Green Security Plugin for Pervasive Computing using the HADAS toolkit

Daniel-Jesus Munoz, José A. Montenegro, Mónica Pinto, Lidia Fuentes
CAOSD Group, Departamento de Lenguajes y Ciencias de la Computación,
University of Málaga, Andalucía Tech, Málaga, 29071, SPAIN
Email: {danimg,monte,pinto,lff}@lcc.uma.es

***Abstract*—Energy is a critical resource in pervasive computing devices. However, information about energy consumption is not directly accessible through software development environments, making it difficult to reuse the knowledge provided by existing energy-consumption experimental studies. To address this limitation, this paper presents a solution to enrich Android Studio with energy consumption information. We have developed a Green Security Plugin that provides energy-aware information to developers that make use of Android Security API. This plugin has been developed taking advantage of the functionalities provided by the HADAS toolkit. HADAS is a repository of energy consuming concerns in which researchers can store the energy measures obtained during their experimental studies and developers can perform a sustainability analysis to make green design/implementation decisions.**

1. Introduction

Energy is critical in mobile pervasive computing devices, which usually have scarce hardware resources [1]. For this reason, it is highly important that software developers are aware of the impact that their design and implementation decisions have on the energy expenditure of pervasive computing applications [2], [3]. Although there have recently been many experimental studies [4], [5], [6], [7], [8], [9], [10], [11] providing information about the energy consumed by different programming APIs, frameworks, etc., this information is not directly accessible through existing software development environments. This means that energy information can go unnoticed by software developers, making it difficult reusing this knowledge in their applications [2], [3].

A straightforward solution to make this information accessible to software developers is to include it into integrated development environments (IDEs). It is desirable that software developers can be aware of energy expenditure information while programming their applications, without any extra effort. Our motivating scenario is a software developer that is using Android Studio to develop an application that requires security. She, the developer, decides to enable a "green security plugin" that enhances the Android security API[1] with energy consumption information for different Android security primitives. Then, when she is coding the application and writes *KeyPairGenerator.getInstance("DSA","SC",* to get an instance of the DSA algorithm implemented by the SC provider, the Green plugin will provide energy information about that particular configuration, as well as information about other alternative configurations that may be more energy efficient. Moreover, if some of the algorithm's parameters are not explicitly provided (e.g. the key size in this example), the "green security plugin" will provide information as to which combination of provider/key allows the generation of the greenest configuration for the DSA algorithm.

There are approaches that focus on how to design APIs to make them more usable [12]. Others, focus not only on usability but on other quality attributes like security, considering the trade-off between usability and security when designing the API [13]. In our approach, we need displaying information to software developers, which is more related with implementing the APIs than with designing, since we measure the energy consumption of concrete APIs implementations. However, the final goal is similar as we wish to provide information that may help software developers to make more sound decisions about the different alternatives provided by the API considering energy, usability, or any other quality attribute. Concretely, in this paper we focus on providing energy-awareness for the Android Security API.

In this paper we present a Green Security Plugin for Android Studio providing energy information if the Android Security API is used. Three requirements were identified:

- **Req 1. Measure the energy consumption.** A systematic experimental study to measure the energy consumption of all the primitives provided by the API must be carried out. This study has to take into account all the variability of the different security primitives, –i.e. there are different providers for the Android Security API, key generation algorithms can be used with different key sizes, and cipher algorithms can be used with different modes, padding and key sizes. *We have already performed this experimental study and the information is available at http://www.lcc.uma.es/~monte/CryptoConsumption*
- **Req 2. Use of a repository with support for sustainability analysis.** The large amount of information gathered during the experimentation phase has to be formatted and stored so that it can be easily

1. https://developer.android.com/reference/java/security/package-summary.html

other hand, it is a common practice that some APIs do not offer public implementation, i.e. IAIK cryptographic APIs. In this situation the usability of the interface may indeed be appropriate and even the implementation may correspond to expected functionality, but its execution produces an elevated energy consumption. Nowadays, developers have tools to check API usability and implementation when available. However, there are not tools like the plugin presented in this paper for improving and in software development.

6. Conclusions

In this paper, we presented a prototype of a plugin for Android Studio, which supports software developers in making more sound decisions regarding the energy consumption of the security primitives used in their applications. Specifically, we have enhanced the Android Security API so that software developers can receive contextual help about energy consumption information previously calculated and stored in HADAS repository. This process is transparent to the software developer who can continue to program their applications in the usual way. The one thing to do is just activate our Green Security plugin, so the information is provided as they are writing the code in Android Studio.

Our next steps follow two different directions. On the one hand, more experimental data has to be gathered and integrated into the HADAS toolkit to provide energy information for a larger variety of pervasive devices. The main idea is to upgrade the plugin, so that it can provide not just 'recommendations', but also precise information for concrete devices in which the applications will run. Additionally, a fully functional version of the plugin will be developed to include all the existing security profiles for Android. The final goal is to provide a highly configurable plugin that can be adapted to different users' levels of expertise regarding energy consumption and to different sustainability analysis scenarios. Information on the performance of cryptographic primitives and the trade-off between energy and performance will also be incorporated into the plugin.

Acknowledgments

Work supported by the projects Magic P12-TIC1814 and HADAS TIN2015-64841-R (co-financed by FEDER funds).

References

[1] Q. Li and M. Zhou, "The survey and future evolution of green computing," in *Proceedings - 2011 IEEE/ACM International Conference on Green Computing and Communications*, 2011, pp. 230–233.

[2] C. Pang, A. Hindle, B. Adams, and A. Hassan, "What do programmers know about software energy consumption?" *IEEE Software*, vol. 33, no. 3, pp. 83–89, may 2015.

[3] G. Pinto, F. Castor, and Y. D. Liu, "Mining questions about software energy consumption," *Proceedings of the 11th Working Conference on Mining Software Repositories - MSR 2014*, pp. 22–31, 2014.

[4] A. Banerjee and A. Roychoudhury, "Automated re-factoring of android apps to enhance energy-efficiency," in *Proceedings of the International Conference on Mobile Software Engineering and Systems*. New York, NY, USA: ACM, 2016, pp. 139–150.

[5] M. Bokhari and M. Wagner, "Optimising energy consumption heuristically on android mobile phones," in *Genetic Improvement 2016 Workshop*, J. Petke, D. R. White, and W. Weimer, Eds. Denver: ACM, Jul. 20-24 2016, pp. 1139–1140.

[6] A. Merlo, M. Migliardi, and L. Caviglione, "A survey on energy-aware security mechanisms," *Pervasive and Mobile Computing*, vol. 24, pp. 77–90, 2015.

[7] D. S. A. Minaam, H. M. Abdual-Kader, and M. M. Hadhoud, "Evaluating the effects of symmetric cryptography algorithms on power consumption for different data types," *International Journal of Network Security*, vol. 11, no. 2, pp. 78–87, 2010.

[8] A. S. Wander, N. Gura, H. Eberle, V. Gupta, S. C. Shantz, University of California, Santa Cruz, and S. M. Laboratories, "Energy analysis of public-key cryptography for wireless sensor networks," in *Third IEEE International Conference on Pervasive Computing and Communications*, vol. 53, no. 9. IEEE, 2005, pp. 324 – 328.

[9] A. Fallis, "Energy Profiles Java Collections Classes," *Journal Chemical Information and Modeling*, vol. 53, no. 9, pp. 1689–1699, 2013.

[10] D. Kim, J.-Y. Choi, and J.-E. Hong, "Evaluating energy efficiency of Internet of Things software architecture based on reusable software components," *International Journal of Distributed Sensor Networks*, vol. 13, no. 1, p. 155014771668273, 2017.

[11] I. Manotas, L. Pollock, and J. Clause, "SEEDS: a software engineer's energy-optimization decision support framework," in *International Conference on Software Engineering (ICSE)*. New York, New York, USA: ACM Press, 2014, pp. 503–514.

[12] B. A. Myers and J. Stylos, "Improving api usability," *Commun. ACM*, vol. 59, no. 6, pp. 62–69, May 2016.

[13] M. Coblenz, J. Sunshine, J. Aldrich, B. Myers, S. Weber, and F. Shull, "Exploring language support for immutability," in *Proceedings of the 38th International Conference on Software Engineering*, ser. ICSE '16. New York, NY, USA: ACM, 2016, pp. 736–747.

[14] D.-J. Munoz, M. Pinto, and L. Fuentes, "HADAS and web services : Eco-efficiency assistant and repository use case evaluation," in *Energy and Sustainability in Small Developing Economies (ES2DE)*. Funchal: IEEE, 2017.

[15] L. Zhang, B. Tiwana, Z. Qian, Z. Wang, R. P. Dick, Z. M. Mao, and L. Yang, "Accurate online power estimation and automatic battery behavior based power model generation for smartphones," in *Proceedings of the Eighth IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis*, ser. CODES/ISSS '10. New York, NY, USA: ACM, 2010, pp. 105–114.

[16] K. Antkiewicz Michal, A. Murashkin, R. Olaechea, J. H. J. Liang, K. Czarnecki, M. Antkiewicz, K. Bąk, A. Murashkin, R. Olaechea, J. H. J. Liang, and K. Czarnecki, "Clafer tools for product line engineering," *Software Product Line Conference*, p. 130, 2013.

[17] A. Castiglione, F. Palmieri, U. Fiore, A. Castiglione, and A. De Santis, "Modeling energy-efficient secure communications in multi-mode wireless mobile devices," *Journal of Computer and System Sciences*, vol. 81, no. 8, pp. 1464–1478, 2015.

[18] D. González, O. Esparza, J. L. Muñoz, J. Alins, and J. Mata, "Evaluation of Cryptographic Capabilities for the Android Platform." Springer, Cham, 2015, pp. 16–30.

[19] M. A. Ferreira, E. Hoekstra, B. Merkus, B. Visser, and J. Visser, "Seflab: A lab for measuring software energy footprints," *Green and Sustainable Software (GREENS)*, pp. 30–37, 2013.

[20] A. Hindle, A. Wilson, K. Rasmussen, E. J. Barlow, J. C. Campbell, and S. Romansky, "GreenMiner: a hardware based mining software repositories software energy consumption framework," *MSR*, 2014.

[21] K. Djemame, D. Armstrong, R. Kavanagh, A. J. Ferrer, D. G. Perez, D. Antona, J. C. Deprez, C. Ponsard, D. Ortiz, M. Macias, J. Guitart, F. Lordan, J. Ejarque, R. Sirvent, R. Badia, M. Kammer, O. Kao, E. Agiatzidou, A. Dimakis, C. Courcoubetis, and L. Blasi, "Energy efficiency embedded service lifecycle: Towards an energy efficient cloud computing architecture," *CEUR Workshop*, vol. 1203, pp. 1–6, 2014.