MASHaBLE

Mobile Applications of Secret Handshakes over Bluetooth LE

Yan Michalevsky (Stanford University)

Suman Nath, Jie Liu (Microsoft Research)
CONTEXT AND MOTIVATION

- Communities & Groups
- Proximity-based communication
- Privacy & Security
EXAMPLES OF ANDROID AND IOS APPS

- Telegram
- Signal
- After School
- LEGATAK
- Yik Yak
SECRET COMMUNITIES

- Members want to identify each other
- Do not want to be discovered by anyone not in the community
- Geo-location privacy
  - Proximity-based communication
- Anonymous messaging and notifications dissemination (gossip)
STATE OF THE ART - “TRUSTED” CENTRAL SERVER

The server becomes a target for attacks
“TRUSTED” CENTRAL SERVER – REQUIRES CONNECTIVITY

Internet connectivity is not always available
“TRUSTED” CENTRAL SERVER – ENERGY CONSUMING COMMUNICATION

GPS and cellular consume a lot of energy

Suspended state

Idle state

<table>
<thead>
<tr>
<th>State</th>
<th>Power (mW)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enabled (internal antenna)</td>
<td>143.1 ± 0.05 %</td>
</tr>
<tr>
<td>Enabled (external antenna)</td>
<td>166.1 ± 0.04 %</td>
</tr>
<tr>
<td>Disabled</td>
<td>0.0</td>
</tr>
</tbody>
</table>
OUR GOALS

▪ Avoid a central server
  ▪ Use P2P communication

▪ Energy efficiency
  ▪ Use Bluetooth LE

• Low cost / low bandwidth
• Minimizes power consumption
• Operates on short ranges
• Supported by many smartphones

But, with no central server, there is no root of trust...
THE PROBLEM WITH TRUST

I DON’T WANT TO IDENTIFY MYSELF BEFORE HE DOES...

I DON’T WANT TO IDENTIFY MYSELF BEFORE SHE DOES...

ALICE

ROBERT
he hates being called Bob.
SECRET HANDSHAKES

- 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)
SECRET HANDSHAKES 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
- Tracking is addressed by unlinkability techniques
PAIRINGS

We have elements $X \in G_1$ and $Y \in G_2$ where $G_1, G_2$ are algebraic groups

A pairing $e$ has the following property

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

Where $e(X, Y) \in G_T$
SECRET HANDSHAKE FROM PAIRINGS

Master secret
\( t \in \mathbb{Z}_Q \)

\((P_A = "p93849", T_A)\)

\(T_A = t \cdot H(P_A)\)

\((P_B = "p12465", T_B)\)

\(T_B = t \cdot H(P_B)\)
SECRET HANDSHAKE FROM PAIRINGS

Master secret
\( t \in \mathbb{Z}_Q \)

\( (P_A = "p93849", T_A) \)

\( T_A = t \cdot H(P_A) \)

\( (P_B = "p12465", T_B) \)

\( T_B = t \cdot H(P_B) \)
SECRET HANDSHAKE FROM PAIRINGS

\[ P_B = "p12465" \]

\[ P_A = "p93849" \]

\[ K_A = e(H(P_B), T_A) = e(H(P_B), H(P_A))^t \]

\[ K_B = e(T_B, H(P_A)) = e(H(P_B), H(P_A))^t \]

\[ Enc_{K_A}(\text{challenge}_A) \]

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
- Cryptographic unlinkability
UNLINKABLE SECRET HANDSHAKE

Master secret
\( t \in \mathbb{Z}_Q \)

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

(\( P_B \in G, T_B = t \cdot P_B \))
UNLINKABLE SECRET HANDSHAKE

Master secret
$t \in \mathbb{Z}_Q$

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

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

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

\[ K_B = e(s \cdot T_B, r \cdot P_A) = e(P_B, P_A)^{rst} \]
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 [Ryan’13]
- Out-of-Band (OOB) – credentials provided by some other method
PROPOSAL: NEW PAIRING MODE

A

Selection of pairing method

Pairing Confirm (Mconfirm) - $P_M$

Pairing Confirm (Sconfirm) - $P_S, Challenge_S$

128-bit only!!!

B

Parties calculate shared key using pairings – serves as STK

Pairing Random (Mrand) – $Response_S, Challenge_M$

Pairing Random (Srand) $Response_M$
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
BLUETOOTH LE ADVERTISEMENTS

- Scanning is supported by
  - Windows phone
  - Android
  - iOS

- Publishing advertisements is supported on
  - Windows phone 10
  - Possibly future Android phone versions
  - Kits such as Cypress and Dialog
CHOICE OF PLATFORM

▪ Easy implementation of pairings
  ▪ JPBC – Java port of Stanford PBC library

▪ iOS and Android did not support publishing
  ▪ Android exposed the API but did not support advertising in practice

▪ Windows Phone
  ▪ Supports scanning and advertising
  ▪ Possible to scan and advertise at the same time
IMPLEMENTATION

▪ Windows Phone OS 10

▪ 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
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
- Experiment duration: 3 hours
- Results are stated as drain % per hour

Enables >12 hours of operation
COMMUNICATION OVERHEAD

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

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

How do we prevent car and driver tracking?

Secret handshakes provide the desirable anonymity and unlinkability
FUTURE WORK

- Computational speed-up
  - 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*
CONCLUSION

▪ We presented a SH protocol on top of BLE
  ▪ Provides provable security and privacy
  ▪ Practical and energy efficient
  ▪ Enables new applications
THANKS
Questions?
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