NDSS 2019 @ San Diego, US

Nearby Threats: Reversing, Analyzing, and Attacking Google's 'Nearby Connections' on Android

Daniele Antonioli[1], Nils Ole Tippenhauer[2], Kasper Rasmussen[3]

[1]Singapore University of Technology and Design (SUTD)
[2]CISPA Helmholtz Center for Information Security
[3]University of Oxford

What are Google Nearby Connections?

- Public API for Android and Android Things
  - In-app proximity-based services
  - E.g. peer-to-peer file editing
- Implemented in the Google Play Services
  - Available across different Android versions
  - Applications use it as a shared library

Why Analyzing Nearby Connections?

- Wide attack surface
  - Android (version $\geq$ 4.0) and Android Things
  - Uses Bluetooth and Wi-Fi (at the same time)
- Proprietary technology
  - No public specifications
  - Implementation is closed-source and obfuscated

Our Core Contributions

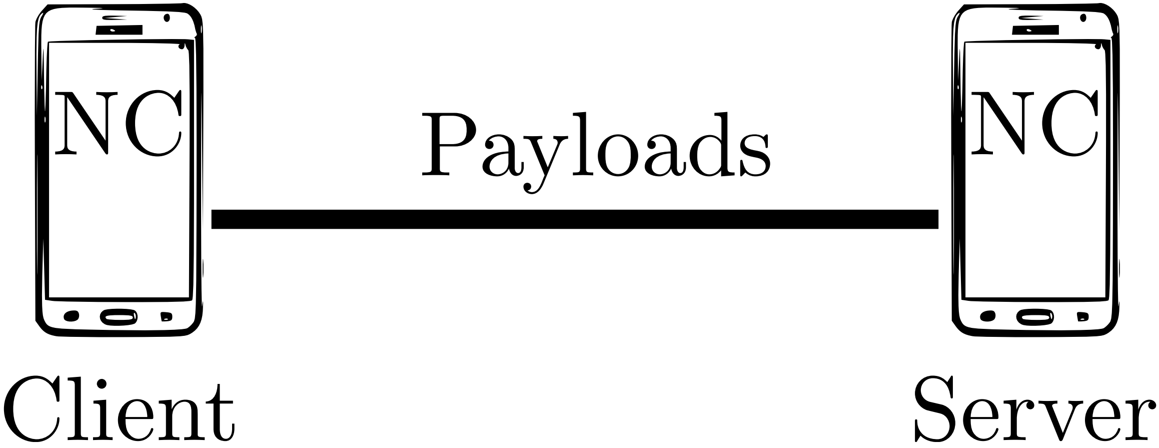
- **First (security) analysis of Nearby Connections**
  - Uncovers its proprietary mechanisms and protocols
  - Based on reversing its Android implementation
- **Re-implementation of Nearby Connections (REarby)**
  - Exposes parameters not accessible with the official API
  - Impersonates nearby devices from any application
- **Attacking Nearby Connections on Android**
  - Connection manipulation and range extension attacks
  - Responsible disclosure with Google

Nearby Connections Public Information

NC
NC
Client
Server

- Server advertises a service, client discovers it (`sid`)
- Connection strategies: `P2P_STAR` and `P2P_CLUSTER`

Nearby Connections Public Information 2



NC

Payloads

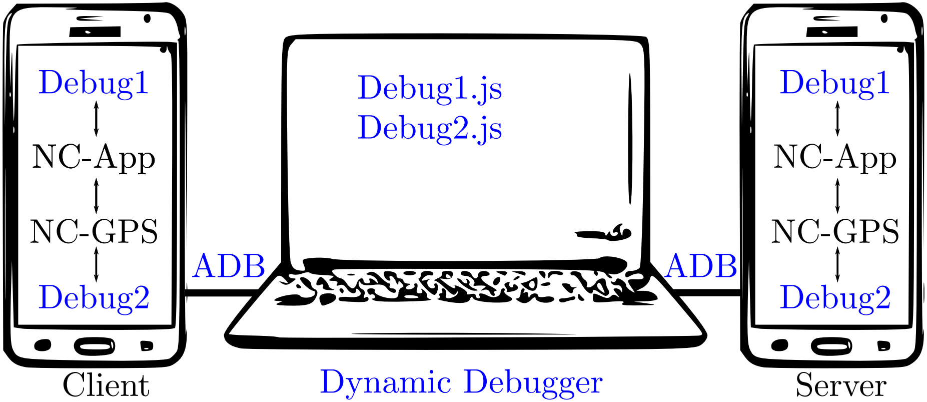
NC

Client

Server

- Client and server connect using Bluetooth and/or Wi-Fi
- Nodes exchange encrypted payloads (peer-to-peer)

Our Dynamic Binary Instrumentation



- Workhorse: Frida, `https://www.frida.re`
  - Profiling of processes, e.g. NC-App, NC-GPS
  - Hook function and methods calls
  - Override parameters and return values
  - Read and write processes' memory

Reversed Phases of a Nearby Connection

Reversed Phases of a Nearby Connection

1. **Discovery**: Bluetooth BR/EDR name and BLE reports

Reversed Phases of a Nearby Connection

1. **Discovery**: Bluetooth BR/EDR name and BLE reports
2. **Connection Request**: Bluetooth BR/EDR, not authenticated

Reversed Phases of a Nearby Connection

1. **Discovery**: Bluetooth BR/EDR name and BLE reports
2. **Connection Request**: Bluetooth BR/EDR, not authenticated
3. **Key Exchange Protocol**: establishment of a shared secret

Reversed Phases of a Nearby Connection

1. **Discovery**: Bluetooth BR/EDR name and BLE reports
2. **Connection Request**: Bluetooth BR/EDR, not authenticated
3. **Key Exchange Protocol**: establishment of a shared secret
4. **Optional Authentication**: based on the shared secret

Reversed Phases of a Nearby Connection

1. **Discovery**: Bluetooth BR/EDR name and BLE reports
2. **Connection Request**: Bluetooth BR/EDR, not authenticated
3. **Key Exchange Protocol**: establishment of a shared secret
4. **Optional Authentication**: based on the shared secret
5. **Application Layer Connection Establishment**: interactive

Reversed Phases of a Nearby Connection

1. **Discovery**: Bluetooth BR/EDR name and BLE reports
2. **Connection Request**: Bluetooth BR/EDR, not authenticated
3. **Key Exchange Protocol**: establishment of a shared secret
4. **Optional Authentication**: based on the shared secret
5. **Application Layer Connection Establishment**: interactive
6. **Key Derivation Functions**: session, AES and HMAC keys

Reversed Phases of a Nearby Connection

1. **Discovery**: Bluetooth BR/EDR name and BLE reports
2. **Connection Request**: Bluetooth BR/EDR, not authenticated
3. **Key Exchange Protocol**: establishment of a shared secret
4. **Optional Authentication**: based on the shared secret
5. **Application Layer Connection Establishment**: interactive
6. **Key Derivation Functions**: session, AES and HMAC keys
7. **Optional Physical Layer Switch**: Bluetooth BR/EDR to Wi-Fi

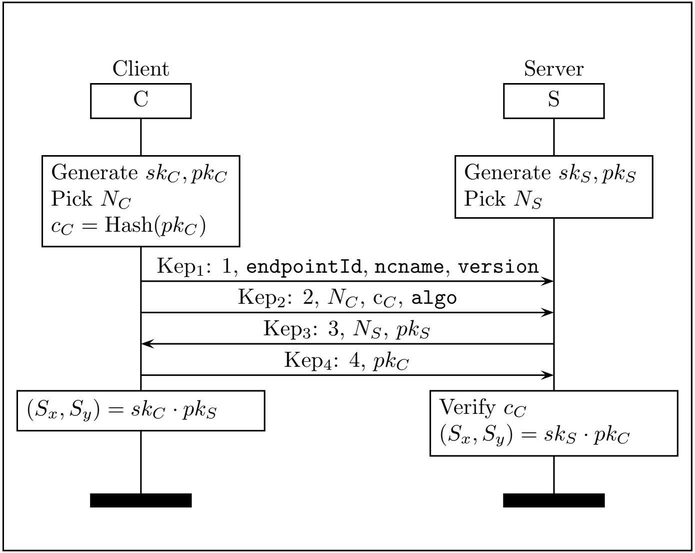
Reversed Phases of a Nearby Connection

1. **Discovery**: Bluetooth BR/EDR name and BLE reports
2. **Connection Request**: Bluetooth BR/EDR, not authenticated
3. **Key Exchange Protocol**: establishment of a shared secret
4. **Optional Authentication**: based on the shared secret
5. **Application Layer Connection Establishment**: interactive
6. **Key Derivation Functions**: session, AES and HMAC keys
7. **Optional Physical Layer Switch**: Bluetooth BR/EDR to Wi-Fi
8. **Exchange Encrypted Payloads**: 30 seconds timeout

Reversed Phases of a Nearby Connection

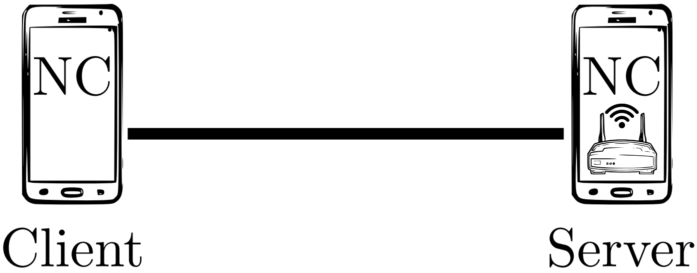
1. **Discovery**: Bluetooth BR/EDR name and BLE reports
2. **Connection Request**: Bluetooth BR/EDR, not authenticated
3. **Key Exchange Protocol**: establishment of a shared secret
4. **Optional Authentication**: based on the shared secret
5. **Application Layer Connection Establishment**: interactive
6. **Key Derivation Functions**: session, AES and HMAC keys
7. **Optional Physical Layer Switch**: Bluetooth BR/EDR to Wi-Fi
8. **Exchange Encrypted Payloads**: 30 seconds timeout
9. **Disconnection**

Key Exchange Protocol (KEP)


Client
C
Server
S
Generate sk_C, pk_C
Pick N_C
$c_C = \text{Hash}(pk_C)$
Generate sk_S, pk_S
Pick N_S
Kep_1: 1, `endpointId`, `ncname`, `version`
Kep_2: 2, N_C, c_C, `algo`
Kep_3: 3, N_S, pk_S
Kep_4: 4, pk_C
$(S_x, S_y) = sk_C \cdot pk_S$
Verify c_C
$(S_x, S_y) = sk_S \cdot pk_C$

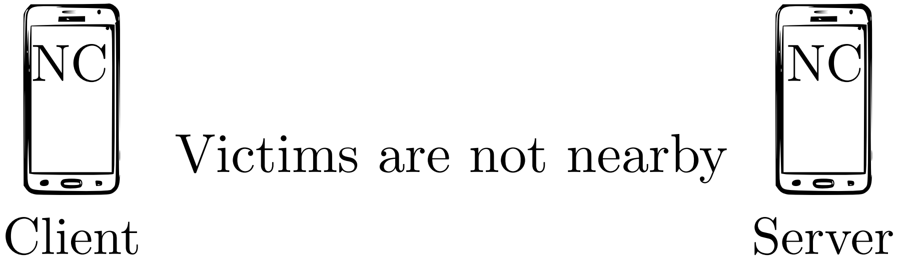
- Based on ECDH, NIST P256 curve, shared secret is S_x

Optional Physical Layer Switch



- **Bluetooth to soft access point (Wi-Fi Direct, hostapd)**
  - Server instructs the client over Bluetooth
  - Client contacts the server over Wi-Fi

Range Extension MitM Attack



NC

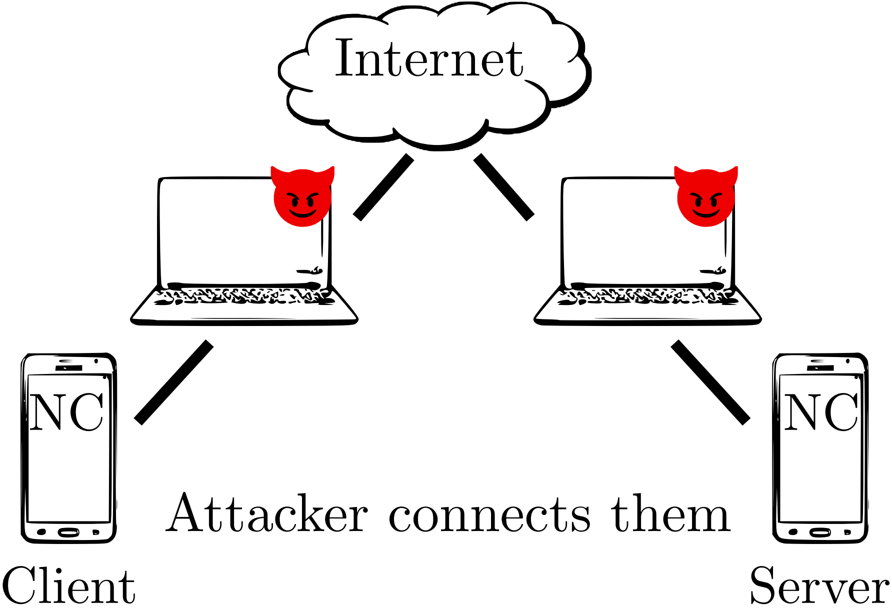
NC

Victims are not nearby

Client

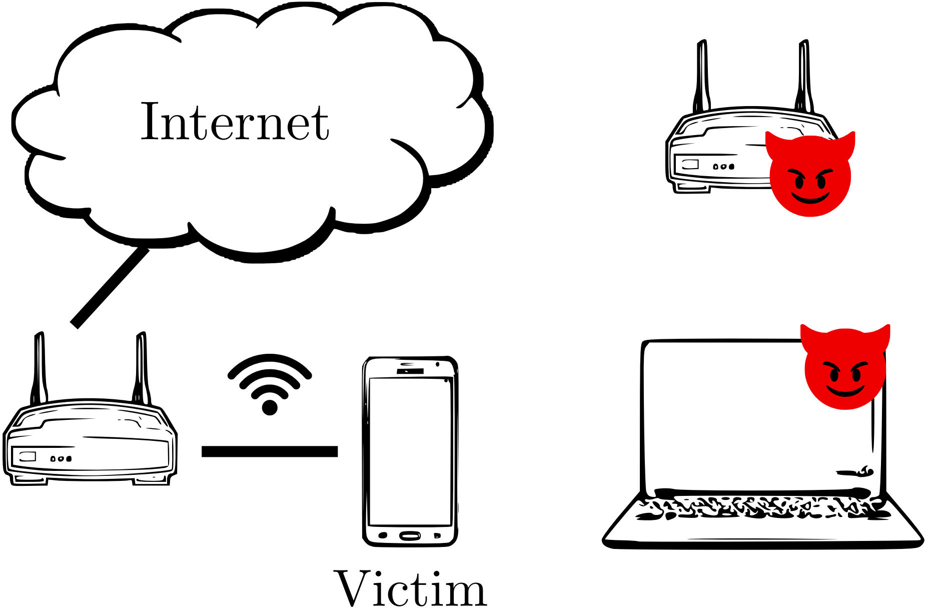
Server

Range Extension MitM Attack

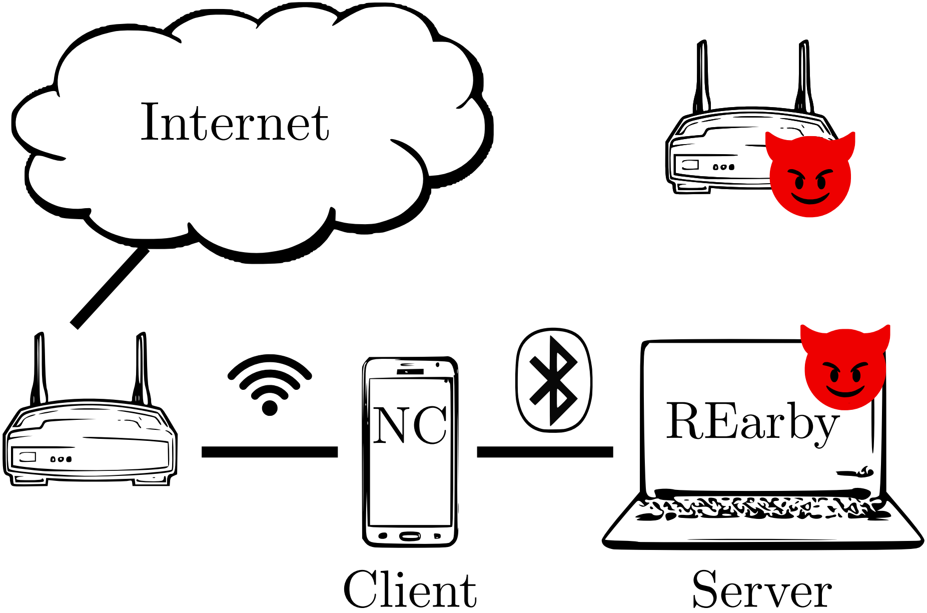


Attacker connects them

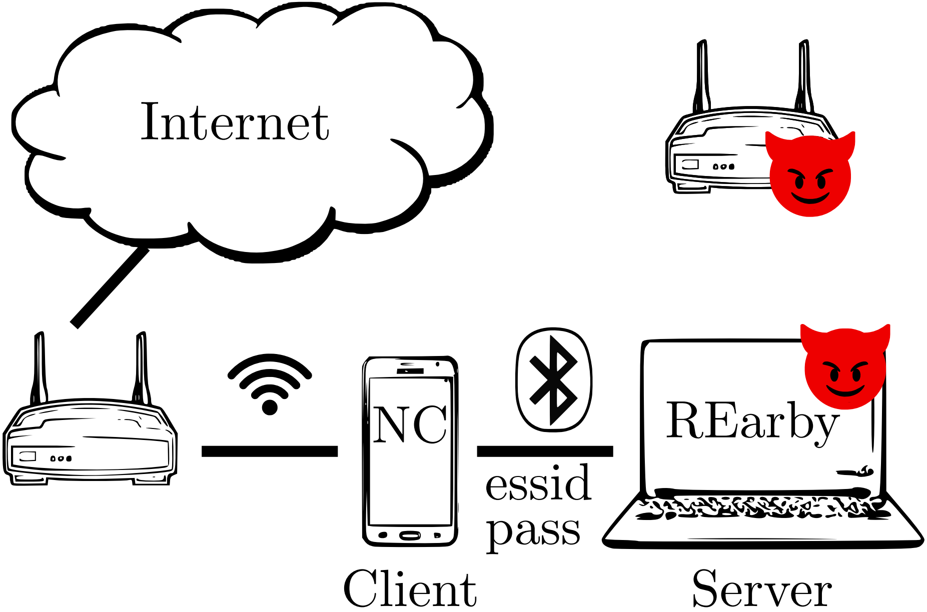
Soft Access Point Manipulation Attack



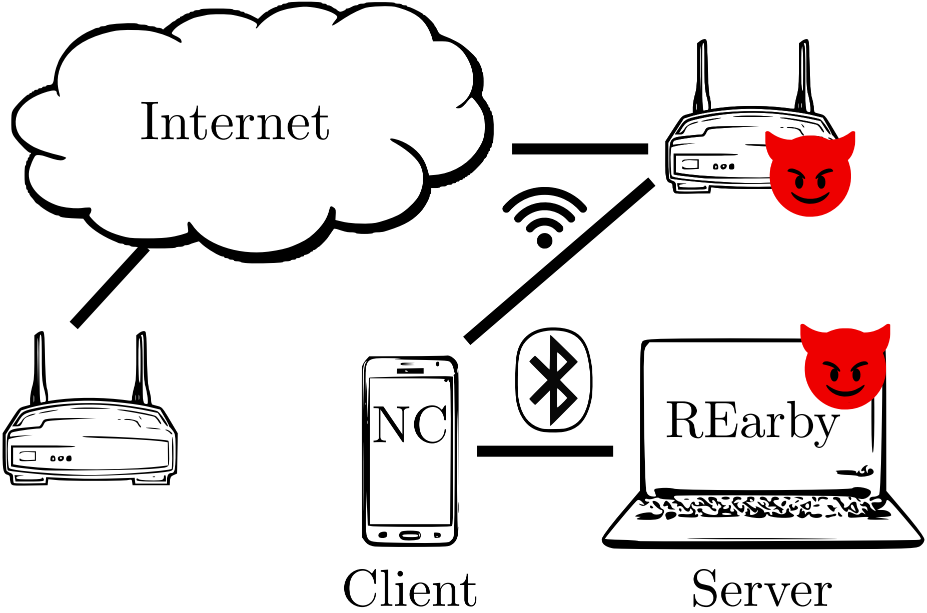
Victim Connects to Attacker’s REarby Server



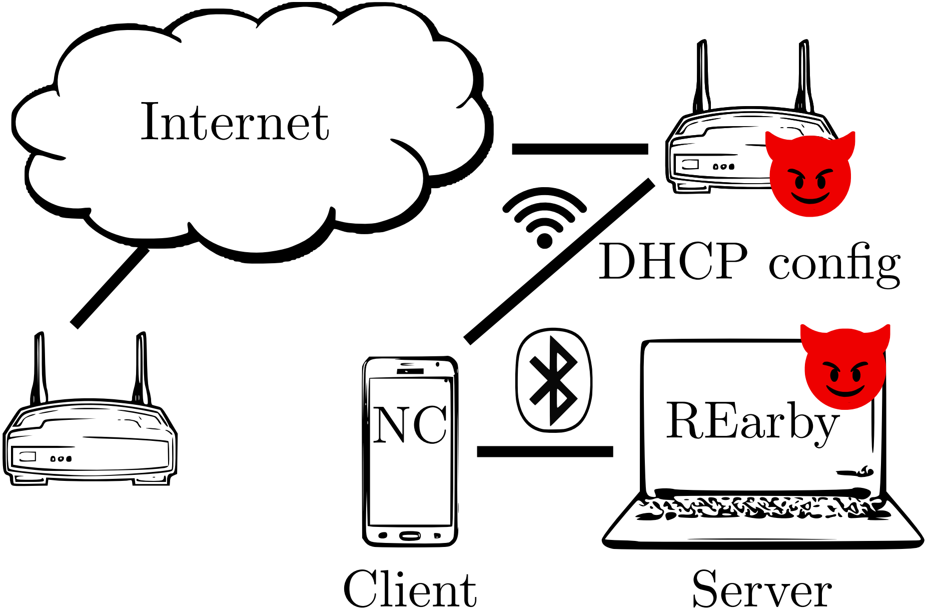
Attacker Manipulates Bluetooth to Wi-Fi Switch



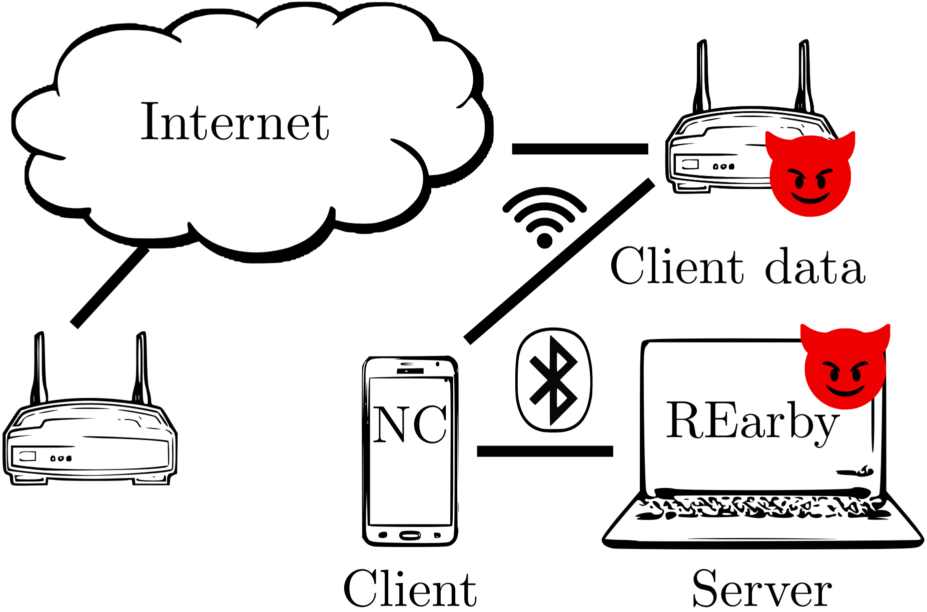
Victim Connects to Attacker's Wi-Fi AP



Attacker Configures Victim’s Network Interface



Attacker Eavesdrops All Wi-Fi Traffic



Conclusions

- First security analysis of Nearby Connections
- Reversed its Android implementation and re-implemented it (REarby)
- Range extension and soft access point manipulation attacks
- Try the Soft Access Point Manipulation attack: `https://github.com/francozappa/REarby/tree/master/poc-hostapd`

Conclusions

- First security analysis of Nearby Connections
- Reversed its Android implementation and re-implemented it (REarby)
- Range extension and soft access point manipulation attacks
- Try the Soft Access Point Manipulation attack:
  https://github.com/francozappa/REarby/tree/master/poc-hostapd

- Thanks for your time! Questions?