooth not in the classic, designed-for way but rather intensively and for long periods, thus generating chart results.

The chart in Figure 4b reflects the considerations above simply changing the point of view; the expected battery life of a device is the other face of its average power consumption, as the latter grows, the former decreases. It is yet of practical interest to have an estimation of how many hours a device may last before running out of energy when employing BlueFall. We have considered declared storage capacity of off-the-shelf batteries and, as we can see, depending on the considered profile, a whole battery charge will last from 3 to 6 hours.

6. CONCLUSION

Swarming protocols have been proposed as a viable solution to offload the Internet infrastructure from traffic that receivers can share. However, proposed solutions have been generally verified only through simulations. In this paper, we presented BlueFall, an application platform that allows a rapid development and testing of swarming protocols using off-the-shelf smartphones or PCs. Through BlueFall, we have also been able to discuss results from experiments involving different topologies and technologies.

7. REFERENCES

[1] Cisco VNI - Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2013–2018.

[2] Study on Architecture Enhancements to Support Proximity Services. *TR Standard 23.703*. March 2014.

[3] C. E. Palazzi, A. Bujari, G. Marfia, M. Roccetti, An Overview of Opportunistic Ad Hoc Communication in Urban Scenario. In Proc. of the *13th Annual Mediterranean Ad Hoc Networking Workshop (MEDHOC-NET 2014)*. June 2014.

[4] C. E. Palazzi, S. Ferretti, M. Roccetti, G. Pau, M. Gerla, What's in that Magic Box? The Home Entertainment Center's Special Protocol Potion, Revealed. *IEEE Transactions on Consumer Electronics*, 52(4): 1280-1288, November 2006.

[5] D. Calin, H. Claussen, H. Uzunalioglu, On femto deployment architectures and macrocell offloading benefits in joint macro-femto deployments. *IEEE Communications Magazine*, 48(1):26-32, January 2010.

[6] L. Keller, A. Le, B. Cici, H. Seferoglu, Ch. Fragouli, A. Markopoulou, MicroCast: Cooperative Video Streaming on Smartphones. In Proc. of *International Conference on Mobile Systems, Applications, and Services*. 2012.

[7] S. Hoteit, S. Secci, G. Pujolle, A. Wolisz, C. Ziemlicki, Z. Smoreda, Mobile Data Traffic Offloading over Passpoint Hotspots. *Computer Networks*, 84(0):76-93, 2015.

[8] A. Nandan, S. Das, G. Pau, M.Y. Sanadidi, M. Gerla, Cooperative Downloading in Vehicular Ad Hoc Wireless Networks. In Proc. of *IEEE/IFIP International Conference on Wireless On demand Network Systems and Services*. January 2005.

[9] U. Lee, J.S. Park, J. Yeh, G. Pau, M. Gerla, Code Torrent: Content Distribution Using Network Coding in VANET. In Proc. of *International Workshop on Decentralized Resource Sharing in Mobile Computing and Networking*. September 2006.

[10] K. Yasumoto, Y. Takamatsu, W. Sun, M. Ito, Energy-Efficient Cooperative Download for Smartphone Users through Contact Time Estimation. In Proc. of *IEEE Intl. Conf. on Wireless and Mobile Computing, Networking and Communications*. October 2012.

[11] A. Hamidian, C.E. Palazzi, T.Y. Chong, J.M. Navarro, U. Korner, M. Gerla, Deployment and Evaluation of a Wireless Mesh Network. in Proc. of *IARIA/IEEE International Conference on Advances in Mesh Networks (MESH 2009)*. June 2009.

[12] M. Pozza, C.E. Palazzi, A. Bujari, BlueFall: Testing Swarming Protocols through Mobile Phones. in Proc. of *International Conference on Computing, Networking and Communications (ICNC)*. February 2015.

[13] C. Casetti, C.F. Chiasserini, L. Curto Pelle, C. Del Valle, Y. Duan, P. Giaccone, Content-centric Routing in Wi-Fi Direct Multi-group Networks. In *arXiv:1412.0880v1 [cs.NI]*. December 2014.

[14] Monsoon Solutions Inc - PowerMonitor: `https://www.msoon.com/LabEquipment/PowerMonitor/`.

[15] S. Kopparty, S.V. Krishnamurthy, M. Faloutsos, S.K. Tripathi, Split TCP for mobile ad hoc networks. In Proc. of *IEEE Global Telecommunications Conference*. November 2002.