Your Money or Your Life—Modeling and Analyzing the Security of Electronic Payment in the UC Framework

Dirk Achenbach[2], Roland Gröll[2*], Timon Hackenjos[2], Alexander Koch[1**], Bernhard Löwe[1**], Jeremias Mechler[1], Jörn Müller-Quade[1], Jochen Rill[2*]

[1] Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany
[2] FZI Research Center for Information Technology

Abstract. EMV, also known as Chip and PIN, is the world-wide standard for card-based electronic payment. Its security wavers: over the past years, researchers have demonstrated various practical attacks, ranging from using stolen cards by disabling PIN verification to cloning cards by pre-computing transaction data. Most of these attacks rely on violating certain unjustified and not explicitly stated core assumptions upon which EMV is built, namely that the input device (e.g. the ATM) is trusted and all communication channels are non-interceptable. In addition, EMV lacks a comprehensive formal description of its security.

In this work we give a formal model for the security of electronic payment protocols in the Universal Composability (UC) framework. A particular challenge for electronic payment is that one participant of a transaction is a human who cannot perform cryptographic operations. Our goal is twofold. First, we want to enable a transition from the iterative engineering of such protocols to using cryptographic security models to argue about a protocol's security. Second, we establish a more realistic adversarial model for payment protocols in the presence of insecure devices and channels.

We prove a set of necessary requirements for secure electronic payment with regards to our model. We then discuss the security of current payment protocols based on these results and find that most are insecure or require unrealistically strong assumptions. Finally, we give a simple payment protocol inspired by chipTAN and photoTAN and prove its security.

Our model captures the security properties of electronic payment protocols with human interaction. We show how to use this to reason about necessary requirements for secure electronic payment and how to develop a protocol based on the resulting guidelines. We hope that this will facilitate the development of new protocols with well-understood security properties.

Keywords: EMV, Universal Composability, Security Models, Human-Server-Interaction, Electronic Payment

[*] This work was supported by grants from the Federal Ministry for Economic Affairs and Energy of Germany (BMWi) for the EDV Project.

[**] This work was supported by the German Federal Ministry of Education and Research within the framework of the projects KASTEL_IoE, KASTEL_SVI and KASTEL_Base in the Competence Center for Applied Security Technology (KASTEL).

1 Introduction

"Your money, or your life!"—surrender your belongings or face death. This threat was used by bandits in England until the 19th century [25]. As people often needed to carry all their valuables with them when traveling, banditry was a lucrative (albeit dangerous) endeavor. Today, electronic money transfer (EMT) systems alleviate the need to have one's valuables at hand, but introduce new threats as well. Instead of resorting to violence, modern thieves may compromise their victim's bank account. Once they are widely deployed, insecure EMT systems are notoriously difficult to transition away from—magnetic stripes are still in use today. The current state-of-the-art payment standard EMV (short for *Europay International, MasterCard and VISA*, also known as "Chip and PIN") improves on this, but falls short of providing a secure solution to payment (or money withdrawal), as shown by its many weaknesses described in literature.

Among these are practical attacks, such as (i) "cloning" chip cards by precomputing transaction messages (so-called "pre-play attacks") [4], (ii) disabling the personal identification number (PIN) verification of stolen cards by intercepting the communication between chip card and point of sale (POS) device [24], (iii) tricking an innocent customer into accepting fraudulent transactions by relaying transaction data from a different POS (so-called "relay attacks") [17].

Upon close examination of these attacks one finds that these issues mainly stem from *two major false assumptions* which are baked into the design of the EMV protocol: (i) that the communication between all protocol participants (e.g. between the chip card and the POS) cannot be intercepted and (ii) that the POS (or the automated teller machine (ATM)) itself is trustworthy. Even though these assumptions are critical for the security of EMV, they are not explicitly stated in the standardization documents [19, 20, 21]. We suggest that this is mainly because EMV has been created by a functionality-focused engineering process in which problems are fixed as they occur and features are added when necessary, rather than a design process that uses formal models and techniques. Modern cryptographic protocols in contrast are designed by first providing a formal description of the protocol, explicitly stating all necessary assumptions and then giving a proof of security. This does not make cryptographic protocols unbreakable, but it does make their potential breaking points explicit. Therefore, we argue that it is necessary to start developing electronic payment protocols by using the same methodology of rigorous formal modeling as has already been established in cryptography.

1.1 Our Contribution

In this work, we give a novel formal model for electronic payment based on the Universal Composability (UC) framework by Canetti [6], which incorporates a stronger, but also more realistic adversarial model than has been used for the design of EMV. We first give a *formal description* of electronic payment which works for both payment at a POS and for the withdrawal of cash at an ATM. Second, we provide an ideal functionality for electronic payment, which captures the desired security guarantees for such protocols. Our model can also be used in the case where one participant is human.

We then prove a set of *general requirements* for designing such protocols. These requirements can act as a guideline for future protocol designers. Based on these results, we argue that a number of current payment systems are insecure already on a conceptual level. Inspired by this analysis, we propose a simple electronic payment protocol which mainly requires secure communication between the bank and the initiator of a transaction. We propose to realize this with a smartphone, as is common in many modern payment protocols. However, unlike these protocols, our protocol can be proven secure if *either* the smartphone *or* the ATM/POS device behaves honestly, whereas all other protocols we analyzed need to trust at least one of them exclusively.

1.2 Related Work

Secure Human-Server Communication. Basin, Radomirovic, and Schläpfer [3] give an enumeration of minimal topologies of channels between a human (restricted in its abilities), a trusted server, a possibly corrupted intermediary and a trusted device, that realize an authenticated channel between the human and the server. Our work differs in two main aspects: Their model uses either fully secure or untrusted channels only and cannot account for just authenticated or just confidential communication, which is important in our setting due to the presence of CCTV cameras or shoulder-surfing. For example, we assume that everything displayed at the ATM or a user's smartphone is not confidential, while entering a PIN at the PIN pad can be done in a confidential way, by suitably covering the pad in the process. Second, our model is based on the UC framework, which gives stronger guarantees and composability, as well as security for concurrent and interleaved execution, compared to the stand-alone setting they consider.

Alternative Hardware Assumptions. As we will see later in Section 2.2, the *confirmation of payment information* by the user is an important sub-problem we aim to solve for achieving secure payment. A possible solution is "Display TAN" [5] providing a smartcard with a display to show the transaction data. Smart-Guard [15] uses such smartcards with a display together with an encrypting keyboard fixed to the card to achieve a functionality which may be used for payment. These strong hardware assumptions allow for flexible trust assumptions, accounting for several combinations of trusted/hacked status of the involved devices. Their protocol comes with a formally verified security proof, albeit not in the UC framework. For our construction we do not propose a new kind of hardware device, but rely on the user's smartphone.

Ecash and Cryptocurrencies. Besides human-server payment protocols, there is also electronic cash, first invented by [9], and modern decentralized cryptocurrencies, such as Bitcoin [22], which can be used to transfer money. In general, these have very different design goals, as they care to establish an electronic money system with certain anonymity/pseudonymity properties, without the possibility to double-spend and in particular, without a trusted bank. In contrast, we are concerned with the authenticated transmission of the transaction data from a human user to the bank. To the best of our knowledge, there is no UC-based model of electronic payment as presented in our work.

EMV. EMV is not only a single payment protocol, but a complete protocol suite for electronic payment (cf. [19, 20, 21]). Protocols that are EMV-compliant might just implement the EMV interface while using another secure protocol. This means that, While there are multiple attacks against the EMV *payment protocol*, not every protocol with EMV in its name is automatically insecure. In addition to the attacks mentioned previously, there are other attacks as described by Chothia et al. [10] and Emms et al. [18].

Anderson et al. [2] discuss whether EMV is a monolithic system reducing the possibilities for innovation. Since we use the UC framework for our model, we inherently support non-monolithic, modular systems. Sub-protocols that UC-realize each other can be exchanged for one another. Furthermore, [2] explore the possibility to use smartcards (as used by EMV) for other applications. Following a similar goal, we give a formalization of signature cards within our model in the full version [1] and show limits to using such cards.

Degabriele et al. [14] investigate the joint security of encryption and signatures in EMV using the same key-pair. A scheme based on elliptic curves (as it is used in EMV) is proven secure in their model. However, as they conclude, their proof does not eliminate certain kinds of protocol-level attacks. Cortier et al. [13] present an EMV-compliant protocol using trusted enclaves and prove the security of their protocol using TAMARIN [29]. Both approaches lack the modularity, composability and security for parallel execution provided by the UC framework.

1.3 The (Generalized) Universal Composability Framework

The Universal Composability (UC) framework, introduced by Canetti in 2001 [6], is a widely established tool for proving the security of cryptographic protocols based on the real-world–ideal-world paradigm. The desired security properties of a protocol are described in terms of a so-called *ideal functionality*, which can be seen as an incorruptible third party carrying out the desired task by definition. The ideal functionality *explicitly* captures the allowed influence an adversary can have and the knowledge he can gain during an execution of the protocol. Informally, a protocol π is said to UC-realize an ideal functionality $\mathcal{F}$ if there is no interactive distinguisher $\mathcal{Z}$ (the so-called *environment*) that can distinguish between the execution of π and the execution of (the ideal protocol of) $\mathcal{F}$.

The framework is specifically well-suited for our case, as it already incorporates an adversary that can control all communication between the protocol parties. If one wants to deviate from this (e.g. when secure communication is available) one must explicitly add new functionalities for communication to the model (so-called *hybrid functionalities*), making the security assumptions of the protocol explicit.

The UC framework's security definition does not capture shared state between several protocol instances. Canetti et al. [7] proposed an extension—the so-called Generalized Universal Composability (GUC) framework—which introduces globally shared functionalities. They can be used by multiple protocols, allowing to share state between different executions of protocols. This extension can be used to model smartcards as used in EMV, allowing us to capture e.g. pre-play attacks in our model. One of the main advantages of the (G)UC framework is that it, unlike stand-alone security models, brings a strong composition theorem. This allows for breaking protocols into smaller components and proving their security individually. A comprehensive description of the framework and its extension can be found in the the full version [1].

2 A Formal Model for Electronic Payment

As a basis for our model, observe the process of withdrawing cash at an automated teller machine (ATM). First, there is the bank and its customer, Alice. Second, there is the money dispensing unit inside the ATM. Assuming authenticated communication from Alice to the bank and from the bank to the money dispensing unit, secure payment is easy: Alice communicates the amount of cash she needs and the identity of the money dispensing unit she expects to receive the cash from. The bank then instructs the money dispensing unit to dispense the money. However, Alice is a human and therefore cannot perform cryptographic operations required for a classical channel establishment protocol. Thus, Alice needs another party which offers a user interface to her and communicates with the bank, namely an ATM.

This does not only apply to cash withdrawal but can be extended to electronic money transfer (EMT) in general. To this end, think of Alice as the *initiator* of a transaction and the money dispensing unit as the *receiver*. The process of money withdrawal can now be framed as a payment of money from Alice's account to the account of the money dispensing unit (which, upon receiving money, promptly outputs cash) using the ATM as an (input) *device*. The same works for the point of sale: here, the device's owner (e.g. the supermarket) is the receiver.

Regarding our adversarial model, as discussed earlier, we make no assumption about the trustworthiness of the ATM whatsoever and do assume that the adversary has control over all communication. We do make certain assumptions regarding the trustworthiness of different protocol participants. First, we assume the money dispensing unit (or receiver in general) to be trusted. If it is under adversarial control, the adversary could simply dispense money at will. Second, since our work focuses on the challenges that arise from the interaction of humans with untrustworthy devices over insecure communication, we do not model the bank's book-keeping and therefore assume the bank to be incorruptible. Third, for reasons of simplicity, our model only considers a single bank, even though in practice most transactions involve at least two banks. This is justified, however, as banks in general can communicate securely with each other.

2.1 Modeling Electronic Payment in the UC framework

In the following, to simplify the model, we consider the case of *static corruption*, where parties may only be corrupted prior to protocol execution. Extending our work to adaptive corruption is left for future work.

We denote the set of initiators as S_{I}, the set of receivers as S_{R}, the set of devices as S_{D} and the bank as B. We also define a mapping $D\colon S_{\mathrm{R}} \to S_{\mathrm{D}}$ of receivers to single devices ($D(R)$) to explicitly name which device belongs to which receiver.

In order to model the adversary's probability of successfully attacking credentials like PINs, we introduce a parametrized distribution $\mathcal{D}$. Let X denote the event of a successful attack. Then $\mathcal{D} : A \to F_X$ maps a value d (e.g. the amount) from a domain A (e.g. $\mathbb{Q}$) to a probability mass function $f_{d,X} \in F_X$ over $\{\texttt{confirm}, \texttt{reject}\}$. An adversary's success probability of correctly guessing a four-digit PIN chosen uniformly at random with one try could be modeled as follows: $\mathcal{D}(m_\$) = f_X$ for all $m_\$ \in \mathbb{Q}$ with $f_X(\texttt{confirm}) = \frac{1}{10000}$, $f_X(\texttt{reject}) = \frac{9999}{10000}$. $\mathcal{D}$ could also map different $d \in A$ to different $f_{X,d}$, modeling that transactions with small amounts require less protection than ones with bigger amounts. $\mathcal{F}_{\mathcal{D}}$ is the ideal functionality $\mathcal{F}$ parametrized with $\mathcal{D}$. Ideal functionalities may have additional parameters, either implicit or explicit ones passed as arguments, e.g. $\mathcal{F}_{\mathcal{D}}(A, B)$.

In the best possible scenario, ideal payment would work as follows: the initiator submits his desired transaction data to an ideal functionality, which then notifies the bank and the receiver about who paid which amount of money to whom without involvement of the adversary whatsoever. In our adversarial model, no payment protocol realizes this strong ideal functionality: an attacker who controls all communication will at least be able to observe that a transaction takes place, even if he cannot see or change its contents. What is more, such a strict security definition would ignore the fact that in all payment protocols which rely on the initiator being protected by a short secret (like a PIN), an attacker always has a small chance of success by guessing the secret correctly.

Our ideal functionality for electronic payment is thus designed with regards to the following principles: (i) The adversary always gains access to all transaction data. An electronic payment operation can be secure (that is all participants of the transaction get notified about the correct and non-manipulated transaction data) without the transaction data being secret. (ii) The adversary can always successfully change the transaction data at will with a small probability (e.g. if he guesses the PIN correctly). (iii) The payment operation occurs in three stages. In the first stage, the initiator inputs his intended transaction data which the adversary can change at will. This models that a corrupted input device will always be able to change the human initiator's transaction data, even if it will be detected at a later stage. In the second and third stage, the receiver and the bank are notified about the transaction data. The resulting functionality is depicted in Fig. 1.

2.2 Confirmation is Key

Since the human initiator of a transaction can never be sure that an input device correctly processes his transaction data, he needs a way of confirming the transaction data with the bank before the transaction is processed. We formalize this confirmation mechanism within the ideal functionality $\mathcal{F}_{\mathrm{CONF}}$ (specified in

The Ideal Functionality for Electronic Payment $\mathcal{F}_{\text{PAY},\mathcal{D}}(I, B, R)$.

Parametrized by a set of receivers S_{R}, a designated receiver $R \in S_{\text{R}}$, a set of initiators S_{I}, an initiator $I \in S_{\text{I}}$, the bank B and a parametrized distribution $\mathcal{D}$.
Initialize $I' = I$, $R' = R$, $attacked = \texttt{no}$.
Assertion: At any time, $I, I' \in S_{\text{I}}$ and $R, R' \in S_{\text{R}}$. If the assertion is violated, halt.

Phase 1: Collecting Information

1. Upon receipt of message $(\texttt{transfer}, sid, R, m_{\$})$ from I: Send $(sid, I, R, m_{\$})$ to the adversary, receive $(sid, I', R', m'_{\$})$ and output $(\texttt{input-received}, sid, I', R', m'_{\$})$ to B.

Phase 2: Confirmation and Execution

2. Resume upon instruction by the adversary.
3. If I' is honest, $(I', R', m'_{\$}) \neq (I, R, m_{\$})$ and $attacked = \texttt{no}$, halt.
4. Make a public delayed output of $(\texttt{received}, sid, I', m'_{\$})$ to R'.

Phase 3: Ensuring Consistency

5. Resume upon instruction by the adversary and make a public delayed output of $(\texttt{processed}, sid, I', R', m'_{\$})$ to B. Halt upon confirmation by the adversary.

Attack

– Upon receiving an input $(\texttt{attack}, sid)$ in Phase 2 from the adversary, sample an element $b \in \{\texttt{confirm}, \texttt{reject}\}$ according to $\mathcal{D}(m'_{\$})$. If $b = \texttt{confirm}$, set $attacked = \texttt{yes}$, otherwise set $attacked = \texttt{no}$. Return $(\texttt{attack}, sid, attacked)$ to the adversary. Ignore all further $\texttt{attack}$ queries.

Fig. 1. The ideal functionality $\mathcal{F}_{\text{PAY}}$ for electronic payment.

Fig. 2). $\mathcal{F}_{\text{CONF}}$ is a two-party functionality which allows a sender to transmit a message and the receiver of the message to *confirm* or *reject* it. As with the ideal payment functionality, the adversary gets the chance to force a confirmation with a certain probability, modeling the insecurity inherent to real-world protocols which use short secrets. Note that he can always force the confirmation to be rejected.

To realize $\mathcal{F}_{\text{PAY}}$, we need authenticated communication from the bank to the receiver, so that the receiver can be notified of the transaction. For most real-world payment protocols, this authenticated communication is easy to establish, since receivers are electronic devices and not humans. In the case of cash withdrawal, the bank owns the money dispensing unit and can pre-distribute cryptographic keys to establish authenticated communication.

Using $\mathcal{F}_{\text{CONF}}$ and $\mathcal{F}_{\text{AUTH}}$ [6, Sect. 6.3], we propose a protocol π_{PAY} which realizes $\mathcal{F}_{\text{PAY}}$. This protocol is informally depicted in Fig. 3. The comprehensive formal description of the protocol can be found in the full version [1]. Having defined all required protocols and functionalities, we are now ready to state our theorem. For the proof, see the full version [1].

Theorem 1. *Let I, B, R, and $D(R)$ ITMs, where I is human, and B and R are honest. Then, π_{PAY}, informally depicted in Fig. 3, UC-realizes $\mathcal{F}_{\text{PAY},\mathcal{D}}(I, B, R)$ in the $\mathcal{F}_{\text{AUTH}}(B, R), \mathcal{F}_{\text{AUTH}}(R, B), \mathcal{F}_{\text{CONF},\mathcal{D}}(B, I)$-hybrid model.*

Even though this might seem unsurprising at first, this allows us to break down the complexity of realizing $\mathcal{F}_{\text{PAY}}$ into two easier problems: realizing a confirmation mechanism between the initiator and the bank and realizing authenticated communication between the receiver and the bank.

The Ideal Functionality for Confirmation $\mathcal{F}_{\text{CONF},\mathcal{D}}(S, C)$

Parameters: The message sender S, the respective confirmer C and a parametrized distribution $\mathcal{D}$.
Initialize *attacked* = `no`, *initiated* = `no`, *completed* = `no`.

- Upon receiving (`initiate`, sid, C, m) from ITI S, make a public delayed output of (`initiate`, sid, S, m) to C and set *initiated* = `yes`. Ignore all subsequent `initiate` messages.
- Upon receiving (`reply`, sid, S, b) from ITI C when *initiated* = `yes`, *completed* = `no` and $b \in$ {`confirm`, `reject`}: Make a public delayed output of (`answer`, sid, C, b) to S. Upon confirmation from the adversary, set *completed* = `yes` and halt.
- Upon receiving (`force-confirm`, sid) from the adversary, assert that *initiated* = `yes`, *completed* = `no` and *attacked* = `no`. If this holds, set *attacked* = `yes` and sample an element $b \in$ {`confirm`, `reject`} according to $\mathcal{D}(m)$. If b = `confirm`, set *completed* = `yes` and make a public delayed output of (`answer`, $sid, C,$ `confirm`) to S and halt upon confirmation by the adversary. Otherwise, return (`fail`, sid) to the adversary.
- Upon receiving (`force-reject`, sid) from the adversary, assert that *initiated* = `yes`, *completed* = `no` and *attacked* = `no`. If this holds, set *attacked* = `yes` and *completed* = `yes`, make a public delayed output of (`answer`, $sid, C,$ `reject`) to S and halt upon confirmation by the adversary. Otherwise, return (`fail`, sid) to the adversary.

Fig. 2. The ideal functionality for confirmation of messages.

2.3 How Our Model Captures Existing Attacks

One of our main motivations for establishing a new formal model for electronic payment is to make trust assumptions explicit in order to detect unrealistic ones which enable practical attacks like [4], [24] and [17]. Thus, our model needs to be able to capture these kinds of attacks. Protocols analyzed within our framework must be insecure if they allow for these attacks. In the following, we explain how this is achieved.

Changing Transaction Data. The adversary controlling all communication can easily change transaction data. Protocols which allow this unconditionally are insecure in our model, since $\mathcal{F}_{\text{PAY}}$ only allows to change the transaction data successfully if the adversary mounts a successful attack (i.e. guesses the initiator's PIN in the real world) or the (possibly changed) initiator is corrupted.

Relay Attacks. The aim of a relay attack [17] is to get Alice to authorize an unintended transaction, which benefits the attacker, by relaying legitimate protocol messages between the point of sale (POS) device she uses to pay for goods to another POS device which Alice uses at the same time. If Alice's input device is corrupted, she cannot know with certainty which transaction data she authorizes. Depending on the point of view, this amounts to either changing the *receiver* of a transaction initiated by Alice or changing the *initiator* of a transaction initiated by a third party Carol. Thus, in our model, this attack is just a special case of *changing transaction data*.

Pre-Play Attacks. Pre-play attacks [4] basically rely on two facts: (i) once unlocked, smartcards, as used in the EMV protocol, can be coerced into generating message authentication codes (MACs) for arbitrary transaction messages and (ii) that even honest ATMs use predictable "unpredictable numbers". Cards interacting with a corrupted ATM can be used to easily generate additional MAC tags. This attack can be modeled by using a global smartcard functionality which

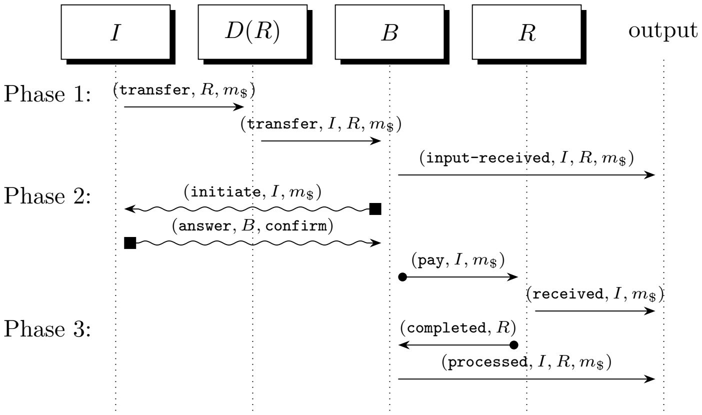

Fig. 3. The protocol π_{PAY} realizing $\mathcal{F}_{\mathrm{PAY},\mathcal{D}}(I, B, R)$ using $\mathcal{F}_{\mathrm{CONF},\mathcal{D}}(B, I)$, $\mathcal{F}_{\mathrm{AUTH}}(B, R)$ and $\mathcal{F}_{\mathrm{AUTH}}(R, B)$, the latter two depicted as ●⟶. The use of an imperfect $\mathcal{F}_{\mathrm{CONF},\mathcal{D}}$ is depicted via ■⇝. The protocol is between the human initiator I, the ATM $D(R)$, the bank B and the money dispenser R. The protocol proceeds in three phases, namely (1) the information collection phase, (2) the confirmation and execution phase and (3) the phase which ensures a consistent view on what happened.

we present in the full version [1] within the Generalized Universal Composability (GUC) extension of the basic Universal Composability (UC) framework. In the GUC framework, the environment (and thus indirectly the adversary) can even access the smartcard in the name of *honest* parties in protocol sessions different from the challenge session. Thus, a payment protocol that GUC-realizes $\mathcal{F}_{\mathrm{PAY}}$ must in particular be secure against all kinds of attacks that result from injecting pre-calculated (sensitive) data into other sessions. Protocols which do not prevent these kinds of attacks (e.g. by enforcing some sort of freshness on the protocol messages) cannot be secure in our model.

3 Towards Realizing Secure Electronic Payment

The core challenge when realizing $\mathcal{F}_{\mathrm{PAY}}$ is the authenticated transmission of transaction data from the (human) initiator to the bank. This can also be captured formally: the functionality $\mathcal{F}_{\mathrm{PAY}}$ can be used to implement the ideal authenticated communication functionality $\mathcal{F}_{\mathrm{AUTH}}$ between initiator and bank (up to the attack success probability captured by the distribution $\mathcal{D}$) by encoding the message as an amount to be transmitted. We use this insight to establish several guidelines for the design of secure payment protocols: First, we state a *necessary* condition for protocols that realize $\mathcal{F}_{\mathrm{PAY}}$: they must use setups that are strong enough to realize authenticated communication between the (human) initiator and the bank. Protocol designers can use this condition as an easily checkable criterion for the *insecurity* of payment protocols. Second, we state several setups that are *sufficient* for realizing $\mathcal{F}_{\mathrm{PAY}}$.

For the proofs, we define an ideal functionality $\mathcal{F}_{\text{AUTH},\mathcal{D}}$, analogous to $\mathcal{F}_{\text{CONF}}$ and $\mathcal{F}_{\text{PAY}}$, that allows the adversary to change the message to be sent with a certain probability parametrized by $\mathcal{D}$. For a formal description, see the full version [1]. For the sake of an easier exposition, we consider ideal functionalities like $\mathcal{F}_{\text{AUTH},\mathcal{D}}$ that model the transmission of *only one* message. This is in line with the protocols we consider. If multiple messages have to be transmitted over the same "channel", this model does not adequately capture reality, as an adversary would be able to attack *each* transmission independently. In this case, ideal functionalities for channels like $\mathcal{F}_{\text{SC}}$ (cf. [8]) can be adapted the same way.

3.1 Requirements for Secure Electronic Payment

In this section, we establish necessary and sufficient criteria for secure electronic payment. Let $\mathcal{F}_{\text{AUTH},\mathcal{D}}(I, R)$ denote the imperfect ideal authenticated communication functionality between parties I and R, and $\mathcal{F}_{\text{SMT},\mathcal{D}}(I, R)$ the corresponding ideal secure message transfer functionality (where correct guessing according to $\mathcal{D}$ results in loss of secrecy and authenticity). For a formal description, see the full version [1]. Throughout this section, let I, B, R be ITMs, where I is human[3], B is honest and $\mathcal{D}$ a parametrized distribution. We obtain the following theorem:

Theorem 2. *There exists a protocol that UC-realizes $\mathcal{F}_{\text{AUTH},\mathcal{D}}(I, R)$ in the $\mathcal{F}_{\text{PAY},\mathcal{D}}(I, \star, R)$-hybrid model, where $\star$ is an arbitrary protocol party.*

In particular, Theorem 2 implies that protocols without any authenticated communication or only between the bank and the receiver cannot realize $\mathcal{F}_{\text{PAY}}$:

Corollary 1. *Let π be a protocol that is in the $\mathcal{F}_{\text{AUTH}}(B, R)$, $\mathcal{F}_{\text{AUTH}}(R, B)$-hybrid model only (in particular, there is no authenticated communication between I and B). Then there is no protocol ρ in the bare model such that ρ^{π} UC-realizes $\mathcal{F}_{\text{PAY},\mathcal{D}}(I, B, R)$ if $\mathcal{D}$ admits the adversary at least a non-negligible successful attack probability.*

This insight can be generalized and gives a *necessary condition*: A protocol π that realizes $\mathcal{F}_{\text{PAY},\mathcal{D}}(I, B, R)$ must use setups that can be used to realize $\mathcal{F}_{\text{AUTH},\mathcal{D}}(I, B)$.

Theorem 3 (Necessary Requirements for Setups). *Let F be a set of ideal functionalities. Let Π be the set of all subroutine-respecting protocols with the set of protocol parties $P \subseteq \{I, R, B\}$ that use only ideal functionalities in F. If there is no protocol $\pi \in \Pi$ such that π^{F} realizes $\mathcal{F}_{\text{AUTH},\mathcal{D}}(I, B)$, then there is no protocol $\rho \in \Pi$ such that ρ^{F} realizes $\mathcal{F}_{\text{PAY},\mathcal{D}}(I, B, R)$.*

Conversely, it is easy to see that $\mathcal{F}_{\text{PAY},\mathcal{D}}$ can be realized by (also) using e.g. $\mathcal{F}_{\text{AUTH},\mathcal{D}}(I, B)$. Several sufficient requirements are stated in the following theorem:

[3] Note that our results hold for arbitrary I.

Theorem 4 (Sufficient Requirements). *Let π be a protocol that UC-realizes (i) $\mathcal{F}_{\mathrm{AUTH},\mathcal{D}}(I, B)$,* or *(ii) $\mathcal{F}_{\mathrm{SMT},\mathcal{D}}(B, I)$,* or *(iii) $\mathcal{F}_{\mathrm{CONF},\mathcal{D}}(B, I)$. Then, there exists a protocol ρ such that ρ^{π} UC-realizes $\mathcal{F}_{\mathrm{PAY},\mathcal{D}}(I, B, R)$ in the $\mathcal{F}_{\mathrm{AUTH}}(B, R)$, $\mathcal{F}_{\mathrm{AUTH}}(R, B)$-hybrid model.*

The proofs of Theorems 2 to 4 and Corollary 1 are in the full version [1].

3.2 No Authentication Using Smartcards Without Additional Trust

By default, EMV uses smartcards containing shared secrets with the bank in order to authenticate transactions. However, this only works if the input device which accesses the smartcard (e.g. the automated teller machine (ATM)) can be trusted. Otherwise, after the initiator enters his personal identification number (PIN) to authorize a seemingly legitimate transaction, the input device can present false (transaction) data to the smartcard (cf. e.g. [4]). We can prove the intuition that smartcards are not sufficient for realizing $\mathcal{F}_{\mathrm{PAY}}$. In the full version [1], we give a global signature card functionality $\overline{\mathcal{G}}_{\mathrm{SigCard}}$, closely modeled after similar functionalities in the literature. We then prove that no protocol which uses only this functionality (and authenticated communication between the bank and the receiver) can realize (transferrable) authenticated communication between the initiator and the bank. Using Theorem 3, we can conclude that $\overline{\mathcal{G}}_{\mathrm{SigCard}}$ is insufficient to realize $\mathcal{F}_{\mathrm{PAY}}$ even in the presence of bidirectional authenticated communication between the bank and the receiver:

Theorem 5. *There exists no protocol π in the $\overline{\mathcal{G}}_{\mathrm{SigCard}}, \mathcal{F}_{\mathrm{AUTH}}(B, R), \mathcal{F}_{\mathrm{AUTH}}(R, B)$-hybrid model that GUC-realizes $\mathcal{F}_{\mathrm{PAY}}(I, B, R)$ if I is human.*

The proof of Theorem 5 is in the full version [1].

3.3 Realistic Assumptions

Protocols build on assumptions to achieve security. However, there often is a huge discrepancy regarding to how realistic these assumptions are. EMV relies on the security of the ATM which is often publicly accessible and offers a large attack surface. Unpatched operating systems and exposed Universal Serial Bus interfaces are only two examples for vulnerabilities that have been exploited successfully. As explained in Section 2.2, a secure protocol can be constructed by establishing a confirmation mechanism. However, if the input device is corrupted, an additional device is required.

Such additional devices could for example be transaction authentication number (TAN) generators or smartphones. In principle these allow for the creation of protocols that are secure in our model. However, smartphones, which are increasingly used to replace smartcards, regularly call attention because of vulnerabilities. They are complex systems connected to the Internet and are thus more vulnerable to attacks—especially if they are operated by people without expertise in IT security. However, this dilemma can be resolved by requiring trust in only *one of the two devices*. We call this property 1-of-2 (*one-out-of-two*) security (which is, in the case of authentication, also known as multi-factor authentication). This means that a protocol is still secure if one of the two devices is corrupted, no matter which one of them. We argue that, in addition to realizing $\mathcal{F}_{\text{PAY}}$, payment protocols should support this property in order to further reduce the attack surface.

4 On the Security of Current Payment Protocols

In this chapter, we use our acquired insights to analyze current protocols for withdrawing cash, paying at the point of sale (POS), and online banking. Table 1 summarizes our findings. Our model allows for a structured and fast categorization of payment protocols on a conceptual level, even without a detailed protocol description. Even though EMV is the most widely used standard for payments, we do not elaborate on its security in this chapter. As mentioned before, its design incorporates at least two assumptions that do not hold, as several attacks have been demonstrated. Current payment protocols such as Google Pay, Apple Pay, Samsung Pay, Microsoft Pay and Garmin Pay provide an app that uses the EMV contactless standard to communicate with existing POS devices via near-field communication [28, 30]. Since they rely on Consumer Device Cardholder Verification Method, the user is authenticated by the mobile device exclusively. Currently, these apps use a personal identification number (PIN), a fingerprint or face recognition and thus do not incorporate a second device such as the POS device for authentication. Therefore the security of the protocol is solely based on the mobile device.

The protocols discussed in this section make additional implicit assumptions, which we believe to be plausible, but want to make explicit. These include the following: (i) An additional trusted device beside the input device. This is a plausible assumption if the device is simple, less so if it is a smartphone. However, using an additional device could enable protocols to be 1-of-2-secure. (ii) Authenticated communication between the initiator of a transaction and an additional personal device. This is a realistic assumption, since the initiator owns the device. Likewise the initiator can authenticate themselves to the device, e.g. by unlocking the screen of a mobile device. (iii) Confidential communication from the initiator to the automated teller machine (ATM), which can be realized by covering the PIN pad with one's hand if the ATM is not compromised. (iv) Confidential communication from the ATM to the bank. This can be realized using public-key cryptography.

In the following, we examine multiple protocols for cash withdrawal and online banking.

Cardless Cash. Cardless Cash [12] is an app-based protocol for cash withdrawal offered by numerous banks in Australia. In its most simple variant, it works as follows: After registration, the app can be used to create a "cash code" by entering the desired amount and a phone number. The phone number is used to send a PIN via SMS and allows to permit someone else to withdraw cash. To dispense the cash, the PIN has to be entered at the ATM alongside the cash code. The security of the protocol is solely based on the ATM, since all relevant information is entered there and no additional confirmation mechanism is established.

VR-mobileCash. VR-mobileCash [31] is another app-based protocol for cash withdrawal offered by Volks- und Raiffeisenbanken, a German association of banks. Upon registration, the user receives the mobile personal identification number (mPIN), which has to be entered on the ATM later on to confirm a transaction. To withdraw cash, the user has to enter the desired amount in the app. After selecting mobile payment at the ATM, the ATM shows a mobile transaction identification number (mTIN) which has to be entered in the app. The ATM then shows the requested amount and asks the user to enter the mPIN. If the mPIN is correct the ATM dispenses the requested amount of cash.

Although not stated explicitly in the public documentation, the mobile device has to be online during the transaction, as the ATM is informed about the transaction data. If the mobile device is corrupted but the ATM is honest, a user can detect an attack because he has to confirm the transaction by entering the mPIN at the ATM and thus verifies the location of the ATM. However, a *corrupted ATM* can employ a relay attack by displaying the mTIN of another corrupted ATM and forwarding the entered mPIN to it thus allowing the second corrupted ATM to dispense the cash. This could be fixed by adding a serial number imprinted on the ATM which is also displayed in the app after entering the mTIN. Thereby VR-mobileCash could potentially realize $\mathcal{F}_{\mathrm{PAY}}$ and even be 1-of-2-secure.

chipTAN comfort. ChipTAN comfort [26] is a protocol for online banking widely used in Germany. Here, the initiator uses a computer as an input device and possesses two additional personal devices: a transaction authentication number (TAN) generator and a smartcard. The TAN generator is used to confirm transactions and thus realizes a confirmation mechanism. This works as follows: First, a transaction has to be requested in the browser. Then, the banking website shows a flickering code. The user puts the smartcard into the TAN generator and scans the flickering code. After reviewing the transaction data presented on the personal device, he presses a button which reveals a TAN that has to be entered into the website.

This protocol satisfies all requirements for a secure realization of $\mathcal{F}_{\mathrm{PAY}}$ by establishing a confirmation channel that allows a user to detect tampering of the transaction data. What is more, the protocol potentially provides a form of 1-of-2 security, since as long as either the input device or alternatively the TAN generator together with the smartcard are uncorrupted, there exists a confirmation mechanism from the bank to the initiator. This is only true for single transactions, however (see [27] for details).

photoTAN. PhotoTAN (or QR-TAN) is a variant of chipTAN comfort, where the code to transmit data to the TAN generator is encrypted by the bank. Furthermore, a smartphone can be used as an alternative to a special-purpose TAN generator. In our model, this encryption does not have an impact on security, since the transaction data is not confidential and is displayed on the smartphone

Table 1. Comparison of different payment protocols. A protocol is marked as offline, if the additional device does not require an Internet connection during the payment process. The security of a protocol is put in parentheses if it meets our requirements for a secure protocol but has not been proven secure.

Protocol	Offline	Secure	Applicable for
Cardless Cash	✓	×	Withdrawal
VR-mobileCash	×	×	Withdrawal
chipTAN comfort	✓	(✓: 1-of-2)	Online banking
photoTAN	✓	(✓: 1-of-2)	Online banking
L-Pay (our scheme)	✓	✓: 1-of-2	Withdrawal, PoS

nonetheless. However, some banking apps for photoTAN [16, 11] show the TAN immediately after scanning the code and before the transaction data have been confirmed by the user. Thus, in the scenario of cash withdrawal, an attacker that corrupted an ATM and deploys a camera monitoring the ATM could change the submitted transaction data at the ATM, read the TAN from the victim's display and confirm the transaction without the initiator's consent.

5 Realizing Secure Electronic Payment

In Section 2.2, we gave a protocol π_{PAY} that realizes $\mathcal{F}_{\text{PAY},\mathcal{D}}(I,R,B)$ in the $\mathcal{F}_{\text{AUTH}}(B,R)$, $\mathcal{F}_{\text{AUTH}}(R,B)$, $\mathcal{F}_{\text{CONF},\mathcal{D}}(B,I)$-hybrid model. While realizing $\mathcal{F}_{\text{AUTH}}$ between the bank and the receiver is simple, realizing $\mathcal{F}_{\text{CONF},\mathcal{D}}(B,I)$ in a way suitable for humans is a challenge under realistic trust assumptions.

The protocols in Section 4 use one or more additional devices, such as smartphones, smartcards or optical code readers to give the initiator a confirmation capability. Yet all cash withdrawal protocols still need a trusted automated teller machine (ATM). In the following, we improve on this by presenting a simple offline protocol called L-Conf (informally described by $\pi_{\text{L-Conf}}$ in Fig. 4). It is inspired by chipTAN and photoTAN which use similar mechanisms. Our protocol is secure even if either the additional device A, such as the initiator's smartphone, or the input device is compromised. We call this property *one-out-of-two security*, formally defined as follows:

Definition 1 (One-out-of-two security). *Let X_1, X_2 be Boolean variables, π a protocol and $\mathcal{F}$ an ideal functionality. We say that π* UC-realizes $\mathcal{F}$ with one-out-of-two security relative to X_1 and X_2, *if $X_1 \vee X_2$ implies that π UC-realizes $\mathcal{F}$.*

$\pi_{\text{L-Conf}}$ can be used with π_{PAY} to realize $\mathcal{F}_{\text{PAY}}$. We call the resulting protocol L-Pay. The protocol starts with a setup phase: The bank B and the initiator I agree on a personal identification number (PIN) and the initiator's smartphone shares keys with the bank for an authenticated secret-key encryption scheme.

The main part, depicted in Fig. 4, consists of the execution of two protocols π_1 and π_2, each realizing $\mathcal{F}_{\text{CONF}}(B,I)$ under different assumptions. By combining their results, the composed protocol $\pi_{\text{L-Conf}}$ realizes $\mathcal{F}_{\text{CONF}}(B, I)$ even if either the input device or the additional device is compromised.

In π_1, the bank first encrypts the transaction data together with a fresh one-time transaction authentication number (TAN). The ciphertext is then transmitted to the initiator's input device, displayed appropriately, transferred to the smartphone (e.g. by scanning a QR code) and is decrypted. The TAN is only shown after the transaction data has been checked and *explicitly confirmed* by the initiator. Afterwards, the initiator enters the TAN into the input device.

In order to achieve security even if the initiator's smartphone is corrupted, π_2 requires the initiator to also check and confirm the transaction by entering his PIN into the input device (confidentially over $\mathcal{F}_{\text{Confid}}$), which is then sent to the bank confidentially. Only if the bank receives both the correct TAN and PIN, it considers the transaction to be confirmed. Now, if only the initiator's smartphone is corrupted, the adversary is able to present false transaction data to them or even to perform the confirmation himself. However, this would be noticed immediately, since the transaction data shown on the input device would be wrong and the initiator would not enter his PIN. Conversely, if only the input device is malicious and displays wrong transaction data, the initiator will notice this using their smartphone.

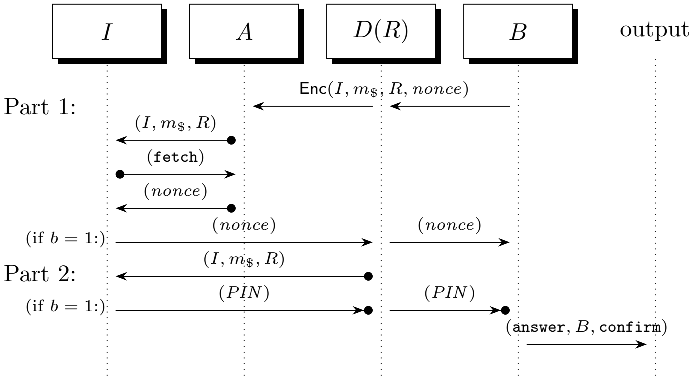

Fig. 4. Main phase of $\pi_{\text{L-Conf}}$ realizing $\mathcal{F}_{\text{CONF},\mathcal{D}}(B, I)$ using authenticated and confidential channels drawn as ●⟶ and ⟶●, resp. The protocol is between the human initiator I, his personal device A, the ATM $D(R)$ and the bank B. The bit $b \in \{0, 1\}$ indicates, whether I wants to confirm, hence $(nonce)$ and (PIN) are only sent in this case.

Theorem 6. *Let I, B, $D(R)$ and A be ITMs, where I is human. Let S be the domain of $\mathcal{D}_1, \mathcal{D}_2$, let π_1 UC-realize $\mathcal{F}_{\text{CONF},\mathcal{D}_1}(B, I)$ if A is honest and let π_2 UC-realize $\mathcal{F}_{\text{CONF},\mathcal{D}_2}(B, I)$ if $D(R)$ is honest. Then, $\pi_{\text{L-Conf}}$ UC-realizes $\mathcal{F}_{\text{CONF},\mathcal{D}_3}(B, I)$ in the $\mathcal{F}_{\text{AUTH}}(A, I)$, $\mathcal{F}_{\text{AUTH}}(I, A)$, $\mathcal{F}_{\text{AUTH}}(D(R), I)$,*

$\mathcal{F}_{\text{Confid}}(I, D(R))$, $\mathcal{F}_{\text{Confid}}(D(R), B)$-hybrid model where for all $x \in S$:

$$\mathcal{D}_3(m_\$)(x) := \begin{cases} \max\left(\mathcal{D}_1(m_\$)(\texttt{confirm}), \mathcal{D}_2(m_\$)(\texttt{confirm})\right) & x = \texttt{accept} \\ 1 - \max(\mathcal{D}_1(m_\$)(\texttt{confirm}), \mathcal{D}_2(m_\$)(\texttt{confirm})) & x = \texttt{reject} \end{cases}$$

Proof (Sketch). The protocol $\pi_{\text{L-Conf}}$ (Fig. 4) can be interpreted as the sequential composition of two confirmation protocols π_1 (Part 1) and π_2 (Part 2). It holds that π_1 realizes $\mathcal{F}_{\text{CONF}}(B, I)$ if A is honest, and that π_2 realizes $\mathcal{F}_{\text{CONF},\mathcal{D}}(B, I)$ if $D(R)$ is honest (omitting the unnecessary message from B to $D(R)$ to initiate Part 2). Let $b \in \{\texttt{confirm}, \texttt{reject}\}$ denote the initiator's input and let $b_1, b_2 \in \{\texttt{confirm}, \texttt{reject}\}$ denote the outputs of π_1 and π_2 as received by B, respectively. After having received b_1 and b_2, B outputs b', which is `confirm` if $b_1 = b_2 = \texttt{confirm}$, and `reject` otherwise. By definition, $b' = \texttt{confirm}$ while $b = \texttt{reject}$ holds with probability upper-bounded by $\max\left(\mathcal{D}_1(m_\$)(\texttt{confirm}), \mathcal{D}_2(m_\$)(\texttt{confirm})\right)$.

Thus, $\pi_{\text{L-Conf}}$ UC-realizes $\mathcal{F}_{\text{CONF},\mathcal{D}_3}(B, I)$ with one-out-of-two security relative to the assumptions that A or $D(R)$ is honest, respectively. For a formal proof, see the full version [1]. □

6 Conclusion and Future Work

Designing secure payment protocols poses a particular challenge. They typically involve a human user who is not capable of performing cryptographic operations and therefore needs an intermediate device (e.g. an automated teller machine (ATM)) to interface with the protocol, which can not always be trusted. In this work we introduce a formal model for the security of such protocols. In particular, we do not assume all intermediate devices as trusted. We use the Universal Composability (UC) framework, guaranteeing strong security and composability even in concurrent and interleaved executions.

With our model, we develop a set of basic requirements for electronic payment protocols without which no protocol can be considered secure. Based on these results, we discuss different current payment protocols and find that most do not realize these requirements. We then specify a protocol called L-Pay (based upon chipTAN and photoTAN), which uses an additional smartphone and which is secure in our model even if either the ATM or the smartphone is honest.

One important security mechanism missing in our model is time (e.g. for arguing about the security of timestamps), which is impossible to model in the standard (G)UC framework however. Extensions exist that model time [23] which could be incorporated in our model in the future. Since we assume the bank to be trusted, we limited our model to a single bank and disregarded the problem of book-keeping. Future work could expand our model to include these features.

References

1. Achenbach, D., Gröll, R., Hackenjos, T., Koch, A., Löwe, B., Mechler, J., Müller-Quade, J., and Rill, J.: Your Money or Your Life—Modeling and Analyzing the Security of Electronic Payment in the UC Framework, Full version of the paper. (2019). https://crypto.iti.kit.edu/fileadmin/User/Mechler/AGHKLMMQR19.pdf
2. Anderson, R.J., Bond, M., Choudary, O., Murdoch, S.J., and Stajano, F.: Might Financial Cryptography Kill Financial Innovation? – The Curious Case of EMV. In: Danezis, G. (ed.) Financial Cryptography and Data Security, FC 2011. LNCS, vol. 7035, pp. 220–234. Springer, Heidelberg (2011)
3. Basin, D.A., Radomirovic, S., and Schläpfer, M.: A Complete Characterization of Secure Human-Server Communication. In: Fournet, C., Hicks, M.W., and Viganò, L. (eds.) IEEE 28th Computer Security Foundations Symposium, CSF 2015, pp. 199–213. IEEE Computer Society (2015)
4. Bond, M., Choudary, O., Murdoch, S.J., Skorobogatov, S.P., and Anderson, R.J.: Chip and Skim: Cloning EMV Cards with the Pre-play Attack. In: 2014 IEEE Symposium on Security and Privacy, SP 2014, pp. 49–64. IEEE Computer Society (2014)
5. Borchert IT-Sicherheit UG: Display-TAN Mobile Banking: Secure and Mobile, (2018). http://www.display-tan.com/ (visited on 09/18/2018)
6. Canetti, R.: Universally Composable Security: A New Paradigm for Cryptographic Protocols. In: 42nd Annual Symposium on Foundations of Computer Science, FOCS 2001, pp. 136–145. IEEE Computer Society (2001)
7. Canetti, R., Dodis, Y., Pass, R., and Walfish, S.: Universally Composable Security with Global Setup. In: Vadhan, S.P. (ed.) 4th Theory of Cryptography Conference, TCC 2007. LNCS, vol. 4392, pp. 61–85. Springer, Heidelberg (2007)
8. Canetti, R., and Krawczyk, H.: Universally Composable Notions of Key Exchange and Secure Channels. In: Knudsen, L.R. (ed.) Advances in Cryptology - EUROCRYPT 2002, International Conference on the Theory and Applications of Cryptographic Techniques, Amsterdam, The Netherlands, April 28 - May 2, 2002, Proceedings. LNCS, vol. 2332, pp. 337–351. Springer, Heidelberg (2002)
9. Chaum, D., Fiat, A., and Naor, M.: Untraceable Electronic Cash. In: Goldwasser, S. (ed.) CRYPTO '88. LNCS, vol. 403, pp. 319–327. Springer, Heidelberg (1988)
10. Chothia, T., Garcia, F.D., de Ruiter, J., van den Breekel, J., and Thompson, M.: Relay Cost Bounding for Contactless EMV Payments. In: Böhme, R., and Okamoto, T. (eds.) Financial Cryptography and Data Security, FC 2015. LNCS, vol. 8975, pp. 189–206. Springer, Heidelberg (2015)
11. Commerzbank: Das photoTAN-Lesegerät. https://www.commerzbank.de/portal/media/a-30-sonstige-medien/pdf/themen/sicherheit-1/Flyer_Lesegeraet.pdf (visited on 12/13/2018)
12. Commonwealth Bank of Australia: Cardless Cash, (2018). https://www.commbank.com.au/digital-banking/cardless-cash.html (visited on 09/25/2018)
13. Cortier, V., Filipiak, A., Florent, J., Gharout, S., and Traoré, J.: Designing and Proving an EMV-Compliant Payment Protocol for Mobile Devices. In: 2017 IEEE European Symposium on Security and Privacy, EuroS&P 2017, pp. 467–480. IEEE (2017)
14. Degabriele, J.P., Lehmann, A., Paterson, K.G., Smart, N.P., and Strefler, M.: On the Joint Security of Encryption and Signature in EMV. In: Dunkelman, O. (ed.) CT-RSA 2012. LNCS, vol. 7178, pp. 116–135. Springer, Heidelberg (2012)
15. Denzel, M., Bruni, A., and Ryan, M.D.: Smart-Guard: Defending User Input from Malware. In: 2016 Intl IEEE Conferences on Ubiquitous Intelligence & Comput-

ing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld), pp. 502–509. IEEE Computer Society (2016)

16. Deutsche Bank: photoTAN – schnell und einfach aktiviert. https://www.deutsche-bank.de/pfb/data/docs/Photo_TAN_Smartphone_2.pdf (visited on 12/13/2018)
17. Drimer, S., and Murdoch, S.J.: Keep Your Enemies Close: Distance Bounding Against Smartcard Relay Attacks. In: Provos, N. (ed.) Proceedings of the 16th USENIX Security Symposium 2007. USENIX Association (2007)
18. Emms, M., Arief, B., Freitas, L., Hannon, J., and van Moorsel, A.P.A.: Harvesting High Value Foreign Currency Transactions from EMV Contactless Credit Cards Without the PIN. In: Ahn, G., Yung, M., and Li, N. (eds.) 2014 ACM SIGSAC Conference on Computer and Communications Security, pp. 716–726. ACM (2014)
19. EMV: Integrated Circuit Card Specifications for Payment Systems: Book 1. Application Independent ICC to Terminal Interface Requirements, Version 4.3 (2011)
20. EMV: Integrated Circuit Card Specifications for Payment Systems: Book 2. Security and Key Management, Version 4.3 (2011)
21. EMV: Integrated Circuit Card Specifications for Payment Systems: Book 3. Application Specification, Version 4.3 (2011)
22. Garay, J.A., Kiayias, A., and Leonardos, N.: The Bitcoin Backbone Protocol: Analysis and Applications. In: Oswald, E., and Fischlin, M. (eds.) EUROCRYPT 2015. LNCS, vol. 9057, pp. 281–310. Springer, Heidelberg (2015)
23. Katz, J., Maurer, U., Tackmann, B., and Zikas, V.: Universally Composable Synchronous Computation. In: Sahai, A. (ed.) TCC 2013. LNCS, vol. 7785, pp. 477–498. Springer, Heidelberg (2013)
24. Murdoch, S.J., Drimer, S., Anderson, R.J., and Bond, M.: Chip and PIN is Broken. In: 31st IEEE Symposium on Security and Privacy, S&P 2010, pp. 433–446. IEEE Computer Society (2010)
25. Old Bailey Proceedings Online (ed.), ed.: Trial of JOHN BUCKLEY, THOMAS SHENTON, version 8.0. (1781). https://www.oldbaileyonline.org/browse.jsp?div=t17810912-37 (visited on 09/22/2018)
26. Postbank: Postbank chipTAN comfort, (2018). https://www.postbank.de/privatkunden/chiptan-comfort.html (visited on 09/25/2018)
27. RedTeam Pentesting GmbH: Man-in-the-Middle Attacks against the chipTAN comfort Online Banking System, (2009). https://www.redteam-pentesting.de/publications/2009-11-23-MitM-chipTAN-comfort_RedTeam-Pentesting_EN.pdf (visited on 09/25/2018)
28. Smart Card Alliance: Contactless EMV Payments: Benefits for Consumers, Merchants and Issuers, http://www.emv-connection.com/downloads/2016/06/Contactless-2-0-WP-FINAL-June-2016.pdf (visited on 12/17/2018)
29. Tamarin: Tamarin prover, (2018). https://tamarin-prover.github.io/ (visited on 12/19/2018)
30. Visa: Visa Token Service, https://usa.visa.com/partner-with-us/payment-technology/visa-token-service.html (visited on 12/17/2018)
31. Volksbank Mittelhessen eG: VR-mobileCash: Geld abheben ohne Karte, https://www.vb-mittelhessen.de/privatkunden/girokonto-kreditkarten/infos-banking/geld-abheben-ohne-karte.html (visited on 09/25/2018)