/Rooted 2015

On Relaying NFC Payment Transactions using Android devices

Pepe Vila
Ricardo J. Rodríguez

other title candidates…

- Holystic relay attacks on NFC cyber-payments with Android mobile cloud phones
- ENE-FE-SÉ relai con droides movibles

about

- D. Pepe Vila
  - IT security consultant at EY
  - University of Zaragoza
  - @cgvwzq

- Dr. Ricardo J. Rodríguez
  - PhD by University of Zaragoza
  - Senior Researcher at University of León
  - @RicardoJRdez

agenda

1. NFC: what and how
2. EMV (a.k.a. credit card payments)
3. Relay attacks
4. Android NFC history
5. NFC implementation in Android
6. PoC + attack scenarios
7. Limitations and conclusions

agenda

1. **NFC: what and how**
2. EMV (a.k.a. credit card payments)
3. Relay attacks
4. Android NFC history
5. NFC implementation in Android
6. PoC + attack scenarios
7. Limitations and conclusions

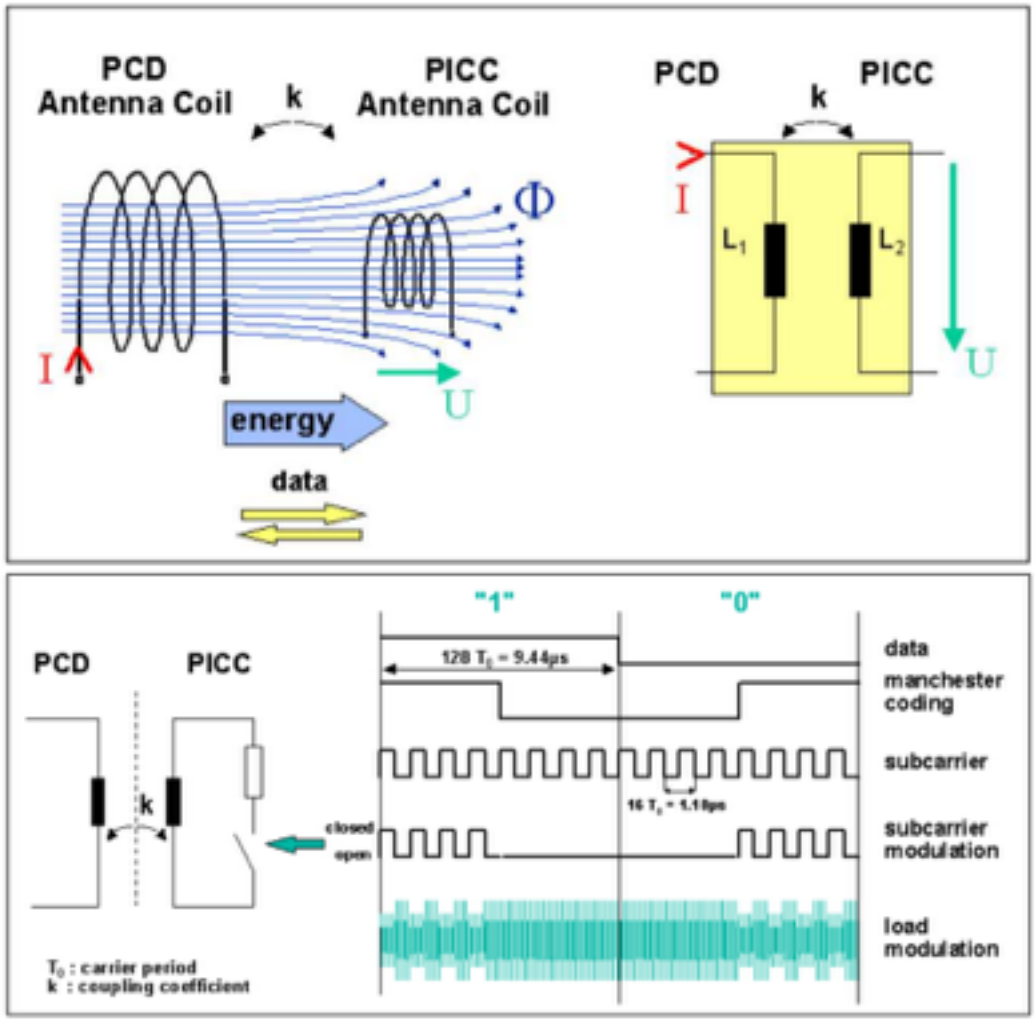
1. Near Field Communication

"Set of technologies that enable radio based communications between two devices in proximity"

William Shakespeare (uncredited)

- Based on RFID: ISO/IEC 14443, FeLiCa, and others
  - 13.56MHz
- Rates: 106 – 424 kbit/s
- Two main actors:
  - PCD (or reader) generates RF field
  - PICC (or tag) active/passive
- Distance: ≤10cm
- Short read range limitation = SECURITY ? (hard eavesdropping)

1. Near Field Communication



http://www.nxp.com/documents/application_note/AN78010.pdf

1. Near Field Communication

ISO/IEC 14443: International Standard for contactless integrated circuit cards

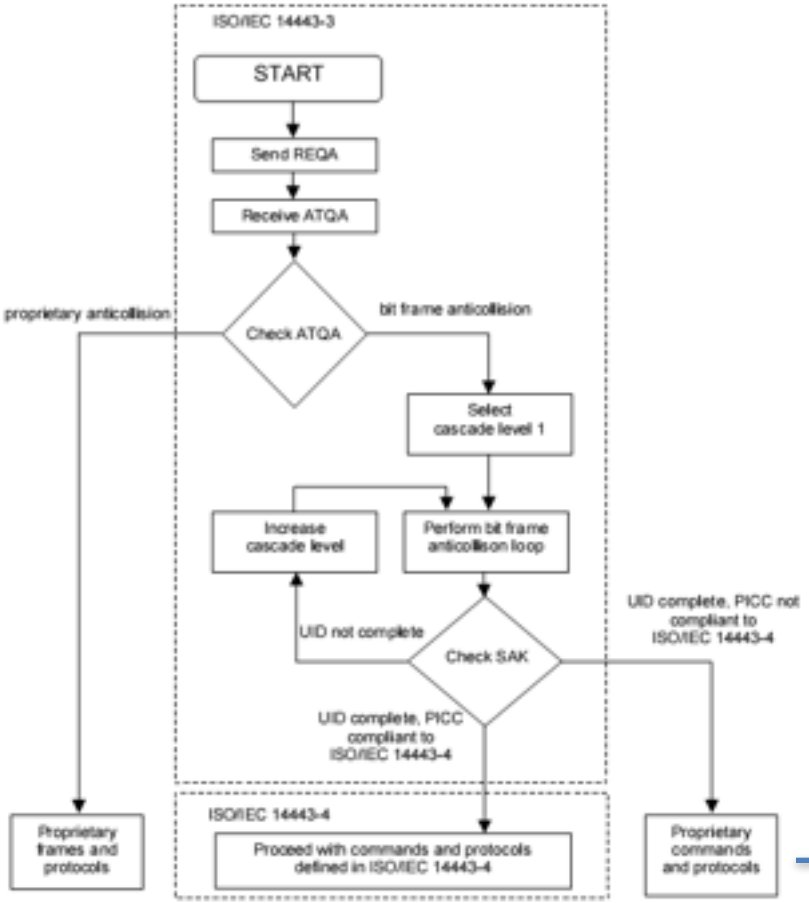
- Part 1: Defines size and physical characteristics
- Part 2: Powering and modulation schemes (type A & B)
- Part 3: Initialisation and anti-collision
- Part 4: Transmission protocol

Tags compliant with all parts are named Type 4 (or *IsoDep*)

Can use ISO/IEC 7816-4 APDUs for communication

1. Near Field Communication

Initialisation and anti-collision (ISO/IEC 14443-3A)



Distinct for Type A and B

Command set:

- `REQA`
- `WUPA`
- `ANTICOLLISION`
- `SELECT`
- `HLTA`

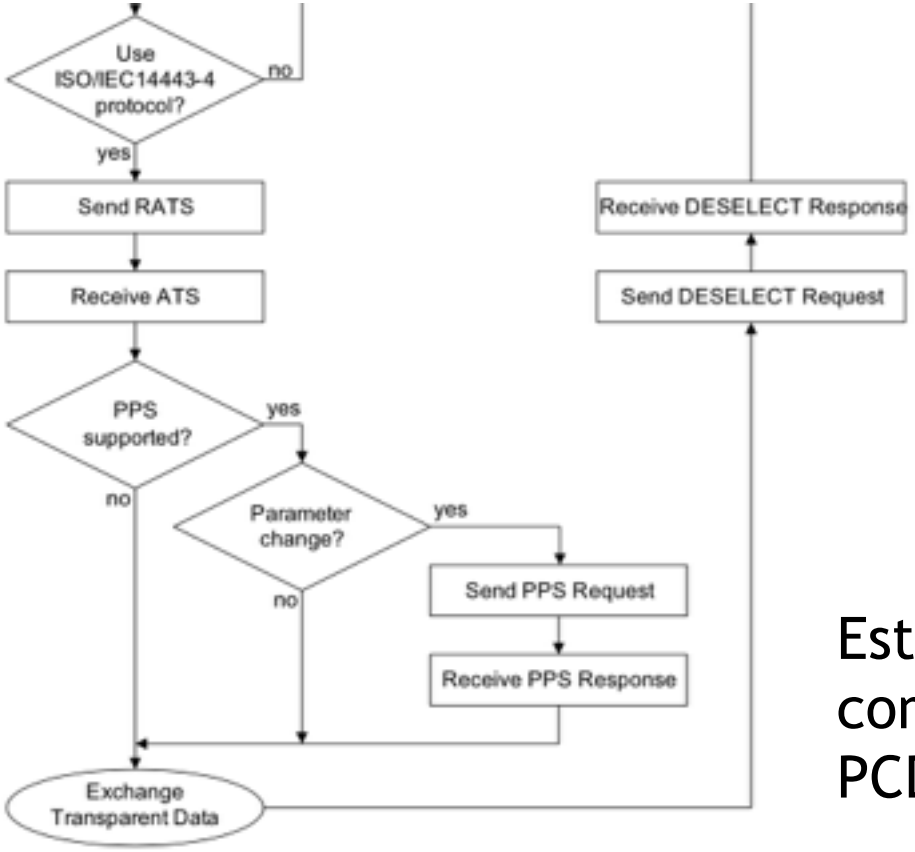
Use of CRCs

Low frame delay timings (FDT)

→ (e.g., MIFARE Classic)

1. Near Field Communication

Transmission Protocol (ISO/IEC 14443-4)



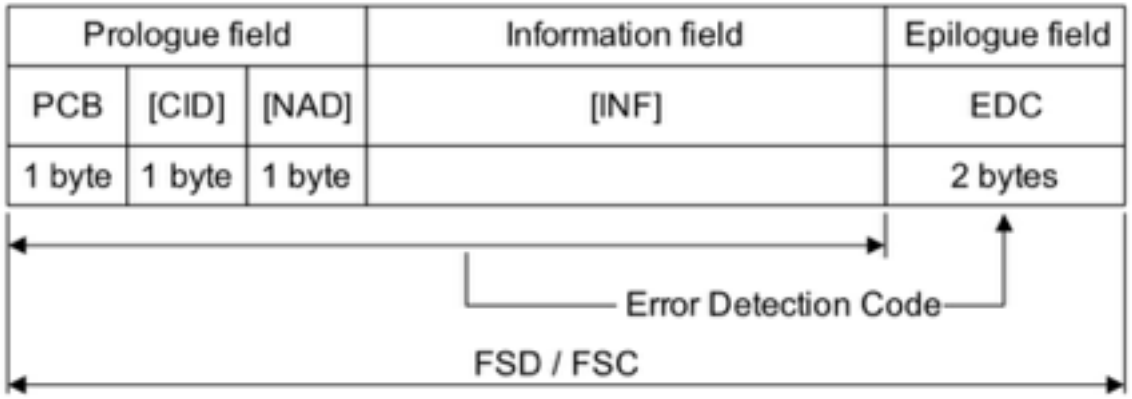
- RATS or *Request ATS*
- ATS or *Answer To Select*: tag connection parameters (e.g., frame size,timeouts…)
- PPS req/resp: allow PCDs to modify some parameters (if tag supports it)

Establishes a logical *half-duplex* communication channel between PCD and PICC

1. Near Field Communication

Transmission Protocol (ISO/IEC 14443-4)

Block Format used



Three types of blocks (depending on Protocol Control Byte):

- I-block: application layer information
- R-block: acknowledgments (empty `INF` field)
- S-block: control information (used for `WTX` and `DESELECT`)

1. Near Field Communication

"ISO/IEC 7816 is an international standard for electronic identification cards, specially smart cards."

Application Protocol Data Unit (or APDU):

- Communication unit between a reader and a SC
- Defined by ISO/IEC 7816-4
- Command-response pair

APDUs are encapsulated in the `INS` field

1. Near Field Communication

- APDU command: 4 byte header + 0..255 bytes data

Command APDU						
Header (required)				Body (optional)		
CLA	INS	P1	P2	Lc	Data Field	Le

- APDU response: 0..65536 bytes data + 2 status bytes

Response APDU		
Body (optional)	Trailer (required)	
Data Field	SW1	SW2

1. Near Field Communication

- Wide set of commands: `SELECT`, `READ/WRITE BINARY`, `GET DATA`, `GENERATE APP CRYPTOGRAM`…
- Used to interact with smart card applications and the File System.
- Special interest on `SELECT`:
  - Smart cards can have multiple applications
  - Each one with its own AID or *Application Identifier*
    - Registered Application Provider Identifier (RID)
    - Proprietary Application Identifier Extension (PIX)
  - Applications are isolated and have different files
  - Before any command the reader have to `SELECT` the specific AID that wants to talk (also exists implicit selection)

agenda

1. NFC: what and how
2. **EMV (a.k.a. credit card payments)**
3. Relay attacks
4. Android NFC history
5. NFC implementation in Android
6. PoC + attack scenarios
7. Limitations and conclusions

3. EMV (a.k.a. credit card payments)

"Standard Europay-Mastercard-VISA defines communication between smart cards, POS, and ATMs to authenticate credit/debit card transactions"

- APDU commands (defined by ISO/IEC 7816-3 and ISO/IEC 7816-4)
- Support for strong cryptographic (chip & PIN)
- Backwards compatibility, always prone to downgrade attacks.
- Few architecture changes with NFC payments compared to chip & PIN

(EMV protocols are another surface attack that we have not covered)

3. EMV (a.k.a. credit card payments)

NFC payments:

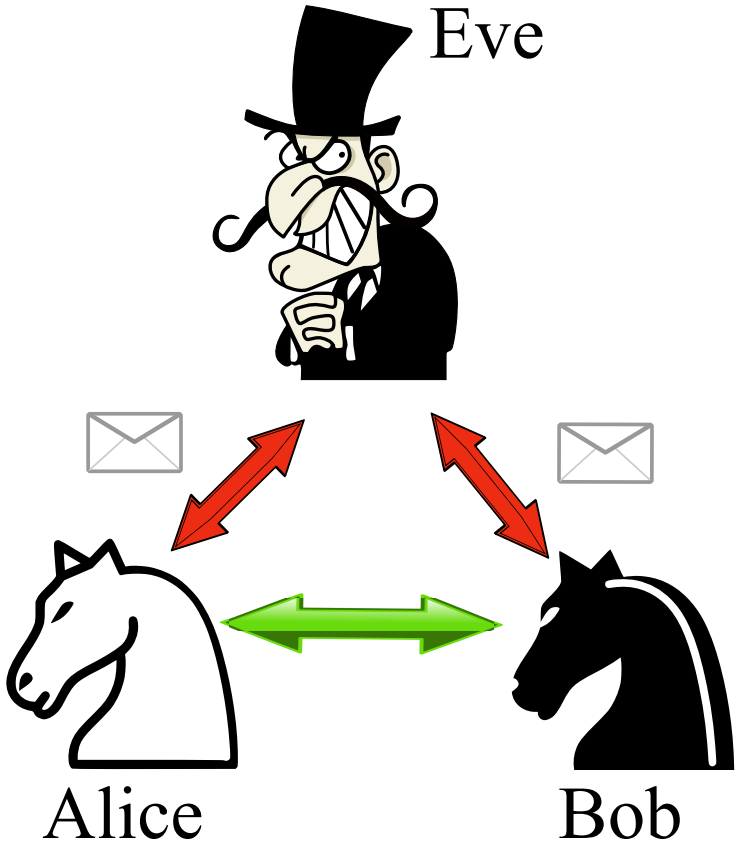
- Up to 20 GBP, 20€, US$50, 50CHF, CAD100 or AUD100
- Limit of using times without PIN verification
- Mag-stripe track emulation (CVV3 or dynamic CVV)

agenda

1. NFC: what and how
2. EMV (a.k.a. credit card payments)
3. **Relay attacks**
4. Android NFC history
5. NFC implementation in Android
6. PoC + attack scenarios
7. Limitations and conclusions

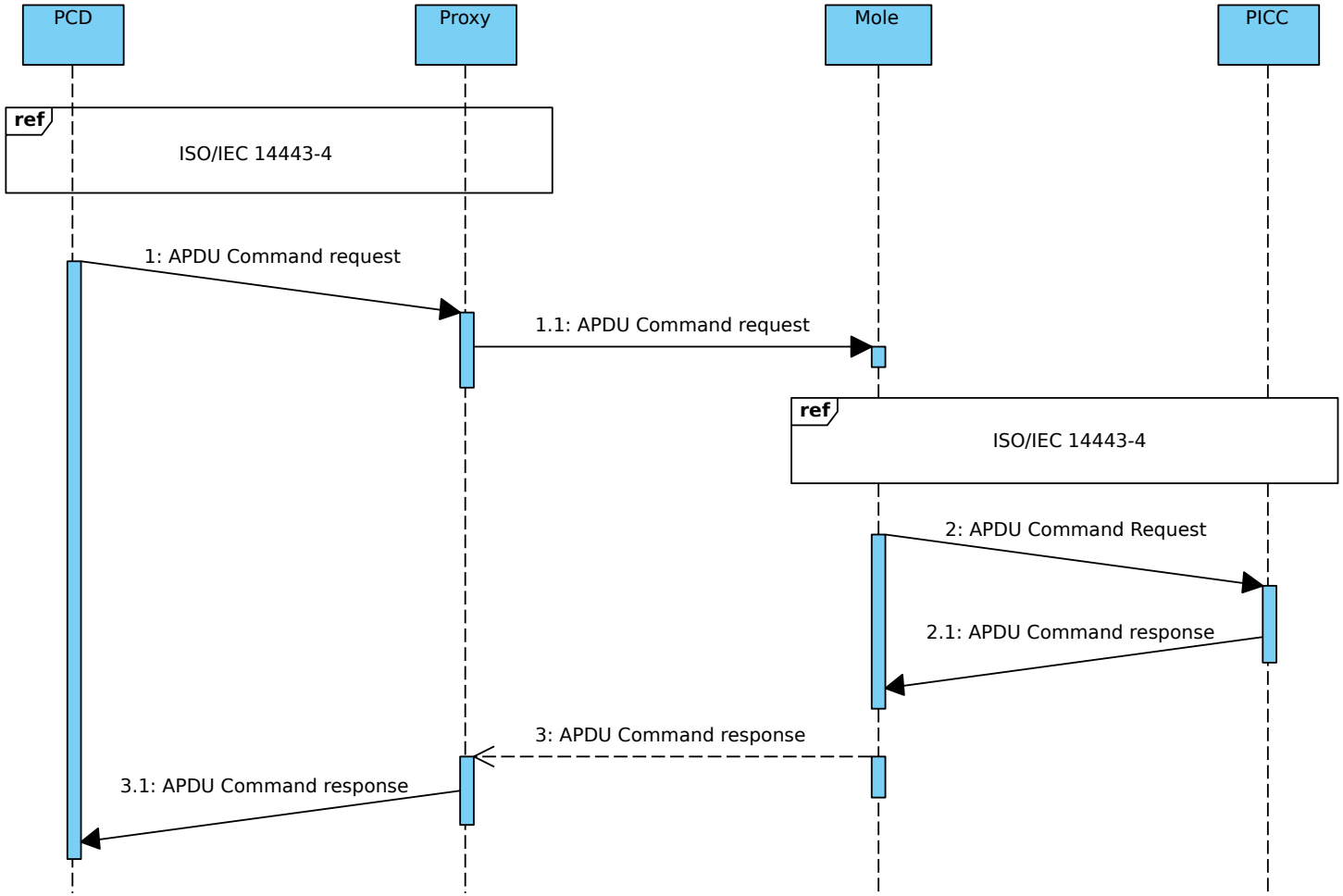
3. Relay Attacks

- Introduced by Conway in 1976
- Used in challenge-response protocol scenarios
- Can exploit security based in proximity concerns



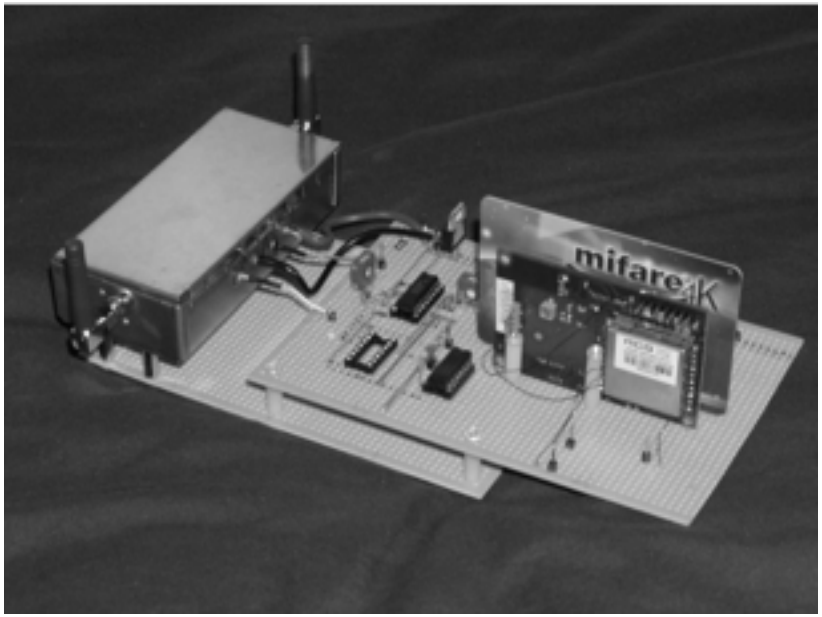
3. Relay Attacks

NFC relay over ISO/IEC 14443-4

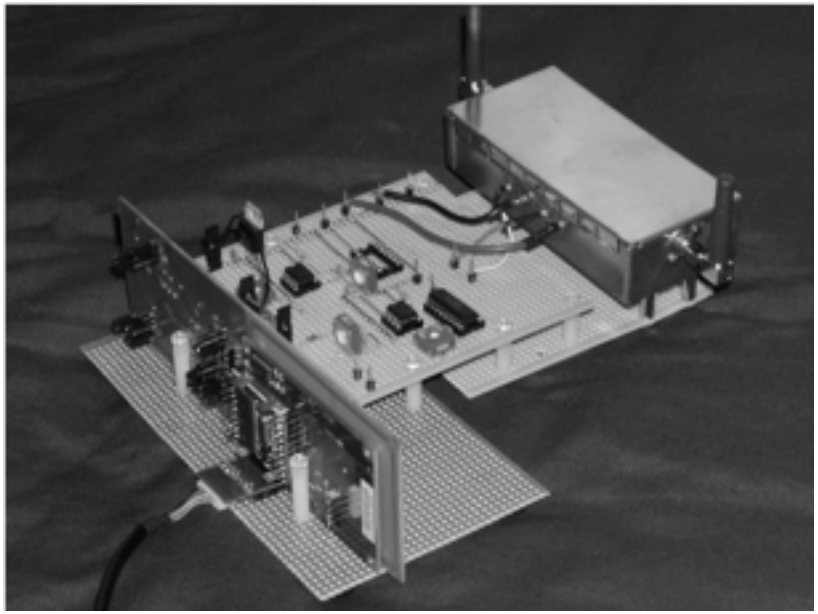


3. Relay Attacks

Some classical examples of NFC relay attacks



Mole



Proxy

Gerhard Hancke. A practical relay attack on ISO 14443 proximity cards. February, 2005

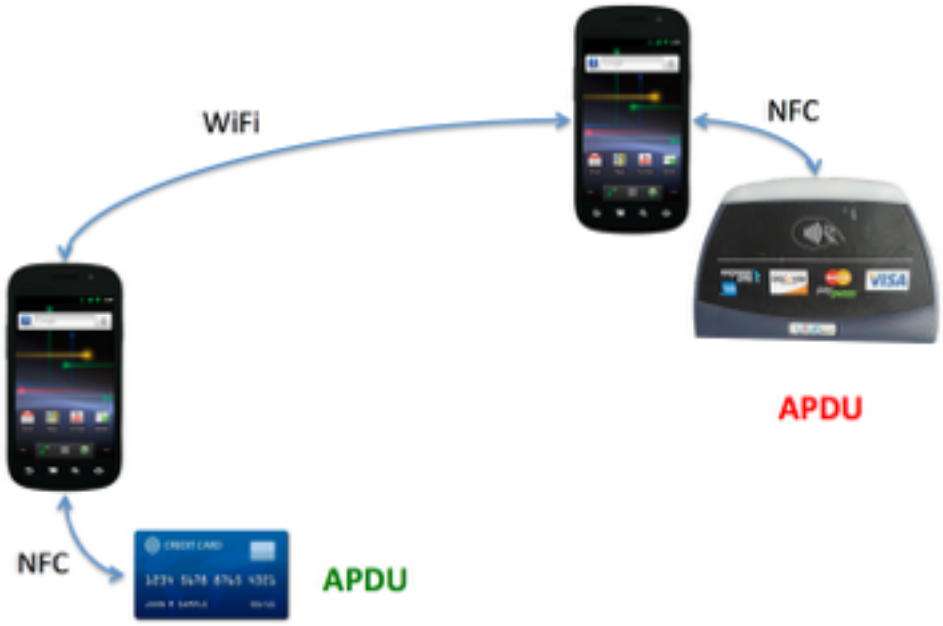
3. Relay Attacks

Some classical examples of NFC relay attacks

Practical Relay Attack on Contactless
Transactions by Using NFC Mobile Phones. February, 2012

3. Relay Attacks

Some classical examples of NFC relay attacks



NFC Proxy with CyanogenMod 9.1. Eddie Lee, DEFCON 20.

3. Relay Attacks

Our contribution?

- NFC relay attacks with *off-the-shelf* Android devices:
  - No root
  - No custom firmware
- Analysis of NFC capabilities on Android

agenda

1. NFC: what and how
2. EMV (a.k.a. credit card payments)
3. Relay attacks
4. **Android NFC history**
5. NFC implementation in Android
6. PoC + attack scenarios
7. Limitations and conclusions

A long time ago in a galaxy far, far away....

4. Android NFC history

NFC support began with Android 2.3
"*Gingerbread*" this is API level 9 and December 2010

- Only two operation modes: read/write and P2P (now Android Beam)
- Card emulation only via hardware SE (Secure Element):
  - Tamper-proof platform to securely store data and execute applications (Global Platform specifications)
  - Isolated from the untrusted host
  - Very restricted environment due to Trusted Service Managers

    Intermediary authority between network operators, manufacturers and service providers
  - OTA updates and installations
  - Virtually exclusive use for Google Wallet (and some banks entities)

4. Android NFC history

As a result, many developers asked for an easier access to this resource

- First solution was BlackBerry 7 OS; which included software card emulation or “soft-SE”
- soft-SE (also called Host Card Emulation) allows the OS to receive APDU commands and to response them by any application instead of by SE’s applets
- In 2011, Doug Year released a set of patches including HCE support for Android Cyanogen Mod (version 9.1)
  - Only for NXP PN544 chipset (Samsung Galaxy S, Nexus 7, etc.)
- **Finally Android officially supported HCE in October 2013 with the Android 4.4 “*KitKat*” release (API level 19)**

agenda

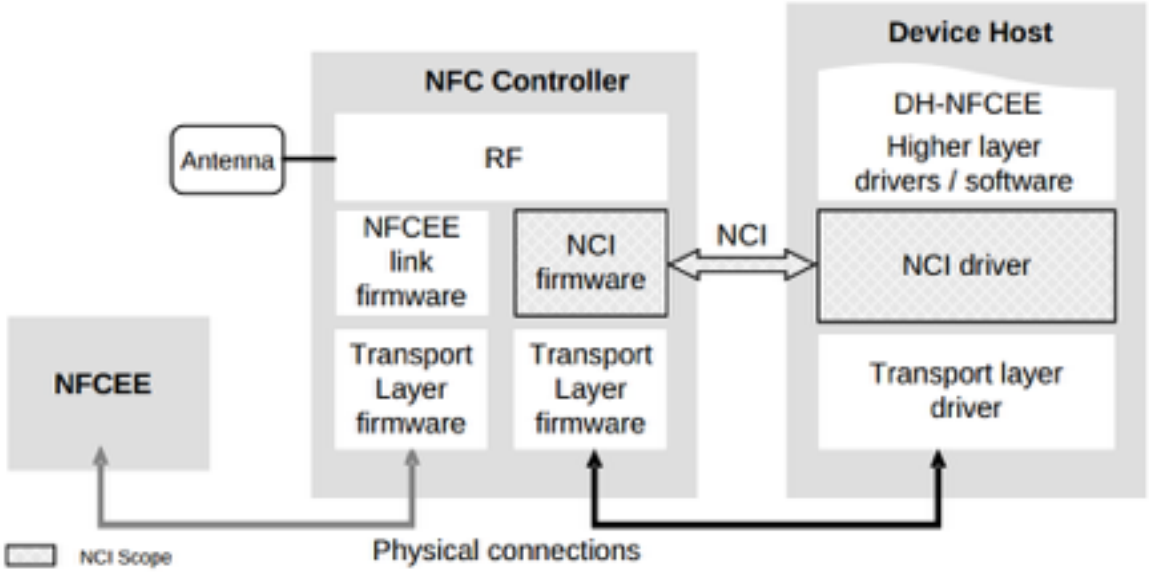
1. NFC: what and how
2. EMV (a.k.a. credit card payments)
3. Relay attacks
4. Android NFC history
5. **NFC implementation in Android**
6. PoC + attack scenarios
7. Limitations and conclusions

5. NFC implementation in Android

Event-driven API with two native implementations, depending on the NFC chip: `libnfc-nxp` and `libnfc-nci`.

NCI (or *NFC Controller Interface*) leads the NFC development:

- provides an open architecture not focused on a single chip
- offers an open interface between the NFC Controller and the Device Host
- has been standardised by the NFC Forum



5. NFC implementation in Android

The NCI defines two types of messages:

- Control messages; subdivided in commands (only from DH to NFCC), responses and notifications
- Data messages; carry the information addressed to (or originated from) the NFC endpoint

We have also the NCI modules, such as the RF Interface Modules:

- define how the DH can communicate through the NCI with a specific NFC endpoint
- each RF interface support a specific RF protocol
- determines how the payload in a *data message*, fits on a RF message

Other modules focused on RF discovery or AID routing (discussed later)

5. Android NFC: R/W mode

Applications are not allowed to directly set the device into **read/write** mode

1. Register NFC tags of interest (in the `AndroidManifest.xml`)
2. NFC service selects and starts the registered app whether a tag is discovered (apps can also ask preference when in foreground mode)

Tags are discovered by the NFCC, which polls the magnetic field

1. The tag protocol and technology is determined
2. An NCI message is sent from NFCC to DH with tag details
3. The DH (or `NfcService`) handles the message and fills a `Tag` object
4. The `NfcService` creates and emits an `Intent` with the `EXTRA_TAG` field with the `Tag` object
5. Android registered application receives the `Intent` and the `Tag` object

5. Android NFC: R/W mode

Android NFC read/write API offers specific classes per tag type:

- `NfcA` and `NfcB` for ISO/IEC 14443-3A and B compliant tags
- **`IsoDep`** for ISO/IEC 14443-4 tags using ISO/IEC 7816-4 APDUs
- `NfcF` for FeLiCa cards (standard JIS 6319-4)
- `NfcV` for ISO/IEC 15693 vecinity cards
- …

Extend `BasicTagTechnology`, which in turn implements the `TagTechnology` interface. All the classes use a high level I/O blocking method:

```
byte[] transceive(byte[] data, boolean raw) {
...
mTag.getTagService().transceive(mTag.getServiceHandle(), data, raw);
...
}
```

To communicate the DH with an NFC remote tag

5. Android NFC: HCE mode

HCE mode is supported by extending the `HostApduService` abstract class to process commands and generate responses, by implementing:

```
byte[] processCommandApdu(byte[] commandApdu, Bundle extras);
```

and

```
void onDeactivated(int reason);
```

methods.

- The application has to the register AIDs of interest in its manifest (since Lolipop also dynamic register support)
- IsoDep compliant readers initiates NFC communication with a `SELECT` command with an specific AID
- After a `SELECT`, the system will route all APDUs to the appropriate service according to its AID, until another application is selected or a `DESELECT` command is received

5. Android NFC arch.: Summary

Description	Language(s)	Dependency	OSS
NFC developer framework (`com.android.nfc` package)	Java, C++	API level	Yes
System NFC library (`libnfc-nxp` or `libnc-nci`)	C/C++	Manufacturer	Yes
NFC Android kernel driver	C	Hardware and manufacturer	Yes
NFC firmware (`/system/vendor/firmware` directory)	(unknown)	Hardware and manufacturer	No

Table 1. NFC architecture levels in Google's Android OS.

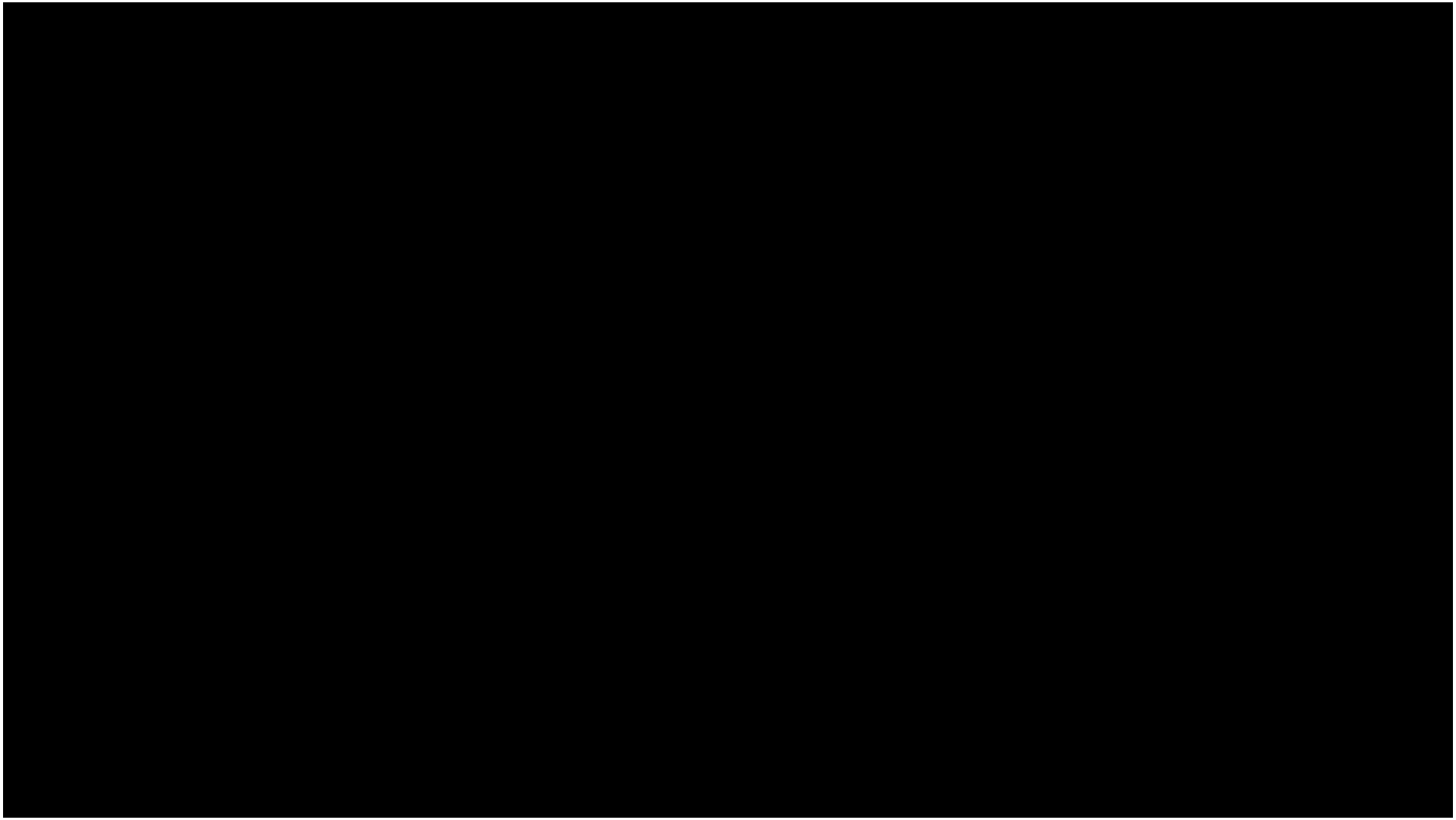
agenda

1. NFC: what and how
2. EMV (a.k.a. credit card payments)
3. Relay attacks
4. Android NFC history
5. NFC implementation in Android
6. **PoC + attack scenarios**
7. Limitations and conclusions

6. PoC + attack scenarios

Demo Time

6. PoC + attack scenarios



6. PoC + attack scenarios

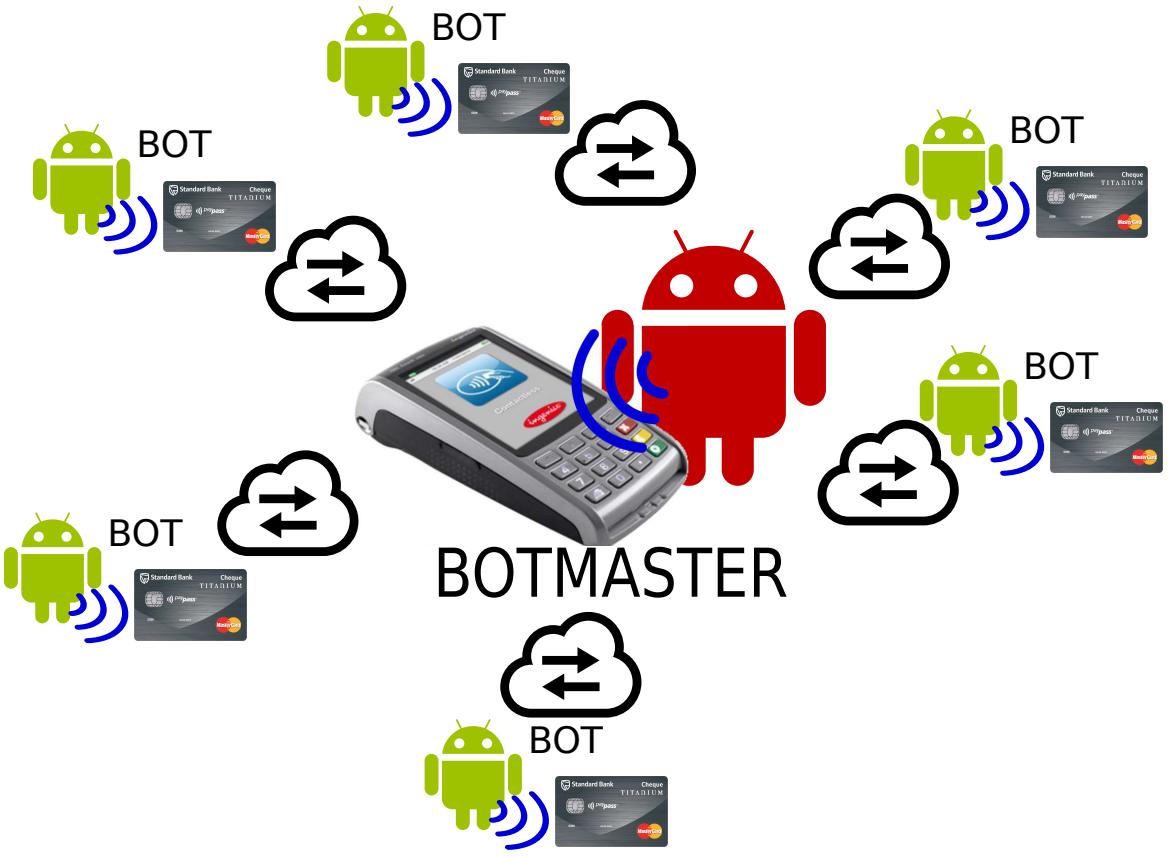


Fig: Distributed mafia fraud

6. PoC + attack scenarios

Fig: Astral payment

agenda

1. NFC: what and how
2. EMV (a.k.a. credit card payments)
3. Relay attacks
4. Android NFC history
5. NFC implementation in Android
6. PoC + attack scenarios
7. **Limitations and conclusions**

7. Limitations and conclusions

Currently, there are several limitations when performing NFC relay attacks with Android *off-the-shelf* devices

- Not support for raw ISO/IEC 14443-3 commands (no read/write mode for MIFARE Classic or other proprietary protocols)
- Emulation constrained to APDUs over ISO/IEC 14443-4
- Pre-registering of AIDs to emulate and explicit SELECTion
- Timing restrictions: maximum delay in the relay channel

$$FWT = 256 \cdot (16/f_C) \cdot 2^{FWI},\ 0 \leq FWI \leq 14,\ \text{where } f_C = 13.56\text{MHz}$$

Frame Waiting Time from 500µs to 5s

Note: about libnfc-nci implementation

7. Limitations and conclusions

Countermeasures against NFC relay attacks

- Distance-bounding protocols
- Hardware or RF fingerprinting identification
- Physical activation (button or switch)
- Secondary authentication methods within the cards (e.g., Zwipe cards)

7. Limitations and conclusions

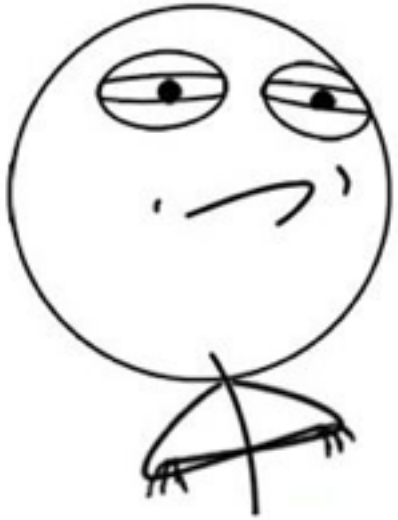
Android NFC off-the-shelf devices (no root nor custom firmware) are able to:

- Perform a relay attack over an ISO/IEC 14443-4 communication
- Contactless payment transactions affected (regardless the EMV security)

Thus, a simple Android app can be used to study NFC transactions without need of custom hardware

But also could be abused by malware…

questions or vinegar?



i also manufacture balsamic vinegar :)