black hat®
ASIA 2017

MARCH 28-31, 2017
MARINA BAY SANDS / SINGAPORE

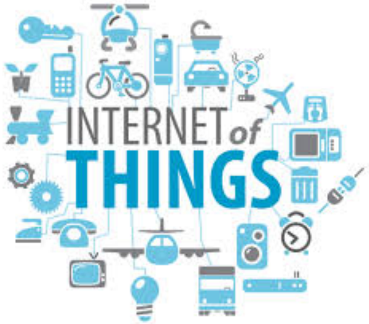
MASHaBLE:

Mobile Applications of Secret Handshakes over Bluetooth Low-Energy

Yan Michalevsky, Suman Nath, Jie Liu

Motivation

- Private communication
- Anonymous messaging
- Secret communities
- Location-based messaging
- Privacy preserving IoT applications



Messaging Applications

Yik Yak

After School

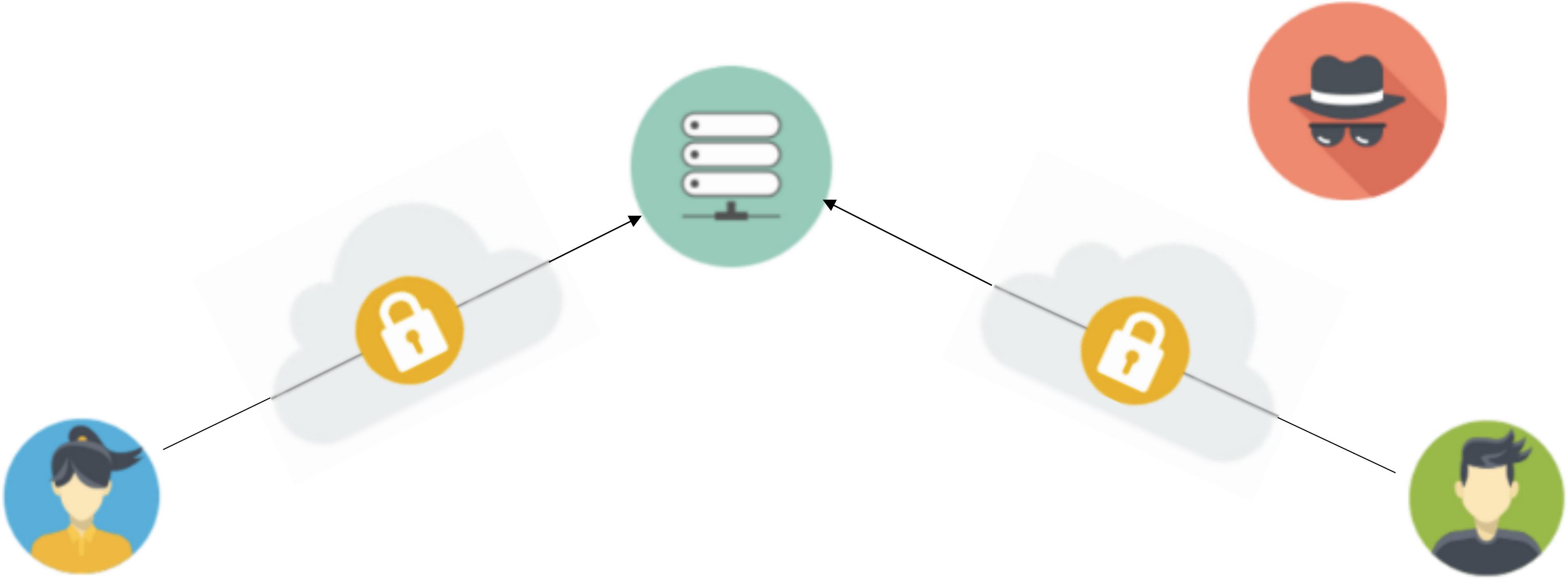
LEGATALK

Yak Server knows everything about the users

Secret communities

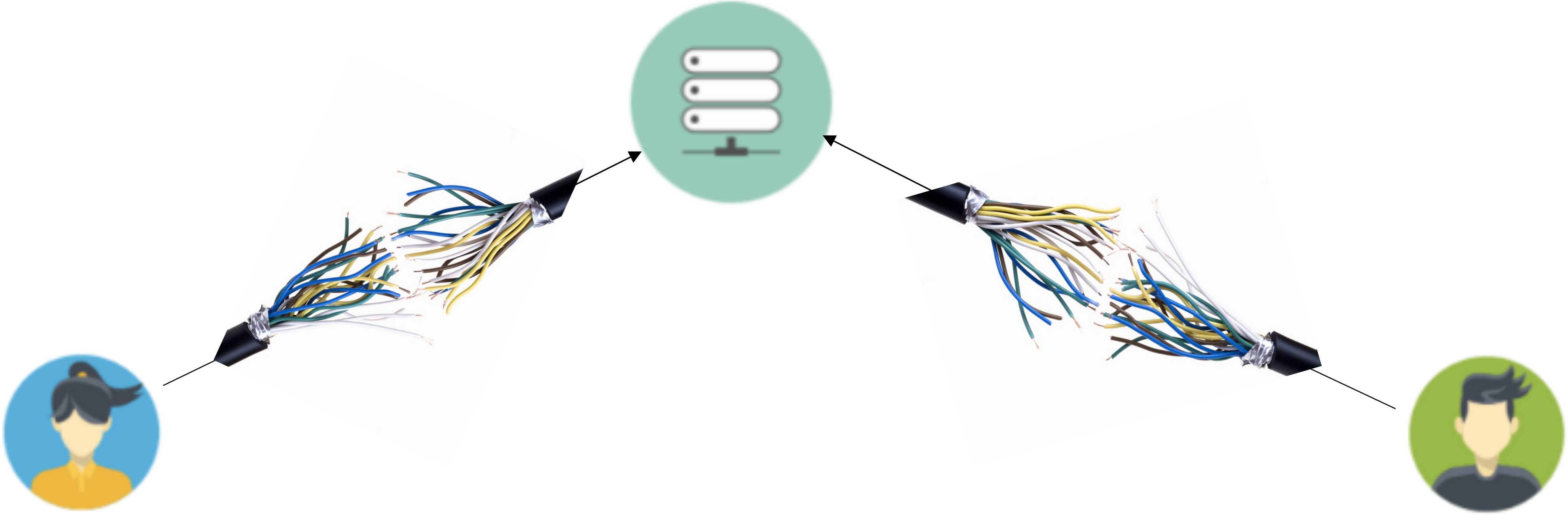
- Members want identify each other
- Do not want to be discovered by anyone not in the community
- Geo-location privacy
- Anonymous messaging and notifications dissemination

“Trusted” Central Server



- The server becomes a target for attacks
- Communicating with the server can reveal affiliation

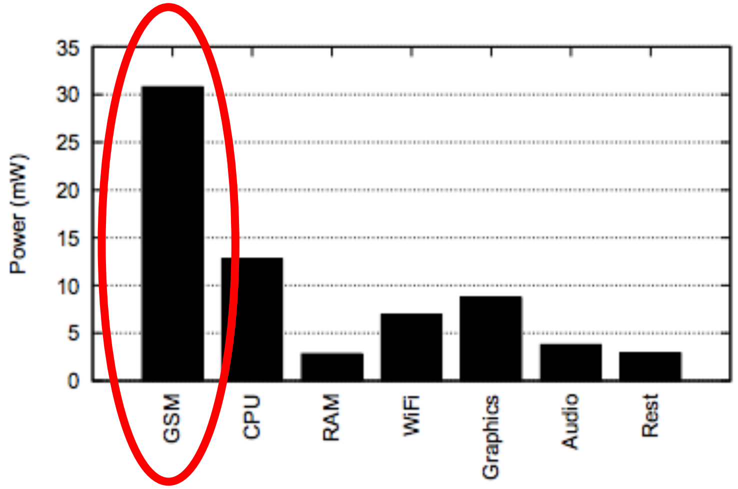
“Trusted” Central Server



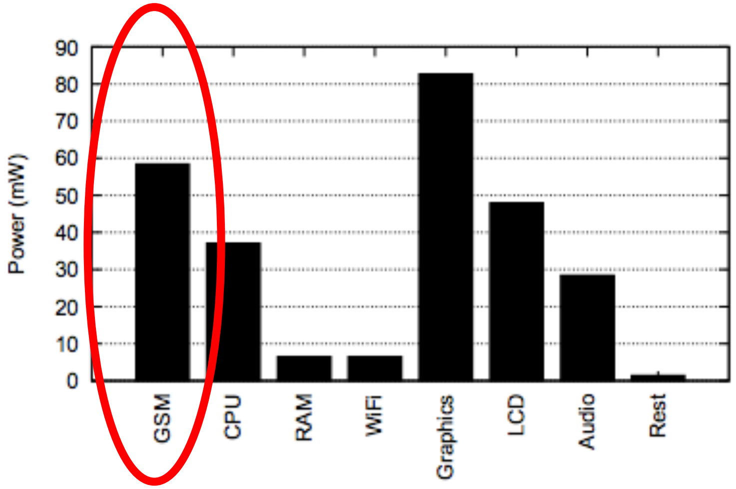
Internet connectivity is not always available

“Trusted” Central Server

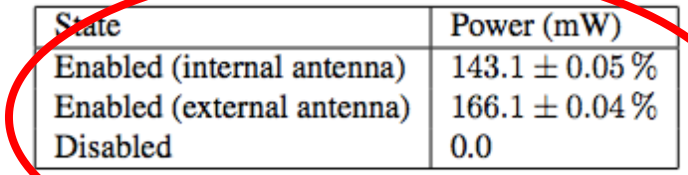
Also… GPS and cellular consume a lot of energy



Suspended state



Idle state



State	Power (mW)
Enabled (internal antenna)	143.1 ± 0.05 %
Enabled (external antenna)	166.1 ± 0.04 %
Disabled	0.0

GPS

We want to...

- Avoid interaction with a server
- Use physical proximity
- Minimize energy consumption

Bluetooth Low-Energy (LE) sounds like a promising solution

Bluetooth LE

But first, the devices need to trust each other…

The problem with negotiating trust

- Alice is willing to reveal its credentials only to another party with certain clearance (needs to verify Bob's identity first)
- Bob is also willing to reveal its credentials only to another party with certain clearance (needs to verify Alice's identity first)
- No party is willing to reveal its credentials and provide a proof of their authenticity first

Properties of a Secret Handshake

- Parties do no know each other
- They perform a procedure that establishes trust
- If it fails – no information is gained by either party
- If it succeeds – parties reveal membership in a group
  - In addition, they can establish respective roles in that group (cryptographic secret handshakes)

More applications of secret handshakes

- Using iBeacon for headcounting
  - Like doubledutch
  - Currently exposes users and event to tracking

Headcounting

- Exposes users to tracking
- Reveals information about the event/gathering
- How do we support private/secret events and provide privacy to attendants?

Secret handshake from pairings

- Based on Balfanz et al. [1]
- If handshake succeeds – both parties have established an authenticated and encrypted communication channel
- If handshake fails – no information is disclosed
- Collusion resistant
    - Corrupted group members cannot collude to perform a handshake of a non-corrupted member
- Compact credentials – important for embedding into small packets

Pairings

We have elements $X \in \mathrm{G}_1$ and $Y \in \mathrm{G}_2$ where $\mathrm{G}_1, \mathrm{G}_2$ are groups over Elliptic Curves

A pairing e has the following property

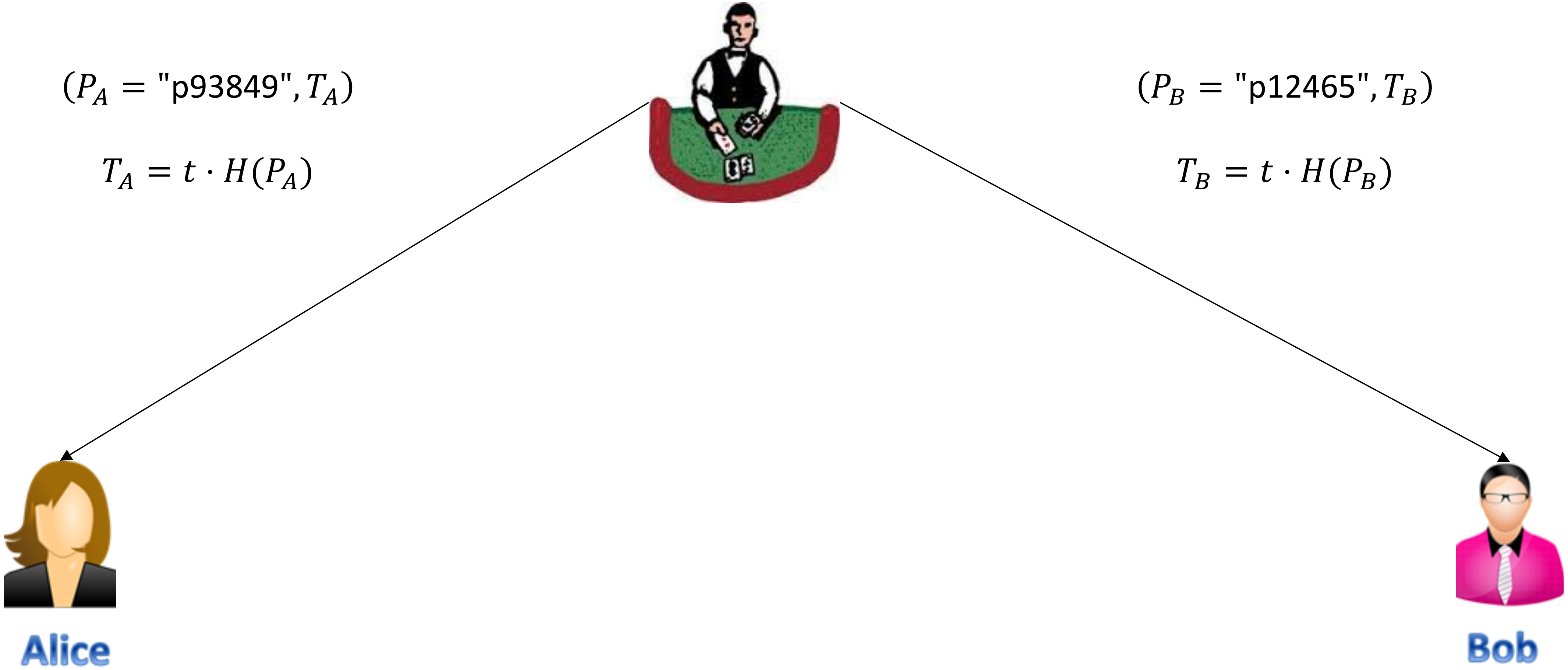
$$e(aX, bY) = e(X, Y)^{ab}$$

Where $\mathrm{e}(X, Y) \in G_T$

Secret handshake from pairings

Master secret

$t \in Z_Q$



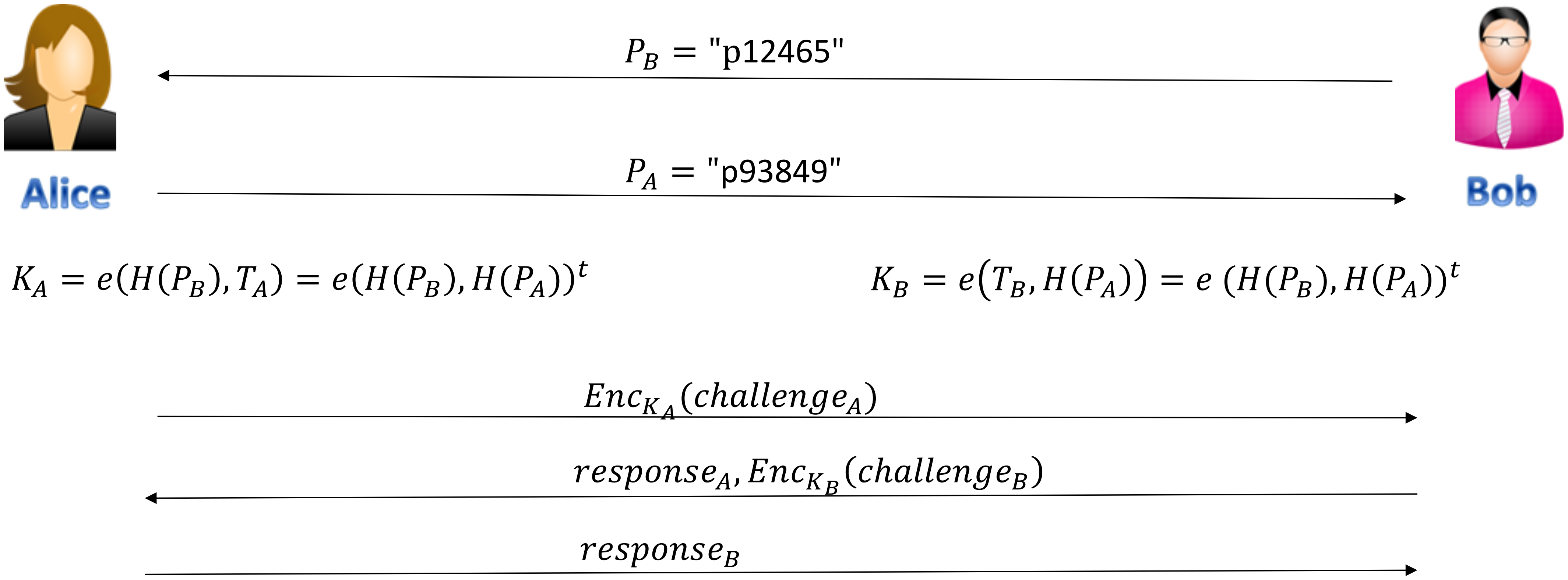
$(P_A = \text{"p93849"}, T_A)$

$T_A = t \cdot H(P_A)$

$(P_B = \text{"p12465"}, T_B)$

$T_B = t \cdot H(P_B)$

Secret handshake from pairings



$P_B =$ "p12465"

$P_A =$ "p93849"

$$K_A = e(H(P_B), T_A) = e(H(P_B), H(P_A))^t$$

$$K_B = e\big(T_B, H(P_A)\big) = e\ (H(P_B), H(P_A))^t$$

$Enc_{K_A}(challenge_A)$

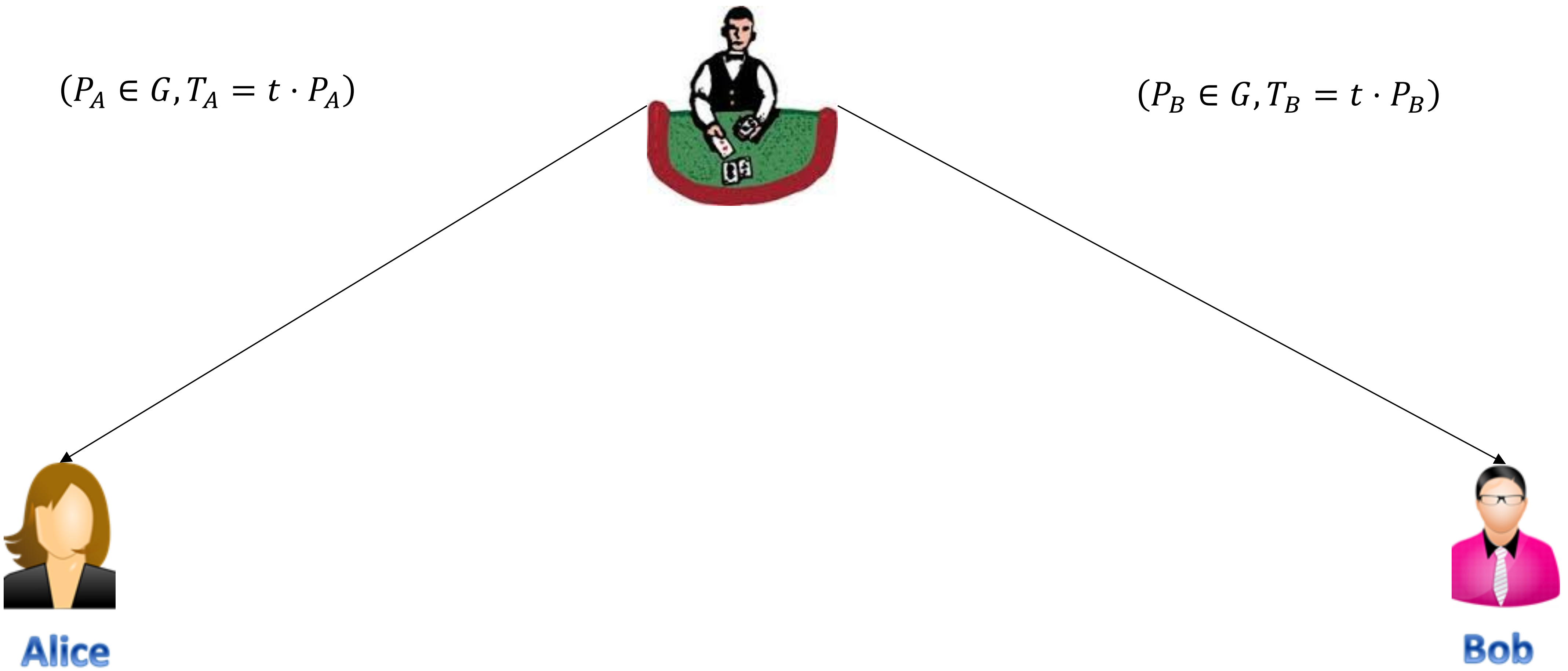
$response_A, Enc_{K_B}(challenge_B)$

$response_B$

Unlinkable Handshakes

- By tracking the pseudonym an attacker can track the user
- Naïve solution:
  - Obtain multiple pseudonyms from master party
  - Use a different pseudonym for each handshake

Unlinkable Secret Handshake



Master secret
$t \in Z_Q$

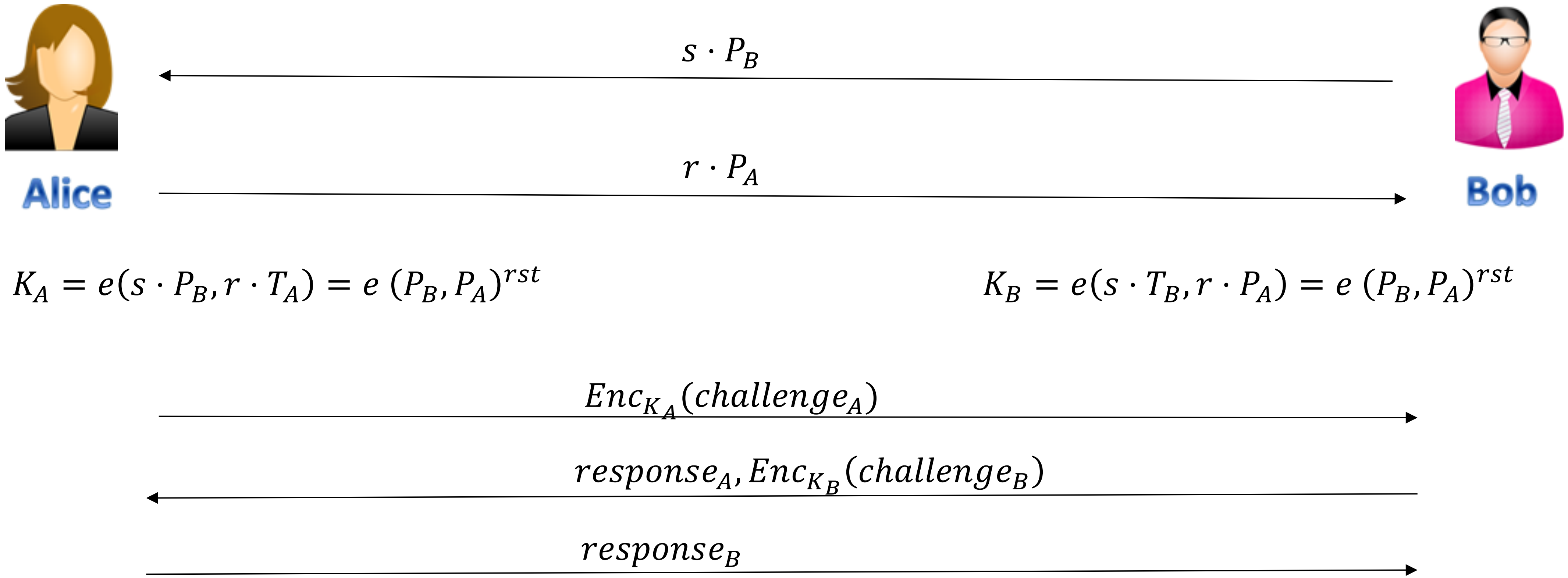
$(P_A \in G, T_A = t \cdot P_A)$

$(P_B \in G, T_B = t \cdot P_B)$

Alice

Bob

Unlinkable Secret Handshake



Alice

Bob

$s \cdot P_B$

$r \cdot P_A$

$K_A = e(s \cdot P_B, r \cdot T_A) = e\ (P_B, P_A)^{rst}$

$K_B = e(s \cdot T_B, r \cdot P_A) = e\ (P_B, P_A)^{rst}$

$Enc_{K_A}(challenge_A)$

$response_A, Enc_{K_B}(challenge_B)$

$response_B$

Some details

- Need to hash arbitrary strings onto G_2
  - Supported by Type 1 or Type 3 pairings
- Group element sizes
  - 128-bit security: 256-bit group element size = 32 bytes
  - 80-bit security: 160-bit element size = 20 bytes

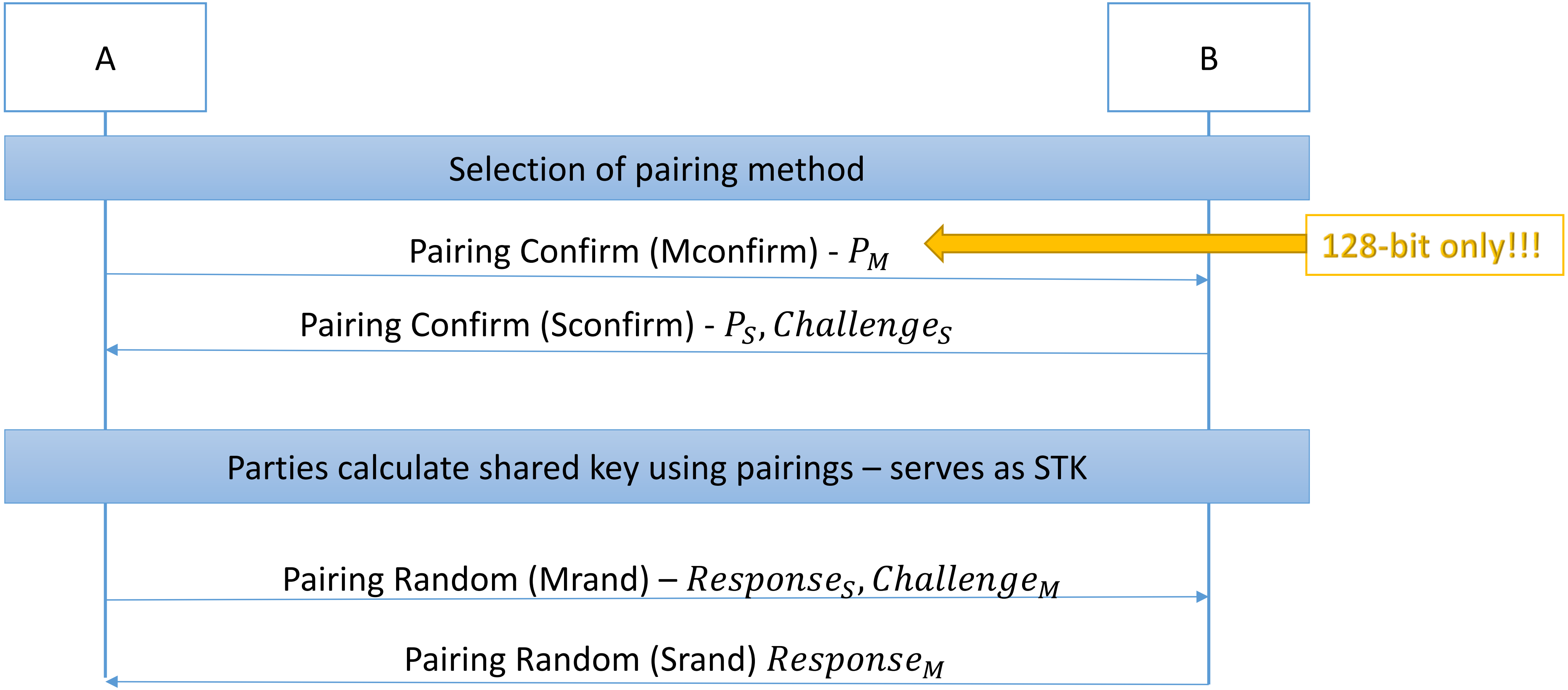
Tracking prevention

- *Random device address* for Bluetooth source address field
    - Set dynamically and changed across different connections

Pairing methods

- Just Works
  - Basically no MITM protection during pairing phase
- Passkey entry
  - Proven to be quite weak [7]
- Out-of-Band (OOB) – credentials provided by some other method

Proposal: New pairing mode

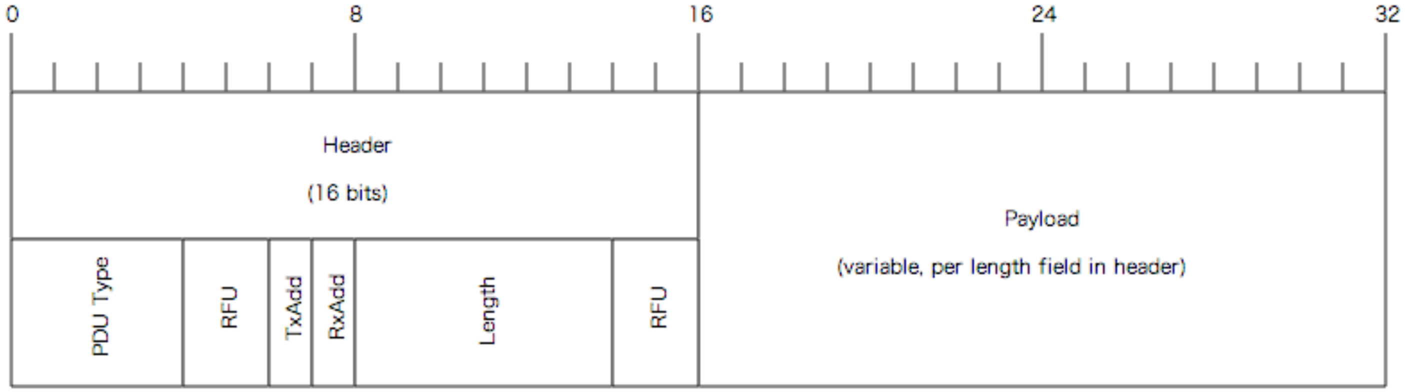


Bluetooth LE Advertisements

- Scanning is supported by
  - Windows phone
  - Android
  - iOS
- Publishing advertisements is supported on
  - Windows phone 10
  - Android: Google Nexus 5x and on
  - Kits such as Cypress and Dialog

Bluetooth LE advertisements

- Bluetooth LE supports broadcasting advertisements
- Clients can scan and filter advertisements of specific types
- A little custom data can be squeezed in – 32 bytes
    - On Windows BTLE stack we currently can only control the Manufacturer Specific Data (AD type 0xFF) – 20 bytes

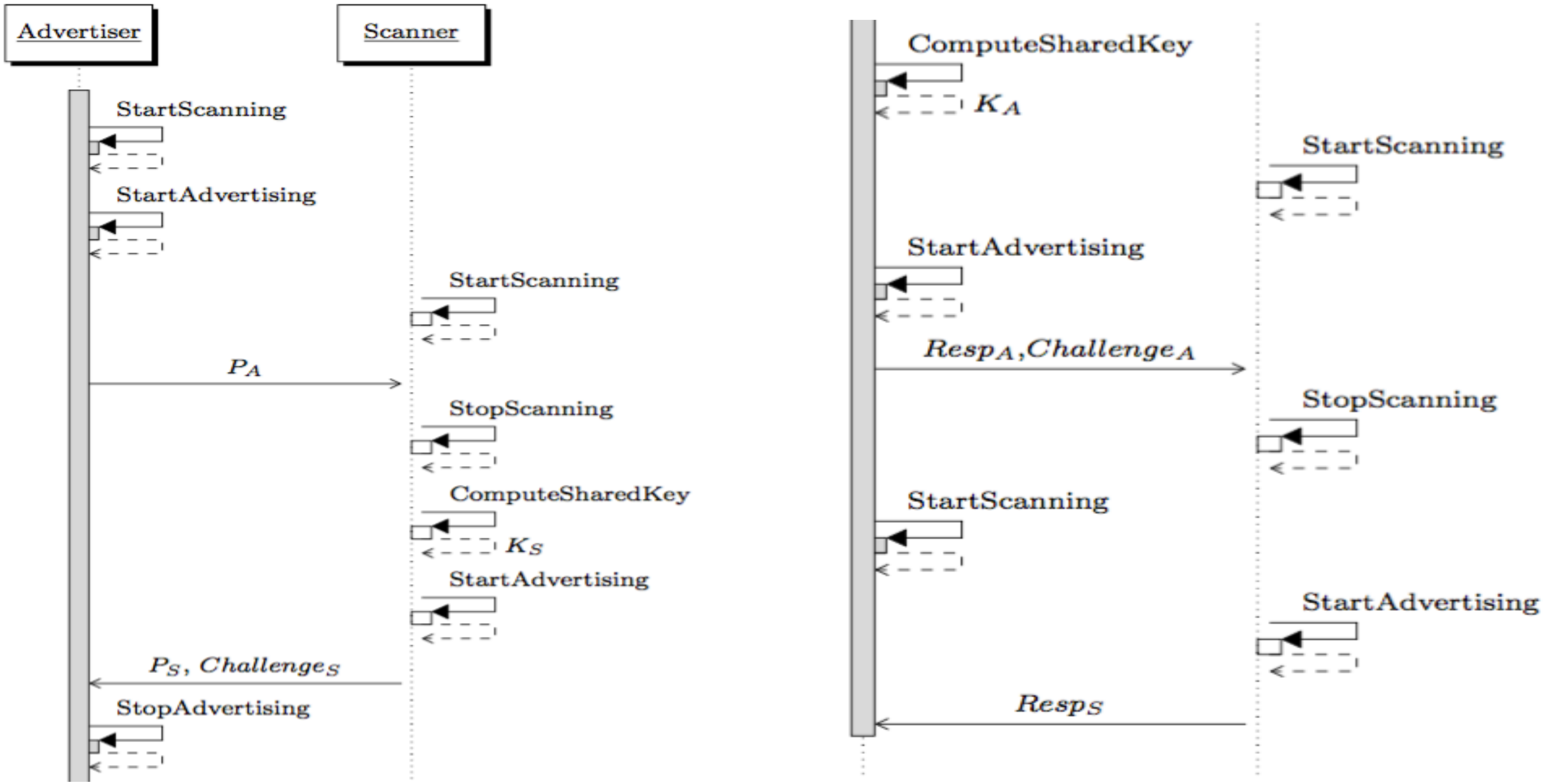


Choice of platform

- Easy implementation of pairings
  - JPBC – Java port of Stanford PBC library
- Support for BLE advertisement publishing
  - Android exposed the API but did not support advertising in practice at the time (but Nexus 5S and on do)
- Windows Phone
  - Supports scanning and advertising
  - Possible to scan and advertise at the same time

Implementation

- Windows Phone OS 10
- Failed attempt: porting JPBC to .NET
- Pairings and group operations using Stanford PBC library
    - Ported to ARM + .NET wrapper (*PbcProxy*)
    - Used MPIR library (Multi-Precision Integers and Rationals, compatible with GMP)
    - Adapted random number generation
- Communication between two phones is based on alternation between advertising and scanning


Advertiser
Scanner
StartScanning
StartAdvertising
StartScanning
P_A
StopScanning
ComputeSharedKey
K_S
StartAdvertising
P_S, $Challenge_S$
StopAdvertising
ComputeSharedKey
K_A
StartScanning
StartAdvertising
$Resp_A$,$Challenge_A$
StopScanning
StartScanning
StartAdvertising
$Resp_S$

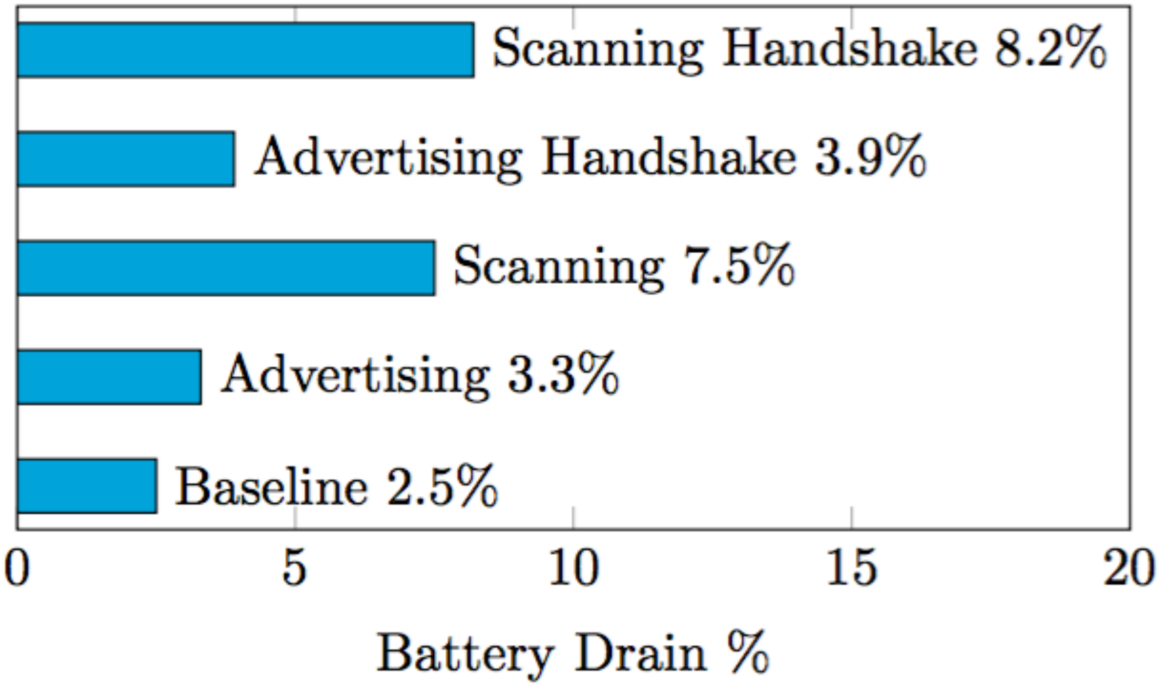
Evaluation: Functionality

- Two mobile phones running our app and performing handshakes
- Experiment duration: 8296 sec = 2 hours 18 sec
- 1 handshakes every 8 seconds
- Total 1068 handshakes
- 1025 succeeded, 43 failed. Success rate: 96%

Evaluation: Energy Consumption

- Nokia Lumia 920 running Windows Phone OS
- Starting with 100% charge, Wi-Fi and GPS off
- Modes:
  - Baseline
  - Advertising
  - Scanning
  - Advertising + handshake
  - Scanning + handshake
- Experiment duration: 3 hours

Evaluation: energy consumption



Percentage of battery drain/hour. Enables >12 hours of operation.

Communication overhead

- Advertisement packet: 47 bytes
- Each party sends 2 packets: 94 bytes

Future work

- Implementation for Android
    - New Nexus devices have sufficient BLE support
- Pairing preprocessing
    - For each handshake using the same credentials preprocessing can be applied
    - Supported by PBC library
- Use BLE specific identifiers as handshake pseudonyms
    - Set a custom *source device address*
    - Would provide additional usable space for longer pseudonyms
- More Windows Universal applications using *PbcProxy*

Black Hat Sound Bytes

- Secret Handshakes – a provably secure primitive with useful applications
- We can easily achieve better security and privacy for mobile and IoT
- Evaluation shows the application is fit for practical use in mobile devices

Thanks for attending!

Questions?

Related work

- Automatic Trust Negotiation (ATN)
- Attribute-Based Encryption (ABE)
  - Decryption is possible if party is certified as possessing certain attributes by an authority
- Secret handshakes [1]
  - Each party receives a certificate from a central authority
- Hidden credentials [2]
  - Protect the messages using policies that require possession of multiple credentials
- Oblivious Signature-Based Envelope (OSBE) [8]
  - Allows certificates issued by different authorities
- Secret handshakes from CA-oblivious encryption [9]
- Unlinkable secret handshakes and key-private group key management schemes [10]

References

1. Secret handshakes from pairing-based key agreements [Balfanz et al. 2003]
2. Hidden credentials [Holt et al. 2003]
3. Authenticated Identity-Based Encryption [Lynn 2002]
4. How tracking customers in stores will soon be norm
5. How retail stores track you using your smartphone (and how to stop it)
6. Apple is quietly making its move to own in-store digital tracking
7. Bluetooth: With Low Energy comes Low Security [Ryan 2013]
8. Oblivious Signature-Based Envelope [Li et al. 2003]
9. Secret handshakes from CA-oblivious encryption [Casteluccia et al. 2004]
10. Unlinkable secret handshakes and key-private group key management schemes [Jarecki et al. 2007]