Open access • Proceedings Article • DOI:10.1145/3097766.3097772

Lessons Learned while Trying to Reproduce the OpenRF Experiment

Mohamed Naoufal Mahfoudi, Thierry Turletti, Thierry Parmentelat, Walid Dabbous

Institutions: French Institute for Research in Computer Science and Automation

Published on: 11 Aug 2017 - ACM Special Interest Group on Data Communication

Lessons Learned while Trying to Reproduce the OpenRF Experiment

Mohamed Naoufal Mahfoudi, Thierry Turletti, Thierry Parmentelat, Walid Dabbous

Lessons Learned while Trying to Reproduce the OpenRF Experiment

Mohamed Naoufal Mahfoudi, Thierry Turletti, Thierry Parmentelat, Walid Dabbous

Université Côte d'Azur, Inria, France

ABSTRACT

Evaluating and comparing performance of wireless systems, like for any other scientific area, requires the ability to reproduce experimental results. In this paper, we describe the specific issues that we encountered when focusing on reproducing the experiments described in a paper related to wireless systems. We selected the OpenRF paper published in SIGCOMM 2013, a very interesting research work allowing to perform beamforming on commodity WiFi devices. We illustrate how reproducibility is strongly dependent on the used hardware, and why an extensive knowledge of the used hardware and its design is necessary. On the basis of this experience, we propose some recommendations and lessons for the design of reproducible wireless experiments.

KEYWORDS

Reproducibility

1 INTRODUCTION

Our initial goals when starting our work around OpenRF [1] were to study and design cross-layer performance enhancements in commodity Multiple-Input Multiple-Output (MIMO) devices. This required the ability to easily modify the network, MAC and physical layers parameters of the wireless system. For example, accessing the channel state information (CSI) and using it to precode the transmission matrix is essential for beamforming and interference nulling. Upon receiving an explicit beamforming request from the transmitter, the receiver computes and sends back a compressed form of the measured CSI matrix respecting the IEEE 802.11n recommendations. In this way, the transmitter is able to precode its transmission matrix by allocating complex weights on each antenna element, which influences the phase as well as the power of the radiated signal. This results in focusing the radiated signal to the receiver's region, which in turn increases the signal-to-noise ratio (SNR) at the receiver and reduces interference for others. This feature was proposed in the IEEE 802.11n draft [2], but very few chip manufacturers enabled it natively. Moreover, it is usually not possible to access the CSI or to precode the transmission matrix in Consumer-Off-The-Shelf (COTS) devices, as firmware is generally locked.

2 OPENRF

In the absence of close collaboration with chip manufacturers, we chose to use the OpenRF extensions[1] of the 802.11n CSI Tool, originally developed by the University of Washington [3] for the Intel Wireless Link 5300 802.11n MIMO cards. Indeed, it is interesting to use OpenRF as it enables commodity Wi-Fi access points to perform several MIMO management techniques: interference nulling, implicit beamforming and interference alignment. Contrary to the explicit beamforming technique described in the IEEE 802.11n draft, OpenRF uses an implicit beamforming mechanism that requires no feedback from the receiver, assuming that the channel conditions are the same in both directions. The OpenRF code includes a MATLAB function that precodes the channel for either nulling interference or beamforming according to the MAC address of the receiver. To do so, this function reads the CSI matrix on the transmitter side, computes the steering or beamforming matrix in user space, and sends it back to the wireless chip driver. We identify the OpenRF system components in figure 1.

To sum up, the perspective of using beamforming within a cross-layer system was the main reason for us to reproduce the OpenRF experiments.

2.1 Reproducibility challenges

The OpenRF code release web page[1] contains installation and testing instructions with an old Ubuntu 10.04.4 LTS ISO image to download. It was late 2015 when we were trying to reproduce the OpenRF experiment, and we faced the problem that this deprecated Ubuntu 10.04.4 LTS release would not support our recent hardware; so we first attempted to install the OpenRF tool on a more recent release (15.04), keeping the same kernel as the one used for the original implementation. Within a 15.04 image, compiling OpenRF raised issues related to deprecated packages. We were unable to conduct and maintain properly our experiments even after applying the necessary changes and improvements. Suspecting that the problem would be coming from the OS distribution, we tried to identify and use the previous Ubuntu long term releases, more precisely 14.04 LTS and 12.04 LTS. Installing OpenRF was much simpler within these distributions, as we were not confronted to compilation problems as before.

However, this setup did not yield satisfactory results as we were unable to reproduce the experiment, which consists in precoding our transmission with a beamforming matrix. When reached directly through email, the original authors of the OpenRF paper suggested that using 12.04 LTS was a potential source of problem. So we took measures to install 10.04 LTS regardless of the numerous incompatibilities - video card, SSD hard drive and network card were not supported - and we finally managed to install it after much effort. Therefore, the following system configuration was adopted for reproducing the experiment:

OS	Kernel	Wireless Cards
Ubuntu 10.04.4 LTS	3.5.4-csitool+	533AN MMW Full / Half

[1] https://www.andrew.cmu.edu/user/swarunk/openrf.html

At that point we had matched exactly the original conditions of the OpenRF paper, but were totally unable to reproduce its results.

More specifically, obtaining the CSI matrix was easy enough with the use of the Intel CSI tool. However, when it came to beamforming using OpenRF, we did not observe any variation in the signal to noise ratio at the receiver. Hence, we adopted a verification process that would eventually allow to pinpoint the problem. The first step of verification was done by checking the CSI at the receiver end, in order to assess the effect of injecting a beamforming matrix at the transmitter. This was done by plotting the SNR for each one of the subcarriers before and after injecting the transmission matrix. Normally, the spatial mapper should use the information from the injected matrix to compute the beamforming vectors. However, we observed that the beamforming matrix is never applied and the spatial mapper is still using Intel's indirect spatial mapping every time we send a packet. This situation is indeed peculiar, since we were getting a positive acknowledgement from the driver that the beamforming matrix was injected. We double-checked every step down the path for sending the CSI report to the wireless card firmware. We resorted to checking the driver messages which report the state of the card and possible problems, but could not spot any message related to a firmware crash or a rejection of the matrix injection by the firmware. In addition, we adopted several transmission scenarios, while injecting the beamforming matrix. At first, we kept the rate adaptation mechanism functioning normally, which did not yield any changes other than those related to the rate variation. Then, we specified the number of space-time streams (between 1 and 3), as well as the modulation and coding scheme (MCS) code, and we did not notice any variations on the SNR plots. As mentioned before, we adopted this verification process on different Ubuntu releases starting all the way from a recent 15.04 Ubuntu, including 14.04 and 12.04, down to a deprecated 10.04 LTS. This effort was done to match as closely as possible the software setup of the original implementation of both Intel CSI tool and OpenRF. In our work, we used almost 45 wireless cards that vary in sizes (Half or Full) and origins, and none of them supported correctly the injection of the beamforming matrix. Further details on our verification process are described in Appendix A, stating as accurately as possible the technical path that we followed. Our goal is to promote such practices that lead to better replicability and to ease the learning process for future experimenters.

From this experience, even though the authors provided their experiment source code, it appears that there was not enough information about the hardware settings. Providing full description of the experimental setup is critical to reproduce the experiment, especially when the used equipment presents some peculiarities, e.g., related to the manufacturers. In fact, we found out several types of the Intel WiFi Link 5300 cards on the market. Some of them were engineering samples, which are generally source of problems and complications. We contacted the authors to seek their support in this matter, however this interaction was not fruitful as none of the proposed solutions was successful. The first author of the present paper has spent 3 weeks in a total period of 4 months trying to reproduce the OpenRF experiment.

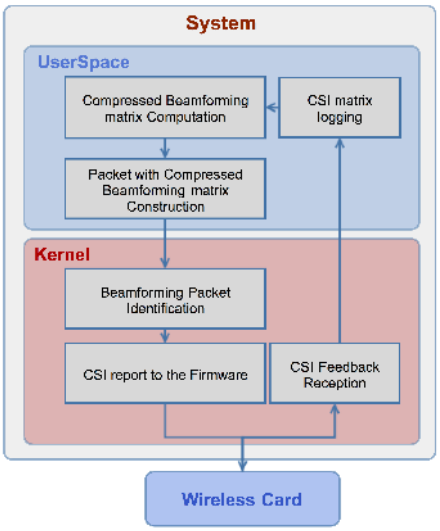

Figure 1: Beamforming System Design

2.2 Recommendations

Although *reproducibility is the ultimate goal* as stated by the ACM Result and Artifact Review and Badging policy[2], the replicability of experimental results done by independent researchers using author-supplied artifacts represents an important intermediate step. The simplest case is when experiments can be done using hardware and software available in open testbeds such as ORBIT[3] and R2lab[4]. In this case, authors only need to ensure they provide enough details on the scenarios and to make available all the code and scripts used to replicate the experiments. Unfortunately, these testbeds are not convenient for all types of experiments and for instance, the experiment repeatability could be tied to a specific hardware setting not available in those testbeds. So, it is important to provide further details on the configuration, with a focus on the specifications and requirements that are necessary for reproducing the experiments. In particular, a simple description of the OS reference is not sufficient, there should be a mention of what makes this version of the OS important for the success of the experiments (packages, drivers, ...). This means the full description of the defining characteristics have to be provided and not just a simple description of the setup used for the experiment. Indeed, this could help future experimenters to be able to find a suitable equivalent when the used hardware or software is not available anymore. This could also allow for a future-proof reproducible experiment. Moreover, it is helpful that authors verify that their solution works with hardware from different manufacturers or OS and, as importantly, that they mention the setups that did not work. Again, mentioning the motivation behind the choice of a certain hardware or software is highly desirable. In case the verification work cannot be done, an explicit warning in this regard should be present, and all the details concerning the hardware used should be provided with the corresponding references (serial number, manufacturer, vendor, etc.). As

[2]https://www.acm.org/publications/policies/artifact-review-badging
[3]ORBIT testbed at WINLAB, URL: http://www.orbit-lab.org/
[4]FIT Reproducible Research Lab (R2lab) at Inria, URL: http://fit-r2lab.inria.fr

for the software used for the system design, authors should avoid when possible licensed software. For example, the use of MATLAB code also hindered our setup due to license management difficulties on experimental machines.

Finally, it is worth considering to manage a research project more like a software development project; indeed in many respects the challenges for reproducibility have strong similarities with the ones of software development, and we believe that tools like source code management tools, maybe even test suite frameworks, as well as interactive computing concepts like notebooks, can be very helpful in building more reproducible research.

3 CONCLUSION

In this paper, we propose a series of recommendations for the reproduction of wireless experiments. In fact, there are generally two types of problems impairing the reproducibility of wireless experiments, one related to the hardware/software and the other to the variability of the wireless channel. We outline that a sufficient knowledge of the hardware/software is essential, and often underestimated in the literature, probably being deemed as too mundane and incidental.

ACKNOWLEDGMENTS

The R2lab wireless testbed at Inria has been funded by the ANR Equipex FIT 6165. This work has been partly funded by the Labex UCN@Sophia. We thank the authors of the OpenRF paper [1] for proposing directions to overcome the encountered issues with the Intel 5300 wireless card. We also thank the reviewers and our shepherd for their valuable comments.

REFERENCES

[1] Swarun Kumar, Diego Cifuentes, Shyamnath Gollakota, and Dina Katabi. Bringing cross-layer mimo to today's wireless lans. *SIGCOMM Comput. Commun. Rev.*, 43(4):387–398, August 2013.

[2] IEEE Standard for Information technology (2009). Local and metropolitan area networks— Specific requirements— Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 5: Enhancements for Higher Throughput, IEEE Std 802.11n-2009, Oct 2009.

[3] Daniel Chaim Halperin, Wenju Hu, Anmol Sheth, and David Wetherall. Tool Release: Gathering 802.11N Traces with Channel State Information. *ACM CCR*, 41(1):53–53, Jan 2011.

A DETAILED VERIFICATION PROCESS

When trying to reproduce the OpenRF results, our symptom was that no matter how hard we try to change, in our Intel WiFi Link 5300 card, the CSI matrix used for emission, we could not measure any noticeable change in the emitted waves. In this appendix, we describe the technical steps that we have taken to try and get our Intel cards to take our modified CSI matrix into account - but to no avail. We are thus left with tracking down data interchange between kernel space - namely the modified `iwlwifi` driver - and userland, and back.

Upon reception of a packet, a CSI report is sent from kernel to userspace using a netlink socket. Then the CSI report is decoded, the CSI matrix is extracted and a singular value decomposition of the matrix is performed. This decomposition results in the computation of the so-called V matrix used for either beamforming, interference nulling or alignment. This matrix is compressed using specific rotations respecting the IEEE 802.11n recommendation for explicit beamforming. These operations are conducted through the MATLAB function `precod_channel` in userspace. After computing the V matrix, we need to send it to the wireless card. Two structures are used when sending this matrix back to the kernel: the MIMO Control subfields and the Compressed Beamforming Report field (`iwl-command.h`), both defined in the standard [2]. More precisely, the MIMO Control field is implemented to handle beamforming feedback information and Compressed Beamforming Report field is to used to carry explicit feedback in the form of angles to be used by a transmitter when computing the corresponding steering matrix.

After storing the matrix in the aforementioned structures, a netlink socket is used to send this CSI report from userland to kernel space with a specific ID (`REPLY_BFER_VCOMP_CONFIG = 0xbc`). This operation is implemented in the `test_send_weight_matrix.c` function. When the CSI report structure is received in kernel space, it is used in function (`iwlagn_send_bfer_config`) defined in file `iwl-agn-lib.c`. In this function we are using the command `iwl_dvm_send_cmd_pdu` for sending this CSI report to firmware. When checking the driver messages after the beamforming operation is completed, we obtain the following messages:

```
$ dmesg
...
iwlwifi 0000:04:00.0: Setting beamforming matrix
iwlwifi 0000:04:00.0: Set bf: Returned (0)
iwlwifi 0000:04:00.0: In iwlagn_send_rxon_assoc_wsdn
```

These driver messages indicate that the beamforming was performed, and that the wireless card is already using the compressed beamforming matrix instead of the indirect mapping matrix. However, as mentioned before, despite all that the SNR plots showed no changes in the signal levels. From our correspondence with the authors, we applied a transmitter reset to verify if it could fix the problem but it has no effect either.

```
$ sudo bash -c "echo 0 >
/sys/kernel/debug/ieee80211/phy0/iwlwifi/debug/bf_flag;
echo 1 >
/sys/kernel/debug/ieee80211/phy0/iwlwifi/debug/bf_flag"
```