Key2Share for Authentication Services

Christoph Busold[1], Alexandra Dmitrienko[2], Christian Wachsmann[1]

[1]Technische Universität Darmstadt/CASED, Germany
{christoph.busold,christian.wachsmann}@trust.cased.de

[2] Fraunhofer SIT Darmstadt, Germany
alexandra.dmitrienko@sit.fraunhofer.de

Key2Share is a smartphone-based solution for authentication services that allows users to manage their access rights to different physical resources on their mobile device. The system is applicable to various application scenarios, such as access control to enterprise facilities, hotel rooms, houses and cars. Key2Share complies with high security standards by providing reliable protection for keys in transit and on the mobile platform. Particularly, it employs modern cryptography to protect keys in transfer and leverages secure hardware (such as smartcards) for storing and handling cryptographic secrets in isolation from untrusted code, such as mobile operating system and apps.

1 Introduction

Existing access control systems for physical resources, such as buildings and rooms typically rely on physical access tokens, such as mechanical keys, smartcards, key fobs or proprietary access control tokens. All these systems require some kind of physical token, such as a smartcard, a key fob or a proprietary token. Typically, each application uses its own token so that users usually have multiple access tokens. Further, the delegation of access rights to other users in these systems typically requires issuing a new physical token for that user.

Smartphone-based access control systems can greatly enhance user experience compared to traditional access control solutions. They combine multiple access tokens in one single device, the smartphone, removing the need for the user to carry them. Tokens stored on a lost or stolen smartphone can be easily revoked and new tokens can be issued remotely, e.g., over the Internet, minimizing the time until the user gets a replacement key. Moreover, users can easily delegate some of their access rights to other users while the key owner keeps full control of the delegated key, e.g., by defining access control policies and remote revocation.

There is a vast number of applications that could benefit from smartphone-based access control systems, including access control to buildings and rooms in enterprises and hotels, rental cars and car sharing applications.

Access control systems in corporate environments typically have multiple physical resources such as buildings and office rooms as well as a large number of users with different access privileges. These environments require flexible access rights management with support for policy-based access control and revocation. Further, enterprise environments are typically demanding in terms of security and require strict control over the key management. Smartphone-based access control systems such as Key2Share can fulfill all these requirements and provide more flexibility than smartcard-based systems which are typically used in these

environments today. For instance, a company could easily grant visitors access rights to the parking garage and a meeting room only for the duration of their stay.

Another widespread application of access control systems is hotels which require a highly dynamic and flexible assignment of access rights to hotel guests and personnel. Smartphone-based access control solutions such as Key2Share greatly enhance the experience of hotel guests. As part of the booking process, guests could receive an electronic room key in advance, which is valid only for the duration of their stay. Hence, guests do not need to check in and can directly proceed to their room on arrival.

An emerging application that could benefit from smartphone-based access control systems is car sharing. While existing car sharing systems typically use smartcards to unlock the car doors or a safe containing the car key, they still use the classical car key fob to unlock the immobilizer and to start the engine. Smartphone-based access control systems promise to enhance user experience in car sharing, car rental and fleet management applications since they allow the user to unlock and to start the car using only his smartphone.

2 Key2Share System

Key2Share is a smartphone-based access control solution which allows usage of a smartphone as a key ring. It can be used in a vast number of applications, including access control to buildings and rooms in enterprises, hotels and multi-tenant buildings, rental cars and cars in car sharing applications, public lockers and post parcel stations. House keys, office keys, garage keys and even car key fobs – all of them can be replaced by electronic access tokens stored in a smartphone. In the following we will introduce key features of Key2Share, describe its system architecture and usage scenarios.

2.1 Key2Share Features

Access control. With Key2Share smartphones can be used to authenticate to and unlock resources, such as a door or car immobilizer, by just holding the phone close to the lock.

Policy-driven access. Key2Share supports different access policies for different users. Access rights can be limited to a specific time frame (e.g., not before and/or not after) or be given for a limited number of times.

Advanced access rights management. Access rights are managed by a central entity (e.g., by the IT department of an enterprise) which specifies access rights of different users to different locks. Users can have different roles, for example, it is possible to differentiate between employees of the enterprise and temporary guests. Access rights can be downloaded by users remotely from the online server.

Delegation of access rights. Key2Share allows users to easily share their access rights with others by creating delegated keys. Delegated keys support policies, i.e., key owners can specify which keys can be delegated and how long they will be valid.

Offline locks. Key2Share supports offline mode of operation for locks, hence, there is no need to connect them to networks, such as LAN or GSM. This lowers deployment costs and makes Key2Share suitable for scenarios where an online connection cannot be easily provided (e.g., for cars).

Battery-powered locks. Key2Share utilizes lightweight cryptographic protocols based on symmetric cryptography which do not require extensive computations on locks. Hence, locks can be battery powered, which makes them independent from wired power supplies and easies installation.

High availability: Key2Share allows users to use their keys even when their smartphone has no online connection. Downloaded or delegated keys can be used immediately with no need to reprogram the locks.

2.2 Key2Share System Architecture

The Key2Share system architecture (Figure 1) involves several entities: an issuer, registered and delegated users, and resources.

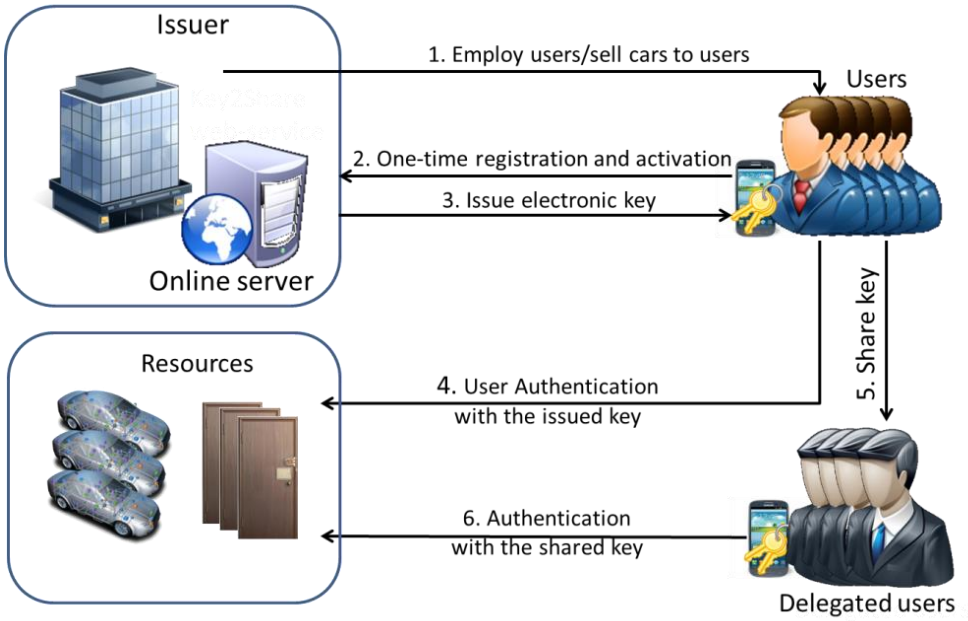

Figure 1. Key2Share System Architecture

Issuer: The issuer is the owner of all resources. This can be for example the enterprise IT department or any other third party. The issuer creates access tokens (electronic keys) for the registered users.

Registered users: A registered user is a user who has registered with the issuer, e.g., an enterprise employee. Registered users can share their keys with other users and guests.

Delegated users: A delegated user is a user who received a shared key from a registered user. Delegated users can be registered with the issuer but cannot share keys delegated to them.

Resource: A resource is a physical object (e.g., the door to a room) protected by the access control system. Registered and delegated users can use their keys to authenticate to and to unlock a resource, e.g., to get access to a room.

Access rights of users are expressed in a form of cryptographic tokens (or electronic keys). Access tokens of registered users are generated by the issuer and transferred to the smartphone of the corresponding registered user during the key issuing procedure. Access tokens of delegated users are generated by the registered users and transmitted to the delegated users during the key sharing (delegation) procedure.

2.3 Usage Scenarios

The Key2Share system is initialized by the issuer, who sets up all resources (e.g., door locks) and the backend system. To use the system, a user must either register with the issuer (registered user) or get a key shared with a registered user (delegated user).

User registration. Frequent users of Key2Share, such as the employees in a corporate environment must register with the issuer, e.g., the IT department of the company, for being able to download their access rights (or keys).

Technically, during registration the smartphone of the user and the issuer exchange authentication and encryption keys that are used later to ensure that only this particular smartphone can access the tokens of this user.

Key issuing. After having registered with the issuer, the user can access and download his access tokens (electronic keys) from the backend system to his smartphone.

Technically, an encrypted authentication and delegation key are downloaded that are used later to authenticate the user to the resource and to securely transfer the access rights to a delegated user, respectively.

Authentication. To authenticate and unlock a resource, e.g., a door, the user just needs to hold his smartphone close to the door lock. The smartphone and the door will automatically run an authentication protocol in the background, which is fully transparent to the user. This protocol uses the authentication key encrypted in the token previously downloaded from the issuer to ensure that only a user authorized to access the resource can actually unlock it.

Key sharing (delegation). Registered users can share their access rights with guest users that are not registered with the issuer and other registered users. The registered user must start the delegation process by selecting the key to be shared as well as the person to share the key with. Optionally, he specifies restrictions on the use of the shared key, such as an expiry date.

The shared key can be transferred to the delegated user's smartphone in different ways. One approach is to use a QR-code that is displayed on the screen of the registered user's phone and scanned with the camera of the delegated user's phone. Alternatively, the key can be transferred using any other technology, such as email, NFC, SMS, instant messaging or the like.

Technically, the phone of the registered user generates a token that contains an authentication key for the delegated user's phone encrypted with the delegation key of the token of the registered user. The phone of the delegated user later uses this authentication key to unlock the resource.

3 Platform Security Framework

Key2Share offers different approaches for protection of the security-sensitive Key2Share application on the smartphone platform, starting from a pure software solution which leverages multi-layer software-based security architecture and does not require hardware security anchors, to a solution which makes use of secure hardware (such as smart cards) for the protection of cryptographic secrets on the device. The latter approach provides higher security compare to pure software-based solutions, and, hence, will be our particular focus in this paper.

3.1 Security Requirements

Mobile platforms typically host a mobile operating system and many third party applications coming from untrusted sources that can potentially be malicious or compromised. Hence, it is essential to protect security-sensitive code and data of the Key2Share application (such as keys) from untrusted code. Particularly, we specify the following security requirements:

R1: Secure storage. Security-sensitive data should not be accessible by untrusted software components while stored on the platform.

R2: Isolation. Security critical code operating on security-sensitive data must be trusted and isolated from the untrusted code, such as an operating system and third party applications.

Further, it is necessary to ensure that the security sensitive operations (e.g., authentication and delegation) are triggered by the user rather than by malware. Moreover, advanced use cases, such as delegation and policy-based access control, rely on security-critical user inputs, such as passwords and user-defined access-control policies (e.g., the validity of the delegated key). Hence, we formulate an additional security requirement:

R3: Secure user interface. The user (either registered or delegated) should be able to securely communicate with the security-sensitive trusted code.

To fulfill requirements R1 and R2, Key2Share leverages the general purpose secure hardware available for commodity mobile platforms. Particularly, it utilizes a hardware-based trusted execution environment (TrEE) which is used to execute security-critical code in isolation from the operating system and other applications. Further, commodity TrEEs typically provide secure storage in a form of dedicated on-chip memory.

However, a secure user interface can be provided only by certain types of secure hardware. We will provide an overview of available secure hardware for mobile systems and discuss their features in Section 3.3.

3.2 Platform Security Architecture

The mobile platform security architecture utilized by Key2Share is depicted in Figure 2. The execution environment of the mobile platform is divided into two independent worlds: An untrusted host and a trusted execution environment TrEE. The host runs on the general purpose processor of the mobile device, while the TrEE is established on top of secure hardware. This secure hardware can be either embedded into the smartphone or attached to the mobile device via the standard communication interfaces, which does not require any changes to commodity platforms.

Depending on the type of secure hardware used, the TrEE can either have a direct (secure) connection to the NFC chip or must rely on the untrusted operating system to handle NFC (illustrated as dashed line in the Figure 2). In contrast, the WiFi or the mobile network interface used for the communication with, e.g., an issuer, is always managed by the operating system. Further, the NFC chip can be either controlled by the smartphone OS or have a direct connection to the TrEE (indicated by dashed lines in Figure 2).

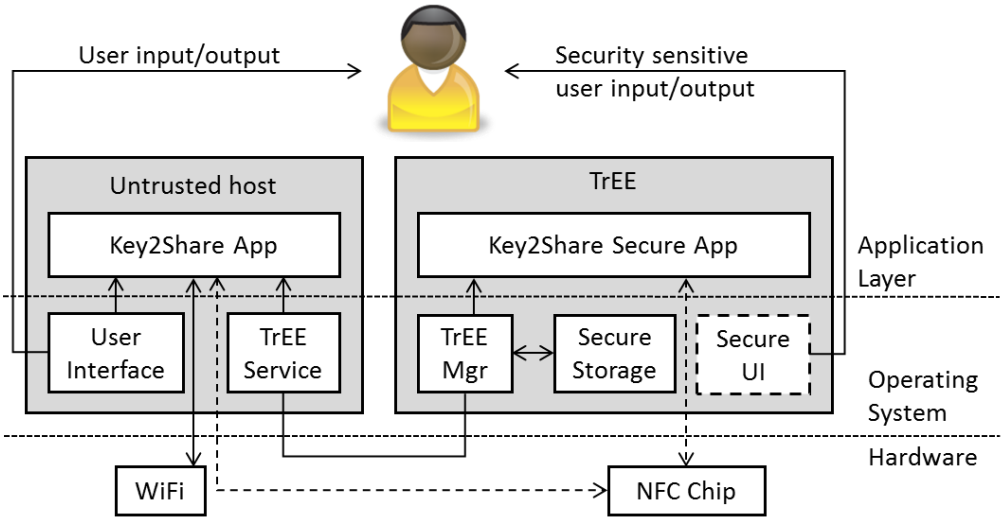

Figure 2. Mobile platform security architecture

The Key2Share functionality is split between the Key2Share App and the Key2Share Secure App. The Key2Share App manages the access rights and handles different usage scenarios (such as registration, delegation and authentication), while the Key2Share Secure App is only invoked to perform the security-critical computations, e.g., cryptographic operations like encryption, decryption, message authentication code (MAC) calculation and MAC verification. Further, all security-sensitive data, e.g., cryptographic keys used by cryptographic protocols, never leave the TrEE in clear text, but rather are securely stored within the Secure Storage component of TrEE.

Key2Share App and Key2Share Secure App communicate via a channel established between both execution environments (host and TrEE). The channel is handled by the TrEE Service and TrEE Mgr components, which are responsible for multiplexing the communication between the different applications running on the phone. TrEE Mgr additionally manages the access of different TrEE applications to the Secure Storage component, so that other applications, possibly residing within TrEE, cannot access the cryptographic secrets of Key2Share Secure App.

User input is handled by two components in the system: User Interface and Secure UI. User Interface is provided by the host and is used for the ordinary interaction with the user, while Secure UI is provided by TrEE and utilized to handle all security-sensitive inputs, such as passwords and access control policies. Secure UI is customized with a background picture or a unique paraphrase only known to the user and the TrEE, allowing the user to distinguish between User Interface and Secure UI.

Notably, Secure UI can be provided only by certain types of secure hardware, as we discuss later in Section 3. Thus, this component is optional in the Key2Share platform security architecture (indicated by a dashed box in Figure 2). Absence of the secure user interface degrades the security of the key delegation mechanism, as it requires the user to provide security sensitive input to the untrusted host. Particularly, the user has to specify whom the key will be delegated to, and define access policies, e.g., validity periods. Hence, instantiation of the platform security architecture without Secure UI can achieve secure delegation only if the User Interface component is not compromised.

4 Secure Hardware

In the following, we analyze the features of the available secure hardware for smartphones and discuss their advantages and disadvantages for using them to instantiate the Key2Share platform security architecture.

Most known commodity security hardware available for modern smartphones includes ARM TrustZone [ATZ04], MShield [MS08], SIM-cards, embedded secure elements (SE) and secure microSD cards from different vendors [GiDe, CgCa12, Tyfo11]. A comparison of their corresponding features is provided in Table 1. Notably, all considered security hardware provides secure storage and isolation, which satisfies security requirements R1 and R2 specified in Section 3.1. However, only a few types of secure hardware provide a secure user interface (and satisfy R3), and those that do so have other disadvantages, such as being not programmable by third party developers and limited availability.

	Secure Storage	**Isolation**	**Secure User Interface**	**Freely programmable**	**Availability**
ARM TrustZone [ATZ04]	+	+	+	-	Some ARM-based phones
TI M-Shield [MS08]	+	+	-	-	Some phones based on Texas Instruments processors
SIM-card	+	+	-	-	Every phone
Embedded SE	+	+	-	-	Phones with NFC
Secure microSD card [GiDe, CgCa12, Tyfo11]	+	+	-	+	Phones with microSD slot

Table 1. Comparison of commodity secure hardware

ARM TrustZone. ARM TrustZone [ATZ04] allows to establish a trusted execution environment (TrEE) that can provide a secure user interface, eventually fulfilling all of our platform-related security requirements. However, TrustZone is available only on a few platforms, such as Apple's iPhone. Further, it is usually deactivated or locked by the phone manufacturer and cannot be used by third party applications running on the phone. Although the TrustZone API is public and software emulators are available, only selected third party developers get access to TrustZone development boards.

M-Shield. M-Shield is available on some phones featuring Texas Instrument processors (e.g., many Nokia phones). However, similarly to TrustZone, it is locked down and not available for programming by third party developers.

SIM-cards. SIM-cards are the most widespread TrEEs and available on every phone. However, SIM-cards are typically closed systems controlled by the network operators. Hence, a solution based on SIM-cards would be available only to customers of the particular network operator controlling the SIM-card.

Embedded secure elements. Embedded secure elements are available on phones featuring an NFC interface, e.g., many Android phones such as the Samsung Nexus S and the Samsung Galaxy Nexus. However, these secure elements are locked down and can only be used with the Google Wallet payment system [GW12].

Secure microSD cards. Promising secure hardware are microSD cards, which are microSD memory cards that include a secure element (e.g., [GiDe, CgCa12, Tyfo11]). They can be used in every smartphone with a microSD card slot. Some of these cards include an NFC chip that uses an NFC antenna integrated in the microSD card [Tyfo11]. Such microSD cards enable solutions that reach a large number of users because they can even be used on phones without an integrated NFC interface. However, although they exist, microSD cards with an integrated NFC antenna are not yet available as commercial product.

Among above discussed secure elements, ARM TrustZone seems to be the most suitable TrEE for the Key2Share platform security architecture (discussed in Section 3), since it satisfies all platform-related security requirements, while all the other TrEE types do not provide a secure user interface. However, developments for TrustZone are currently limited to development boards, thus it is more practical to consider other types of TrEEs for the prototype.

5 Prototyping

Key2Share was prototyped using an NFC-enabled smartphone with a secure microSD card [GiDe], which seems to be the most applicable configuration in practice.

Smartphone. Key2Share was implemented for Android smartphones using an NFC-enabled Samsung Galaxy S3 smartphone. The NFC hardware of the Galaxy S3 comes with a built-in secure element used for the Google Wallet electronic payment system [GW12]. However, as mentioned before, this secure element is locked and cannot be used for custom applications such as our Key2Share Secure application. Therefore, another secure element, particularly a Giesecke & Devrient Mobile Security Card 1.0, was utilized. This is a microSD smart card that allows installation of custom applications. The underlying smart card operating system complies with the Java-Card 2.2.2 and Global Platform 2.2.1 specifications and provides all cryptographic primitives required.

The User Interface component of the platform security architecture is instantiated by the keyboard and display drivers already present in the Android OS. The TrEE Service implementation is based on the smartcard API provided by the Seek-for-Android project [SefA]. It enables access to smartcards via APDUs as defined in ISO7816. The Galaxy S3 stock firmware already contains this smartcard API for the built-in secure element; however, it is not enabled for the Mobile Security card. Therefore, a custom build of CyanogenMod9 [CM9] based on Android 4.0.3 was utilized, which included the smartcard API patches (version 2.3.2).

However, Seek-for-Android plans to release a plugin-in terminal for the Mobile Security Card, which can be used with the existing API and thus would not require a custom firmware.

Key2Share App is implemented as a standard Android app. Further, the functionality of the Key2Share Secure App is realized in a JavaCard applet that uses the secure storage of the smartcard. Moreover, an interface between the smartcard and the Key2Share application is implemented based on the Seek-for-Android API which communicates with the Java Card operating system.

Lock. For the lock prototype a typical setup for electronic locks was applied. Specifically, an Arduino Uno [AUno] was used, which is a commercial development board based on an 8 bit Atmel AVR microcontroller with 32 KB memory clocked at 16 MHz. The Arduino is connected via a Serial Peripheral Interface (SPI) to an NFC shield [NShi] based on the PN532 controller [PN532]. The lock firmware uses AVR-Crypto-Lib [AVRC], an open source library optimized for AVR microcontrollers, to realize the necessary cryptographic primitives. Furthermore, the NFC library provided with the PN532 NFC controller was adapted so that the NFC hardware emulates a contactless smartcard according to the NFC Forum type 4 [NFCF] and ISO14443-4 specifications [ISOS08].

6 Conclusion

In this paper we presented Key2Share, a smartphone-based solution for different authentication services, and described one possible approach for protection of security-sensitive code and data on a mobile platform. This approach is to leverage secure hardware, particularly, a microSD security card (that can be plugged into a standard microSD card slot of a mobile device), which allows to establish an isolated execution environment on the mobile device. This environment is used to process and to store the cryptographic secrets of the Key2Share application in isolation from the mobile operating system and mobile applications. However, Key2Share is not limited to this particular type of secure element and can be instantiated, e.g., on top of other Java-based secure co-processors, such as NFC-based embedded secure elements or SIM cards, or based on ARM TrustZone.

Literature

1	ARM TrustZone	[ATZ04]
2	J. Azema and G. Fayad. M-Shield mobile security technology: Making wireless secure. Texas Instruments white paper, 2008.http://focus.ti.com/pdfs/wtbu/ti_mshield_whitepaper.pdf	[MS08]
3	Giesecke & Devrient Secure Flash Solutions. The Mobile Security Card SE 1.0 o_ers increased security. http://www:gd-sfs:com/the-mobile-security-card/mobile-security-card-se-1-0/.	[GiDe]
4	Certgate products. cgCard. http://www.certgate.com/wp-content/uploads/2012/09/20131113_cgCard_Datasheet_EN.pdf	[CgCa12]
5	Tyfone. Tyfone to license SideTap MicroSD NFC and Secure Element Card technologies to AboMem, 2011. http://tyfone:com/newsroom/?p=541	[Tyfo11]
6	Google Wallet. http://www.google.com/wallet/, 2012	[GoWa12]
7	CyanogenMod. http://www:cyanogenmod:com/	[CM9]
8	Secure Element Evaluation Kit for the Android platform. http://code.google.com/p/seek-for-android/	[SefA]
9	Arduino. http://www.arduino.cc	[AUno]
10	NFC Shield. Near Field Communication interface for Arduino. http://www.seeedstudio.com/wiki/NFC_Shield	[NShi]
11	PN532 Near Field Communication (NFC) controller. NXP Semiconductors. http://www.nxp.com/products/identification_and_security/reader_ics/nfc_devices/series/PN532.html	[PN532]
12	AVR cryptographic library. Set of cryptographic primitives for Atmel AVR microcontrollers. https://www:das-labor:org/wiki/AVR-Crypto-Lib	[AVRC]
13	Near Field Communication Forum. http://www.nfc-forum.org/home/	[NFCF]
14	International Organization for Standardization. International Standard ISO/IEC 14443-4. Identification cards -- Contactless integrated circuit cards -- Proximity cards. Part 4: Transmission protocol	[ISOS08]

M.Sc. Christoph Busold

M.Sc. Christoph Busold joined CASED as a research assistant and PhD student after obtaining his master degree in computer science from TU Darmstadt in May 2012. He is working at the recently founded Intel Collaborative Research Institute for Secure Computing which focuses on research in IT security for mobile and embedded devices. His research interests are mobile and platform security, including the integration of security hardware like smartcards and trusted execution environments into mobile systems and applications.

Contact
Christoph Busold
Mornewegstrasse 32
D-64293 Darmstadt
GERMANY

Tel. +49 6151 16 7 55 8
Fax. +49 6151 16 7 21 35
E-Mail: christoph.busold@trust.cased.de

M.Sc. Alexandra Dmitrienko

M.Sc. Alexandra Dmitrienko is a research assistant at Fraunhofer Institute for Secure Information Technology in Darmstadt. She obtained her M.Sc. in IT-Security from the Saint-Petersburg State Polytechnical University in Russia. Currently she is pursuing a PhD degree in area of Secure Mobile Computing at TU Darmstadt. Her academic achievements within the PhD program were honored by the Intel Doctoral Student Honor Award. Her research is mainly focused on security aspects of mobile operating systems and secure mobile applications, such as online banking, mobile payments, and ticketing.

Contact
Alexandra Dmitrienko
Rheinstraße 75
D-64295 Darmstadt
GERMANY

Tel. +49 6151 16 7 55 66
Fax. +49 6151 16 7 21 35
E-Mail: alexandra.dmitrienko@sit.fraunhofer.de

Dr.-Ing. Christian Wachsmann

Dr.-Ing. Christian Wachsmann is a postdoctoral researcher at the Intel Collaborative Research Institute for Secure Computing (CRI-SC) at Technische Universität Darmstadt. He received his Ph.D. in Computer Science from Technische Universität Darmstadt and his diploma in IT Security from Ruhr-Universität Bochum in Germany. His research focuses on the formalization and security analysis of security primitives based on physical properties of hardware components, in particular Physically Unclonable Functions (PUFs), and the formalization, development and design of practical cryptographic protocols for mobile and resource-constrained embedded devices, such as device authentication and attestation schemes.

Contact
Christian Wachsmann
Mornewegstrasse 32
D-64293 Darmstadt
GERMANY

Tel. +49 6151 16 7 55 63
Fax. +49 6151 16 7 21 35
E-Mail: christian.wachsmann@trust.cased.de