A Hybrid Testbed for Secure Internet-of-Things

Ştefan-Ciprian Arseni(✉), Alexandru Vulpe(✉), Simona Halunga, and Octavian Fratu

Telecommunications Department, University Politehnica of Bucharest, 1-3 Iuliu Maniu Blvd., 061071 Bucharest, Romania
{stefan.arseni, alex.vulpe}@radio.pub.ro, {shalunga, ofratu}@elcom.pub.ro

Abstract. The need for insertion of technology in everyday tasks has brought an increase in new methodologies and concepts used to accomplish such objectives. By trying to make technology an enabler for an increasing number of personal or work-related activities, we allow devices to collect data about our way of being, that, if not properly protected and used, can prove a vulnerability for our personal security. This is why new means of securing information, even by the tiniest or low-resource devices, need to be implemented and, in many cases, they take the form of cryptographic algorithms, classic or lightweight. Assessing these algorithms can sometimes become difficult, depending on the targeted system or on the environment where the device will be deployed. To address this issue and help developers, in this paper we present a hybrid testbed, comprised of three hardware architectures, that will ensure a general environment in which users can test their security solutions, in order to have an idea of what changes need to be made to provide optimal performances.

Keywords: Internet-of-Things · Security · Hybrid testbed
Software middleware · Hardware architectures

1 Introduction

Ever since the industrial revolution, humanity has been searching for methods of creating better technologies that can improve the way humans not only interact with the environment, but also how they make use of the resources provided. Recent years have brought a new technological revolution, in terms of miniaturization of devices and their embedment in all layers of society. Earlier proposed concepts, such as Internet of Things [1, 2], Cloud Computing [3, 4] or Deep Learning [5, 6], have become today's trending technologies, being in a continuous process of development [7] and integration [8–10]. Yet, this fast pace that is characteristic to any recent offer-demand pair in nowadays' economy [11, 12] introduces also some vulnerabilities regarding mainly the security aspects [13, 14]. Focusing on the Internet of Things (IoT) concept [15, 16], the embedment of sensors or smart devices in the environment [17] surrounding us sets new thresholds that need to be passed before any data is captured, processed and/or transmitted to a sink device or to a Cloud service.

O. Fratu et al. (Eds.): FABULOUS 2017, LNICST 241, pp. 3–8, 2018.
https://doi.org/10.1007/978-3-319-92213-3_1

Addressing the need of security in transmitting information has been generally made by using cryptographic algorithms to encrypt the data that needs to be sent. These cryptographic algorithms have different key features that make them reliable under certain conditions or in specific environments. The miniaturization of devices brought a problem for security, given that classic cryptographic algorithms require a certain amount of resources to function with an acceptable performance. The constraint resources of a sensor or embedded device lead to the introduction of a new branch of cryptographic algorithms, namely lightweight cryptographic algorithms that give a reasonable degree of security, without requiring too many resources [18–20].

Being a relatively new study domain, when compared to classic cryptography, lightweight cryptographic algorithms can, sometimes, prove to be difficult to implement or create, thus requiring a strict phase of testing in which any minor vulnerability or performance drop can be resolved. The hybrid testbed presented in this article was first introduced in [21] and it addresses the problem of testing the implementations of lightweight cryptography, by giving developers a unified platform to conduct their tests on. This testbed is comprised of three different types of hardware architectures, so that the behavior of the implementation on specific environments can be observed. The access to these hardware architectures is done through a middleware that acts as a unique point-of-entrance, enabling developers to write their implementation once and test it simultaneously or consecutively, on each one of the three base architectures. This paper continues the testbed presentation initiated in the previously mentioned paper, by adding information regarding the middleware layer of the testbed, on how it makes use of the hardware architectures and how it enables users to interact with the testbed.

The present paper is organized as follows. Section 2 gives an overview of the proposed testbed, while in Sect. 3 we focus on the software layer of the testbed and give a description of the method of interfacing users with the middleware. In Sect. 4, some conclusions are drawn.

2 Overview of the Testbed

In scientific literature, multiple papers, [22–26], have presented the benefits given by the hardware implementations of cryptographic algorithms as compared to the software counterparts. By integrating in our testbed an SDSoC (Software Defined System-On-a-Chip) that contains also a number of logical gates that can be programmed, we enable users to test their implementation in this type of environment also. The other two hardware architectures of the testbed allow only a software implementation of an algorithm.

In order for users to interact with these hardware architectures, they need to interact with the middleware layer of the testbed. This layer is composed of two sub- layers: one consisting of the drivers or APIs (Application Programming Interface), required to communicate with the hardware layer, and one consisting of the integrator APIs which developers will use and integrate in their implementations.

The main architecture of the testbed, with an emphasis on the hardware layer, was presented in papers [21, 27] and Fig. 1 depicts its high-level design. The current paper continues the description through an initial validation of the middleware layer,

emphasizing on the connection of it with the hardware layer and the means that users can access its functions.

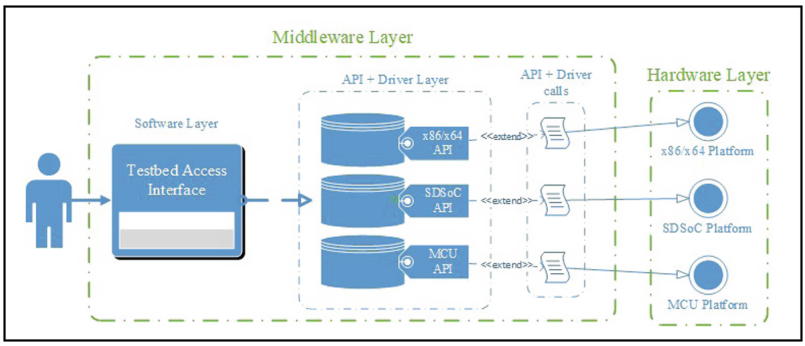

Fig. 1. Main architecture of testbed

3 Middleware Layer of the Testbed

As described in [27], the main challenge is to create a unified access point for all three hardware systems, while maintaining control over the operating characteristics that represent criteria when assessing an algorithm. In order to achieve this point of integration the platforms were customized to support a self-contained operating environment, either under the form of an operating system or a scheduler-service, as will be described in the remaining part of the section.

Each one of the hardware architectures presents its own methods of interaction that are made available to a user or developer. Figure 2 presents an overview of forming and receiving, by the corresponding architecture, the API and driver calls, shown in Fig. 1. The UI (User Interface) is the component that sends calls to the middleware and customizes the task sent to each one of the hardware elements, as follows:

- For the standard x64 processor that can run also x86 applications, a simple task dispatcher is implemented as a service that waits for the user to send a task to be executed. The task represents an algorithm implementation that is done by the user and sent through the testbed UI. After the completion of the task, collected metrics are parsed and sent to the UI to give the user a performance view of its implementation or algorithm.
- The SDSoC architecture is comprised of two hardware elements that can be used either separately or as a whole. In our proposed testbed, these elements are used as one, by establishing the required communication bridge between them. The FPGA (Field Programmable Gate Array) element contains some defined cryptographic functions, such as permutation, standard AES (Advanced Encryption Standard) S-Boxes or substitutions, which can increase the performance of an algorithm. The implementation in FPGA was taken into consideration given a few performance tests made between a hardware and software implementation of the same algorithm,

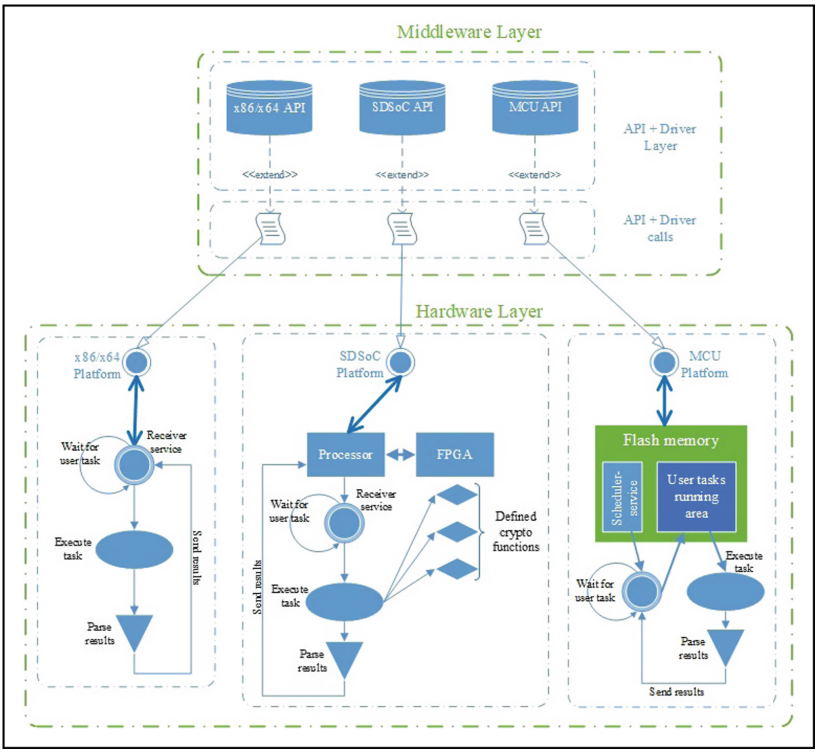

Fig. 2. Hardware layer customization

tests that were briefly presented in [27]. The second element, the processor, is the point in which the dispatcher service resides and the task is mainly being executed. After the successful completion of the task, results are parsed and transmitted to the UI.

- The third hardware element takes advantage of the possibility of dynamically writing an application into the Flash memory that can be executed afterwards. Implemented as a bootloader, the scheduling-service waits for a task to be deployed through the UI. After receiving a task, the service writes it into the Flash at a separate address and launches it into execution. In this case, the service will return in the waiting state, but will not have the functionality of parsing and sending the results. This functionality will reside in the task itself and will be attached to the user code at deployment.

After terminating all the deployed tasks, the UI will act also as a collector of data from the underlying dispatcher-services, by formatting and grouping the data and presenting it to the user as performance metrics.

4 Conclusions

The paper introduces a part of the elements that the proposed hybrid testbed contains, with a focus on the software layer and on method of interaction between users and it. Given that this testbed will enable simultaneous testing on three different hardware architectures, it can prove to be an important factor in the process of large-scale integration of the IoT concept. An initial validation of the proposed testbed has been made with high-level tests, through which observations were made on how the testbed can be completely integrated and how multiple functionalities can be developed for users.

Acknowledgments. This work was supported by University "Politehnica" of Bucharest, through the "Excellence Research Grants" Program, UPB – GEX, Identifier: UPB–EXCELENTA–2016, project "Platform for Studying Security in IoT", contract number 96/2016 (PaSS-IoT) and by a grant of the Ministry of Innovation and Research, UEFISCDI, project number 5 Sol/2017 within PNCDI III and partially funded by UEFISCDI Romania under grant no. 60BG/2016 "Intelligent communications system based on integrated infrastructure, with dynamic display and alerting - SICIAD".

References

1. Wortmann, F., Flüchter, K.: Internet of Things - technology and value added. Bus. Inf. Syst. Eng. **57**(3), 221–224 (2015)
2. Gubbia, J., Buyyab, R., Marusica, S., Palaniswamihuang, M.: Internet of Things (IoT): a vision, architectural elements, and future directions. Future Gener. Comput. Syst. **29**(7), 1645–1660 (2013)
3. Qian, L., Luo, Z., Du, Y., Guo, L.: Cloud computing: an overview, cloud computing. In: Proceedings of The First International Conference CloudCom 2009, Beijing, China, 1–4 December 2009, pp. 626–631 (2009)
4. Buyyaa, R., Yeoa, C.S., Venugopala, S., Broberga, J., Brandic, I.: Cloud computing and emerging IT platforms: vision, hype, and reality for delivering computing as the 5th utility. Future Gener. Comput. Syst. **25**(6), 599–616 (2009)
5. Schmidhuber, J.: Deep learning in neural networks: an overview. Neural Netw. **61**, 85–117 (2015)
6. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature **521**(7553), 436–444 (2015). https://doi.org/10.1038/nature14539
7. Miorandia, D., Sicarib, S., De Pellegrinia, F., Chlamtaca, I.: Internet of Things: vision, applications and research challenges. Ad Hoc Netw. **10**(7), 1497–1516 (2012)
8. Botta, A., Donato, W., Persico, V., Pescapé, A.: Integration of cloud computing and Internet of Things: a survey. Future Gener. Comput. Syst. **56**, 684–700 (2016)
9. Lane, N.D., Bhattacharya, S., Georgiev, P., Forlivesi, C., Kawsar, F.: An early resource characterization of deep learning on wearables, Smartphones and Internet-of-Things devices. In: Proceedings of the 2015 International Workshop on Internet of Things Towards Applications, 01 November 2015, Seoul, South Korea (2015). https://doi.org/10.1145/2820975.2820980

10. Ma, X., Yu, H., Wang, Y., Wang, Y.: Large-scale transportation network congestion evolution prediction using deep learning theory. PLoS ONE **10**(3), e0119044 (2015). https://doi.org/10.1371/journal.pone.0119044
11. Porter, M.E., Heppelmann, J.E.: How smart, connected products are transforming competition. Harvard Bus. Rev. **92**(11), 64–88 (2014)
12. Weber, R.H.: Internet of Things – new security and privacy challenges. Comput. Law Secur. Rev. **26**(1), 23–30 (2010)
13. Subashini, S., Kavitha, V.: A survey on security issues in service delivery models of cloud computing. J. Netw. Comput. Appl. **34**(1), 1–11 (2011)
14. Zissis, D., Lekkas, D.: Addressing cloud computing security issues. Future Gener. Comput. Syst. **28**(3), 583–592 (2012)
15. Sicaria, S., Rizzardia, A., Griecob, L.A., Coen-Porisini, A.: Security, privacy and trust in Internet of Things: the road ahead. Comput. Netw. **76**, 146–164 (2015)
16. Yana, Z., Zhangc, P., Vasilakos, A.V.: A survey on trust management for Internet of Things. J. Netw. Comput. Appl. **42**, 120–134 (2014)
17. Romana, R., Zhoua, J., Lopezb, J.: On the features and challenges of security and privacy in distributed Internet of Things. Comput. Netw. **57**(10), 2266–2279 (2013)
18. Poschmann, A.Y.: Lightweight cryptography: cryptographic engineering for a pervasive world. Ph.D. thesis (2009)
19. Masanobu, K., Moriai, S.: Lightweight cryptography for the Internet of Things. Sony Corporation (2008). https://pdfs.semanticscholar.org/9595/b5b8db9777d5795625886418d38864f78bb3.pdf
20. Manifavas, C., Hatzivasilis, G., Fysarakis, K., Rantos, K.: Lightweight cryptography for embedded systems – a comparative analysis. In: Garcia-Alfaro, J., Lioudakis, G., Cuppens-Boulahia, N., Foley, S., Fitzgerald, William M. (eds.) DPM/SETOP -2013. LNCS, vol. 8247, pp. 333–349. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-54568-9_21
21. Arseni, S., Mițoi, M., Vulpe, A.: PASS-IoT: a platform for studying security, privacy and trust in IoT. In: 11th International Conference on Communications (COMM 2016), Bucharest, Romania, 9–11 June 2016 (2016). ISBN: 978-1-4673-8196-3
22. Eisenbarth, T., Sandeep, K.: A survey of lightweight-cryptography implementations. IEEE Des. Test Comput. **24**(6), 222–533 (2007)
23. Panasayya, Y., Kaps, J.-P.: Lightweight cryptography for FPGAs. In: 2009 Proceedings of International Conference on Reconfigurable Computing and FPGAs, ReConFig 2009, pp. 225–230. IEEE (2009)
24. Cédric, H., Kamel, D., Regazzoni, F., Legat, J.-D., Flandre, D., Bol, D., Standaert, F.-X.: Harvesting the potential of nano-CMOS for lightweight cryptography: an ultra-low- voltage 65 nm AES coprocessor for passive RFID tags. J. Cryptograph. Eng. **1**(1), 79–86 (2011)
25. Aydin, A., Gulcan, E., Schaumont, P.: SIMON says: break area records of block ciphers on FPGAs. IEEE Embed. Syst. Lett. **6**(2), 37–40 (2014)
26. Vivek, V., Shila, D.M.: High throughput implementations of cryptography algorithms on GPU and FPGA. In: 2013 IEEE International Instrumentation and Measurement Technology Conference (I2MTC), pp. 723–727. IEEE (2013)
27. Vulpe, A., Arseni, Ș.-C., Marcu, I., Voicu, C., Fratu, O.: Building a unified middleware architecture for security in IoT. In: Rocha, Á., Correia, A.M., Adeli, H., Reis, L.P., Costanzo, S. (eds.) WorldCIST 2017. AISC, vol. 570, pp. 105–114. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-56538-5_11