Another Look at Relay & Distance-based Attacks in Contactless Payments

Ioana Boureanu and Anda Anda

University of Surrey
i.boureanu@surrey.ac.uk, da00448@surrey.ac.uk

Abstract. Relay attacks on contactless e-payments were demonstrated in 2015. Since, countermeasures have been proposed and Mastercard has recently adopted a variant of these in their specifications. These relay-counteractions are based on the payment-terminal checking that the card is close-by. To this end, several other EMV-adaptations have emerged, with the aim to impede dishonest cards cheating on their proximity-proofs. However, we argue that both the former and the latter measures are ineffective.

We only sketch possible designs in the right directions, with the idea to pass on the message that these problems should be look at much more carefully.

We shortly debate what should and should not be the case w.r.t. confirmation of EMV contactless payments.

We also discuss alternative views onto making contactless payments secure against relay-attacks via proximity-checking.

1 Introduction

Relay Attacks. Relaying between two legitimate parties A and B is an attack whereby a man-in-the-middle C simply sends A's messages to B and/or B's messages to A, unbeknown to them. In so doing, C aims to get a privilege that was meant for A or for B. I.e., C could fraudulently spend A's funds at a contactless terminal represenedted B. Indeed, the EMV (Europay, Mastercard and Visa) payment protocols, in their contactless version, are prone to relay attacks [7].

Proximity Threat-model. Imposing an upper-bound on the round-trip times (RTTs) of message-exchanges is a known, physical-layer mechanism [5] for effectively lowering the probability of successful relay attacks. This mechanism is often referred to as *distance-bounding (DB)* or sometimes *proximity-checking*. But, DB comes with its own threat-model, which we summarise next. In a distance-fraud (DF) attack, a far-away malicious card aims to forge his proximity proof; traditionally, this was intended to be such that the dishonest far-away prover acts on his own in this attack. Instead of having the DF attacker mount the attack without exploiting on any other entity, one can imagine an augmentation of DF called distance-hijacking [8]. In distance-hijacking (DH), a far-away malicious prover P^* takes advantage of on honest close-by prover to mount P^*'s false proximity-proof. In terrorist fraud (TF), to make the verifier accept, a far-away malicious card P^* colludes with a malicious man-in-the-middle (MiM) A, positioned close to the reader, but in such a way that A fraudulently authenticates as the card P once yet P^* does not favour future, individual attempts by A to authenticate as P^* . Generalisations of these DB-threats exist (see [3]).

This Document. Herein, **(1).** we will describe how contactless EMV was enhanced with a proximity-checking dimension into a protocol called *PaySafe*, to protect against relaying; **(2).** we will explain that this may be unsuccessful despite formal claims to the contrary; **(3).** linked to the latter, we will discuss the somewhat philosophical aspect of what "proximity-checking"

and "payment authorisation" does and/or should really mean in contactless EMV; **(4).** we will describe how PaySafe was further enhanced to protect against DB-attacks; **(5).** we will show that the latter EMV-enhancement was also unsuccessful in reaching its goals despite formal claims to the contrary; **(6).** we will give a direction towards remedies to point (2) and (5) above; **(7).** we will discuss, at high level, some security-alternatives for relay-resilient contactless EMV.

2 Relaying in EMV & Countermeasures

There are two main flavours of the contactless EMV protocol: *PayWave* and *PayPass*, by Mastercard and Visa respectively [9]. Herein, it suffices that we explain just the transactions between the card and the terminal in PayWave, given in Fig. 1.

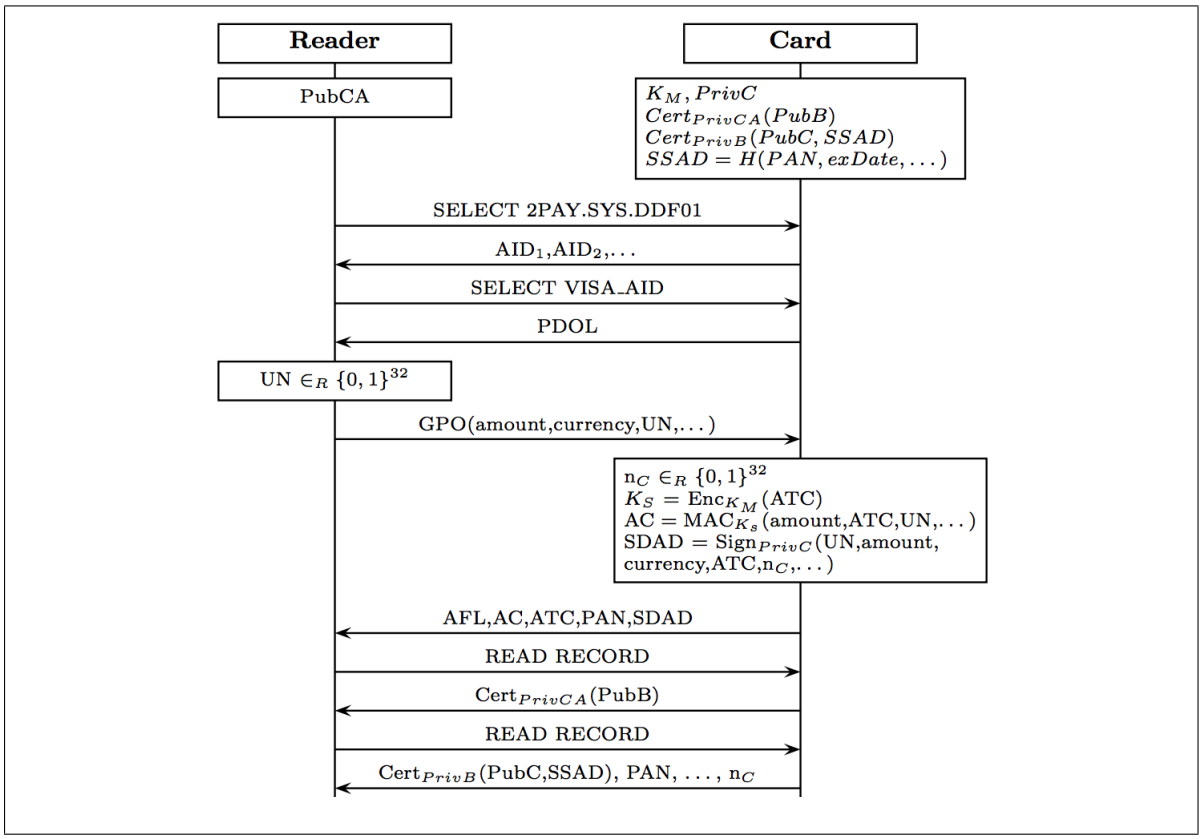

Fig. 1: Visa's PayWave qVSDC Protocol (as from [9])

The card includes: a private key $Priv_C$; a symmetric key K_M that it shares with the bank; a certificate $Cert_{Priv_{CA}}(Pub_B)$ on the bank's public key Pub_B signed by a certificate-authority CA; a certificate signed by the bank $Cert_{Priv_B}(Pub_C, SSAD)$ on the card's public key Pub_C and on a hash on the card's static-data which is called SSAD and includes e.g. the number on the card (PAN) and the expiry date. The reader has the public key Pub_{CA} of the certificate-authority CA, so it can extract Pub_B out of $Cert_{Priv_{CA}}(Pub_B)$, and then use Pub_B to extract the card's public key Pub_C and its static details out of $Cert_{Priv_B}(Pub_C, SSAD)$.

The reader initiates the communication, and, in the 2nd message, the card sends the list of payment-application identities (AIDs) that it supports. Upon the reader having selected one, the card sends the PDOL (Processing Options Data Object List); this is information enabling the reader to send the right instructions latter on. The reader generates a 32-bit nonce UN. It then sends the GPO (GET PROCESSING OPTIONS) command, which we can conceptually equate to "here is my nonce UN, you need to pay amount x in currency y". Then, the card first generates a 32-bit nonce n_C, then it generates a session key K_S as the encryption of its application transaction counter (ATC) under K_M. The card is supposed to generate a "cryptogram" (a.k.a. AC), which the reader can send to the bank to signal a due payment; the cryptogram AC is a MAC keyed under K_S of a series of data including the ATC, the nonce UN, and the amount. The card also produces the "Signed Dynamic Application Data (SDAD)": the card's signature on a message including UN, amount, currency, ATC, n_C, etc. Finally, in the 6th message in our figure, the card sends the cryptogram AC, the SDAD, as well as the ATC, the PAN and the AFL (i.e., location of the records on the card) to the reader. Thereafter, by using AFL, the reader is able to request the certificates. The reader uses them to the check the signature SDAD and the consistency of the sub-messages inside (e.g., UN, amount, etc.)

In [7], Chothia et al. showed an effective relay attack against PayPass and PayWave. As such, they suggested a thuswise improved version of contactless EMV called *PaySafe* (see Fig. 2). The main idea is that the reader check the proximity of the card to the reader, in a timed challenge-response phase. So, the PayWave GPO challenge-response is split in two: in a first instance, the card responds with the static data plus its nonce is sent, and then the card generates the cryptogram, the SDAD and sends these. The round-trip time of the first part of this is measured by the reader and if it is larger than an established bound, the card is deemed to be afar (hence, relaying being likely) and the protocol is terminated. As Fig. 2 shows, there is little other change from e.g. PayWave to PaySafe. (The generation of the nonce n_c being done earlier in PaySafe than in PayWave etc. is mainly down to achieving accurate time-measurements in the timed challenge-response round.)

In [7], the PaySafe protocol was formally verified using ProVerif [1] and shown correct/secure w.r.t. relay attacks. Mastercard appears to have adopted a relay-counteraction measure in its new, but not public, v3.1specifications for EMV [11]; this relay-countermeasure appears to equate to PaySafe[1]. But, as we explain below, we find that PaySafe is not secure against *all types* of relaying; so, if Mastercard indeed deploys its current PaySafe-like relay-counteraction as is, then this *may* not be sufficient relay-protection on their new e-payment, contactless systems.

3 Possibly Failing Relay-Counteraction in EMV

The Question of Payment-Authorisation in Contactless EMV. One question is when the respective authorisations of the payment from the card's and the card-holder's viewpoints do actually take place in contactless EMV.

The first argument to make is that perhaps the point when the payment-authorisation is given from a card-holder's perspective is when this card-holder brings the card in the range

[1] The EMV specifications by Mastercard v3.1 are not publicly available. Details of the relay-protection herein were conveyed to the authors by 3rd parties. As such, the attacks and discussions herein refer primarily/directly to PaySafe [7] and not to these EMV specifications.

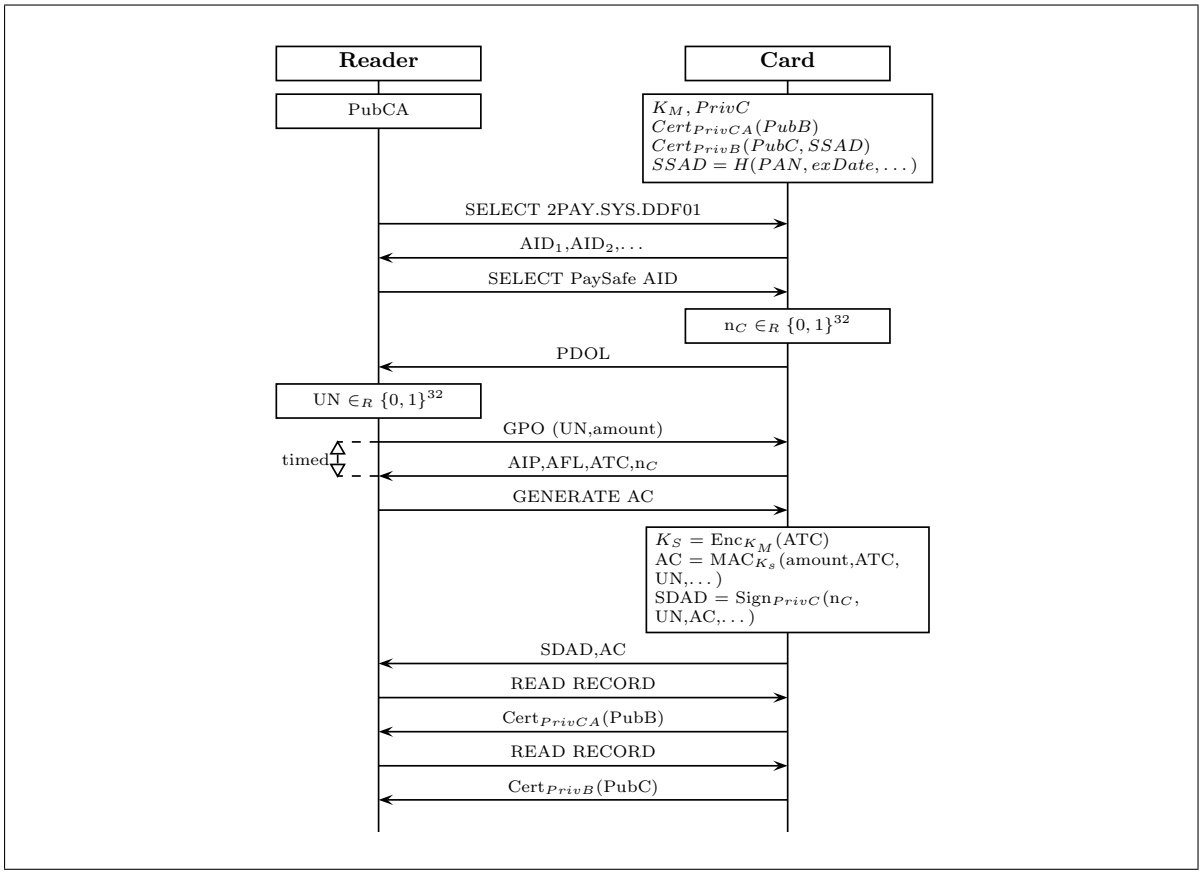

Fig. 2: The PaySafe Protocol [7]

of the reader. But, note that a card can be brought in the range of the reader without the knowledge/realisation of the card-holder. Nonetheless, everyone arguably wishes to maintain the contactless nature for this version of EMV, i.e., that the card-holder not be asked for any input.

As such, the second argument is precisely from the card's perspective (and not the card-holder's), or rather from the viewpoint of the protocol-design. And, the case that can be made here is that there should be *some explicit, clear point in the actual EMV-contactless protocol specification* when the paying-card gives "technical authorisation" that the payment should be taken by the card (that is, w.r.t. the protocol steps undertaken in the card-side of the EMV protocol). By "technical authorisation", we mean the point of no-return, with respect to the payment being taken. An appropriate point of this type is not easy to find, since there is no physical input from the card. Note that this point could be indeed the moment when the card is brought in the range of the reader, but *–in the way of proximity being actually checked–* then it makes sense that this point be later on in the protocol, once the proximity was indeed checked. To this end, we do note that [9] states that "as in EMV Contact, [...] the interaction with the card is completed *after the card returns a cryptogram*"; this is stated in the EMV standard as well. So, one can take the view that the "technical authorisation" of the paying-card to release the funds is made as the card sends the cryptogram.

However, we note that in PaySafe, there is no guarantee that the card is in the reader's proximity when the card actually emits the payment (i.e., when it sends SDAD, AC). The

PaySafe protocol technically ensures only that a card is (likely to be) in the proximity of the reader just as the latter sends the GPO request, i.e., as the "first request for payment is made". Because of the fact that the proximity-checking is present only before the AC (i.e, the cryptogram) is sent over by the card, *a relay attack resulting in a fraudulent payment is still possible*, as we detail below.

Our Relay & Payment-Fraud in PaySafe. Let us describe a scenario of such a fund-stealing, relay attack against someone's whose EMV-capable card runs PaySafe. At a busy time, in a supermarket, the attacker and his accomplice are queueing to pay at a payment-terminal $T1$. The victim is also waiting to pay, on the immediately neighbouring conveyor belt, at a different terminal. The accomplice pretends to wish to pick up some carrier bag from a stand adjacent to the victim's line; in so doing, the accomplice makes the victim move slightly such that the victim comes for a short period much closer to the terminal $T1$, on the attacker's line. For the PaySafe protocol between the victim and any reader to initiate, the distance between these two needs to be of 10cm of less. Some adversarial amplification equipment should be in place close to the reader $T1$, such that the actual distance between the victim and the reader $T1$ can still remain larger than 10cm, yet the protocol between the victim and the reader $T1$ would still initiate. In this way, the victim's contactless card starts running the protocol with the reader $T1$, including the timed round such that this round fall within the time-bounds. Then, the victim moves back further away from the reader $T1$, e.g., is walking up to some meters further away. As this point the NFC-based protocol should fail. Instead, the attacker would keep the connection alive (impersonating the reader to the card and vice-versa); the attacker relays the last three messages from the victim's card (SDAD, AC, and the certs) to payment terminal $T1$. Note that the protocol finishes correctly: the relay attack will succeed, meaning that an attacker can pay with someone else's funds even if they were meters away from the terminal $T1$ when their card emitted the final, payment-conceding cryptogram AC and authenticating data SDAD. So, this attacker just needs to get the victim be close to the terminal some seconds before the fund-stealing relaying starts.

On Relaying in PaySafe or Relay-protecting EMV. Whilst the scenario above is not tested in practice, it is not quantified by any measurements or approximations in the way of feasibility and it may even seem far-fetched, the message here is that fraudulently taking payments via relaying in relay-protecting PaySafe may still be possible. This equates to the fact that it is still possible to break one of the main goals of EMV, i.e., to protect against fake payments.

The original, large-distance relay attacks in [7] may indeed no longer be possible in PaySafe. But as we explained above, *PaySafe may not fully solve the problem of relaying in EMV.*

4 More Attacks on Proposed Relay-Protecting EMV Versions

We are aware of the fact in the original paper [7], PaySafe set out to protect just against relaying. However, when one combines two cryptographic primitives, the attacker-model from the resulting construction is generally a composition of the threats in the respective initial primitives. I.e., if one combines proximity-checking with authentication, then the potentially malicious prover in the former should be accounted for in the entire composition. This was amply discussed, e.g., in [2], specifically in the context of composing proximity-checking with authentication, as it is the case in PaySafe too. This would mean that PaySafe should now be analysed against a threat-model considering dishonest provers as well.

Whilst the above is the view taken in provable-security, one could envisage that Mastercard (or EMVCo – the consortium behind EMV) may not be not interested (for now) in attacks mounted by dishonest provers. However, if EMVCo/Mastercard implements a protocol like PaySafe as suggested by [11], then it means that –with each contactless payment recorded in a banking-ledger– they are issuing to the corresponding card-holder a proximity receipt, i.e., a confirmation that the payee was close to a payment-terminal. Therefore, if a card-holder can forge the proximity proof then there can be liabilities for Mastercard/EMVCo. Indeed, in [12], Sjouke Mauw *et al.* have recently looked at distance-bounding attacks on PaySafe. So, on balance, we are discussing this here too, in direction relation with [12].

4.1 Proximity-based (In)security in PaySafe

PaySafe did not mean to resist distance-fraud (DF), yet one such attack is extremely obvious. Indeed, the card's response in PaySafe's timed-round does not depend on the challenge sent by the reader. So, the card can send his answer early and thus appear to be closer than he is. The virtue of [12] is to have exhibited this DF via the use of a symbolic security-verification tool [13]. And, [12] proposed a solution to this DF, which we herein dub PaySafe+. In PaySafe+, compared to PaySafe, the card's timed response additionally contains the nonce UN which the reader sent, i.e., the card replies to GPO(...) via AIP, AFL, ATC, n_C, UN .

Failing Distance-Fraud-Counteraction [12] on top of PaySafe. It may appear that, in PaySafe+, the card has to wait to receive UN before answering. However, the card can sample UN bit by bit, from the MSB (least significant bit) to the LSB (least significant bit). So, a cheating prover does not need to wait to read all the bits of UN: he can do a distance-fraud by sending UN's bits back one by one, as soon as each is read at his end. Or, he can do a distance-fraud attack whereby he waits the first x MSBs of UN ($x < 32$) and guesses the rest of $32 - x$. By choosing x, the cheating prover will tune his success probability but also the degree of distance-lowering. This type of distance-frauds have been known to the DB community for a while [6].

However, we do acknowledge that the *formal* model [12] used to determine whether PaySafe+ was secure against distance-fraud does not capture the aforementioned bit-by-bit sampling of the UN nonce, and –instead– assumed that the entire UN is received at once. So, as such, the results [12] were correct in their model: it is just that the threat-model in [12] does not encode certain real-life risks such as the one described herein and/or in [6].

The Most "Naive" Solution to Distance-fraud in PaySafe/PaySafe+. This type of attack is not at all new in DB: its nature is described in 2005 in [6] In the spirit of [6], we suggest that if distance-fraud is what PaySafe+ aims to counteract, then –in the timed challenge-response– the card should send back a simple transformation on UN: e.g., reply with UN in reverse-bit order from how the reader send it, from LSB to MSB. This clearly lowers the probability of success of the DF we presented above, by intuitively forcing the card to really wait for (all or most of) UN before sending it back. Note that more sophisticated measures can be put in place to counteract early-sending attacks by a prover: a Challenge Reflection with Channel Selection (CRCS) can be implemented [14] (which is very efficient). Other, very recent solutions (presented from the hardware-implementation perspective) are given in [15].

Other Proximity-Related Security Issues in PaySafe-like Protocols. PaySafe+ fails to achieve DF-security, but also, as a distance-based primitive, PaySafe+ has other security-issues w.r.t. distance-based attacks.

For instance, it is clear that a distance-hijacking (DH) attack [8] and terrorist-fraud attacks exist.

We believe that in context of EMV, a dishonest prover may wish to literally pay something for the attacker to get the receipt that proves that he was close, when in fact he was not. So, we believe that a terrorist-fraud attack makes sense in EMV. To mount a terrorist-fraud, a dishonest prover generates n_c and gives it to the attacker, before the timed phase starts. An attacker $\mathcal{A}$ found closer to the terminal then pass the timed challenge-response Then, over the untimed part, this attacker $\mathcal{A}$ sends all the necessary info to the prover so that this one generates SDAD, AC, and hands it all back to $\mathcal{A}$. Then, $\mathcal{A}$ relays this to the reader. This is a valid TF, under the assumption of nonce freshness. In all nonces are checked for freshness, then once the accomplices stops helping the terrorist, the latter has no way of passing the protocol on his own (in any case, not under reasonable cryptographic assumptions such as unforgeable signatures, etc.). In this document, we offer no immediate solution to this TF attack, i.e., not one solution based on the very structure of PaySafe+ (like we had for PaySafer which is moulded on PaySafe, or for the distance-fraud in PaySafe+). Solutions exist, but not based on this "almost EMV-identical" type of design where we just add one proximity-checking round inside the standard contactless EMV.

Yet, w.r.t. DH attacks, we leave this aside. This is due to the fact that a DH attack would essentially mean that some far-away prover would pay with its funds for a purchase of an honest prover found close to the verifier. Since the latter is honest and does not collude with the far-away prover (as in TF), the former will never get the receipt in order to prove that he was close when he was not. So, in EMV, we do not really see the need to analyse DH.

Our PaySafe-like Solution for Relay-counteraction in EMV. In Fig. 3, we suggest a possible direction for a better counteraction against the aforementioned relay-attack on PaySafe. We call this protocol *PaySafer*. In PaySafer, after the reader sends the "GENERATE AC" command, it waits an amount δ of time, which is apriori fixed to be the time it takes cards on the market to generate AC and SDAD. The drawback would be that *all* cards on the market would have to implement the generation of the AC and SDAD in this particular permitted period δ; this would need to be enforced by EMVCo universally. We acknowledge that this is a big ask, but we believe that it is not impossible to achieve. Further, one can say that an attacker can overclock a card and thus gain time, yet we believe that this threat-model is clearly out of scope for relay attacks.

Moreover, it is only after this waiting period elapses that the reader issues our new command to the card denoted "SEND AC", accompanied by a nonce $UN2$. The card responds with AC, SDAD and $UN2^*$ which is $UN2$ sent back processed slightly by the prover: i.e., $UN2^*$ is obtained by at least reversing the bit-order in $UN2$. Note that we require the same of UN: i.e., the sent in reverse bit-order as UN^* after the GPO-request. The sending of $UN2^*$ and UN^* in this way is not essential for relay-protection, but we included them for the sake of the distance-fraud attacks presented above on PaySafe. In other words, if in our PaySafer we aimed for DF-counteraction, then UN needs to be sent back processed slightly by the prover, i.e., at least in reverse bit-order, as UN^*, during the first timed challenge-response round, and $UN2$ also needs to be is also sent back in reverse bit-order as $UN2^*$ in the 2nd timed challenge-response round. If no DF-counteraction is envisaged, them we can remove both UN and $UN2$ from PaySafer (and their starred counterparts).

PaySafer is the same as PaySafe, apart from the following: (a) using a 2nd timed round (which EMV-wise would mean a new command for what we denoted "SEND AC, SDAD"); (b) using $UN^*, UN2, UN2^*$ as per explained above.

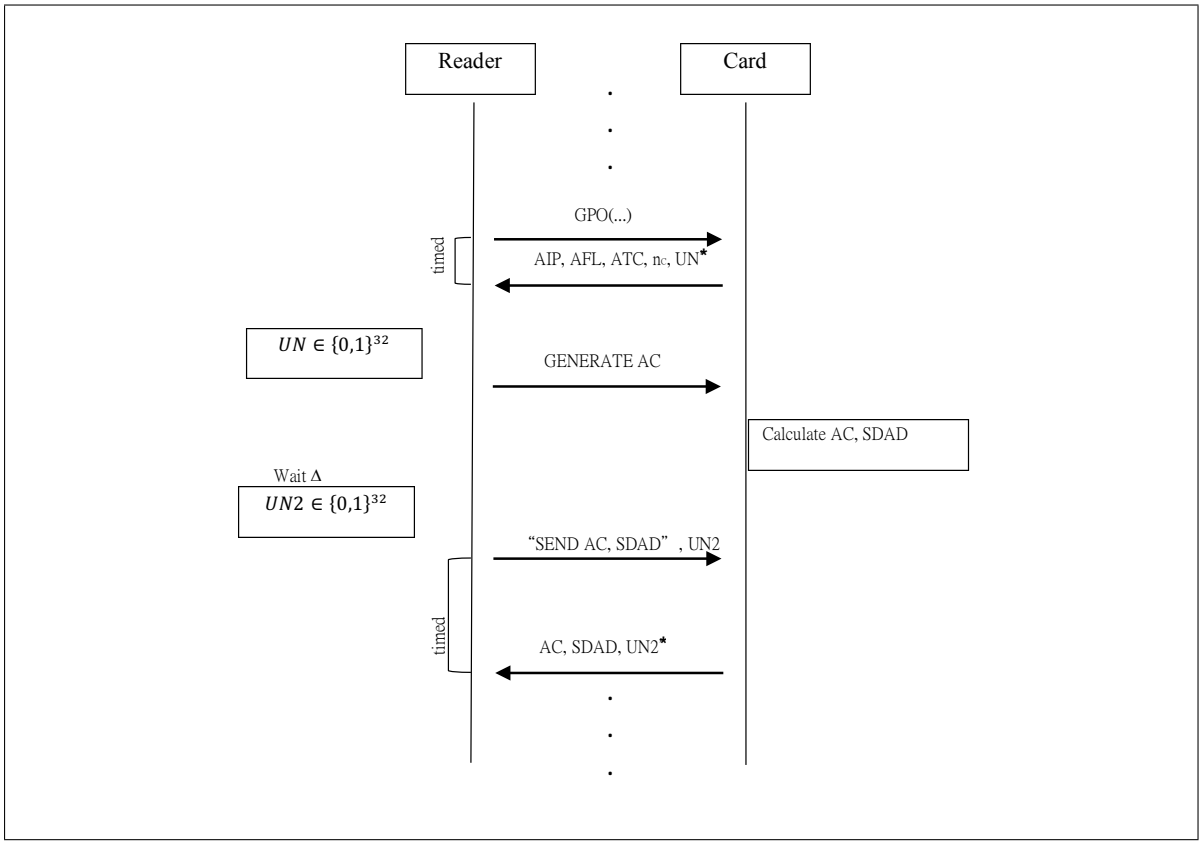

Fig. 3: PaySafer: A Primer for Better Relay-Counteraction in EMV

So, to PaySafe, we essentially added a timed challenge-response round at the point in which AC, SDAD are sent, such that we hinder relaying these two after the AC-generation (as per our attack above). In essence, this means that we check that the payee is close to terminal when the payment is concluded from the card's viewpoint (i.e., when AC, SDAD are sent), not just when the payment is "GPO-initiated" as per PaySafe.

However, note that, with PaySafer, we keep as close as possible to PaySafe and therefore to contactless EMV, mainly aiming to improve PaySafe w.r.t. achieving *only* its original goal: relay-counteraction.

5 Proximity inside EMV...: Far-away?!

We saw that PaySafe [7], which appears to be adopted in Mastercard v3.1 of EMV specs [11], does not seem to meet its aims of full relay-counteraction in EMV We can debate whether distance-based attacks (beyond relaying) are a valid concern for EMV. Meanwhile, we also saw that the current EMV-enhancements against distance-based attacks [12] also fail at achieving their goals of protecting against proximity-frauds.

Herein, we suggested directions in the way of the first adjustments to mend both failures in silo, one of which is inspired by "old-school measures" [6] against proximity-faking attacks.

So, all these can only suggest the following: (1) adding relay-counteraction to EMV in the shape of proximity-checking may not be fully accomplished as of yet; (2) it arguably opens for new distance-cheating security-threats.

Alternative Relay-Protection Mechanisms for EMV. As such, in Fig. 4, we recall the mechanism in [4], which is different to PaySafe. This design intends that a symmetric-key DB protocol is run based on a shared key *sym*. This key is established between the card and reader beforehand, by running a key-agreement protocol (KA), based on the card's a secret key *sk* and a public key *pk*. In [4], it was suggested that after this DB is run the payment is made. Since, Vaudenay has also instantiated this DB protocol above [16].

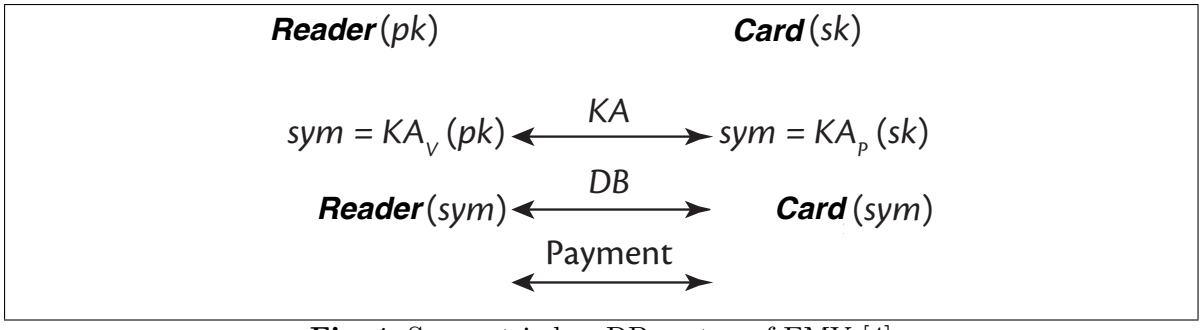

Fig. 4: Symmetric-key DB on top of EMV [4]

There are two advantages to this: (a) one can choose a DB protocol which is also provably secure against all DB threats, e.g., [3], or make little variations on Fig. 4 to achieve enhanced security [10]; (b) one can achieve some level of privacy on the card's side [10].

That said, we are not totally satisfied by this solution either: like with PaySafe, the proximity-proof happens only before the payment and, during it, relaying could still be possible. So, to succeed, one would need to embed the DB protocol more intrinsically in the payment scheme (à la PaySafer). Another issue with the solution in Fig. 4 or its embodiments as per [10] is that all the EMV readers/cards would have to implement the DB protocol to be run in combination with the payment, which is a big ask. In this deployment-driven sense, PaySafe –which keeps extremely close to EMV– is preferable.

6 Conclusions

Whilst we touched on some security-failures of EMV-enhancements with proximity checking, many aspects remain not explored herein: e.g., active men-in-the-middle, concrete privacy issues on cards' side. Our aim was not to formally assess or profess, but rather to draw *immediate* attention to this fact : as proximity-checking within EMV may be embraced by MasterCard and EMVCo, many emerging solutions are failing to different extends. Overall, we encourage an closer look into this topic, all the while calling for solutions that are as in-keeping as possible with the original EMV designs and architectures.

References

1. B. Blanchet. An Efficient Cryptographic Protocol Verifier Based on Prolog Rules. In *Proc. of the 14th IEEE Computer Security Foundations Workshop*, pages 82–96, Novia Scotia, Canada, 2001. IEEE Computer Society Press.
2. I. Boureanu, D. Gerault, P. Lafourcade, and C. Onete. Breaking and fixing the HB+DB protocol. In *Wisec 2017 - Conference on Security and Privacy in Wireless and Mobile Networks*, pages 241 – 246, Boston, United States, July 2017.

3. I. Boureanu, A. Mitrokotsa, and S. Vaudenay. Practical and Provably Secure Distance-Bounding. *Journal of Computer Security*, 23(2):229–257, 2015.
4. I. Boureanu and S. Vaudenay. Challenges in Distance Bounding. *IEEE Security & Privacy*, 13(1):41–4, 2015.
5. S. Brands and D. Chaum. Distance-bounding protocols (extended abstract). In *Proc. of EUROCRYPT*, pages 344–359. Springer, 1993.
6. S. Capkun and J. P. Hubaux. Secure positioning of wireless devices with application to sensor networks. In *Proceedings IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies.*, volume 3, pages 1917–1928 vol. 3, March 2005.
7. T. Chothia, F. D. Garcia, J. de Ruiter, J. van den Breekel, and M. Thompson. Relay cost bounding for contactless EMV payments. In *Financial Cryptography and Data Security - 19th International Conference, FC 2015, San Juan, Puerto Rico, January 26-30, 2015, Revised Selected Papers*, pages 189–206, 2015.
8. C. Cremers, K. B. Rasmussen, and S. Čapkun. Distance hijacking attacks on distance bounding protocols. Cryptology ePrint Archive, Report 2011/129, 2011. `http://eprint.iacr.org/`.
9. E. P. Jordi van den Breekel, Diego A. Ortiz-Yepes and J. de Ruiter Manuscript. Emv in a nutshell. `http://www.cs.ru.nl/~erikpoll/papers/EMVtechreport.pdf`, 2016.
10. H. Kilinc and S. Vaudenay. Efficient public-key distance bounding protocol. In *Proceedings, Part II, of the 22Nd International Conference on Advances in Cryptology — ASIACRYPT 2016 - Volume 10032*, pages 873–901, New York, NY, USA, 2016. Springer-Verlag New York, Inc.
11. MasterCard. Contactless paypass reader specifications v3.1,. not publically available, 2017. Details of relay-protection herein were conveyed to the authors by 3rd parties.
12. S. Mauw, Z. Smith, J. Toro-Pozo, and R. Trujillo-Rasua. Distance-bounding protocols: Verification without time and location. In *IEEE Symposium on Security and Privacy (Oakland S&P), May 21–23, 2018, San Francisco, California, USA*, pages 1–18, may 2018. (to appear).
13. S. Meier, B. Schmidt, C. Cremers, and D. Basin. The TAMARIN Prover for the Symbolic Analysis of Security Protocols. In *Computer Aided Verification*, pages 696–701, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.
14. K. B. Rasmussen and S. Čapkun. Realization of rf distance bounding. In *Proceedings of the 19th USENIX Conference on Security*, USENIX Security'10, pages 25–25, Berkeley, CA, USA, 2010. USENIX Association.
15. M. Singh, P. Leu, and S. Capkun. Uwb with pulse reordering: Securing ranging against relay and physical layer attacks. Cryptology ePrint Archive, Report 2017/1240, 2017. `https://eprint.iacr.org/2017/1240`.
16. S. Vaudenay. Private and secure public-key distance bounding - application to NFC payment. In *Financial Cryptography and Data Security - 19th International Conference, FC 2015, San Juan, Puerto Rico, January 26-30, 2015, Revised Selected Papers*, pages 207–216, 2015.