Towards Privacy-Preserving Data Dissemination in Crowd-Sensing Middleware Platform

Romain Sommerard
Univ. Lille / Inria
France
romain.sommerard@inria.fr

Romain Rouvoy
Univ. Lille / Inria
France
romain.rouvoy@inria.fr

ABSTRACT

Crowd-sensing, also known as *mobile crowdsourcing*, is a growing research topic, which consists in engaging end users in the process of gathering physical measurements in the field. While the democratization of such middleware platforms opens the venue for the observation of phenomenon at scale, it may also raise key issues about the privacy of end users involved in the gathering process.

In this paper, we therefore report on our preliminary steps towards enabling privacy-preserving data dissemination in crowd-sensing middleware platforms by adopting a decentralized dissemination strategy to fuzz the contributions of end users. To assess such a strategy, we also report in this paper on our initiative to deploy a large-scale middleware platform to emulate and control a crowd of devices. In particular, we claim that an assessment by simulation is not acceptable for such a kind of contribution and that the mobile engineering community requires a more realistic evaluation testbed.

Keywords

Mobile computing, Privacy, Simulator, WiFi-Direct

1. INTRODUCTION

Crowd-sensing platforms are new way to collect and aggregate knowledge from the huge number of devices which populate our world. These platforms are betting to exploit and share the information by the smartphone we hold right in our pocket.

APISENSE [9] is a crowd-sensing platform which is going to change the way we use our smartphones. This platform engages end-users to contribute to experiments that help scientists to improve their dataset and their knowledge in the context of their research. For instance, a specific campaign can be deployed by APISENSE to gather measurements about air quality in cities. This, can be insightful to better understand pollution issues and improve urbanization. Because crowd-sensing platforms collect information in the vicinity of their users, privacy is a crucial issue because users share their data only if they trust in the platform. APISENSE therefore evolves to integrate the principles of *privacy-by-design* [16] to increase the privacy of end users, while preserving the quality of the datasets it produces.

Although APISENSE is a mature platform, it suffers from some privacy leaks. First, it adopts a server-side data anonymization technique, which is the most common used approach in companies today.This means that users upload their non-anonymized data to a remote server which anonymise it upon reception. However, users may not trust that their data are really processed by the server. Beyond that, the server hosting the data can be subject to specific attack (*e.g.*, man in the middle) and intercept raw data. Finally, it delegates the storage of collected dataset to secondary nodes, which can be hosted by third parties. Because these nodes cannot be continuously controlled, APISENSE cannot ensure that these third parties will disable the anonymization process.

To address these limitations, this paper introduces two main contributions for crowd-sensing platforms that are:

- A peer-to-peer data dissemination approach, which takes part of WiFi-Direct communications. This paper provides an application skeleton to test different data dissemination algorithm based on WiFi-Direct.
- An large-scale emulation platform, which includes a WiFi-Direct implementation for the Android OS. This platform provides a way to test crowd-sensing Android applications in context of a crowd of device. This solution trends to be as realistic as possible of the field usage of mobile devices.

The reminder of this paper is organised as follows. We first motivates the key challenges to be addressed in the context of privacy (cf. Section 2). Then, we introduce a new mobile peer sampling service as a core building block of privacy-aware data dissemination (cf. Section 3). Because the assessment of such service is not trivial, we report on our effort to build a large-scale emulation platform for Android devices (cf. Section 4). We finally discussess the related work (cf. Section 5) before concluding and exposing our perspectives (cf. Section 6).

UbiMob'16 July, 2016, Lorient, France
© 2016 ACM. ISBN 978-1-4503-2138-9.
DOI: 10.1145/1235

2. MOTIVATION

Preserving privacy in the context of crowd-sensing platforms is a key challenge to address.

In particular, the anonymization of crowdsourced datasets is a priority when it comes to increase the privacy of end-users [4]. To the best of our knowledge, data anonymization techniques are often deployed on the server side [9], [6].

This observation can be correlated with the difficulty of developing mobile apps enabling peer-to-peer mobile communications. For example, none of Android or iOS emulators support proximity wireless interactions and the only way is to test them is to use real devices. However, this method does not scale and cannot reproduce realistic events that can be encountered in real deployments.

Another limitation in the geo-privacy field is the incapacity to reproduce and compare experiments. Geo-privacy focuses its interest on the critical impact of location data on users privacy. In this domains, the evaluation builds on shared datasets, which are used by the community to test and assess their algorithms. However, these datasets are often altered for the purpose of evaluation and this modification of the input data prevents from further comparisons.

Finally, in the mobile computing community, some simulation tools give the opportunity to simulate many devices, network and others. However, these tools are not focused on crowd-sensing concerns like wireless proximity issues or the validation on a very large number of emulators running and moving at the same time.

In this paper, we address these drawbacks by providing a data dissemination approach that leverages peer-to-peer communications on mobile device to anonymize data. Our approach take advantages of the privacy-by-design principles presented by Langheinrich [16].

We also describe the design of an emulation platform, which can be used to reproduce human movement with real location data and test crowd-sensing applications in more realistic settings.

3. TOWARDS MOBILE PEER SAMPLING

We propose a data dissemination approach adopting a peer-to-peer protocol that tries to exploit the proximity of mobile user to initiate interactions. We choose a peer-to-peer approach to exchange data directly with another device without involving a third party server in order to limit the risk of a single point of data leak. With this kind of approach, we need to trust the nearby devices we need to communicate with. We first oriented our work towards data exchanges based on the WiFi-Direct protocol, which is already supported by Android.

As Conti et al. [5] exposed the feasibility of opportunistic network atop of WiFi-Direct, we think that WiFi-Direct is a good candidate to disseminate data in the context of crowd-sensing platforms. The major drawback of the WiFi-Direct implementation is that it requires a human action to pair devices the first time they connect. An alternative to this protocol could be to integrate an implementation of specific opportunistic WiFi protocol as proposed by Trifunovic et al. [26], which is specifically thought for the kind of interaction that we need, but not included by default in Android.

We first developed an Android app to build a skeleton that can be used to implement different kinds of anonymization algorithms. This application uses the WiFi-Direct as the underlying transport layer to exchange data between devices. We focus on the WiFi-Direct protocol instead of the Bluetooth because of the larger communication radius that WiFi provides.

Additionally, we build on the *Network Service Discovery* (NSD) that is provided by default by the system. This discovery protocol leverages the implementation of mobile services and the interaction between devices.

Data dissemination process. The data dissemination process is exposed in the Figure 1.

The environment is composed of an APISENSE server and several devices. The process involves 3 steps: the production, the dissemination and the sending to the server.

1. Data production: devices produce data that are requested for a data collection.

2. Data dissemination: according to the rules implemented by the data dissemination algorithm, each device sends its data to others which are close enough and which provide the data dissemination service. For instance, the devices A, B, C, D and E can send each other data through WiFi-direct. No data are send to the F device because it does not provide the data dissemination service.

3. Sending: after an amount of time and/or a number of exchange for a given data, a device sends it to the server. The server knows the data sender (the last device in the chain), but because of the system implementation, it is unable to recover the data producer.

Packages. Our application skeleton is composed of 3 main packages: `data`, `wifidirect` and `dao`.

The data package provides a `DataManager`, which plays the role of data selector. When the application needs or wants to send data, the `DataManager` is responsible for providing the data to send. A class `Data` is a kind of container for sending data and this class can be extended to provide a specific type of data. `DataManager` provides a way to change the selector instance. The current selector implemented is a random selector, which randomly select a data to send. We don't use the Linear Congruential Pseudo Random Number Generator provides by `java.util.Random` because it can be predicatable [14]. Instead, we choose to use the `java.security.SecureRandom` which is a better one. In the future, we plan to implement selectors, which could select data only after a certain amount of time or any other selection techniques that could improve privacy.

The peer package includes all the classes that manage neighbor peers. It contains a PeerManager, which select and order peers to interact with them. The implementation provides a way to implement other peer selection algorithms. The current selector is a random one. It basically takes randomly a neighbor peer in its list and use the `java.security.SecureRandom` implementation.

We plan to use Gossip protocol in our data dissemination algorithm. This kind of protocol is the best to use for data dissemination in a large-scale environment. A framework, presented by Jelasity et al. [13], is planned to be used in particular.

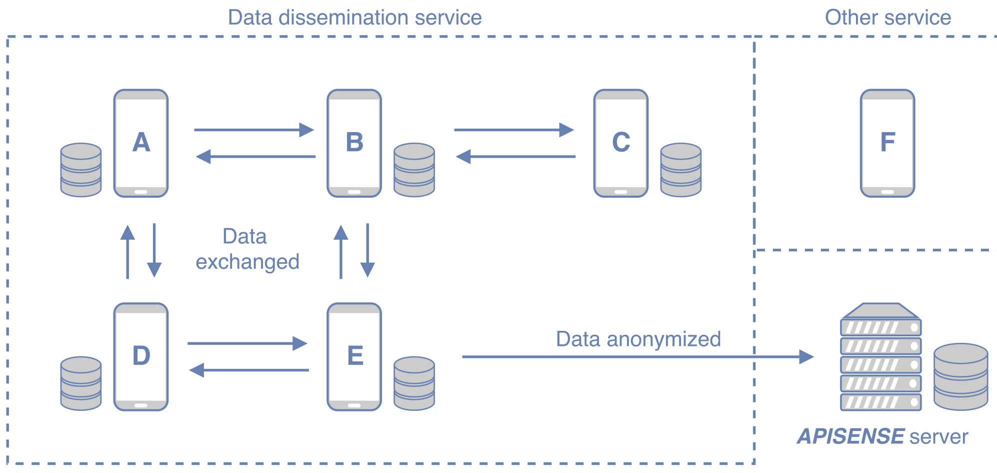

Figure 1: Mobile interactions

4. TOWARDS LARGE-SCALE EMULATION

We are interested in adopting emulation platforms because of the lack of wireless emulator for the Android OS, to the best of our knowledge. In particular, the standard Android emulator does not include a support for WiFi, WiFi-Direct or Bluetooth implementations. This means that it is not possible to test a mobile app that extensively uses these technologies.

GENYMOTION[1] is a commercial emulator that offers the possibility to use the WiFi by binding it to the Ethernet network, but it does not provide any WiFi-Direct or Bluetooth implementations.

WiFi-Direct. Our first goal is to provide a WiFi-Direct implementation to be able to test our data dissemination algorithm by re-implementing an original WiFi-Direct Android API. We aim to emulate WiFi-Direct by making an interface layer which replaces original interactions by a custom homemade. For instance, forward a WiFi-Direct request through an software-defined layer that return the response instead of hardware component that does not exist in the emulator. The external server could interact with the library via standard socket and give the possibility to enable communications between devices that are considered as closed enough.

Nevertheless, integrating our WiFi-Direct mock implementation requires some code change to be made in the mobile app under test. This can be considered as a drawback, but a tool, named XPOSED[2], can be used to get around the need of code modification. XPOSED is a hooking method system for Android, inspired by the principles of *Aspect-Oriented Programming* (AOP), that is automatically launched at the system boot. With XPOSED, one needs to specify the methods that will be hooked and to provide an action to apply before or after the original method is called. It is possible to replace entirely the base implementation to provide a completely new behavior. This is a very interesting way, but using this tool requires a rooted version of the Android system, which is rather complex to achieve in an emulator.

GPS scenario. With this simulation platform, we want to exploit proximity user interactions. This implies that we want to simulate movements of a human crowd. We plan to implement an orchestration engine takes mobility scenarios as an input and orchestrates the virtual location of emulators. A scenario is therefore a suite of timestamped GPS locations and each emulator instance reflects a user who moves across a city, for example. With such an orchestration engine, we are able to mimic real user movements in a city.

System architecture. We propose the system architecture describe in Figure 2. This architecture is composed of 3 main parts that are: one Master communicating with several Machines, themselves orchestrate several Emulators. An UI part is integrated to monitor the simulation, but not required.

Each part of the system is packaged as DOCKER[3] containers. This tool leverages the re-usability and the deployment of our system. By default, each container has a local IP address on a private network on a host. On this network, only containers that are on the same machine can interact with each others. To enable the interaction between containers through different machines, we use WEAVE[4]. This tool allows to create a network where each container has a public IP address that enables us to make interaction between them.

Details. The Master is a specific type of container. The system can have only one container of this type. The Master container always takes the same IP address on the Weave public network. By doing that, each Machine container can easily contact and interact with it without any prior configuration or discovery process. The role of the Master is to orchestrate the entire simulation and to act as a registry, which group each part of the system. It therefore orchestrates the simulation by broadcasting events to Nodes, which are essentially to move to the next GPS position in a scenario or to execute a simulation action (*e.g.*, to discover a nearby device, to start an app)

A Machine is a container initialized on host where are deployed Emulator containers. We have a single Machine by host. A Machine container has a fixed local address in addition to the public address on the Weave network. With this fixed local address, Emulator containers launched on the

[1]https://www.genymotion.com
[2]http://repo.xposed.info
[3]https://www.docker.com
[4]https://www.weave.works

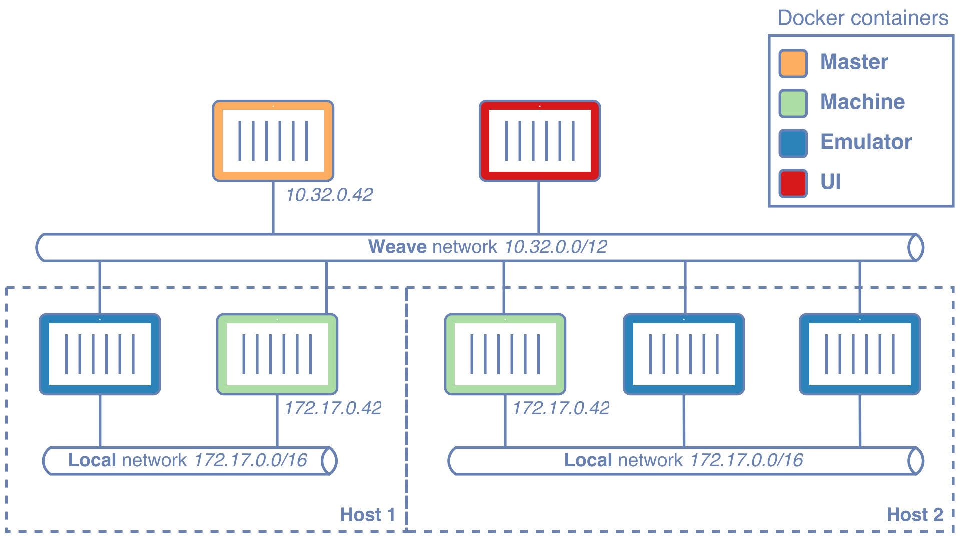

Figure 2: System architecture

same machine can easily interact with it. Machine containers are also responsible for the local deployment of Emulator containers.

Finally, the Emulator container contains the WiFi-Direct server interaction and an Android emulator. The Android emulator is started with the container and to be able to interact with node commands.

5. RELATED WORK

In this section, we discuss the relevant literature focusing on privacy and emulation platforms.

Privacy. There are several approaches related to privacy-preserving algorithms. The first technique, which was used at the beginning of the Internet, is the *pseudonymization* and consists in hiding an user identity behind a fake one. This technique can improve the user anonymity, but cannot be generalized to privacy because of possible linkage of other information. Furthermore, *pseudonymization* is insufficient when used alone, as exposed by Rao and Rohatgi [23], Pfitzmann and Hansen [21] or Christin et al. [4]. Another example of the limits of this method, is the de-anonymization of the AOL search dataset [1]. Because the company was simply replacing user identity by an id, researchers succeeded in recovering user identities from her search requests by linking them with other sources of information.

The second anonymization technique that is used in the field is known as *obfuscation*. In particular, the reference algorithm for data obfuscation is *k-anonymity*, which was introduced by Sweeney [25]. K-anonymity were improved with *l-diversity* introduced by Machanavajjhala et al. [18], and after the work by Li et al. [17], called *t-closeness* came to cover the weaknesses of the previous methods. The obfuscation techniques protect against linkage attack, which consist to take two datasets and link data between them to extract information about users. However, because obfuscation can modify information to be not able to be linked, the loss of information can be troublesome for the applications that use these data. While the obfuscation replaces a data by an inclusive range, *sanitization* is another approaches that systematically deletes the critical data. However, this technique is rougher than obfuscation due to the complete loss of information.

The third approach is based on *randomization* and adds noises to datasets. Like obfuscation techniques, these have the drawbacks of degrading the information. For example, blurring techniques [20] is a kind of randomization that add noises on localization data.

Finally, an approach called *generalization* tries to generalize a data to hide the interesting one, as explained by Huang et al. [12]. This article explains the tessellation approach, which consist to request a larger set of data that will contain the original requested. This approach hides the real intent of a user and increase their privacy.

We can observe that there are several techniques that try to preserve privacy. One of the hottest issue in privacy concerns localization data. Localization data are the most critical data as exposed in many research [4, 15, 22, 8, 7, 3], which are focused on localization privacy issues.

Simulation. There are many kinds of simulation tools that provide a support to simulate different things. However, research made in this field are generally old and cannot be run currently.

Martin and Nurmi [19] provides a simulation platform, which simulates interactions, but does not provide way to plug OS emulation.

Wehrle et al. [27] propose a network emulation architecture, which focuses on network interactions. They provide an Android plug-in, which use the old OS emulator, but their approach is too complex to run.

Hetu et al. [11] present another simulator, but their proposal require to introduce a lot of code refactoring in the tested application.

Richerzhagen et al. [24] work seems to be a good candidate, but there is no source code available and no way to test it.

Henne et al. [10] provide a simulation platform, which focuses on interactions and does not provide Android or iOS simulation possibilities. In particular, it is impossible to run Android application with it.

Finally, Bruno et al. [2] propose a solution which is a good

WiFi-Direct simulator. However, this simulator cannot scale to run a crowd of devices. Moreover, one need to change a lot of code in the tested application to interact with it. Finally, there is no API to control device interactions as only way to control interaction is to pass through a command line interface.

6. CONCLUSION

In this paper, we presented our current and future work. We developed an Android application skeleton that can be used as a basis to test different anonymization techniques using peer-to-peer protocols applied on mobile devices. Because of privacy requirements, we must validate our results by making tests which ensure execution conditions that are closer to the real usage, and thus providing a reproducible and large scale environment. This is the reason why we also plan to deliver a complete emulation platform to test Android applications using WiFi-Direct interactions for crowdsensing applications. In particular, the system intends to play GPS scenarios to simulate human movements. We make a lot of efforts to make the system as simple as possible and we use Docker for the usage and deployment facilities. The simulation platform can be deployed by people who want to make experiments on a large set of mobile devices. Still, our system will require to have application source code to apply minor changes in import libraries, in order to be able to use our WiFi-Direct mock implementation.

In the future, we will continue to develop the emulation platform and to improve our WiFi-Direct library by implementing a larger part of the WiFi-direct stack provided by the Android operating system. We are also interested in avoiding the need for source code changes by using library like XPOSED to be able to test standard binary applications. We plan also to provide a proof of the modularity of our system by implementing features like: the support of other wireless technologies or the degradation of wireless communications. Because localization data are the most sensitive user data, we also focus our interest on privacy solutions that can improve the privacy provided on this kind of data.

7. REFERENCES

[1] M. Barbaro, T. Zeller, and S. Hansell. A face is exposed for aol searcher no. 4417749. *New York Times*, 9(2008):8For, 2006.

[2] R. Bruno, N. Santos, and P. Ferreira. Termite: Emulation testbed for encounter networks.

[3] K. Chatzikokolakis, C. Palamidessi, and M. Stronati. Geo-indistinguishability: A principled approach to location privacy. In *Distributed Computing and Internet Technology*, pages 49–72. Springer, 2015.

[4] D. Christin, A. Reinhardt, S. S. Kanhere, and M. Hollick. A survey on privacy in mobile participatory sensing applications. *Journal of Systems and Software*, 84(11):1928–1946, 2011.

[5] M. Conti, F. Delmastro, G. Minutiello, and R. Paris. Experimenting opportunistic networks with wifi direct. In *Wireless Days (WD), 2013 IFIP*, pages 1–6. IEEE, 2013.

[6] C. Cornelius, A. Kapadia, D. Kotz, D. Peebles, M. Shin, and N. Triandopoulos. Anonysense: privacy-aware people-centric sensing. In *Proceedings of the 6th international conference on Mobile systems, applications, and services*, pages 211–224. ACM, 2008.

[7] Y.-A. de Montjoye, C. A. Hidalgo, M. Verleysen, and V. D. Blondel. Unique in the crowd: The privacy bounds of human mobility. *Scientific reports*, 3, 2013.

[8] S. Gambs, M.-O. Killijian, and M. N. del Prado Cortez. Show me how you move and i will tell you who you are. In *Proceedings of the 3rd ACM SIGSPATIAL International Workshop on Security and Privacy in GIS and LBS*, pages 34–41. ACM, 2010.

[9] N. Haderer. *APISENSE®: une plate-forme répartie pour la conception, le déploiement et l'exécution de campagnes de collecte de données sur des terminaux intelligents.* PhD thesis, Université des Sciences et Technologie de Lille-Lille I, 2014.

[10] B. Henne, C. Szongott, and M. Smith. Towards a mobile security & privacy simulator. In *Open Systems (ICOS), 2011 IEEE Conference on*, pages 95–100. IEEE, 2011.

[11] S. N. Hetu, V. S. Hamishagi, and L.-S. Peh. Similitude: Interfacing a traffic simulator and network simulator with emulated android clients. In *Vehicular Technology Conference (VTC Fall), 2014 IEEE 80th*, pages 1–7. IEEE, 2014.

[12] K. L. Huang, S. S. Kanhere, and W. Hu. Preserving privacy in participatory sensing systems. *Computer Communications*, 33(11):1266–1280, 2010.

[13] M. Jelasity, R. Guerraoui, A.-M. Kermarrec, and M. Van Steen. The peer sampling service: Experimental evaluation of unstructured gossip-based implementations. In *Proceedings of the 5th ACM/IFIP/USENIX international conference on Middleware*, pages 79–98. Springer-Verlag New York, Inc., 2004.

[14] H. Krawczyk. How to predict congruential generators. *Journal of Algorithms*, 13(4):527–545, 1992.

[15] J. Krumm. A survey of computational location privacy. *Personal and Ubiquitous Computing*, 13(6):391–399, 2009.

[16] M. Langheinrich. Privacy by design-principles of privacy-aware ubiquitous systems. In *Ubicomp 2001: Ubiquitous Computing*, pages 273–291. Springer, 2001.

[17] N. Li, T. Li, and S. Venkatasubramanian. t-closeness: Privacy beyond k-anonymity and l-diversity. In *Data Engineering, 2007. ICDE 2007. IEEE 23rd International Conference on*, pages 106–115. IEEE, 2007.

[18] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 1(1):3, 2007.

[19] M. Martin and P. Nurmi. A generic large scale simulator for ubiquitous computing. In *Mobile and Ubiquitous Systems: Networking & Services, 2006 Third Annual International Conference on*, pages 1–3. IEEE, 2006.

[20] A. Moawad, T. Hartmann, F. Fouquet, J. Klein, and Y. Le Traon. Adaptive blurring of sensor data to balance privacy and utility for ubiquitous services. In *Proceedings of the 30th Annual ACM Symposium on Applied Computing*, pages 2271–2278. ACM, 2015.

[21] A. Pfitzmann and M. Hansen. A terminology for

talking about privacy by data minimization: Anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management, 2010.

[22] V. Primault, S. Ben Mokhtar, and L. Brunie. Privacy-preserving publication of mobility data with high utility. In *Distributed Computing Systems (ICDCS), 2015 IEEE 35th International Conference on*, pages 802–803. IEEE, 2015.

[23] J. R. Rao, P. Rohatgi, et al. Can pseudonymity really guarantee privacy? In *USENIX Security Symposium*, 2000.

[24] B. Richerzhagen, D. Stingl, J. Rückert, and R. Steinmetz. Simonstrator: Simulation and prototyping platform for distributed mobile applications. In *Proceedings of the 8th International Conference on Simulation Tools and Techniques*, pages 99–108. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2015.

[25] L. Sweeney. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(05):557–570, 2002.

[26] S. Trifunovic, B. Distl, D. Schatzmann, and F. Legendre. Wifi-opp: ad-hoc-less opportunistic networking. In *Proceedings of the 6th ACM workshop on Challenged networks*, pages 37–42. ACM, 2011.

[27] E. Weingärtner, H. v. Lehn, and K. Wehrle. Device driver-enabled wireless network emulation. In *Proceedings of the 4th International ICST Conference on Simulation Tools and Techniques*, pages 188–197. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2011.