Why is Hard to Secure Mobile Proximity Services

Daniele Antonioli, EURECOM (France)

Greetings :)

- Daniele Antonioli
  - Asst. Prof. at EURECOM (France)
- Current research interests
  - Security of wireless, embedded, industrial, and cyber-physical systems
- More info
  - https://francozappa.github.io/

EURECOM
Sophia Antipolis

Talk Outline

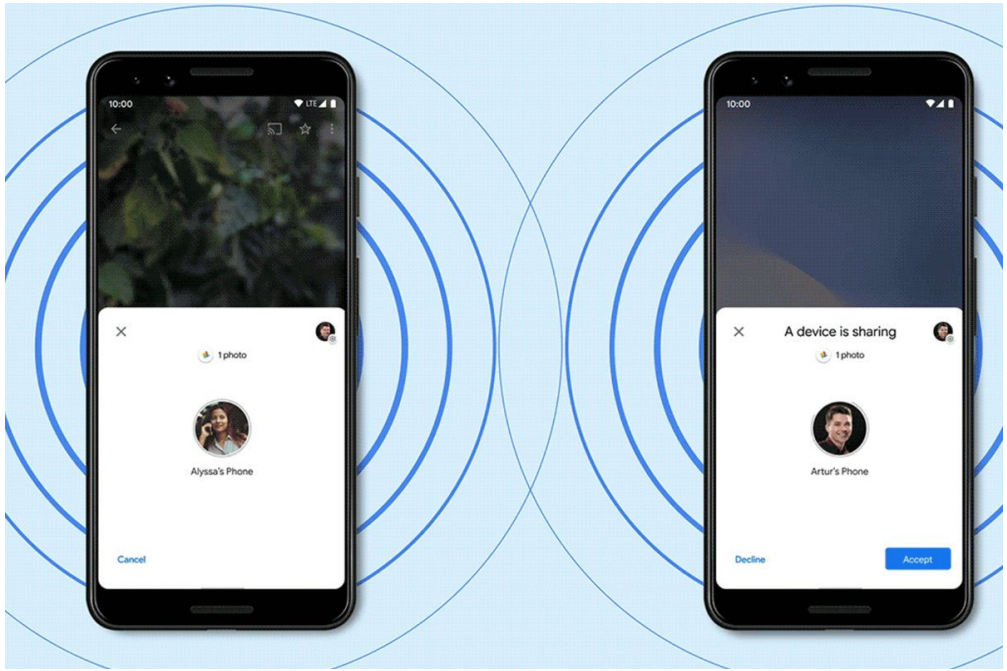
1. Proximity Services
2. Nearby Connections
3. Exposure Notifications
4. Research Trends

Acknowledgments!

- SecMT’21 organizers
  - E. Losiouk (Padua Uni.)
  - O. Gadyatskaya (Leiden Uni.)
- Nearby Connections
  - K. Rasmussen (Oxford Uni.)
  - N. Tippenhauer (CISPA)
- Exposure Notification
  - M. Payer (EPFL)
  - DP3T team (full list)

1 - Proximity Services

Mobile Proximity Service (1)



Mobile Proximity Service (2)

- Components
  - Users carrying mobile devices
  - Devices with radios (Wi-Fi, Bluetooth)
  - Internet-connected backend (optional)

Let’s see some **examples**

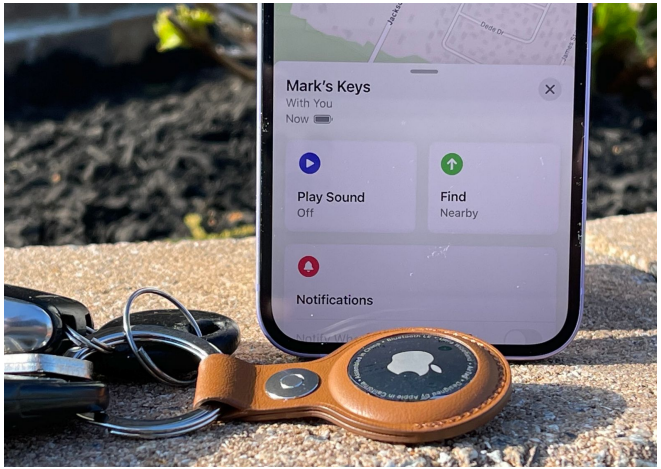
E.g., Sharing (files, contacts, …)

android

Nearby › Connections API

AirDrop

E.g., Tracking (fitness, goods, COVID-19, …)



E.g., Interacting (chat, alerts, …)

berty

Proximity Services' **entangled requirements**?

Requirements: 1. Pervasive

- Software
- Hardware
- Transports

Bluetooth®

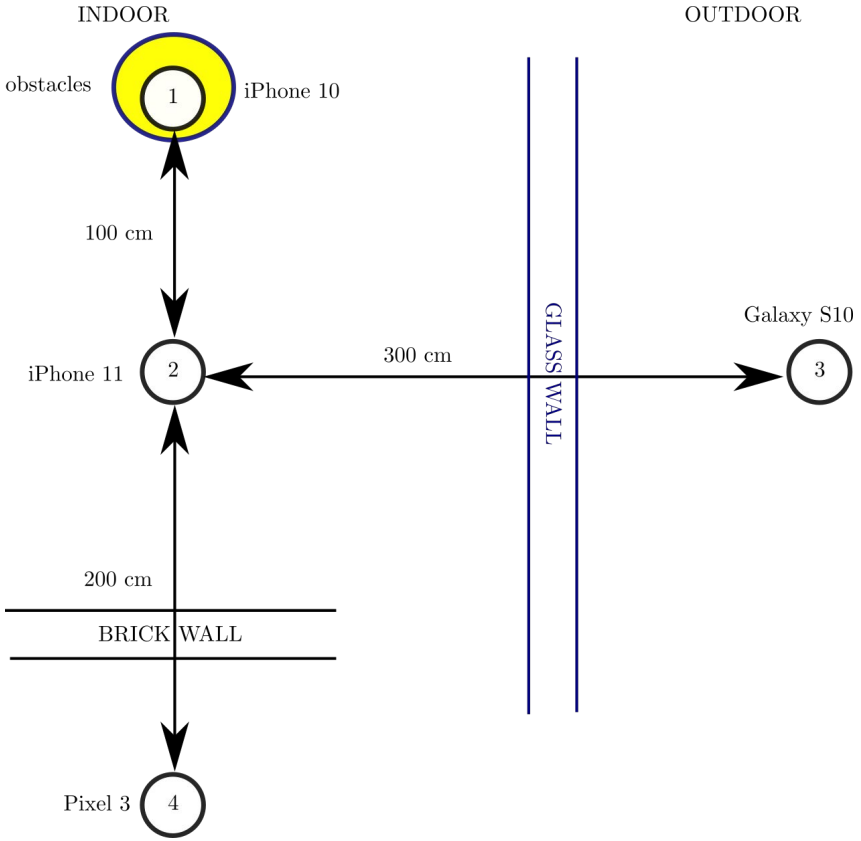
Wi Fi™

Requirements: 2. Performant

- Usable
- Low Energy
- Low Latency
- Scale with users

Requirements: 3. Adaptive

- Indoor
- Outdoor
- Line-of-sight
- Non line-of-sight
- Multipath, fading, ...



Requirements: 4. Secure

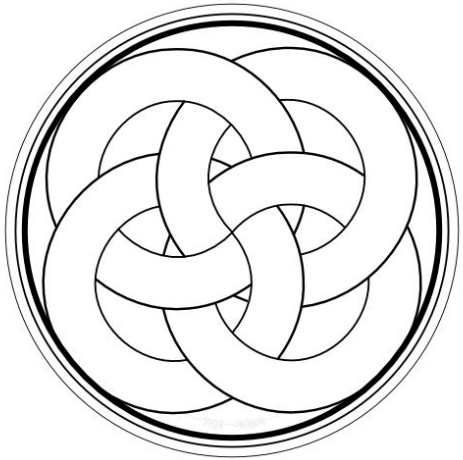
- Confidentiality
- Integrity
- Availability
- Authorization
- ...

Requirements: 5. Privacy-preserving

- Anonymity
- Non-linkability
- Decentralization
- ...

Proximity Services: Entangled Requirements

1. **Pervasive** (software, hardware, transports)
2. **Performant** (usability, energy, scale, latency)
3. **Adaptive** (indoor, outdoor, LoS, NLoS)
4. **Secure** (confidentiality, integrity, availability)
5. **Privacy-preserving** (anonymity, linkability, decentralization)



2 - Nearby Connections (NC)

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

Daniele Antonioli
Singapore University of Technology and Design
daniele_antonioli@mymail.sutd.edu.sg

Nils Ole Tippenhauer
CISPA Helmholtz Center for Information Security
tippenhauer@cispa.saarland

Kasper Rasmussen
Department of Computer Science
University of Oxford
kasper.rasmussen@cs.ox.ac.uk

***Abstract*—Google's Nearby Connections API enables any Android (and Android Things) application to provide proximity-based services to its users, regardless of their network connectivity. The API uses Bluetooth BR/EDR, Bluetooth LE and Wi-Fi to let "nearby" clients (discoverers) and servers (advertisers) connect and exchange different types of payloads. The implementation of the API is proprietary, closed-source and obfuscated. The updates of the API are automatically installed by Google across different versions of Android, without user interaction. Little is known publicly about the security guarantees offered by the API, even though it presents a significant attack surface.**

closed-source and obfuscated library that allows Google to provide the same services to any Android and Android Things application, regardless of the version of the operating systems. The API is compatible with any Android device, version 4.0 or greater, and it is updated by Google without user interaction [1]. An attacker who can exploit this API can target (at least) any application using Nearby Connections in any Android mobile and IoT device. This implies a large attacker surface and represents a critical threat with severe consequences such as data loss, automatic spread of malware, and distributed denial

https://francozappa.github.io/publication/rearby/paper.pdf

Nearby Connections (1)

``Nearby Connections is a peer-to-peer networking API that allows apps to *easily* discover, connect to, and exchange data with *nearby* devices in real-time, *regardless of network connectivity*''

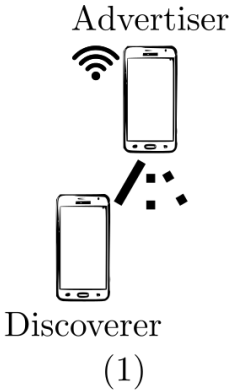
https://developers.google.com/nearby/connections/overview

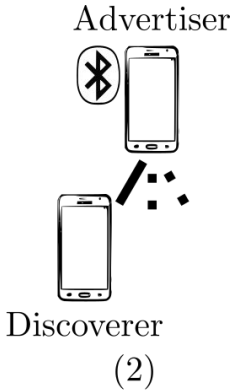
Nearby Connections (2)

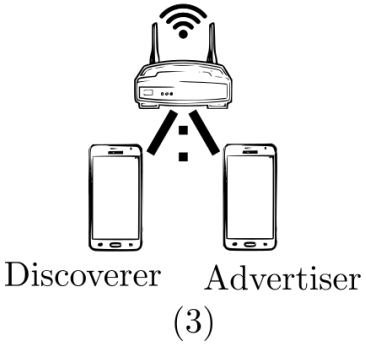
- Android and Android Things
- Wi-Fi, and Bluetooth
- No backend
- Proprietary design and impl
- High-level Android API
- Transparent to users

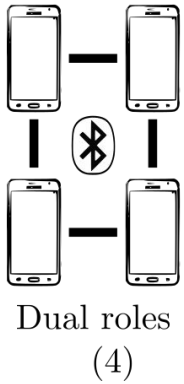
Nearby Connections: Topologies

1. Ad-hoc Wi-Fi
2. Ad-hoc Bluetooth Classic
3. Infrastructure Wi-Fi
4. Mesh Bluetooth







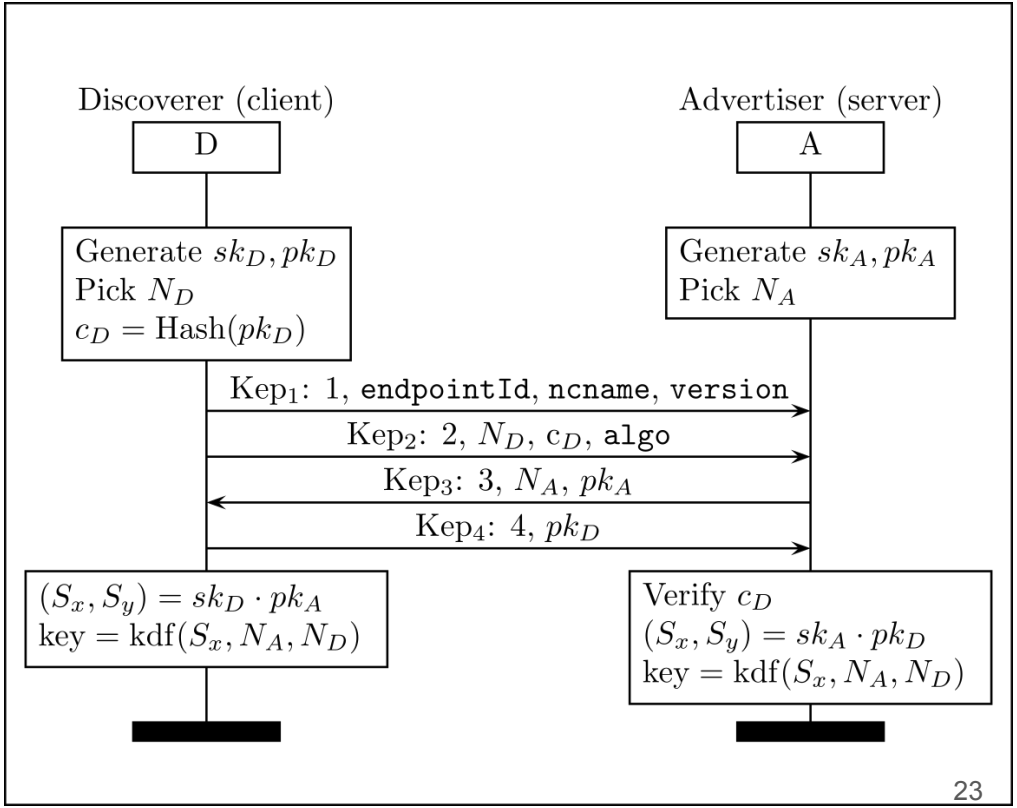


Nearby Connections: Lifecycle (simplified)

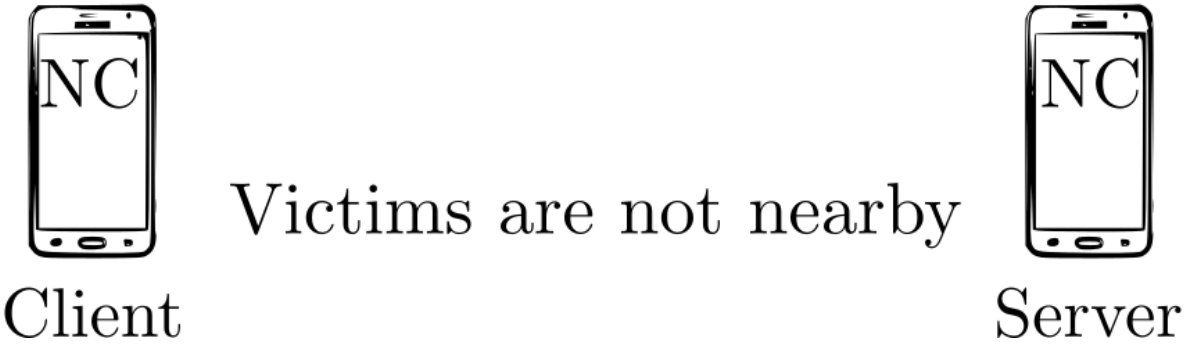
1. **Discovery** over Bluetooth Classic and BLE
2. Link layer **Bluetooth connection** (**insecure**)
3. Application layer **key exchange** (**custom**)
4. Application layer **authentication** (**optional**)
5. Application layer **key derivation** (**custom**)
6. Application layer **encrypted session**
7. Bluetooth to Wi-Fi **automatic switch** (**custom**)

Nearby Connections: appl. layer key exchange

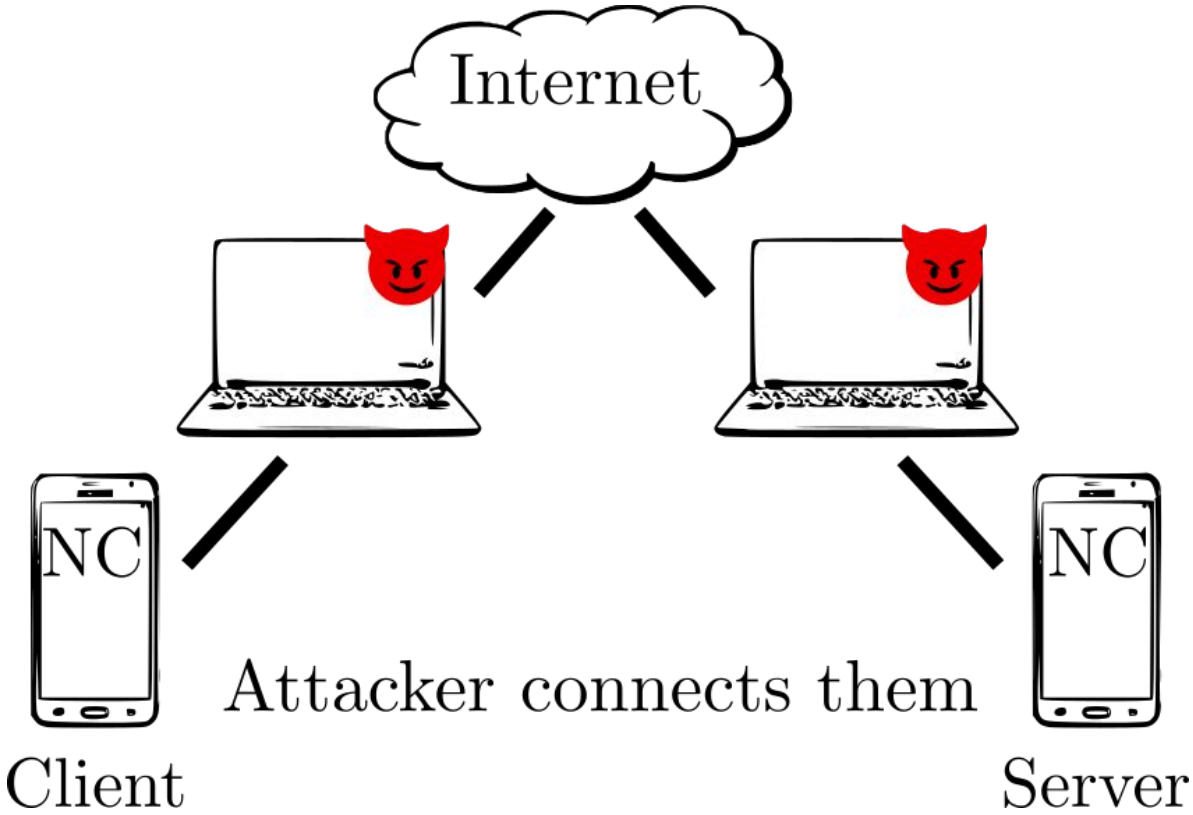
- ECDH-based
- TLS-like `algo` negotiation
- Client commitment
- Custom kdf



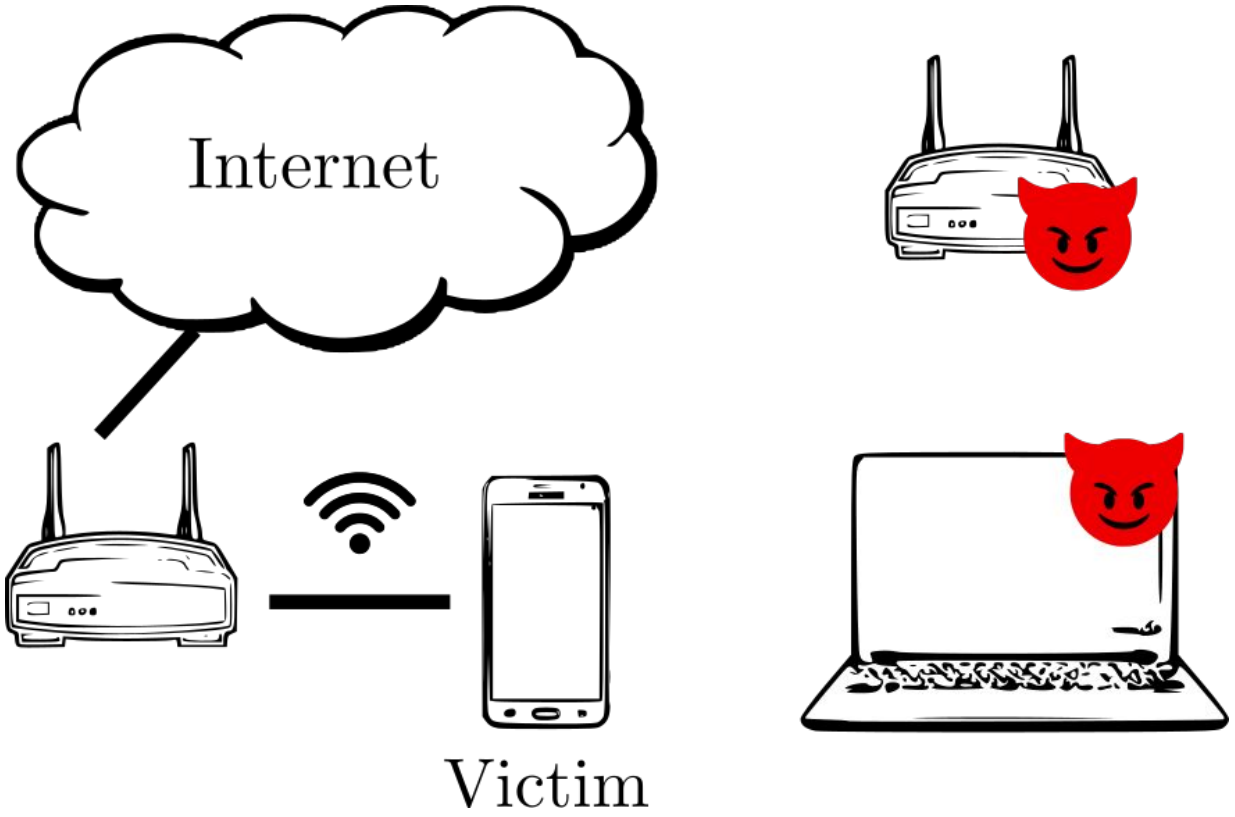
Nearby Connections: Range Extension Attack (1)



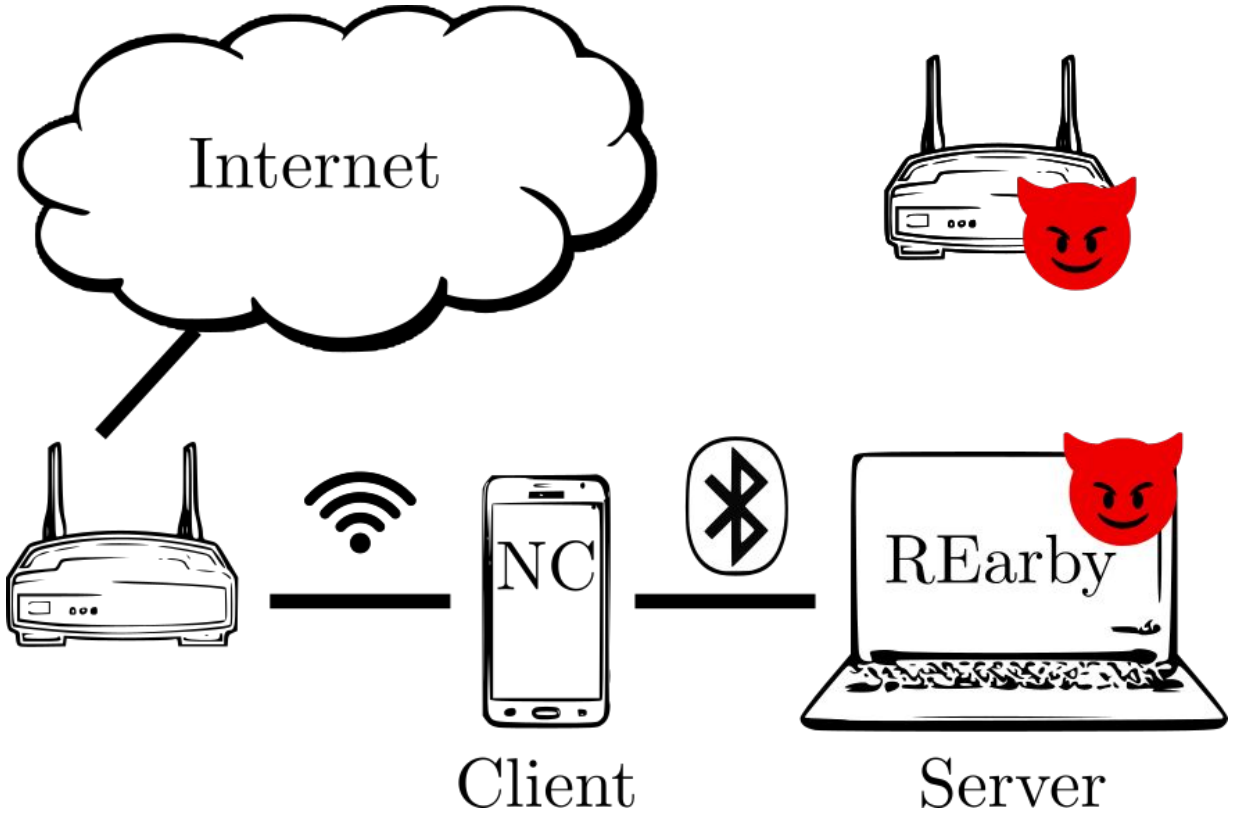
Nearby Connections: Range Extension Attack (2)



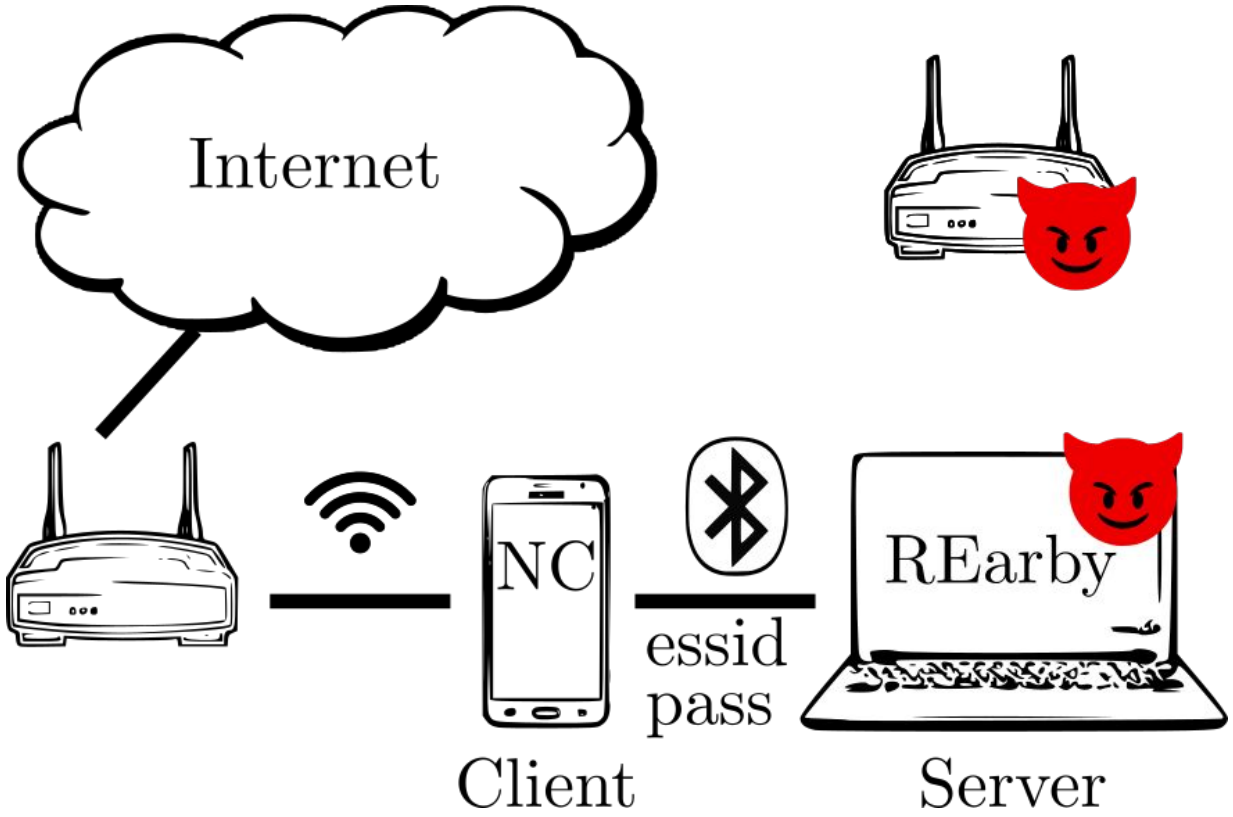
Nearby Connections: Wi-Fi Takeover (1)



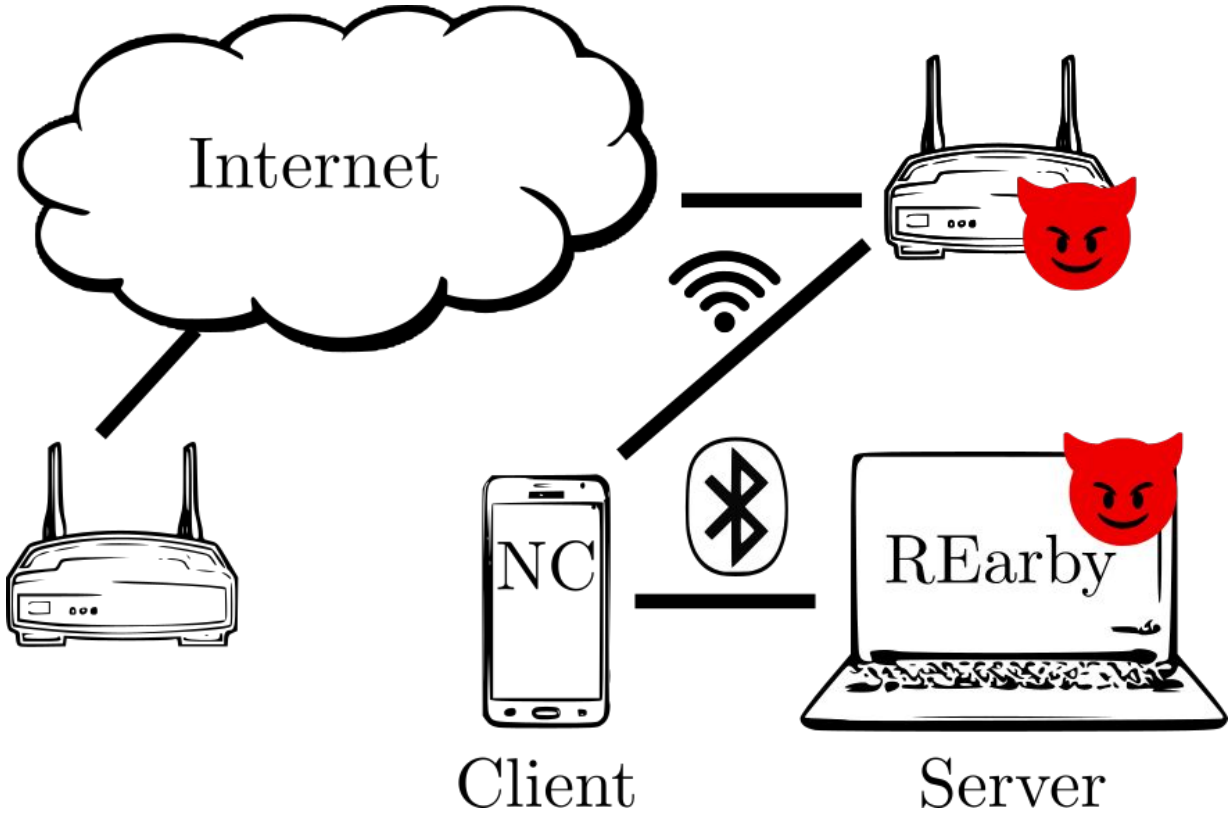
Nearby Connections: Wi-Fi Takeover (2)



Nearby Connections: Wi-Fi Takeover (3)

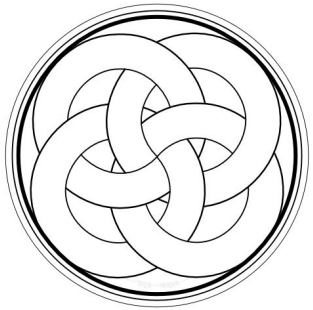


Nearby Connections: Wi-Fi Takeover (4)



Why is hard to secure Nearby Connections

- Android support
- Transparent to end user
- Low and high bandwidth
- Energy/power efficient
- Proprietary Application Layer



3 - Exposure Notification (EN)

Executive Summary

This document describes and analyzes a system for secure and privacy-preserving proximity tracing at large scale. This system provides a technological foundation to help slow the spread of SARS-CoV-2 by simplifying and accelerating the process of notifying people who might have been exposed to the virus so that they can take appropriate measures to break its transmission chain. The system aims to minimise privacy and security risks for individuals and communities and guarantee the highest level of data protection.

https://francozappa.github.io/project/dp3t/

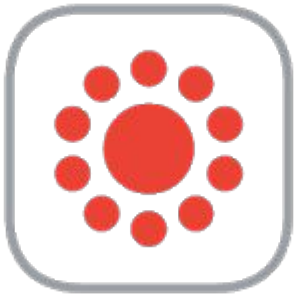
Exposure Notification (1)

``Combat the spread of the coronavirus, by alerting participants about possible exposure, through someone they have recently been in contact with who has subsequently been positively diagnosed''

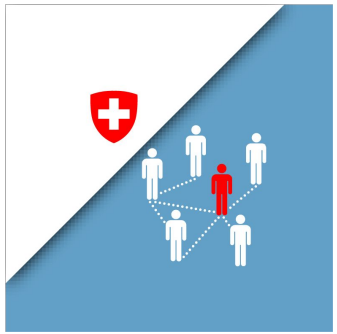
https://www.google.com/covid19/exposurenotifications/

Exposure Notification (2)

- Works for Android and iOS
- Apps use it as a service
- Decentralized with local data
- Backend to check exposure
- Open crypto and BLE specs
- Proprietary implementations
- Privacy-preserving

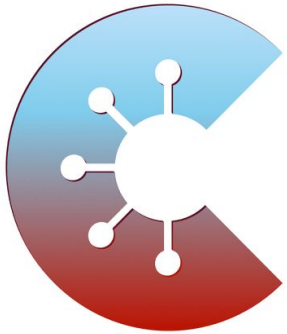


Worldwide adoption of Exposure Notification



SwissCovid (CH)

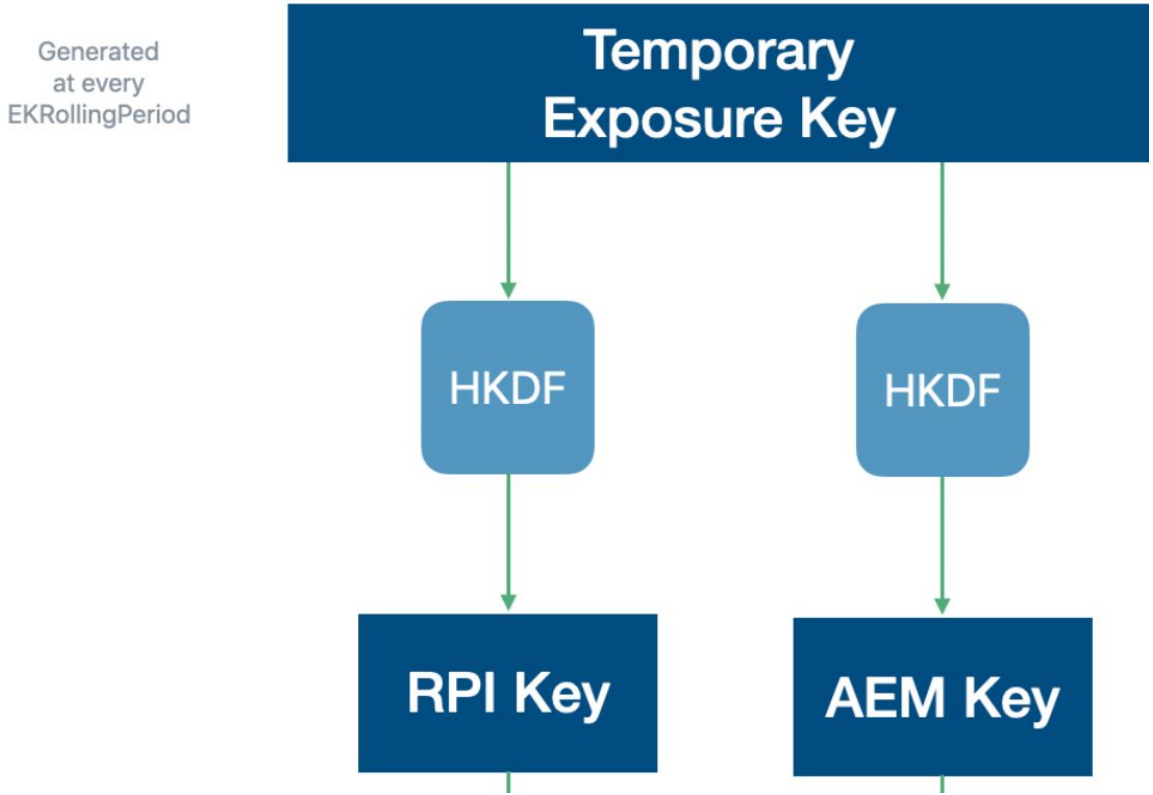
Immuni (IT)



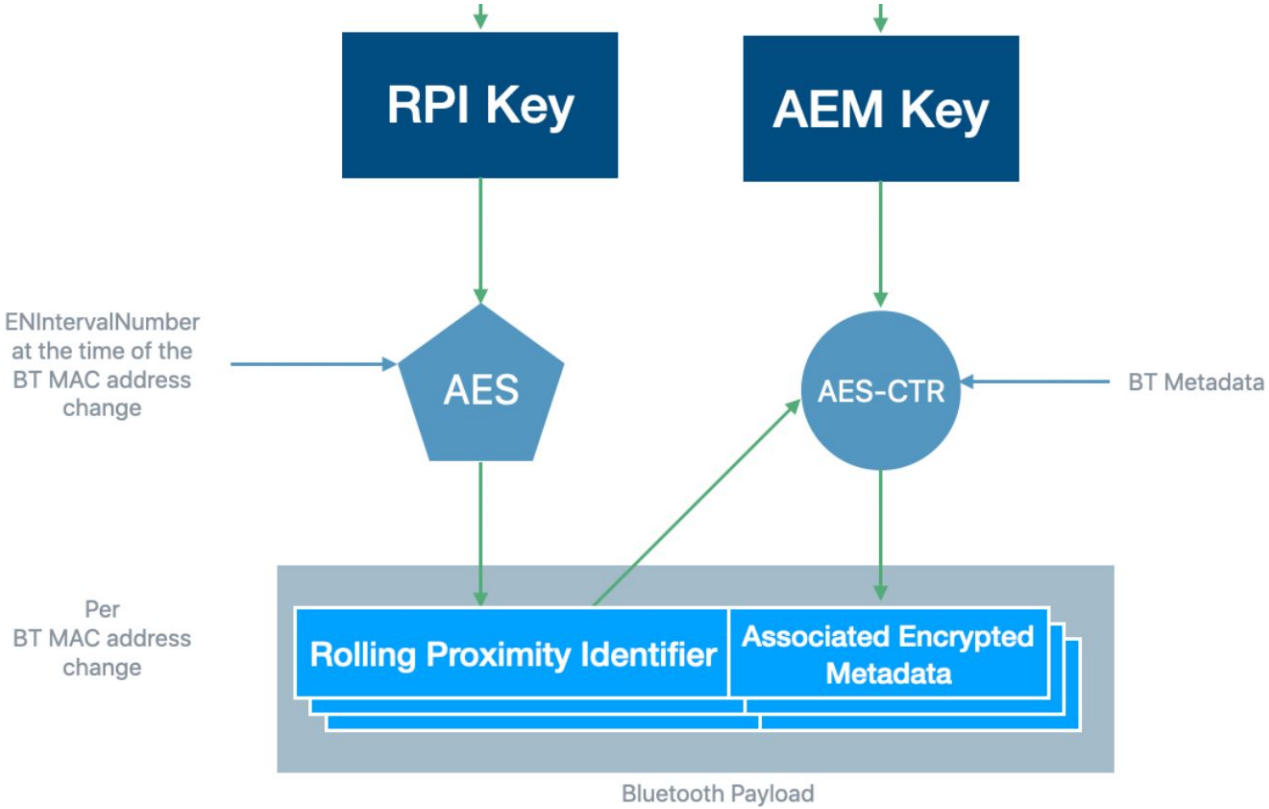
Corona-Warn (DE)

https://en.wikipedia.org/wiki/Exposure_Notification

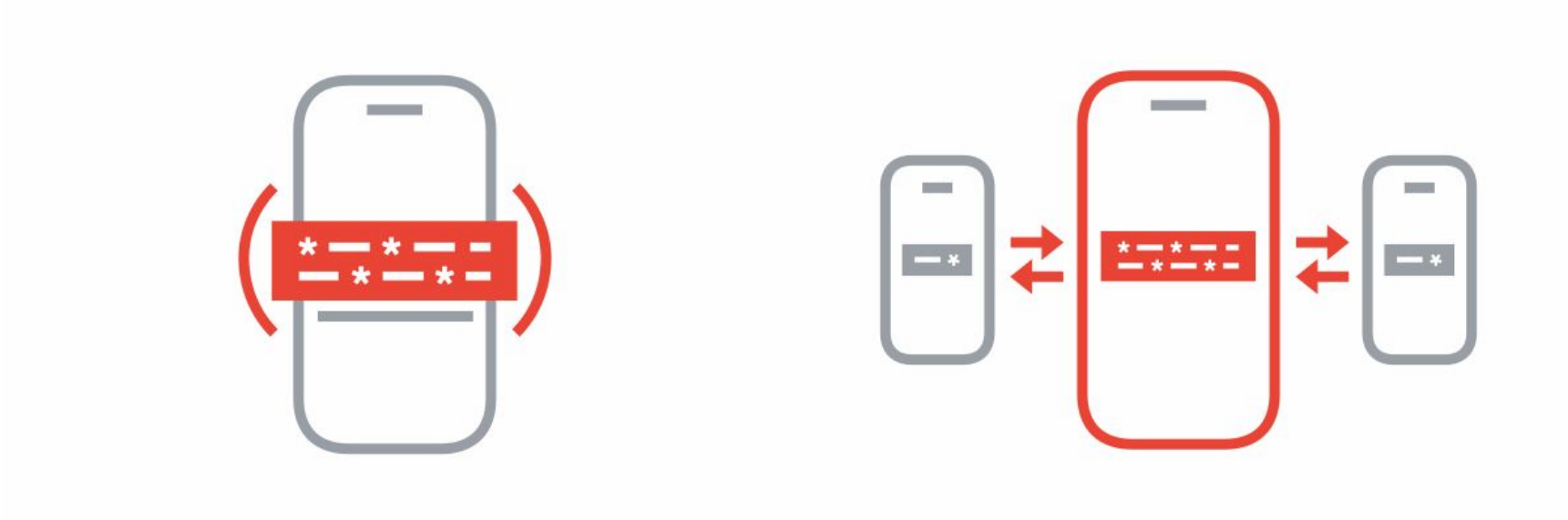
Exposure Notification: Key Schedule (1)



Exposure Notification: Key Schedule (2)



Exposure Notification: Communication (1)



Exposure Notification: Communication (2)

Exposure Notification: Privacy Guarantees

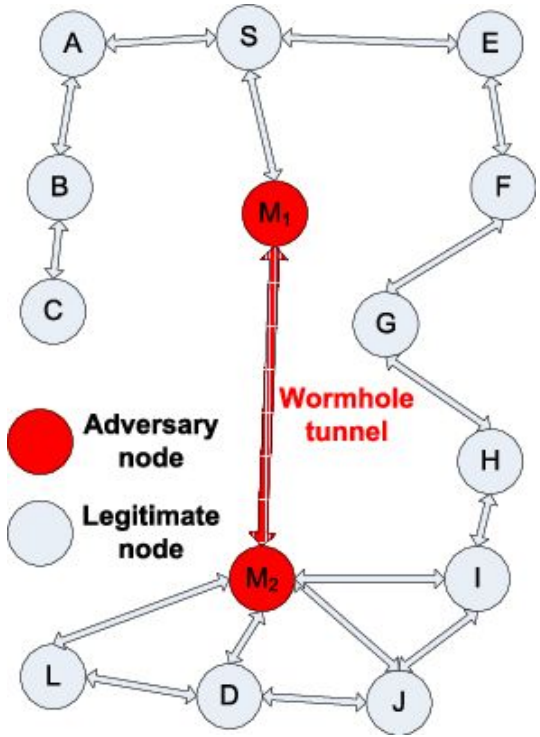
- **Anonymity**
- **Unlinkability**
- **Data collection minimization**

Exposure Notification: Security Issues

- **Proximity IDs sent in clear**
- **Proximity IDs are not integrity protected**

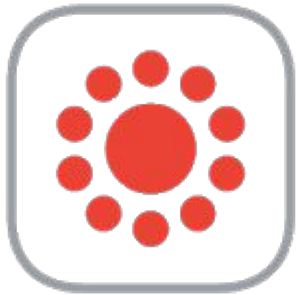
Exposure Notification: Wormhole Attack

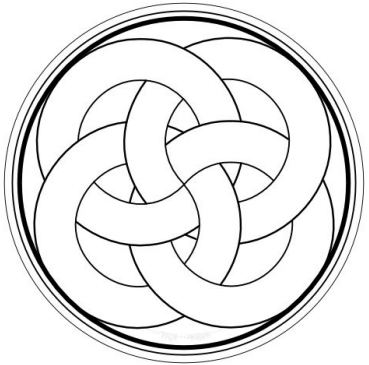
- Take an proximity ID from location A
- Relay it to location B
- Broadcast it in location B



Why is hard to secure Exposure Notifications

- Supported by the Android and iOS
- Connection-less (no key agreement)
- BLE (RSSI) proximity estimation
- Transparent to end user
- Power/energy efficient

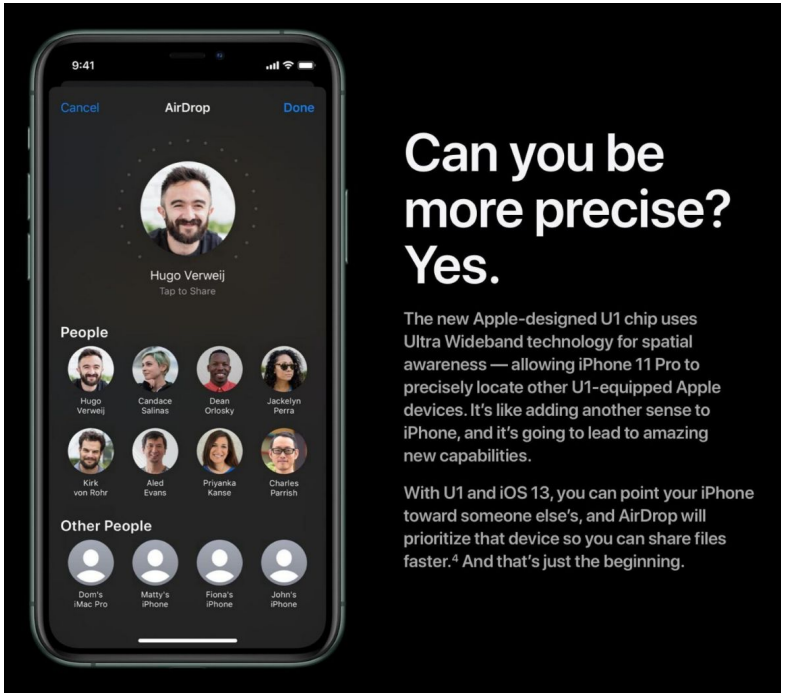




4 - Research Trends

Better Proximity Estimation: UWB

- Ultra Wide Band (UWB)
  - E.g., Apple's U1 chip (see AirTags)
  - Secure data? Availability?

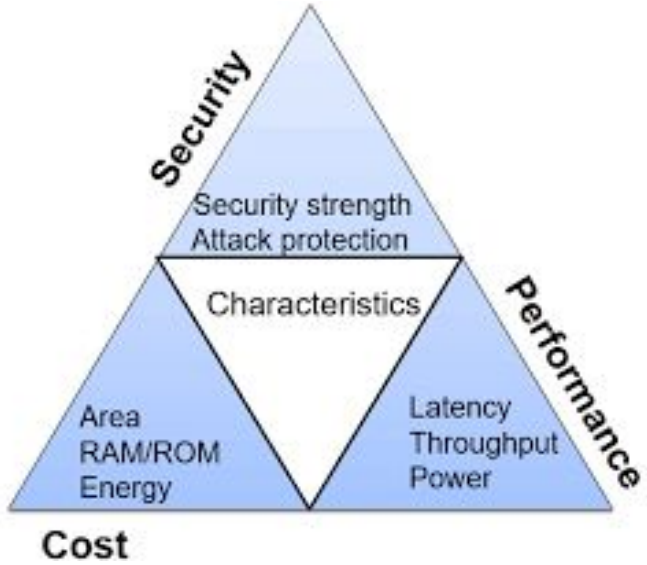


Better Scalability: dedicated HW/SW

- Special-purpose technologies
  - E.g., Exposure Notification wearable
  - Producer? Convince users

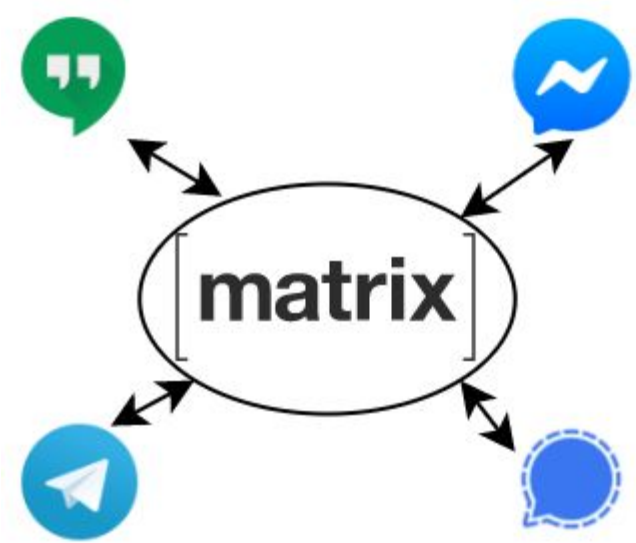
Better Security: Lightweight Crypto

- SW or HW optimized crypto
  - E.g., NIST LC 21 final candidates
  - Asymmetric crypto? Side channels?



Better Privacy: Full Decentralization

- Decentralized proximity services
  - E.g., decentralize the backend
  - Liability?, Privacy vs. Security



This is it! Any Question?

- References
  - D. Antonioli et al, *“Nearby Threats: Reversing, Analyzing, and Attacking Google’s Nearby Connections on Android”* [NDSS19]
  - D. Antonioli et al, *“DP3T Decentralized Privacy-Preserving Proximity Tracing”* [whitepaper]
- More at https://francozappa.github.io/

EURECOM
Sophia Antipolis