RELAY COST BOUNDING FOR CONTACTLESS EMV PAYMENTS

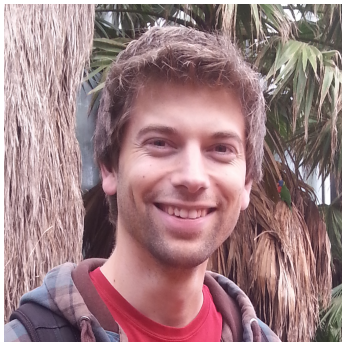
Tom Chothia
Univ. of Birmingham

Introduction

- This talk is about relay attacks against EMV PayWave Cards.
- We build a relay that can be just as fast as real cards, using easily available hardware (phones).
- We show that time bounding of the current protocols is difficult/ impossible.
- We propose a small change to the protocols that would allow time bounding that stops relay attacks using NFC phones.
- We propose a new method to formally verify the correctness of the protocol.

Co-authors

Nijmegen



Joeri de Ruiter

Jordi van den Breekel

Birmingham

Flavio D. Garcia

Matt Thompson

PayWave & PayPass

RFID card basics

- We have past work on e-passports and Mifare Classic.
- Low level is ISO 14443, protocol commands are based on ISO 7816.
- ISO 7816-like cards store data in records, with a few basic commands to authenticate and perform crypto.
- Exactly what crypto the cards do, varies between applications.

Let's Build a Relay

Let's Go Shopping (2010)

Relay 1 field tests

- EAT, Pret, Go Coffee

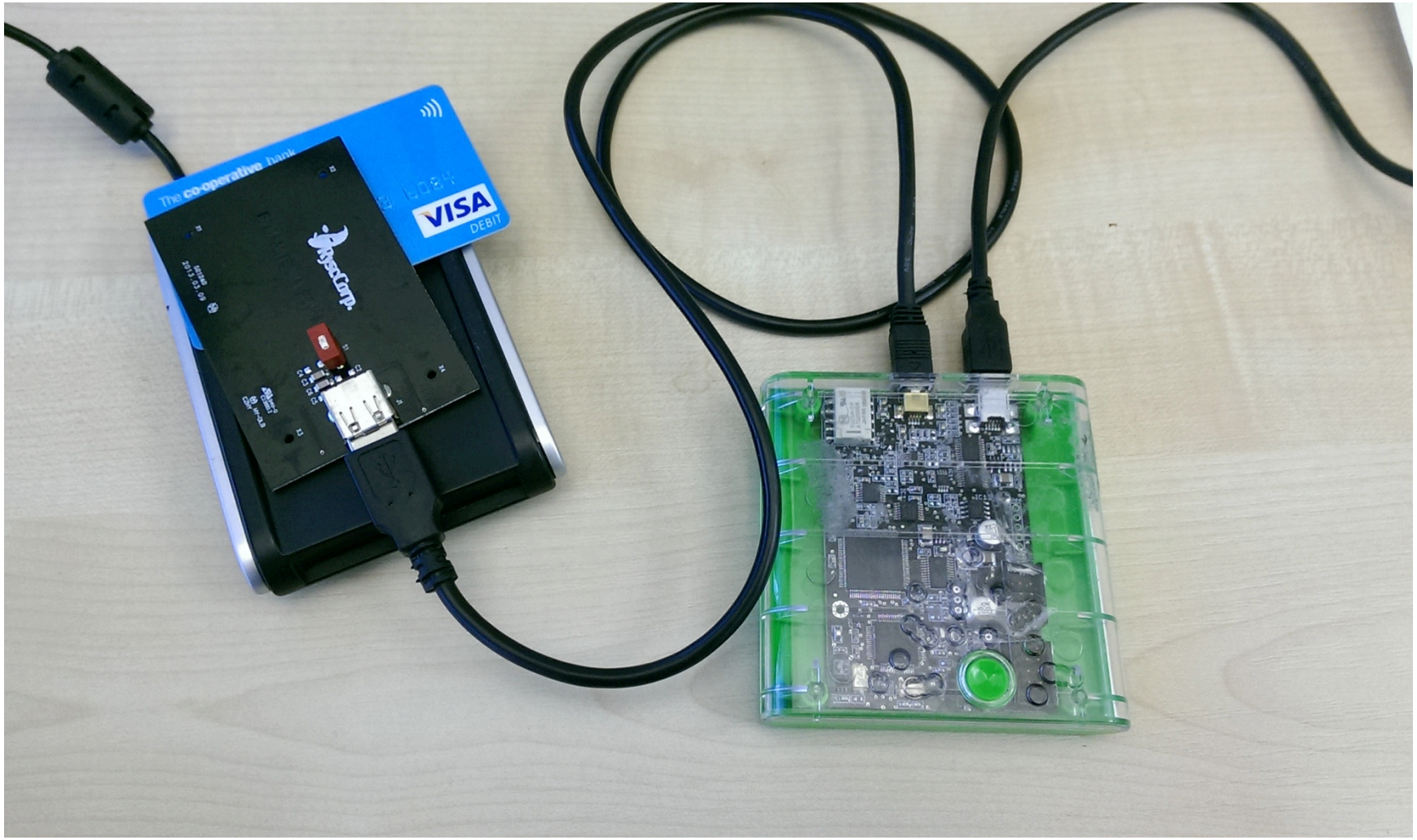
Let's Go Shopping (2010)

Relay 1 field tests

- EAT, Target, Go Coffee

Complete Failure

Proxmark III



ISO 14443: What We Expect:

Reader:	52	WUPA (wake up)
Tag:	44 03	ATQA (Respond)
Reader	93 20	SELECT
Tag:	88 04 34 74 cc	UID (tags unique ID)
Reader	93 70 88 04 34 74 cc	SELECT card via UID

If there is more than one card present the reader picks a UID at random.

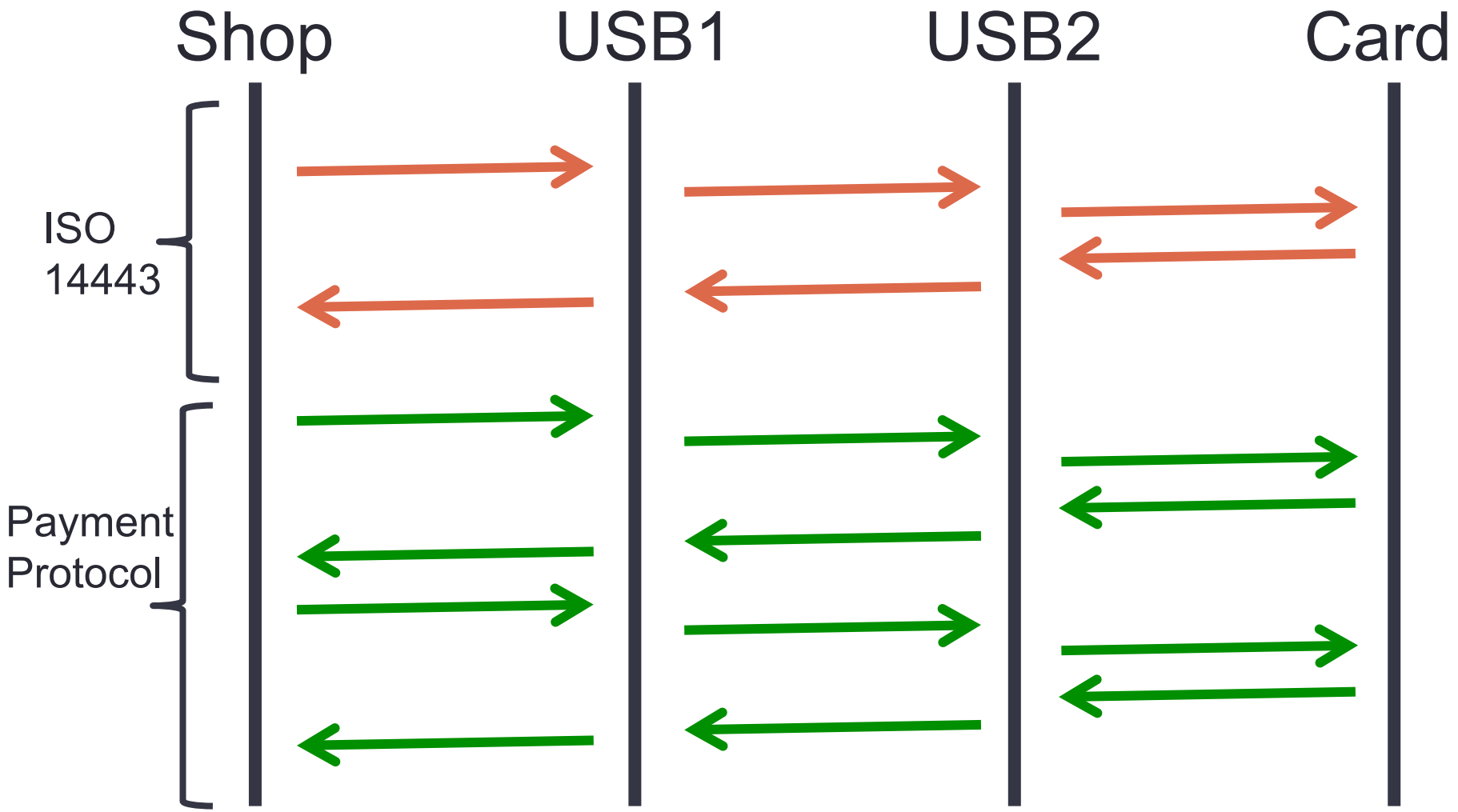
Messages include checksums (not shown). If messages from different cards collide, the reader sends SELECT again.

ISO 14443, What we actually saw:

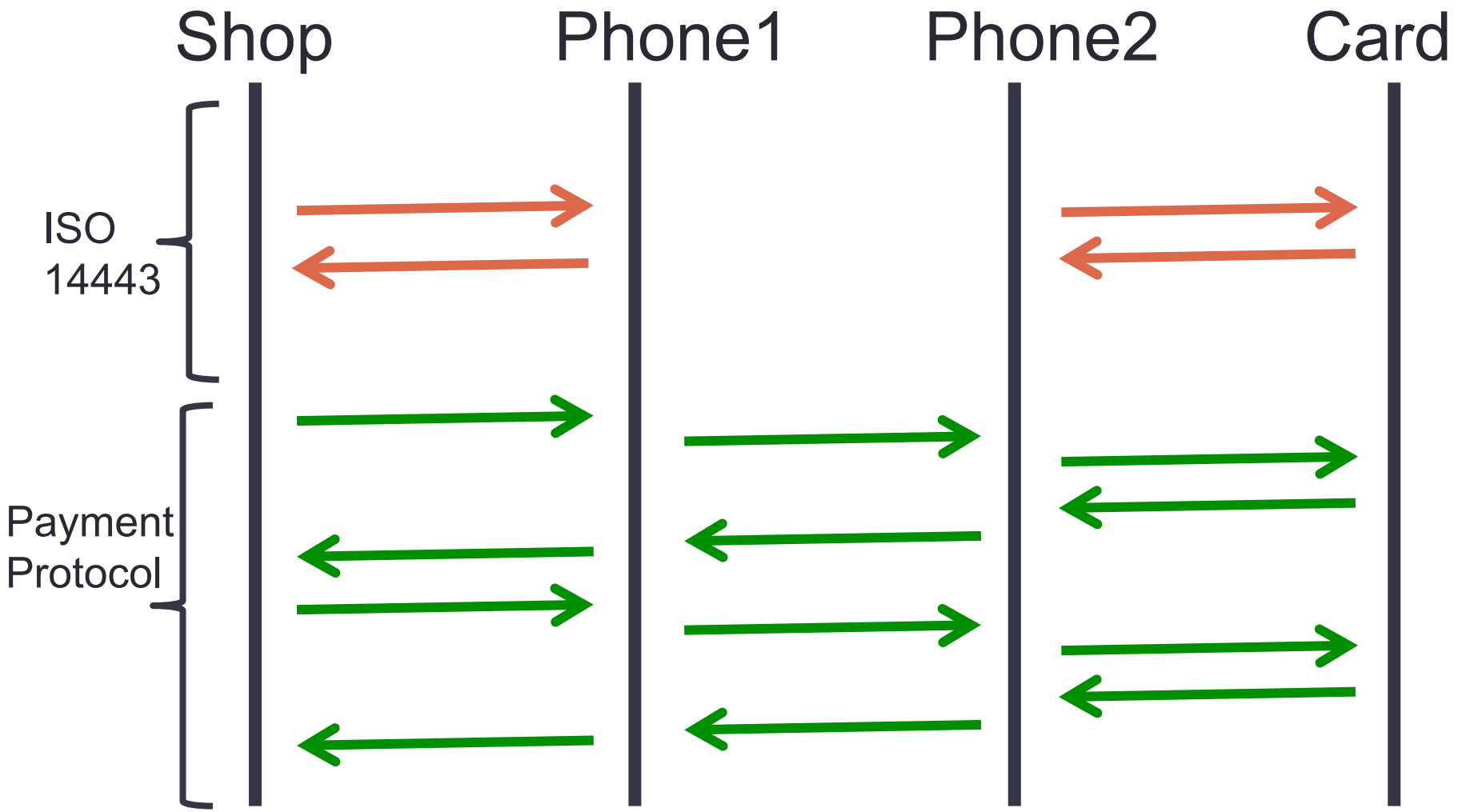
Reader:	52	WUPA (wake up)
Reader:	52	WUPA (wake up)
Tag:	04 00	ATQA (Respond)
Reader:	52	WUPA (wake up)
Tag:	04 00	ATQA (Respond)
Reader:	93 20	*SELECT*
Tag:	d4 fa 50 cb b5	*UID Response*
Reader:	52	WUPA (wake up)

The USB relay is too slow to get the ISO 14443 commands to the reader in time.

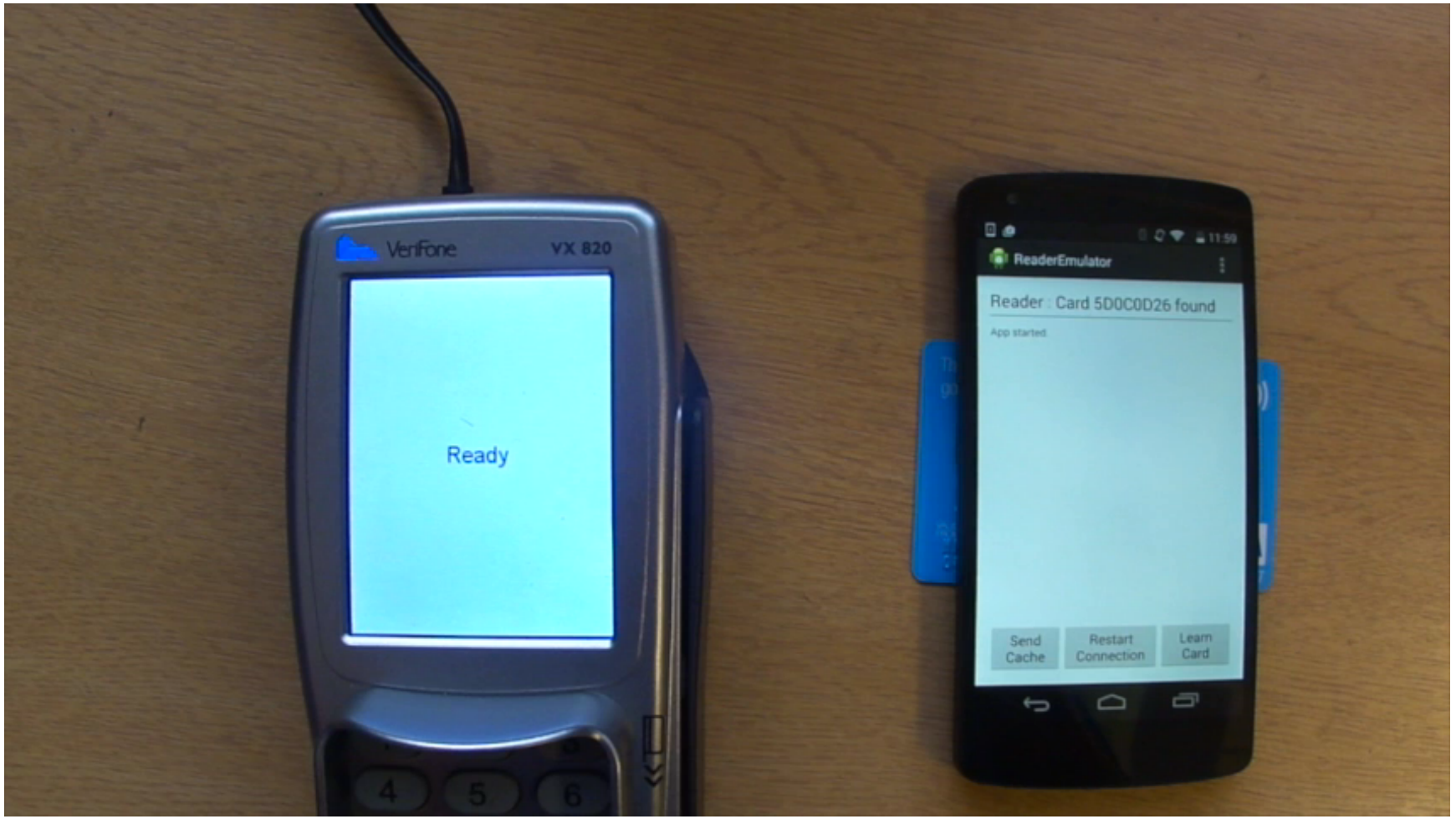
Relay With USBs



Relay With Phones



Relay in Action



Others done relay.

- *Practical NFC peer-to-peer relay attack using mobile phones.*
  - Lishoy Francis, Gerhard Hancke, Keith Mayes, and Konstantinos Markantonakis.
  - Proceedings of the 6th International Conference on Radio Frequency Identification: Security and Privacy Issues, RFIDSec'10,

- *The dangers of verify PIN on contactless cards.*
  - M. Emms, B. Arief, T. Defty, J. Hannon, F. Hao, and A. van Moorsel.
  - Technical report. CS-TR-1332.

- Also use phones for the relay.
  - Run ISO 14443 themselves, don't relay it.
  - Cheap, easily available, not suspicious.

Traffic:

Reader: 00A404000E325041592E5359532E444446303100

Tag: 6F378407A0000000031010A52C500A56495341204445
4249549F38189F66049F02069F03069F1A0295055F2A029A039C019F37045F2D026
56E9000

…

Reader80A800002383213220400000000000030000000000000826000000000082
614091500338F

507800Tag7781C29F4B81804D8EC3F85EB28D9C8828E2238BFE8F922F89D08DE
DA061DE7270CF6EB015109D58DC58B34706CED0BFA24A28ED3E6AE0B2908617
D34199B0A3BD298187376F639F65203C84EEE7BC60B4D14F649E67C62162CAF5
3045E8D5A2A99E39589483A28DF24941C6AF486FEEBA0A8C6DB33978309EFF87
FFF9984C9DECDFDCE6728DB19404100203009F1007060A0A039000005713463565
8326570935D160420151400000001001F820220009F360200579F26083501E6BD09
8562889F6C0210009000

Traffic:

Reader: 00A404000E325041592E5359532E444446303100

Tag: 6F378407A0000000031010A52C500A56495341204445
4249549F38189F66049F02069F03069F1A0295055F2A029A039C019F37045F2D026
56E9000

…

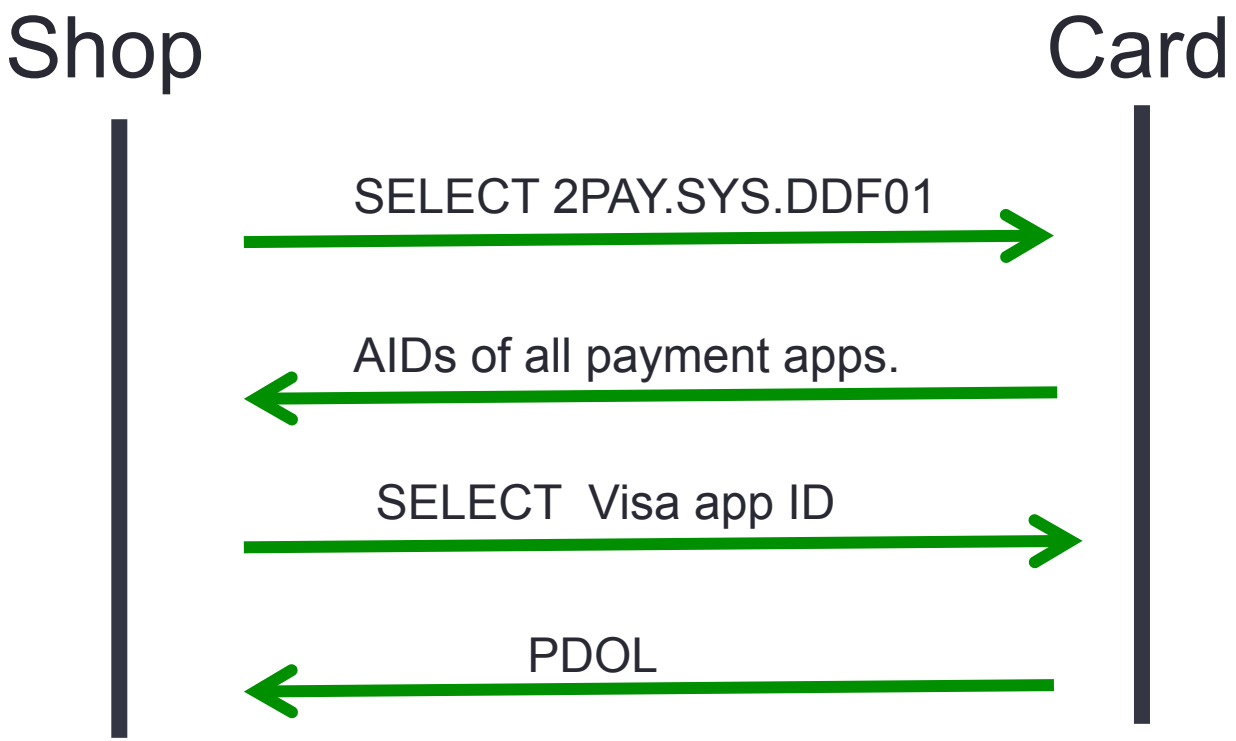
Reader80A800002383213220400000000000030000000000000826000000000082
614091500338F

507800Tag7781C29F4B81804D8EC3F85EB28D9C8828E2238BFE8F922F89D08DE
DA061DE7270CF6EB015109D58DC58B34706CED0BFA24A28ED3E6AE0B2908617
D34199B0A3BD298187376F639F65203C84EEE7BC60B4D14F649E67C62162CAF5
3045E8D5A2A99E39589483A28DF24941C6AF486FEEBA0A8C6DB33978309EFF87
FFF9984C9DECDFDCE6728DB19404100203009F1007060A0A03900000571346356
58326570935D160420151400000001001F820220009F360200579F26083501E6BD09
8562889F6C0210009000

The Specification (over 1600 pages)

- Reader has a CAs public key.
- Card has:
  - Symmetric key shared with bank
  - Certificate for a signing key.
- Static data signed by bank
  - CC no (PAN)., exp. date., etc.
- Card generates
  - A cryptogram (AC) to send to the bank as evidence of the transaction,
  - A signature (SDAD) that is checked by the bank.

Visa's Protocols



PDOL = Processing Options Data Object List

- list of data the reader must provide to the card.

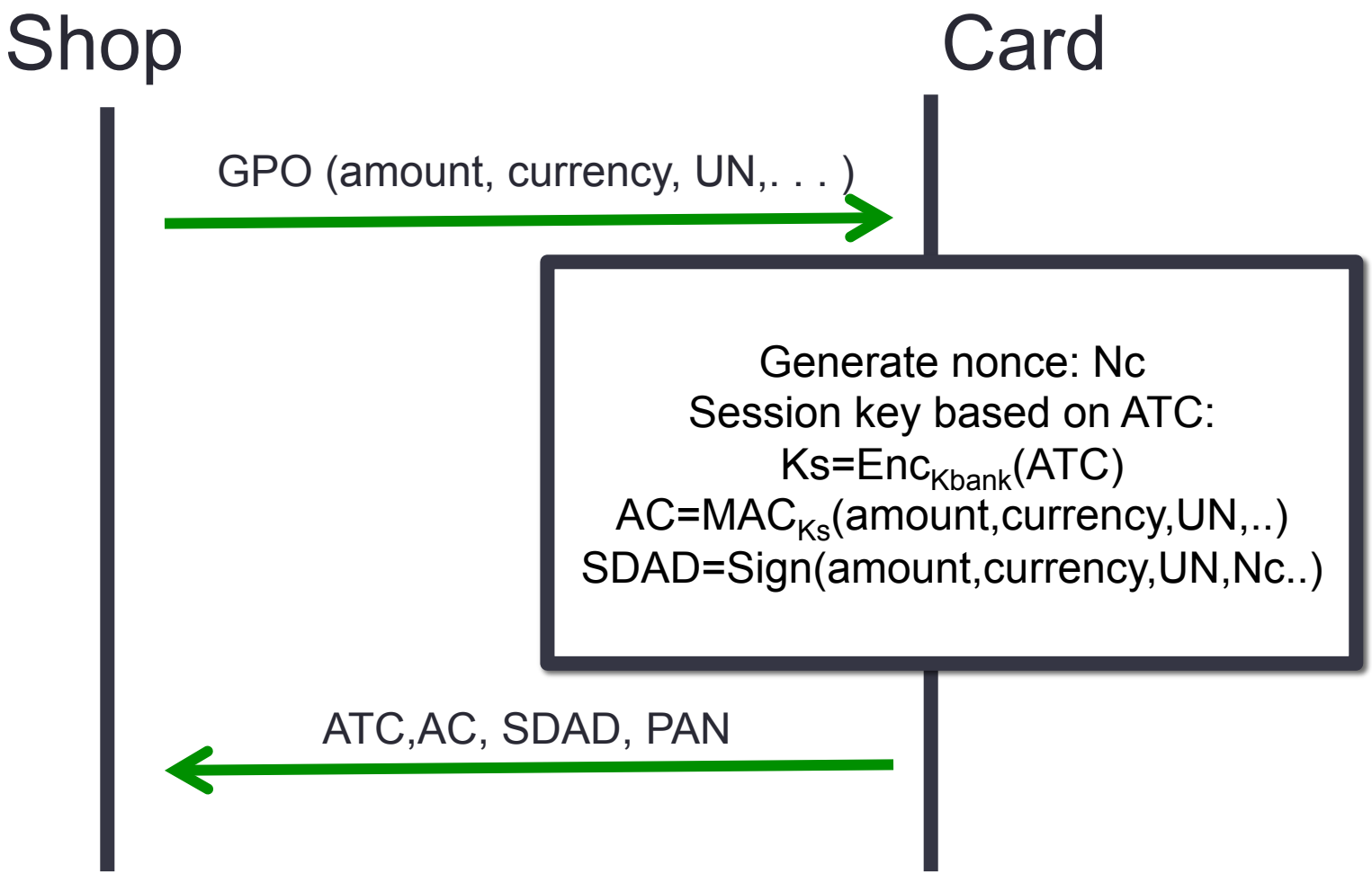
PDOL

9F38189F66049F02069F03069F1A0295055F2A029A039C019F37045F2D02656E9000

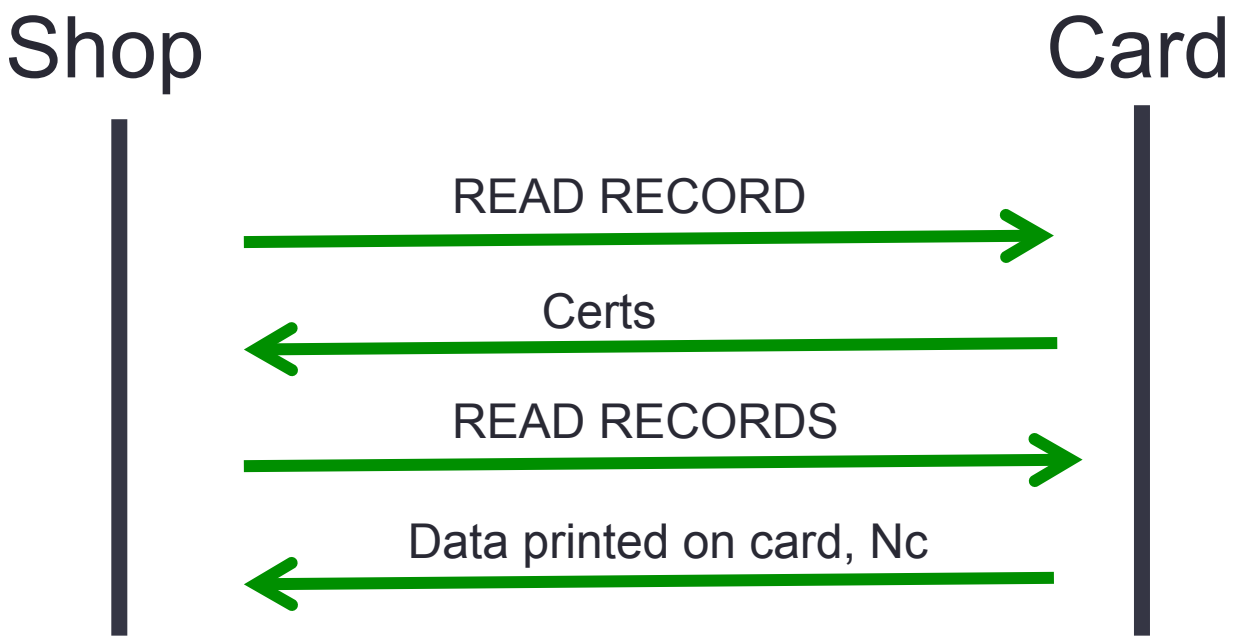
which parses as:

```
9F38 | len:18    Processing Options Data Object List (PDOL)
          9F66   len:04          Card Production Life Cycle
          9F02   len:06          Amount, Authorised (Numeric)
          9F03   len:06          Amount, Other (Numeric)
          9F1A   len:02          Terminal Country Code
            95   len:05          Terminal Verification Results
          5F2A   len:02          Transaction Currency Code
            9A   len:03          Transaction Date
            9C   len:01          Transaction Type
          9F37   len:04          Unpredictable Number
```

Visa's Protocols

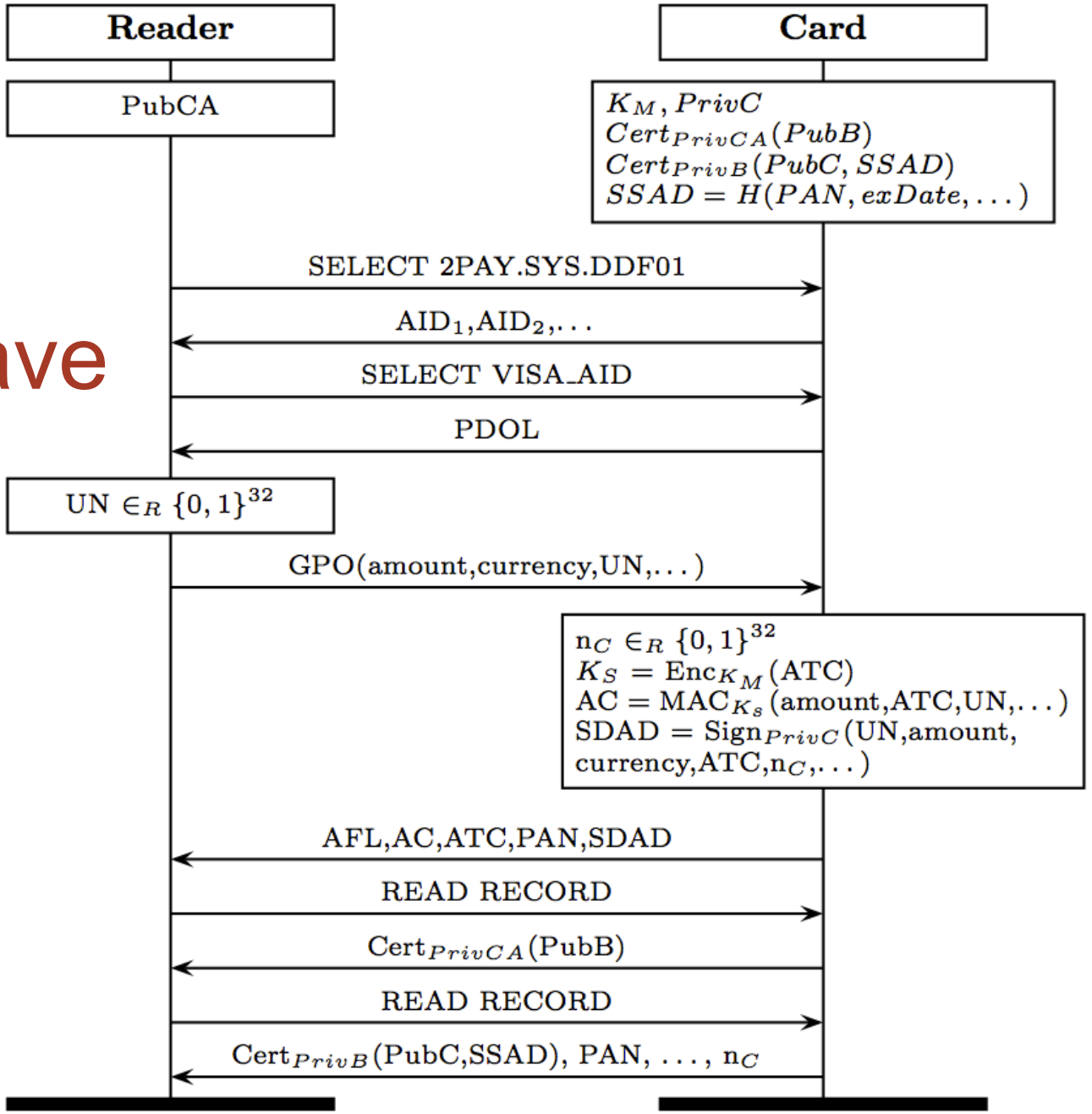


Visa's Protocols

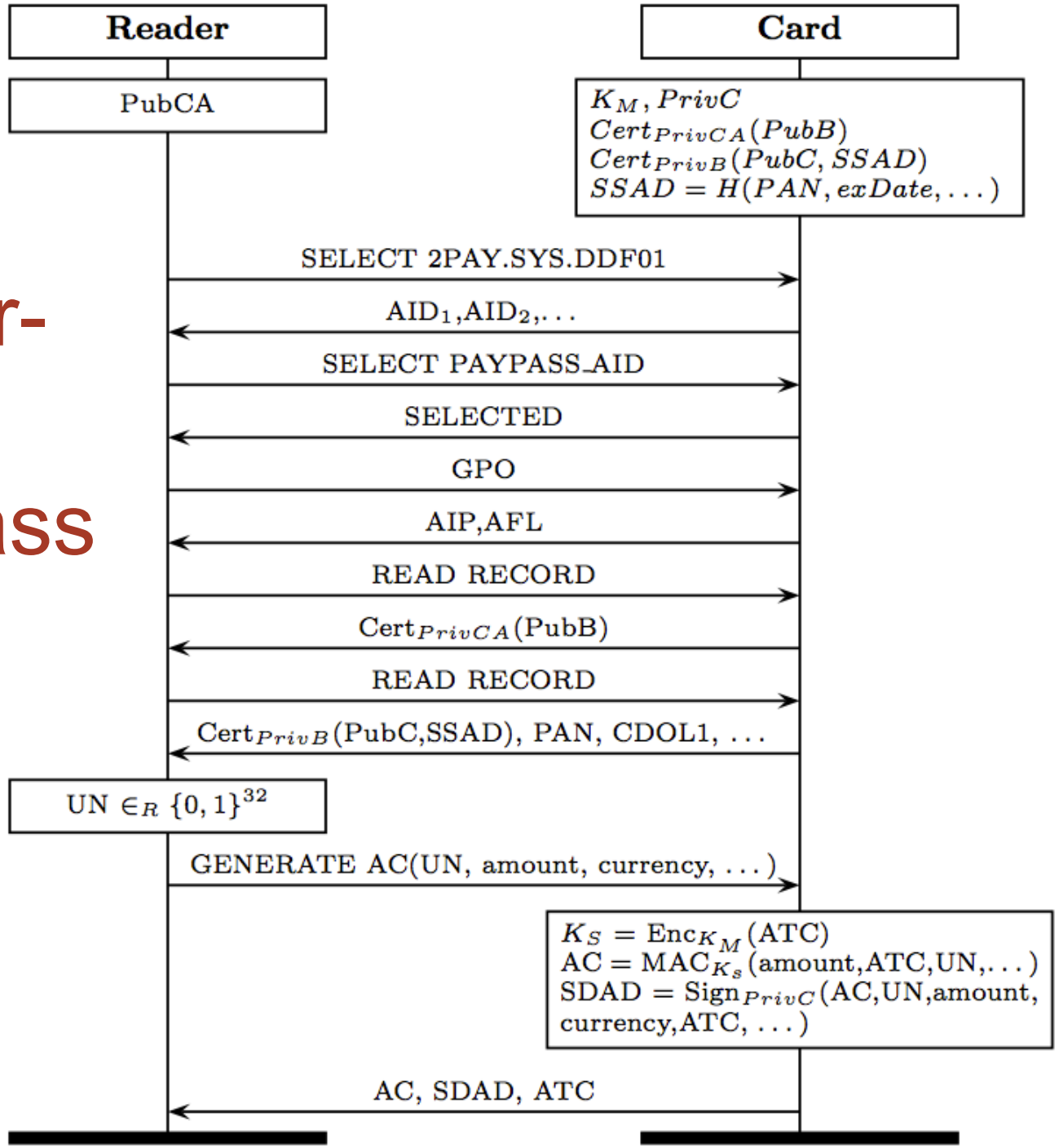


- Shop reader then checks the signature on the SDAD data.
- If this is correct it shop reader accepts the payment and sends the AC to the bank.
- The bank checks the AC and transfers the money.

Visa's PayWave

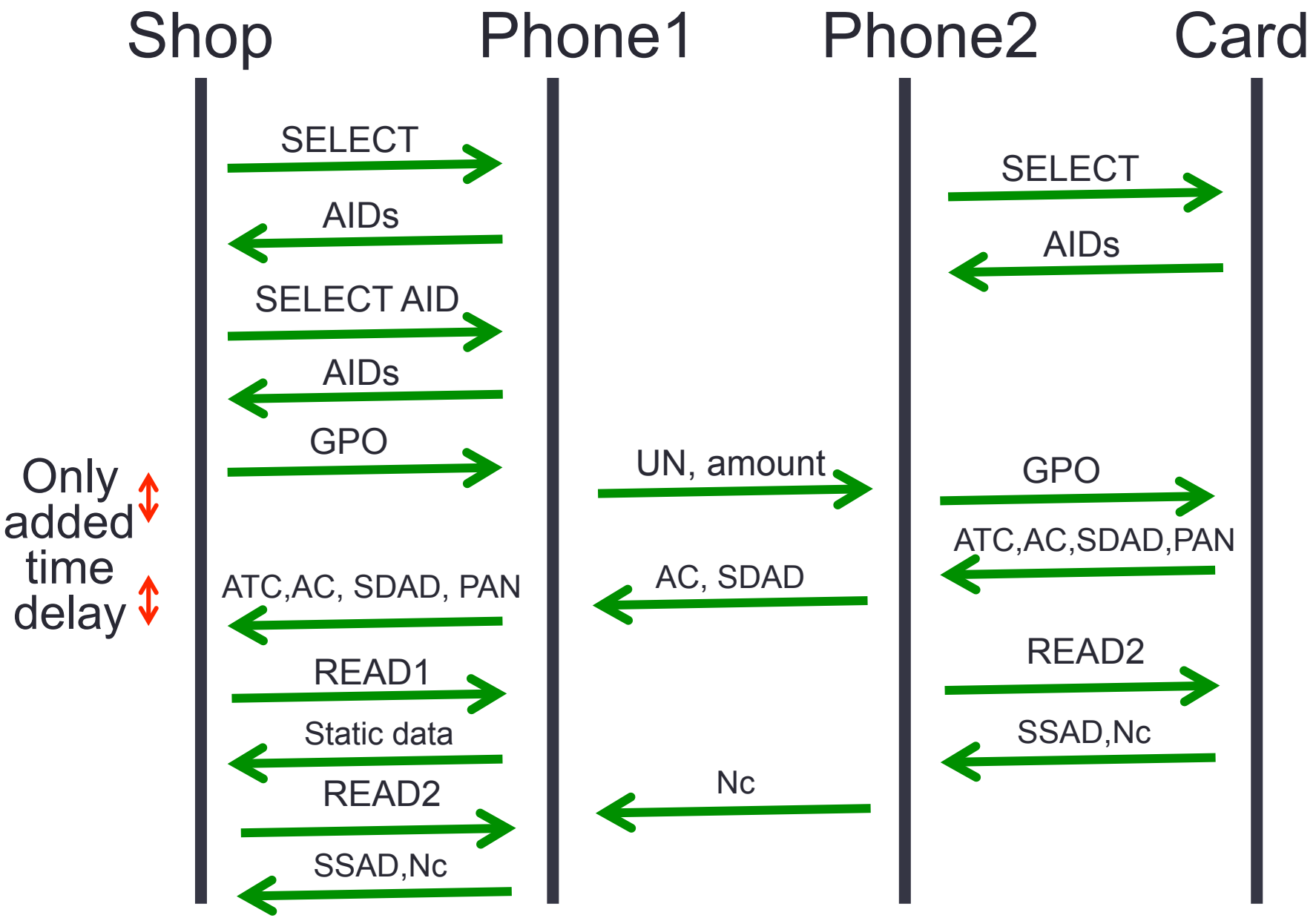


Master-card's PayPass



Stopping Relays: Idea 1

- Relaying all messages takes over a second.
- The spec. says that the transaction *should* complete in under 500ms.
- Can we stop relay attacks by adding a time out to the reader?
- Related question: can we make the relay faster?


Shop
Phone1
Phone2
Card
SELECT
AIDs
SELECT AID
AIDs
GPO
UN, amount
GPO
ATC,AC,SDAD,PAN
AC, SDAD
ATC,AC, SDAD, PAN
READ1
Static data
READ2
SSAD,Nc
Nc
READ2
SSAD,Nc
Only added time delay

Relay timing

- We measured the exact transaction times for a number of cards.
  - Fastest 330ms
  - Slowest 637ms
- Fastest relayed transaction: 485ms
- Placement of card can have an affect > 80ms for longest messages.

- ABN Amro (Dutch)
  - Time for card to complete a purchase: 637ms
  - Time for relay to complete a purchase:627ms.

Stopping Relays: Idea 2

- Why not just time-bound the important crypto message?
  - GPO for Visa's payWave
  - GENERATE AC for Mastercard's PayPass

- **Problem**: these are the steps that require the cards to do crypto, which shows more variance than any other messages.
  - Fastest payWave GPO: 105ms
  - Slowest payWave GPO: 364ms

- We were able to relay the fastest response in 208ms.

“Keep Your Enemies Close: Distance Bounding Against Smartcard Relay Attacks ” Drimer and Murdoch

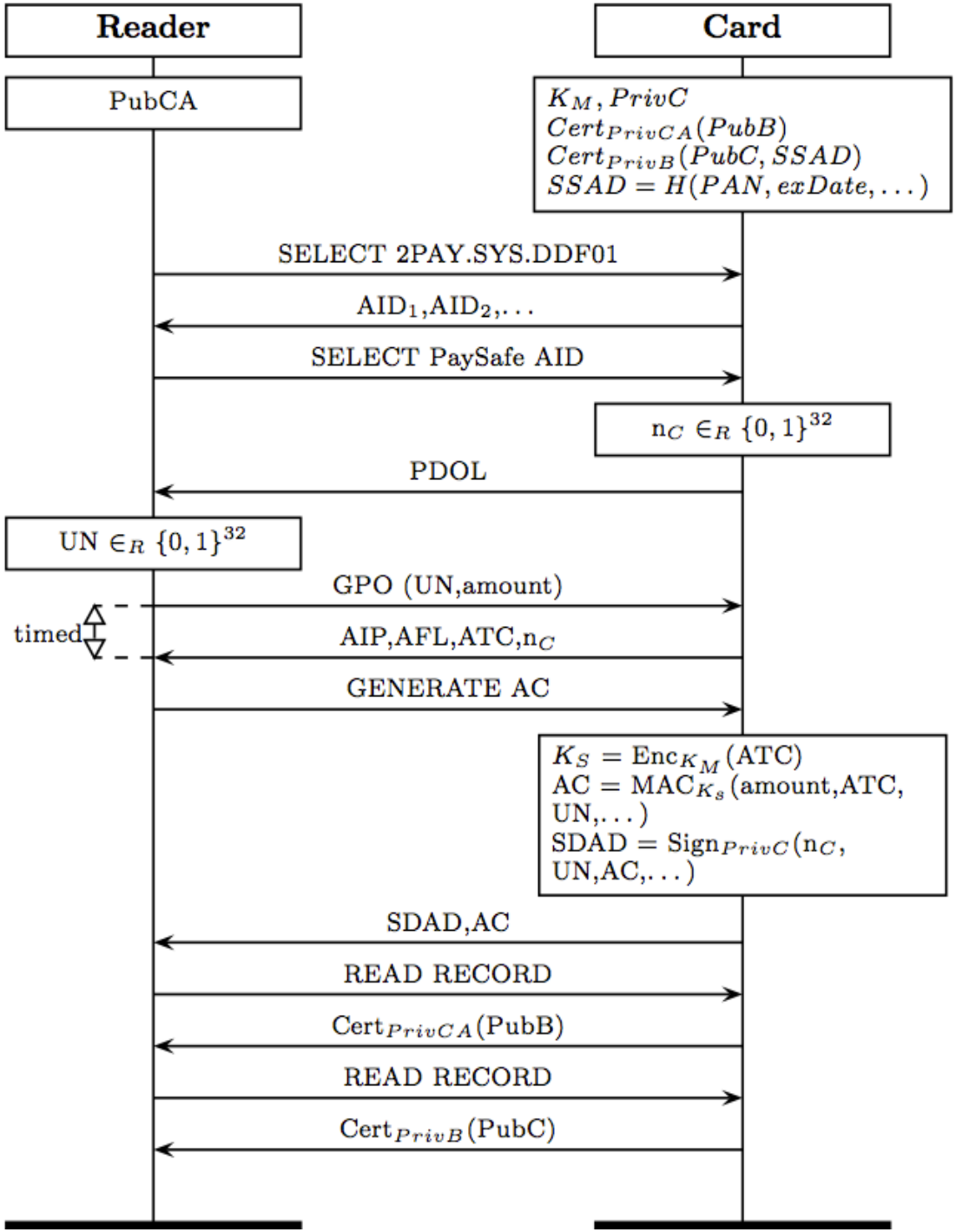
- Reader times to nano second level.
- Uniform card hardware, clock speed known.
- Attacker that can relay close to the speed of light.
- Bounds distance to 100s of meters.
- Major changes to protocol and hardware.

- Reader times to micro second level.
- Variable card hardware
- Attacker uses cheap, slow hardware. e.g. phones.
- Stops attack inside same shop.
- Only change the payloads of existing message.

Key Observation Protocol

- The non-crypto messages are predictable and therefore can be time bound.
- But in the current protocols all none crypto messages can be cached.
- We tweak the protocol, so there is a non-crypto message that can be time-bound.

PaySafe


Reader
Card
PubCA
$K_M, PrivC$
$Cert_{PrivCA}(PubB)$
$Cert_{PrivB}(PubC, SSAD)$
$SSAD = H(PAN, exDate, \ldots)$
SELECT 2PAY.SYS.DDF01
$AID_1, AID_2, \ldots$
SELECT PaySafe AID
$n_C \in_R \{0,1\}^{32}$
PDOL
$UN \in_R \{0,1\}^{32}$
GPO (UN,amount)
timed
AIP,AFL,ATC,n_C
GENERATE AC
$K_S = Enc_{K_M}(ATC)$
$AC = MAC_{K_s}$(amount,ATC, UN,...)
$SDAD = Sign_{PrivC}(n_C$, UN,AC,...)
SDAD,AC
READ RECORD
$Cert_{PrivCA}$(PubB)
READ RECORD
$Cert_{PrivB}$(PubC)

PaySafe Timing

- Time for cards to respond to a message of this length = 28 to 36ms.
- Time to relay a message of this length: 100ms
- So the reader will time out after 80ms.
- No phone or USB reader will be able to relay this message.
- Faster purpose build hardware costs tens of thousands of pounds.

Formal verification

a, b, c, k, s	names
$f(M_1, \ldots, M_n)$	constructor application
$D ::= g(M_1, \ldots, M_n)$	destructor application
$P, Q ::=$	processes
0	nil
$\overline{M}\langle N \rangle.P$	output
$M(x).P$	input
$P \mid Q$	parallel composition
$!P$	replication
$\nu\, a.P$	create new name

Protocol model

$$
\begin{aligned}
\mathit{Reader} = {} & \overline{c}\langle \text{SELECT,PAYSYSDDF}\rangle. \\
& c(=\text{AID}). \\
& \overline{c}\langle \text{SELECT,aid}\rangle. \\
& c(=\text{PDOL}). \\
& \nu\, n_R.\overline{c}\langle \text{GPO,amt},n_R\rangle. \\
& c(n'_C,\text{atc}',\text{PAN}'). \\
& \overline{c}\langle \text{GENERATE AC}\rangle. \\
& c(\text{sdad}',\text{ac}'). \\
& \overline{c}\langle \text{READ RECORD}\rangle.c(\mathit{ssad}'). \\
& \overline{c}\langle \text{READ RECORD}\rangle.c(\text{cert}'). \\
& \text{let cardPub}'_K = \\
& \qquad \mathit{check}(\text{cert}',\mathit{pk}(\text{bank}_K)) \\
& \text{if } \mathit{check}(\text{sdad}',\text{cardPub}'_K) = \\
& \qquad (n_R,n'_C,\text{amt,atc}',\text{ac}') \\
& \overline{c}\langle \text{readerAccepts}\rangle
\end{aligned}
$$

$$
\begin{aligned}
\mathit{Card} = {} & c(=\text{SELECT},=\text{PAYSYSDDF}). \\
& \overline{c}\langle \text{AID}\rangle. \\
& c(=\text{SELECT},=\text{AID}). \\
& \nu\, n_C.\overline{c}\langle \text{PDOL}\rangle. \\
& c(=\text{GPO,amt}',n'_R). \\
& \overline{c}\langle n_C,\text{atc,PAN}\rangle. \\
& c(=\text{GENERATE AC}). \\
& \text{let mac}_K = \mathit{genkey}(\text{atc,bank}_K) \text{ in} \\
& \text{let ac} = \mathit{mac}((\text{amt}',n'_R,\text{atc}),\text{mac}_K) \text{ in} \\
& \text{let sdad} = \\
& \quad \mathit{sign}((n_R,n_C,\text{amt,atc,ac}),\text{card}_K) \text{ in} \\
& \overline{c}\langle \mathit{sdad}\rangle. \\
& c(=\text{READ RECORD}). \\
& \overline{c}\langle \mathit{sign}((\text{PAN,expDate}),\text{bank}_K)\rangle. \\
& c(=\text{READ RECORD}).\overline{c}\langle \mathit{cert}\rangle
\end{aligned}
$$

$$
\begin{aligned}
\mathit{System} = {} & \nu\,\text{bank}_K.(\overline{c}\langle \mathit{pk}(\text{bank}_K)\rangle \mid\ !\nu\,\text{amount}.!\mathit{Reader} \\
& \mid\ !(\nu\,\text{PAN}.\nu\,\text{expDate}.\nu\,\text{card}_K.\text{let cert} = \mathit{sign}(\mathit{pk}(\text{card}_K),\text{bank}_K) \text{ in } !\nu\,\text{atc}.!\mathit{Card}))
\end{aligned}
$$

Key Observation

- The attackers can do anything they want before or after the time-bound step.
- Attackers can reply to the time-bound step with their own message or a replayed message.
- The attacker does not have time to
  - look at the time-bound step,
  - and then send a message to the card
  - and then reply to the reader.

This is equivalent to saying that the attacker cannot talk to the card during the time bound step.

Key Observation

- In our formal model, we lock the card during the time-bound step.
  - It cannot communicate with the attacker or the reader.
- If the attacker can find a sequence of actions that allow the reader to successfully terminate, then there is a relay attack.
- If the reader cannot terminate then the protocol is safe from relay attacks.

Locking the Card Process Using Phases

- Phases enforce order on processes:
  - e.g. 2:P | 2:a(x).3:Q | 3:R
- To model relays we use three phases: 0,1 & 2.
- The reader, attacker and card can all act in phase 0 & 2
- The attacker can act in phase 0,1 & 2
- The reader moves to phase 1 before sending its time action & moves to phase 2 when it gets the reply.

Locking the Card Process Using Phases

- Cards and readers must be able to jump from phase 1 to 3 at any point

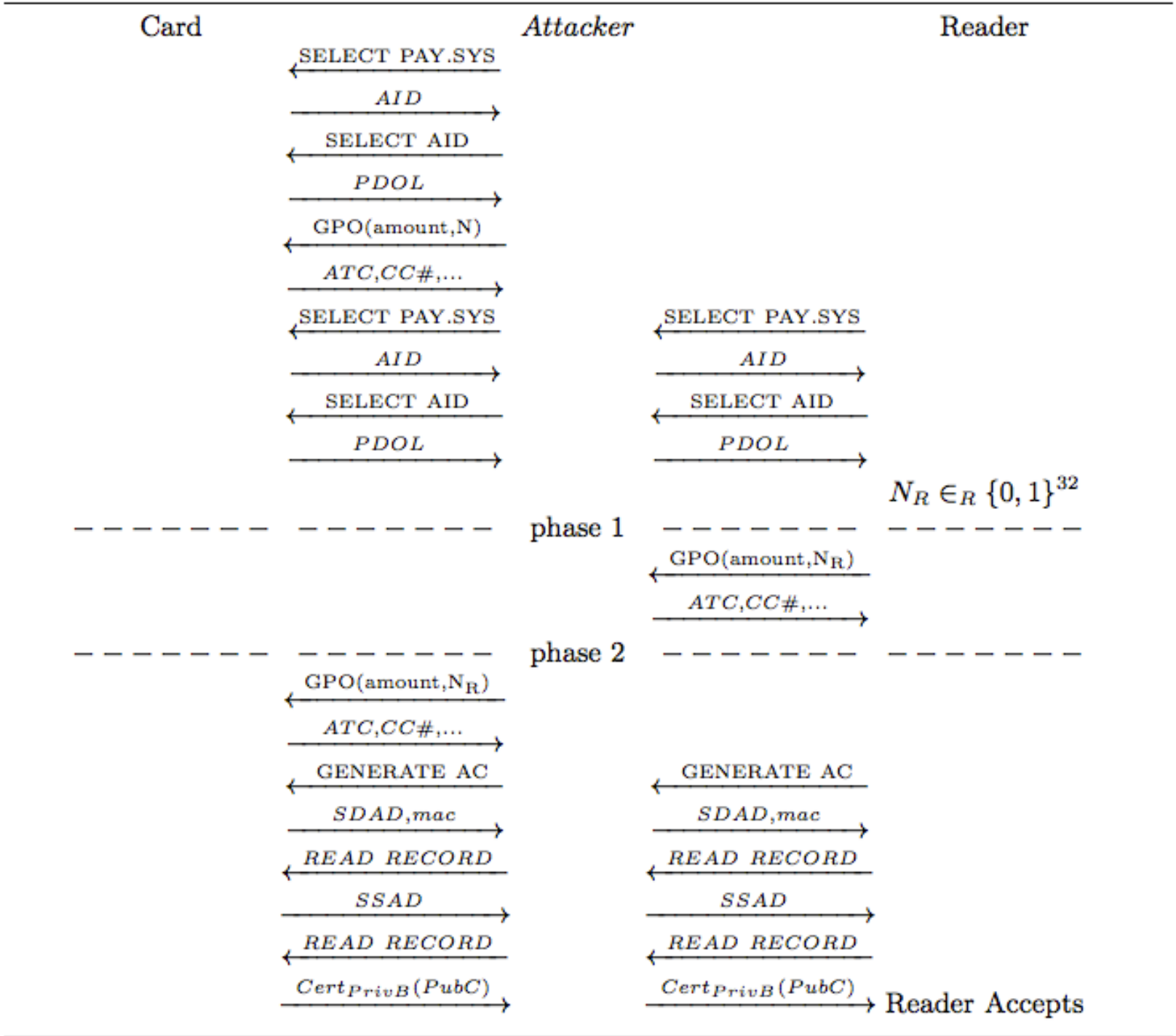
$$phasesSet(P) = \{C[2:M(x).P'] : P = C[M(x).P']\}$$
$$phases(P) = !P_1 \mid !P_2 \mid \cdots \mid !P_n \quad where \{P_1, \ldots, P_n\} = phasesSet(P)$$

$$\begin{aligned}
TestReader = \ldots \\
&c(=\text{PDOL}).\nu\, n_R. \\
&1:\bar{c}\langle \text{GPO,amt}, n_R\rangle. \\
&c(n'_C, \text{atc}', \text{ccNo}'). \\
&2:\bar{c}\langle \text{GENERATE AC}\rangle. \\
&c(\text{sdad}', \text{ac}'). \\
&\ldots \\
&\text{if } check(\text{scad}', \text{cardPub}'_K) = \\
&\qquad (n_R, n'_C, \text{amt}, \text{atc}', \text{ac}') \\
&\bar{c}\langle \text{phaseReaderAccepts}\rangle
\end{aligned}$$

$$\begin{aligned}
SystemP &= \nu\, \text{bank}_K.(\bar{c}\langle pk(\text{bank}_K)\rangle \\
&\mid \nu\, \text{amount}.TestReader \\
&\mid !\nu\, \text{amount}.Readers \\
&\mid !(\nu\, \text{ccNo}.\nu\, \text{expDate}.\nu\, \text{card}_K. \\
&\quad \text{let cert} = sign(pk(\text{card}_K), \text{bank}_K) \\
&\quad \text{in } !\nu\, \text{atc}.Cards))
\end{aligned}$$

where:

$$\begin{aligned}
Cards &= phases(Card) \\
Readers &= phases(Reader)
\end{aligned}$$


Card
Attacker
Reader
SELECT PAY.SYS
AID
SELECT AID
PDOL
GPO(amount,N)
ATC,CC#,...
SELECT PAY.SYS
AID
SELECT AID
PDOL
SELECT PAY.SYS
AID
SELECT AID
PDOL
$N_R \in_R \{0,1\}^{32}$
phase 1
GPO(amount,N_R)
ATC,CC#,...
phase 2
GPO(amount,N_R)
ATC,CC#,...
GENERATE AC
SDAD,mac
READ RECORD
SSAD
READ RECORD
$Cert_{PrivB}(PubC)$
GENERATE AC
SDAD,mac
READ RECORD
SSAD
READ RECORD
$Cert_{PrivB}(PubC)$
Reader Accepts

Conclusion

- We have shown that fast relay attacks are possible against PayWave and PayPass
- These attacks cannot be easily stopped by time-bounding the current protocols.
- We have proposed a very small change to the protocols that will make time-bounding an affective way to stop relays using phones and USB NFC.
- We have shown how these kinds of protocols can be formally verified.