Table 2: Average computation times of cryptographic operations on secure element

	Key length	Data	Encr.	Decr.
AES	128 bits	128 B	51 ms	54 ms
	256 bits	128 B	59 ms	62 ms
RSA	1024	117 B	53 ms	117 ms
3-DES	192	128 B	56 ms	54 ms

	Operation	Time
SHA-256	Hash of 128 bytes	78 ms
ECDH-192	Key-pair generation	76 ms
	Generate SS	103 ms
	Total	196 ms
RSA-1024	Key-pair generation	1,957 ms

4.2 Cryptographic Operations

The secure element comes with a co-processor that can perform multiple cryptographic operations in hardware. One of the major goal of the open ecosystem is to use this feature to establish a secure channel between the application and the applet. To make a feasible statement on the maximum transfer rate in such a secure channel, we measured not only the transfer speed but also the performance limitations of those cryptographic operations on the actual hardware.

Table 2 lists the result of the computation measurements for AES, RSA, 3-DES, SHA-256 and ECDH on the chip. The fastest algorithm for encryption and decryption of 128 bytes of data is AES-128. However, the other two operations gave similar encryption time for the selected key lengths. From the results we can also see that RSA key generation is significantly slower and ECDH is therefore the better choice for asymmetric cryptography. In terms of speed we can state that the secure element is able to encrypt at around 2.51 kB/s with AES-128 and at around 2.169 kB/s with AES-256 and decrypt at 2.37 kB/s and 2.06 kB/s respectively.

5. DISCUSSION

The performed measurements show that there is a significant difference in the transfer speed of the hardware variants. Standard cryptographic operations on the SE are, however, very fast due to the hardware implementation. If we combine the values, we can make a statement on the limitations of an open ecosystem when it comes to user interactions: Assuming for the secure channel between application and applet we use AES-128 to ensure the privacy of the communication. In a use-case where the user wants to encrypt or decrypt data on the SE, we would need a total time of 878 ms (SD) or 179 ms (UICC) for 255 bytes of data (transfer speed + $2 * 128$ bytes AES encryption). We also assume the user is willing to wait a maximum time of one second for the execution of an operation. The maximum data that could be used for such an open ecosystem without annoying the user is around 290 bytes for SD and 1,424 bytes UICC.

While this data rate might be a limitation for applications with high speed requirements, we can assume that the performance would be sufficient for applications like an account manager, a password storage or an access control system.

6. CONCLUSION

In this paper and in our demonstration we present a concept and prototype implementation of our open ecosystem with multiple example applications like a password manager, applet manager, cryptography tester and a performance tester. We will also demonstrate our prototype of a secure element emulator and the transparent access to all possible SE variants on an Android phone.

Future work will consist of further investigation on requirements for an open ecosystem of embedded tamper resistant hardware. We will especially focus on finding solutions to all tasks listed in section 2 to achieve our main goal of setting up a trusted environment for security-critical as well as other applications on mobile devices.

7. ACKNOWLEDGMENTS

This work has been carried out within the scope of u'smile, the Josef Ressel Center for User-Friendly Secure Mobile Environments. We gratefully acknowledge funding and support by the Christian Doppler Gesellschaft, A1 Telekom Austria AG, Drei-Banken-EDV GmbH, LG Nexera Business Solutions AG, and NXP Semiconductors Austria GmbH. The authors would also like to thank Endalkachew Asnake for the implementation of the transfer and cryptography tester.

8. REFERENCES

[1] R. Anderson and M. Kuhn. *Low cost attacks on tamper resistant devices*, pages 125–136. 1998.

[2] N. Ben-Asher, N. Kirschnick, H. Sieger, J. Meyer, A. Ben-Oved, and S. Möller. *On the need for different security methods on mobile phones*, pages 465–473. MobileHCI '11. ACM, 2011.

[3] S. Höbarth and R. Mayrhofer. A framework for on-device privilege escalation exploit execution on android. *Proceedings of IWSSI/SPMU*, 2011.

[4] ISO. *ISO/IEC-7816: Part 4: Interindustry commands for interchange*. International Organisation for Standardisation, Geneva, Switzerland, 2005.

[5] S. Khan, M. Nauman, A. Othman, and S. Musa. *How secure is your smartphone: An analysis of smartphone security mechanisms*, pages 76–81. 2012.

[6] P. Kocher, J. Jaffe, and B. Jun. *Differential power analysis*, pages 388–397. 1999.

[7] C.-T. Li and M.-S. Hwang. An efficient biometrics-based remote user authentication scheme using smart cards. *Journal of Network and Computer Applications*, 33(1):1–5, Jan 2010.

[8] G. Madlmayr, J. Langer, C. Kantner, and J. Scharinger. *NFC Devices: Security and Privacy*, pages 642–647. 2008.

[9] T. Mantoro and A. Milisic. *Smart card authentication for Internet applications using NFC enabled phone*, pages D13–D18. 2010.

[10] C. Miller. Mobile attacks and defense. *IEEE Security & Privacy*, 9(4):68–70, 2011.

[11] W. Rankl and W. Effing. *Smart Card Handbook*. Jun 2010.

[12] P. Urien and C. Kiennert. *A new cooperative architecture for sharing services managed by secure elements controlled by android phones with IP objects*, pages 404–409. 2012.

DOI 10.1145/2536853.2536947