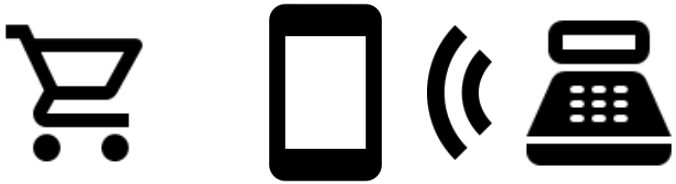
NFCGate

Steffen Klee, Alexandros Roussos, Max Maass, Matthias Hollick

Opening the Door for NFC Security Research with a Smartphone-Based Toolkit

Near-Field Communication (NFC)

While shopping:



At home:

On public transport:

In general:

tag (PICC) *reader (PCD)*

NFC Research Toolkits Requirements

SECURE MOBILE NETWORKING

Interaction with tags

- Read *static tag data* of a tag, e.g. NFC identifier (NFCID)
- Exchange *Application Protocol Data Units (APDUs)* with a tag

Interaction with readers

- Emulate *static tag data*
- Exchange *APDUs* with a reader

Research features

- Analyzing data
- Allow for different attack scenarios

Related Work

Attacks on NFC

- Electronic vehicle charging stations (Dalheimer, 2017): use tag identifiers as authentication
  → Hardware-based off-the-shelf NFC tools

- ReCoil (Sun et al., 2020): NFC range extension up to 49.6 cm
  → Custom hardware

Related Work

NFC toolkits

Tool	Protocols	Availability	Usability and Handling	Price
NFCProxy, other phone-based tools	Only ISO/IEC 7816 APDUs	Android	Inconspicuous, no additional hardware	$
Proxmark3	Any on ISO/IEC 14443	Dedicated Hardware	Suspicious, requires USB host	$$$
ChameleonMini	Any on ISO/IEC 14443	Dedicated Hardware	Suspicious, requires USB host	$$
NFCGate	**Any on ISO/IEC 14443**	**Android (rooted)**	**Inconspicuous, no additional hardware**	**$**

NFC on Android

Reader Mode

- Read *static tag data*
- Transmit and receive arbitrary APDUs to/from tags

NFC toolkit

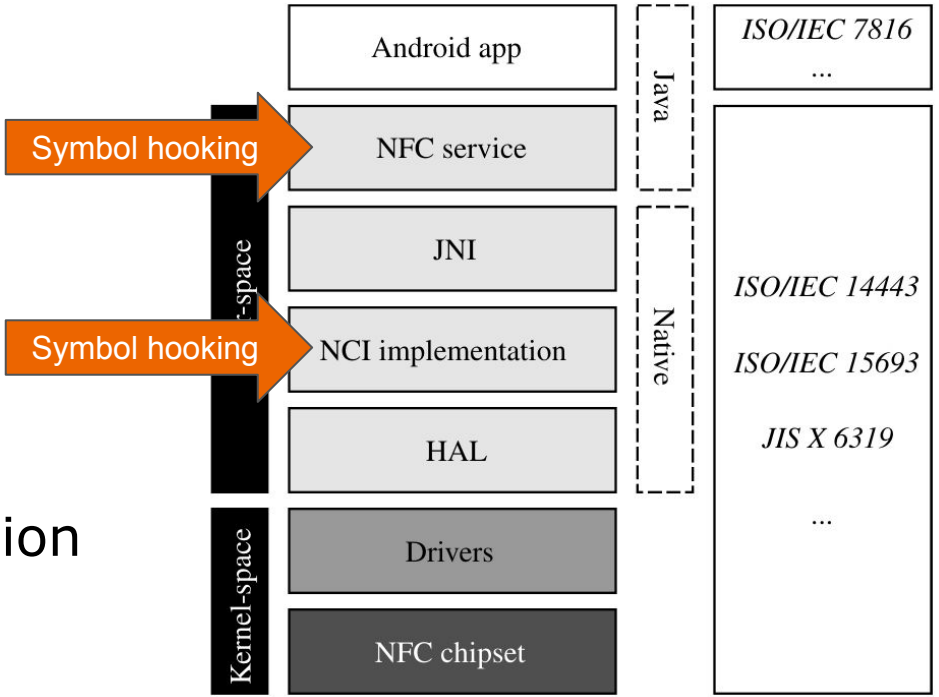
Host Card Emulation (HCE)

- Only access to NFC "application layer"
- No control over *static tag data*
- APDUs restricted to ISO 7816 *Application IDs (AIDs)*

Can we unleash the full power of HCE on Android?

Hardware Limitations? No.

- NFC chipset supports setting *static tag data*
  → Only software limitation
- NFC Controller Interface (NCI): standardized *configuration stream*



Solution: Set custom configuration stream, change software logic

The NFCGate Proof of Concept

Full tag emulation support

- Static tag data
- No APDU AID limitation

Clone mode

- Clones static tag data of a tag
- No APDUs

Relay mode

Logging

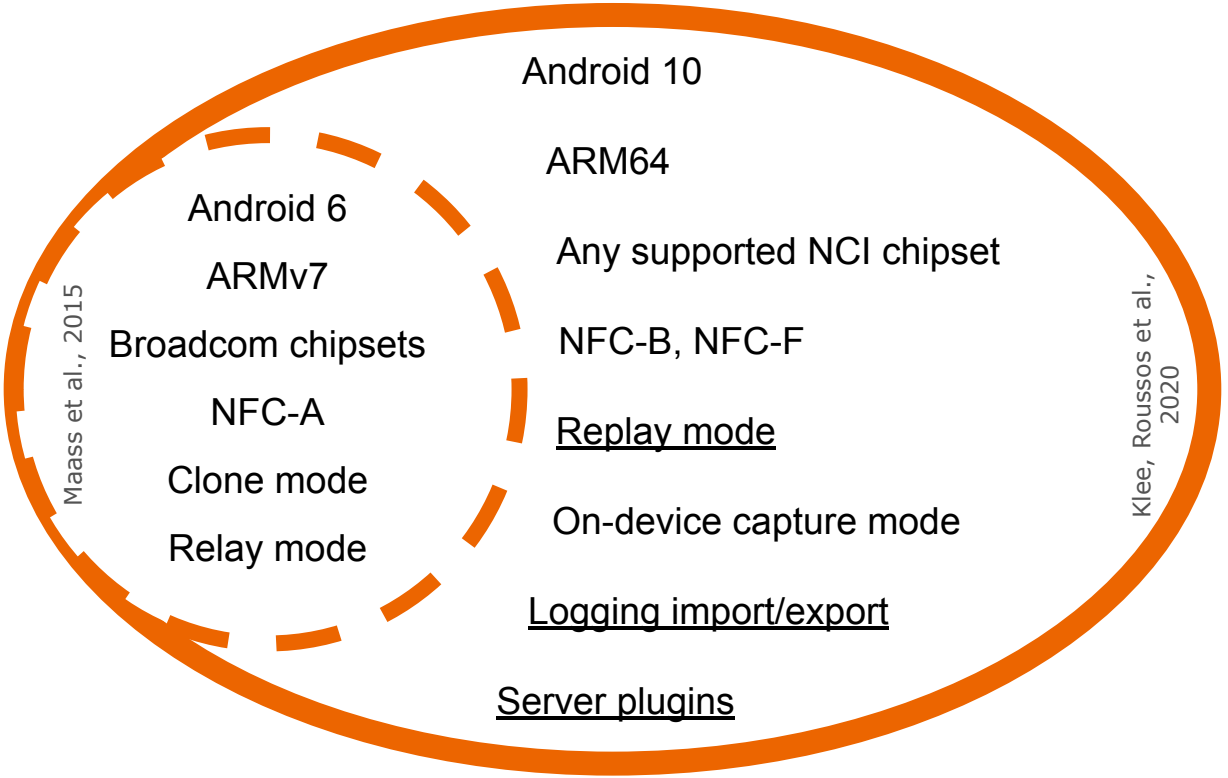
- Display APDUs in app

Just a smartphone

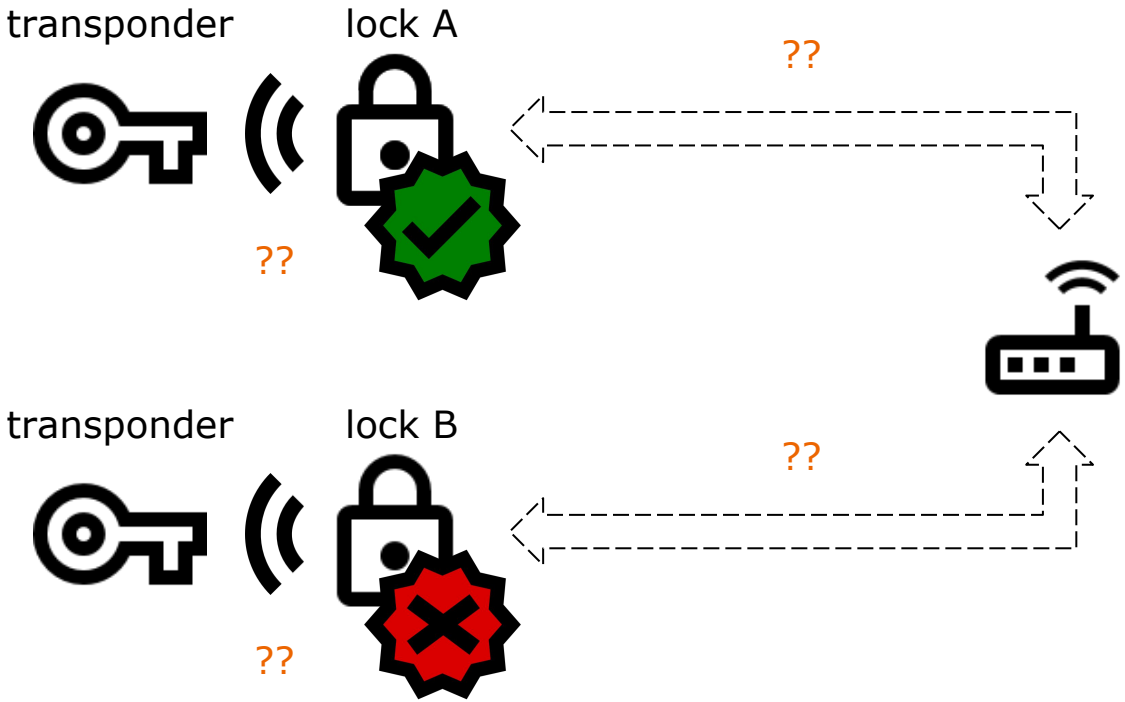
Inconspicuous and cheap

Maass et al., 2015

A New NFCGate



Case Study: Smart Door Lock



Can we break it?

First Look at the Lock

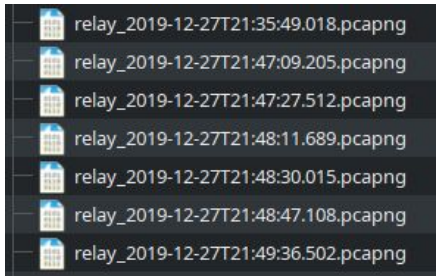
- Expensive, enterprise-level lock
- Made by well-known European vendor
- Mifare DESFire EV1 transponder
- NFCID1 (static tag data): randomized
- Lock requires “random” NFCID1
- Popular PN532 dev board has no support

We use NFCGate, which has no such limitations.

Capturing NFC Traffic

Issue #1:
No protection against relay attacks.

1. Connect smartphones to server
2. Start NFCGate's relay mode in *PICC* and *PCD* role
3. Hold devices in proximity to locking system
4. ...
5. Collect traffic as trace file



Open Sesame!

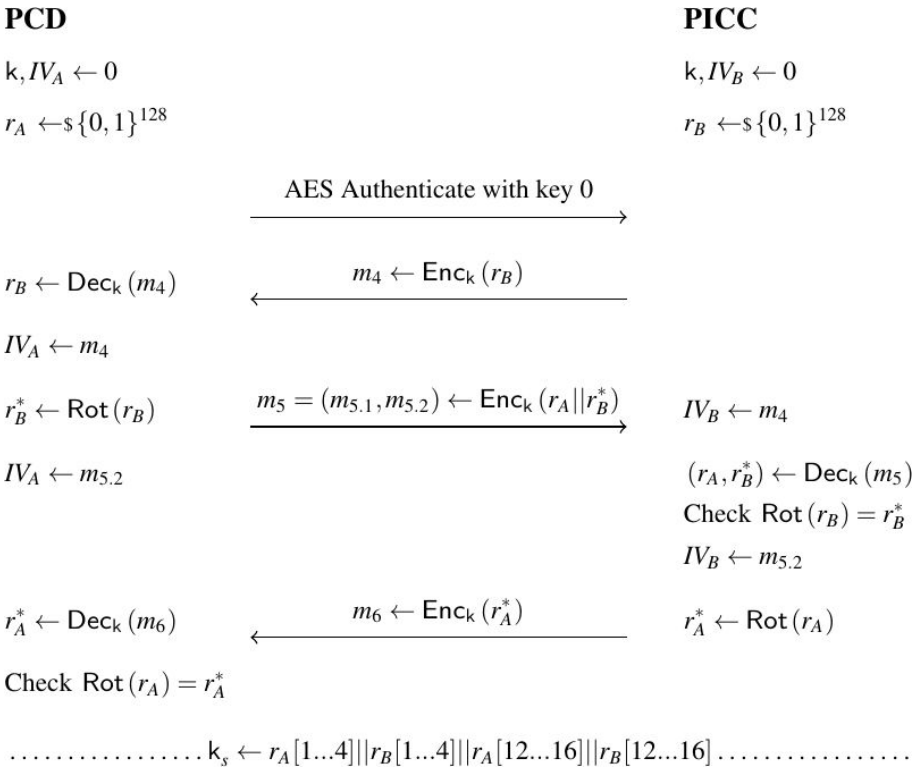
1. Start NFCGate in *replay mode* of PICC role
2. Load previously recorded relay trace
3. Hold smartphone in proximity to lock
4. …
5. Lock opens

Analyzing NFC Traffic

No.	Dir.	Data
1	<--	5a 01 00 00
2	-->	00
3	<--	aa 00
4	-->	af 2b 17 b5 5b 77 4d d2 2d 23 xx xx xx xx xx xx xx
5	<--	af 5b b4 1a 63 8b 30 86 ff 91 xx xx xx xx xx xx xx 29 76 a9 0c fa 44 d6 32 f1 xx xx xx xx xx xx xx
6	-->	00 99 9e 31 43 43 07 0a 18 56 xx xx xx xx xx xx xx
7	<--	51
8	-->	00 ba e9 7f 79 d3 66 de 1f 59 xx xx xx xx xx xx xx

- Not compliant with ISO/IEC 7816-4
  → NFCGate solves Android HCE limitation
- DESFire commands/results:
  - 5a: “Select Application”
  - aa: “AES Authenticate”
  - af: “Additional Frame”
  - 51: “Get tag UID”
  - 00: “Result: OK”

DESFire AES Authentication



- `Rot(x) = x << 8`
- AES-128-CBC
- Establishes encrypted channel
- Ensures both parties have knowledge of same key *k*
- Replay protection through nonces r_A and r_B

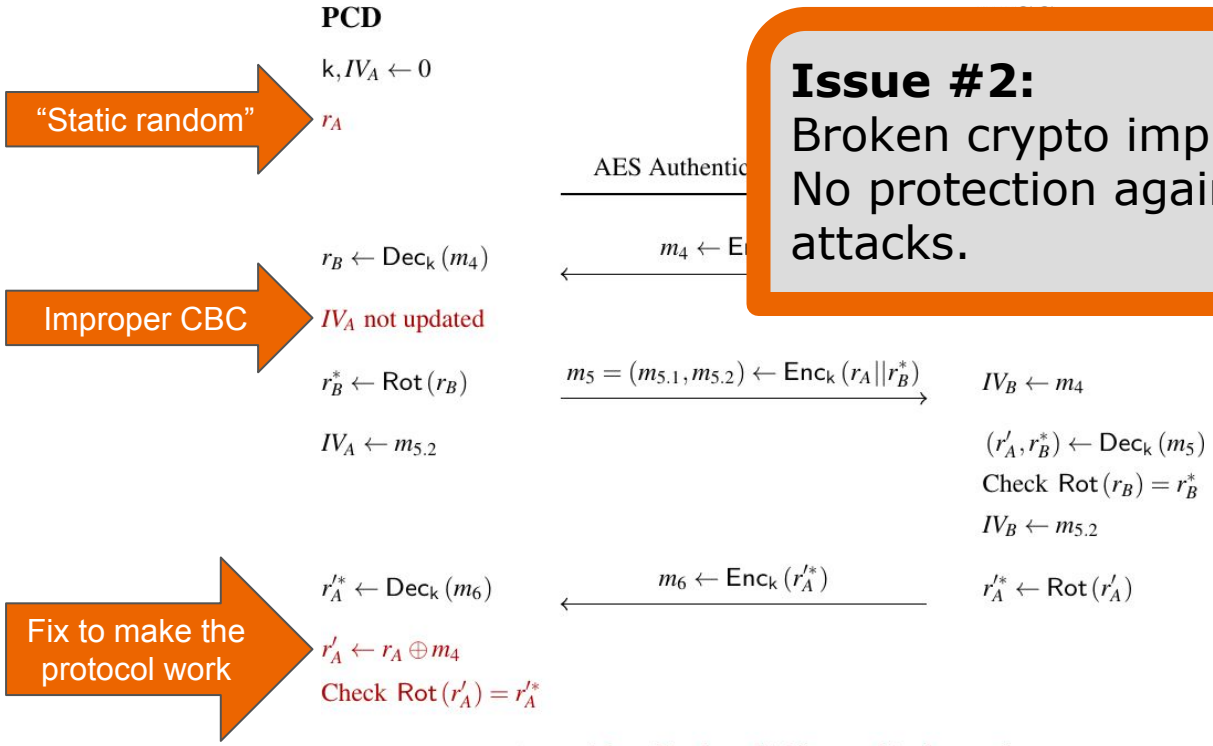
Why does our replay attack work?

Analyzing NFC Traffic

No.	Dir.	Data	
3	<--	aa 00	*AES Authenticate with key 0*
4	-->	af 2b 17 b5 5b 77 4d d2 2d 23 ..	*encrypted rB*
4	-->	af 17 fd f2 4e ef 96 44 39 4d ..	*encrypted rB*
5	<--	af 5b b4 1a 63 8b 30 86 ff 91 .. 29 7[illegible]a9 0c fa 44 ..	*encrypted rA || rB**
5	<--	af 5b b4 1a 63 8b 30 86 ff 91 .. [illegible]	*encrypted rA || rB**
6	-->	00 99 9e 31 43 43 07 0a 18 56 ..	*rA**
6	-->	00 00 b5 d0 af 88 92 ec 64 ab ..	*encrypted rA**
7	<--	51	Get tag *UID*
8	-->	00 ba e9 7f 79 d3 66 de 1f 59 ..	encrypted tag *UID*
8	-->	00 45 7a 66 41 33 b0 4f e0 ce ..	encrypted tag *UID*

Incorrect use of AES-CBC?
Message 4 not chained in message 5

The Protocol in Detail



Issue #2:
Broken crypto implementation: No protection against replay attacks.

More Attacks

- Desktop software: register transponders with the system
- Contains the authentication key *k* (static for entire product series)

Walk-by attack:

- Read real tag *UID* with known key
  → Store the *UID* for later use

Issue #3:
Use of a static key.

Privilege escalation/brute-force attack:

- *UID* (6 bytes) is not random
- Numerical difference of two tags: ≈3500
- Lock does not limit number of tries per time period

Case Study Conclusions

- **Issue #2:** Broken implementation → **Easy to solve**
  → Vendor solution: Update, properly implement protocol

- **Issue #3:** Use of a static key → **Hard to solve**
  → Vendor solution: Use different key, requires redeployment

- **Issue #1:** Vulnerable to relays → **Hard, research topic**
  → Vendor solution: Not possible due to limited hardware

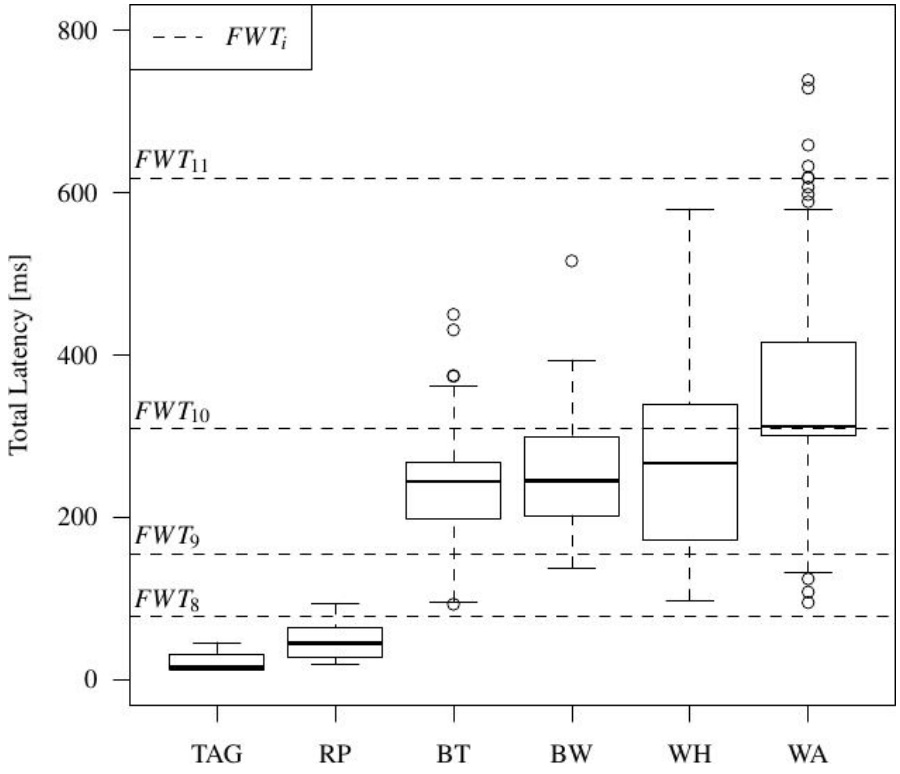
Preventing Relay Attacks

- Naïve idea: upper bound on communication latency
- ISO/IEC 14443 Frame Waiting Time (FWT):
  - Retransmission if no response received within some interval
  - Tag defines interval (max. ≈ 5s)
  - No enforcement in our experiments
  - Safety measure

NFCGate Latency Measurements

- Configurations:
  - TAG: baseline, direct communication with tag
  - RP: local replay using NFCGate (replay)
  - BT: Bluetooth PAN, server hosted on smartphone (relay)
  - BW: Bluetooth tethering to wireless network (relay)
  - WH: Wireless hotspot, server hosted on smartphone (relay)
  - WA: Wireless network, server hosted on computer (relay)

NFCGate Latency Measurements



- Replay almost indistinguishable from original tag
- No general upper bound
- Specific upper bound dependent on use-case
- Crypto operations might compensate network latency

Upper bound on communication latency no general solution

Relay Attack Countermeasures

- Do not use FWT as security feature
- Hard timings only in controlled deployments
- Distance bounding protocols as general solution

Distance bounding

1. Protocol layer: requires standard extension and hardware modifications
2. Application layer: can ensure authenticity, domain-specific

Conclusions

- **Any Android smartphone** with Xposed/EdXposed support
- No changes to system
- Interoperability: **pcapng support**
- Easy attack scenario development: **Python plugins**
- **Finds security issues** in deployed products

Get in Touch

NFCGate is open-source
https://github.com/nfcgate/nfcgate

Contact

sklee@seemoo.tu-darmstadt.de
aroussos@seemoo.tu-darmstadt.de
mmaass@seemoo.tu-darmstadt.de