NORMAN RESEARCH

CVE-2020-13702: catastrophic breach of user privacy in Apple/Google COVID-19 Exposure Notification API

Authors: Norman Luhrmann, V

daughter-of-v@protonmail.com / PGP key DF20D21B349C2A9A

Revision: 1.0 (2020-05-30) Initial draft for CVE code reservation/reviewing
1.1 (2020-06-11) Public disclosure, formatting and minor changes
1.2 (2020-06-16) Dispute rebuttal
1.3 (2020-06-16) Add clarification from Google

Classification

Severity: 10.0 CRITICAL

CVSS v3.1 Vector: /AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:H/A:N/E:H/RL:U/RC:C/CR:H/IR:H/AR:X/MAV:N/MAC:L/MPR:N/MUI:N/MS:C/MC:H/MI:H/MA:X

Vulnerability Type: CWE-359 (Exposure of Private Personal Information to an Unauthorized Actor)

Vendor of Product: Apple, Google

Affected Product Code Base: Android 6.0 or higher, IOS 13,5 or higher

Affected Component: Exposure Notification API, Bluetooth LE

Attack Type: Remote

Attack Vectors: Bluetooth Smart Privacy is broken in API due to the addition of secondary temporary UID. Bluetooth LE discovery mechanism can be used to track individual device movement across a fleet of devices

Update 2020-06-16/2:

As has been aligned between Google internal findings and our tests with the Google Play Services implementation the RPI always rotates the BT MAC address at the same time, while in the other direction the dual rotation is not applied as per specification guidance.

Update 2020-06-16:

Google has responded to the report as WONTFIX since the specification also includes the following remark:

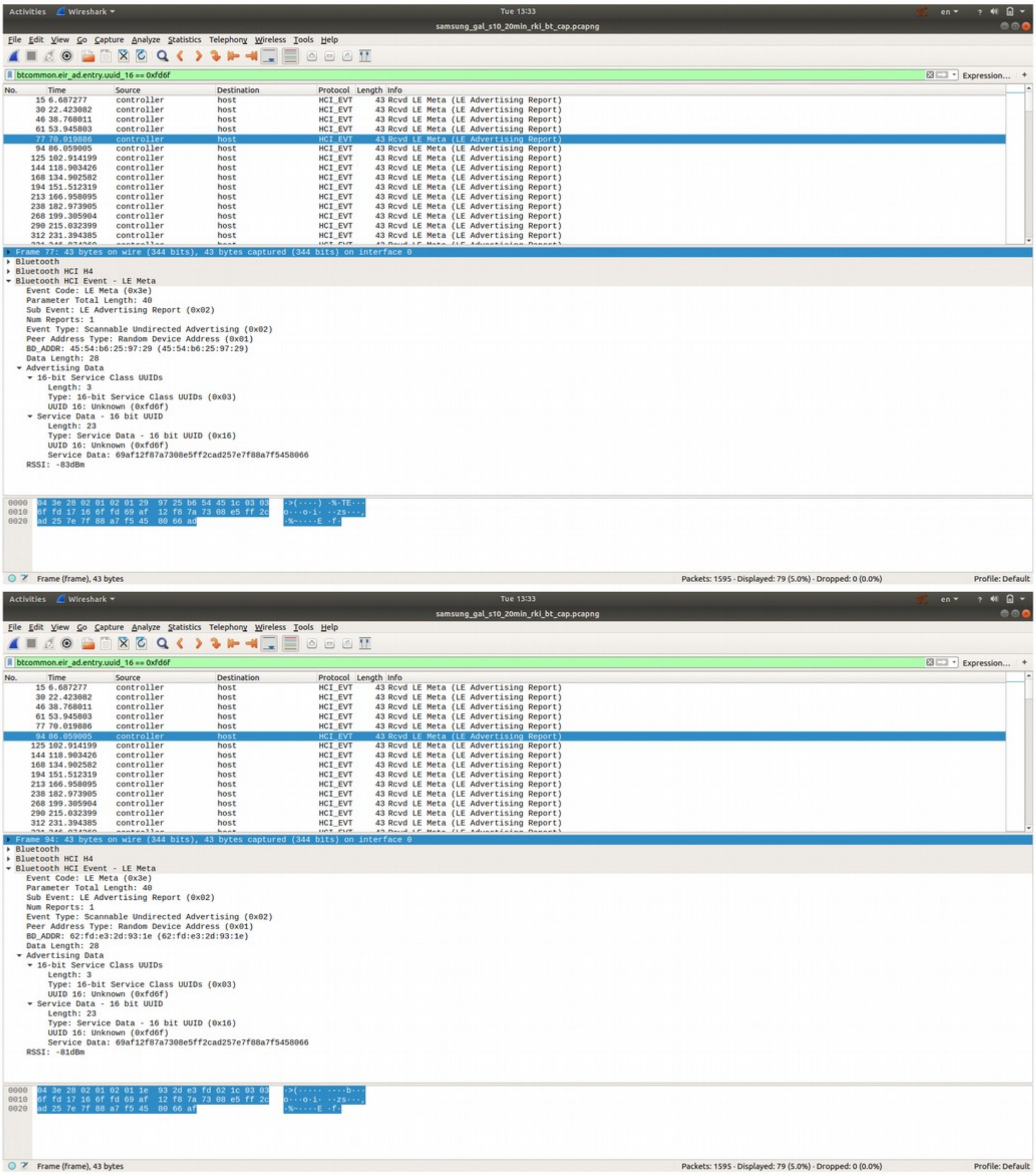
“The advertiser address, Rolling Proximity Identifier, and Associated Encrypted Metadata shall be changed synchronously so that they cannot be linked.”

This remark is in direct conflict with the rolling identifiers timer specs. We have now also traced the official German Corona Warn app by RKI which utilizes the ExposureNotification API.

Traces have shown that the rolling identifier features which are part of the Google Play Services implementation of the Apple/Google specification are in itself faulty, displaying asynchronous rotation. This proofs that the Google implementation of ExposureNotification is prone to the reported issue.

Similar tests with Apple devices are still outstanding.

The dispute of this CVE should be rebutted with this new information since the Google reference implementation itself is observing the information leak.


Activities
Wireshark
Tue 13:33
samsung_gal_s10_20min_rkl_bt_cap.pcapng
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
btcommon.eir_ad.entry.uuid_16 == 0xfd6f
Expression…

```
No.       Time        Source       Destination  Protocol  Length Info
   15 6.687277       controller    host         HCI_EVT       43 Rcvd LE Meta (LE Advertising Report)
   30 22.423082      controller    host         HCI_EVT       43 Rcvd LE Meta (LE Advertising Report)
   46 38.768011      controller    host         HCI_EVT       43 Rcvd LE Meta (LE Advertising Report)
   61 53.945803      controller    host         HCI_EVT       43 Rcvd LE Meta (LE Advertising Report)
   77 70.019886      controller    host         HCI_EVT       43 Rcvd LE Meta (LE Advertising Report)
   94 86.059005      controller    host         HCI_EVT       43 Rcvd LE Meta (LE Advertising Report)
  125 102.914199     controller    host         HCI_EVT       43 Rcvd LE Meta (LE Advertising Report)
  144 118.903426     controller    host         HCI_EVT       43 Rcvd LE Meta (LE Advertising Report)
  168 134.902582     controller    host         HCI_EVT       43 Rcvd LE Meta (LE Advertising Report)
  194 151.512319     controller    host         HCI_EVT       43 Rcvd LE Meta (LE Advertising Report)
  213 166.958095     controller    host         HCI_EVT       43 Rcvd LE Meta (LE Advertising Report)
  238 182.973905     controller    host         HCI_EVT       43 Rcvd LE Meta (LE Advertising Report)
  268 199.305904     controller    host         HCI_EVT       43 Rcvd LE Meta (LE Advertising Report)
  290 215.032399     controller    host         HCI_EVT       43 Rcvd LE Meta (LE Advertising Report)
  312 231.394385     controller    host         HCI_EVT       43 Rcvd LE Meta (LE Advertising Report)

▸ Frame 77: 43 bytes on wire (344 bits), 43 bytes captured (344 bits) on interface 0
▸ Bluetooth
▸ Bluetooth HCI H4
▾ Bluetooth HCI Event - LE Meta
    Event Code: LE Meta (0x3e)
    Parameter Total Length: 40
    Sub Event: LE Advertising Report (0x02)
    Num Reports: 1
    Event Type: Scannable Undirected Advertising (0x02)
    Peer Address Type: Random Device Address (0x01)
    BD_ADDR: 45:54:b6:25:97:29 (45:54:b6:25:97:29)
    Data Length: 28
  ▾ Advertising Data
    ▾ 16-bit Service Class UUIDs
        Length: 3
        Type: 16-bit Service Class UUIDs (0x03)
        UUID 16: Unknown (0xfd6f)
    ▾ Service Data - 16 bit UUID
        Length: 23
        Type: Service Data - 16 bit UUID (0x16)
        UUID 16: Unknown (0xfd6f)
        Service Data: 69af12f87a7308e5ff2cad257e7f88a7f5458066
    RSSI: -83dBm

0000  04 3e 28 02 01 02 01 29  97 25 b6 54 45 1c 03 03   ·>(····) ·%·TE···
0010  6f fd 17 16 6f fd 69 af  12 f8 7a 73 08 e5 ff 2c   o···o·i· ··zs···,
0020  ad 25 7e 7f 88 a7 f5 45  80 66 ad                  ·%~····E ·f·

Frame (frame), 43 bytes                Packets: 1595 · Displayed: 79 (5.0%) · Dropped: 0 (0.0%)      Profile: Default
```

Activities
Wireshark
Tue 13:33
samsung_gal_s10_20min_rkl_bt_cap.pcapng
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
btcommon.eir_ad.entry.uuid_16 == 0xfd6f
Expression…

```
No.       Time        Source       Destination  Protocol  Length Info
   15 6.687277       controller    host         HCI_EVT       43 Rcvd LE Meta (LE Advertising Report)
   30 22.423082      controller    host         HCI_EVT       43 Rcvd LE Meta (LE Advertising Report)
   46 38.768011      controller    host         HCI_EVT       43 Rcvd LE Meta (LE Advertising Report)
   61 53.945803      controller    host         HCI_EVT       43 Rcvd LE Meta (LE Advertising Report)
   77 70.019886      controller    host         HCI_EVT       43 Rcvd LE Meta (LE Advertising Report)
   94 86.059005      controller    host         HCI_EVT       43 Rcvd LE Meta (LE Advertising Report)
  125 102.914199     controller    host         HCI_EVT       43 Rcvd LE Meta (LE Advertising Report)
  144 118.903426     controller    host         HCI_EVT       43 Rcvd LE Meta (LE Advertising Report)
  168 134.902582     controller    host         HCI_EVT       43 Rcvd LE Meta (LE Advertising Report)
  194 151.512319     controller    host         HCI_EVT       43 Rcvd LE Meta (LE Advertising Report)
  213 166.958095     controller    host         HCI_EVT       43 Rcvd LE Meta (LE Advertising Report)
  238 182.973905     controller    host         HCI_EVT       43 Rcvd LE Meta (LE Advertising Report)
  268 199.305904     controller    host         HCI_EVT       43 Rcvd LE Meta (LE Advertising Report)
  290 215.032399     controller    host         HCI_EVT       43 Rcvd LE Meta (LE Advertising Report)
  312 231.394385     controller    host         HCI_EVT       43 Rcvd LE Meta (LE Advertising Report)

▸ Frame 94: 43 bytes on wire (344 bits), 43 bytes captured (344 bits) on interface 0
▸ Bluetooth
▸ Bluetooth HCI H4
▾ Bluetooth HCI Event - LE Meta
    Event Code: LE Meta (0x3e)
    Parameter Total Length: 40
    Sub Event: LE Advertising Report (0x02)
    Num Reports: 1
    Event Type: Scannable Undirected Advertising (0x02)
    Peer Address Type: Random Device Address (0x01)
    BD_ADDR: 62:fd:e3:2d:93:1e (62:fd:e3:2d:93:1e)
    Data Length: 28
  ▾ Advertising Data
    ▾ 16-bit Service Class UUIDs
        Length: 3
        Type: 16-bit Service Class UUIDs (0x03)
        UUID 16: Unknown (0xfd6f)
    ▾ Service Data - 16 bit UUID
        Length: 23
        Type: Service Data - 16 bit UUID (0x16)
        UUID 16: Unknown (0xfd6f)
        Service Data: 69af12f87a7308e5ff2cad257e7f88a7f5458066
    RSSI: -81dBm

0000  04 3e 28 02 01 02 01 1e  93 2d e3 fd 62 1c 03 03   ·>(····· ····b···
0010  6f fd 17 16 6f fd 69 af  12 f8 7a 73 08 e5 ff 2c   o···o·i· ··zs···,
0020  ad 25 7e 7f 88 a7 f5 45  80 66 af                  ·%~····E ·f·

Frame (frame), 43 bytes                Packets: 1595 · Displayed: 79 (5.0%) · Dropped: 0 (0.0%)      Profile: Default
```

Fig. A: screenshot of two advertising events displaying a rotation of MAC address and traceability through stable ServiceData

Description

The Rolling Proximity Identifier used in the Apple/Google Exposure Notification API beta through 2020-05-29 enables attackers to circumvent Bluetooth Smart Privacy because there is a secondary temporary UID. An attacker with access to Beacon or IoT networks can seamlessly track individual device movement via a Bluetooth LE discovery mechanism.

Additional information

The authors of this CVE stay anonymous to avoid personal ramifications. The vulnerability has already been communicated to specific international media outlets in advance. Taking the potential scale of deployment for this API and the omnipresence of Bluetooth beacons/IoT devices this issue is catastrophic if left to roll out. Due to the "beta" test phase of the API a disclosure with the vendors is neither required nor desired - action needs to be taken ASAP to bring this API development on course or find alternative solutions, a vendor reporting cycle delay towards both parties needs to be avoided therefor.

Executive summary

Apple and Google have joined forces to provide the Exposure Notification API as a secure baseline for contact tracing app developers. In their API release (currently in beta) they accidentally destroyed Bluetooth Smart Privacy, allowing seamless device location tracking. Adding current beacon network coverage and broken IoT security to this scenario, and considering the potential acceptance rate of such a technology, this might be an Orwellian nightmare on our doorsteps. This API development needs to be brought on course with international data protection legislations (COPPA, GDPR, PRC Cybersecurity Law, etc.) or alternative solutions should be investigated.

Introduction

This document describes a critical breach of Bluetooth Smart Privacy introduced by Apple and Google in their COVID-19 support infrastructure called Exposure Notification. In a collaboration Apple and Google decided to tackle the difficult problem of enabling a global scale contact tracing app infrastructure while applying cryptographic means to avoid that user privacy is affected by this new technology.

Numerous initiatives have been started early on in the COVID-19 pandemic to provide local solutions for digital contact tracing via smartphone apps. Most of them raising high concerns about the user privacy impact of such a large-scale apps.

Rest assured that it left me with much wonder as I reviewed the specification released by Apple and Google and quickly stumbled upon what looked like a nuclear security fallout – the API spec effectively destroys the integrity of the Bluetooth Smart Privacy functionality established in 2015 by the Bluetooth SIG [1].

In itself this is already a fatal situation, but the Bluetooth LE context of the Exposure Notification protocol renders this even worse. The attack is fully remote, passive and without any need for device or authentication data access. This implies that the endless number of Bluetooth LE enabled devices world-wide are potential attack vectors, resulting in a tracking risk of unprecedented scale affecting every user that installs an IOS or Android contact tracing app which consumes the Exposure Notification API.

The vulnerability lies in the basic protocol design and as such does not need special exploit code. Any IoT device vendor stack is prone to undesired modification to enable this device tracking. Any Bluetooth LE chip vendor is able to add such tracking under the hood.

Vulnerability details

The described vulnerability sits in the Bluetooth LE part of the specification [2] published in v1.2 at the time of this writing. The encryption architecture designates a 16-bit “Rolling Proximity Identifier” to enable the exchange of contact data between nearby devices without exposing the user identity to the peers.

The details of the encryption protocol are not relevant for this exploit, it is only the identifier rotation interval that needs to be considered. The specification proposes that the rotation should take place “on average every 15 minutes”, which would be the duration within which a user is trackable by a UID. After the rotation the UID is refreshed which prohibits the identification across the rotation segment.

In the same specification the Bluetooth Random Private Address feature of Bluetooth LE devices version 4.2 or later is recommended to receive a rotation set randomly in between 10 and 20 minutes. Bluetooth SIG specifies this MAC address randomisation feature supported by the “LE Set Resolvable Private Address Timeout” command to prevent devices to become trackable [3].

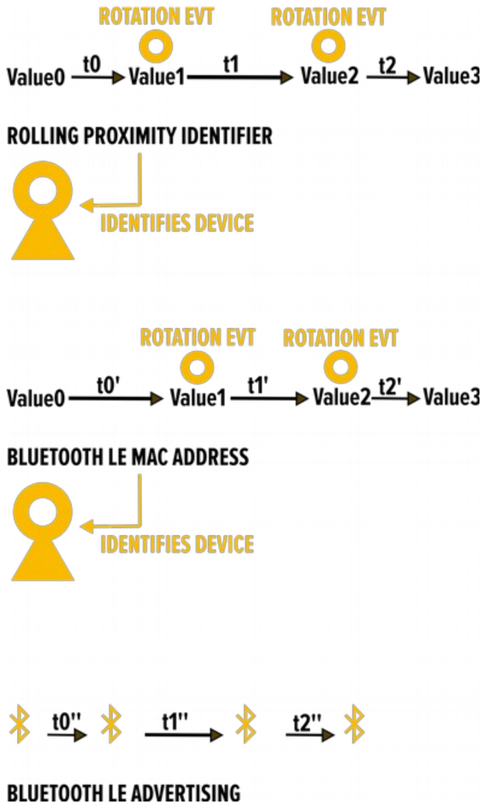


Fig 1.: involved protocol timers (not in scale)

The two differing rotation intervals create the critical flaw in this protocol design. Since both MAC address and proximity identifier are part of the public Bluetooth LE broadcast packets anyone can intercept this data simply by placing a receiver within signal range.

A malicious device network could easily store and correlate both identifiers to track users moving between the device locations. Both Bluetooth Smart Privacy address randomisation and Exposure Notification identifier rotation work as expected independently, but anyone holding both data points can trace devices across rotations.

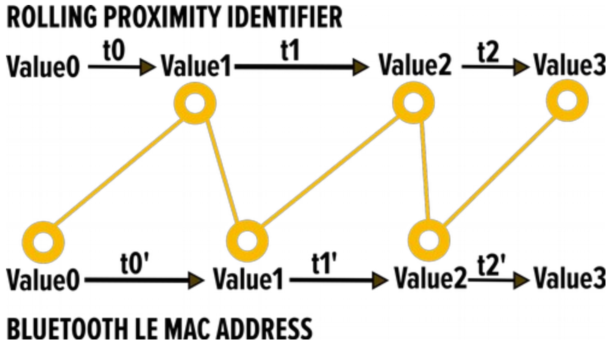


Fig 2.: following device by jumping to dual identifier on rotation event

This results in Bluetooth Smart Privacy getting broken in consequence.

The Exposure Notification specifies an advertising interval of 200-270 milliseconds (presumably in an opportunistic manner). Due to the random nature of the rotations a collision of the is highly unlikely, thus resulting in near-100% tracking accuracy.

Attack vectors

The vulnerability exposes the user identity both on re-scans of a single device and cross-device scans. Re-scanning on a device seems to provide less leverage because the integrity of the encryption protocol still guarantees that no undesired data is exposed, although a follow-a-device scenario is theoretically possible.

The Exposure Notification specifies a scan interval of around 5 minutes (again in opportunistic manner) for API consumer apps, so the granularity is very low for the standard use case. Now a permanently powered device like a beacon can increase scan frequency to capture much more broadcasts.

But even in a high-volume scan scenario the critical impact of this vulnerability occurs when a single malicious actor controls a network of Bluetooth LE devices with central data storage available.

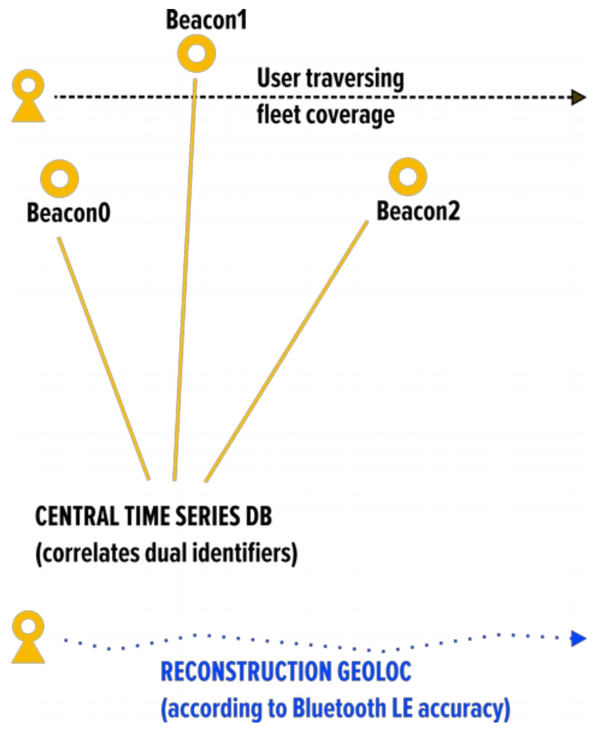


Fig 2. storing both identifiers for all devices in a fleet allows geographic tracking over fleet coverage

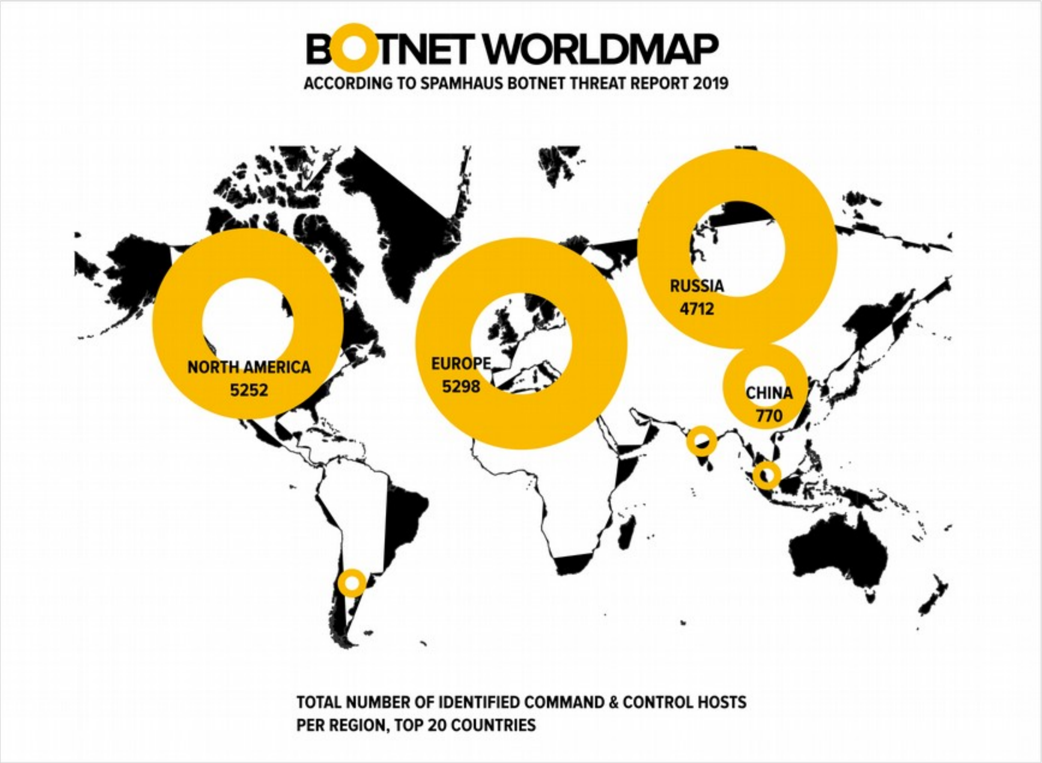
In this case the tracking potential is only limited by the coverage of Bluetooth LE devices within a controlled network. Note that many urban and commercial areas already see full coverage of beacon networks.

IoT security context

In the assessment of this vulnerability the general state of security of IoT devices should be considered. The aforementioned scenario applies to valid Bluetooth LE device network operators, which in itself should not be allowed by Bluetooth Smart Privacy.

The lack of a good security baseline of IoT in general, the numerous product/device variants, and the long runtime/lack of deployment lifecycle multiplies the issue beyond undesired network operator tracking capabilities.

It is a likely scenarios that advanced penetration threat actors like nation states or high profile hacking groups can gain illegit access to Bluetooth LE device fleets. Near global tracking coverage for urban areas is theoretically attainable.


BOTNET WORLDMAP
ACCORDING TO SPAMHAUS BOTNET THREAT REPORT 2019
NORTH AMERICA
5252
EUROPE
5298
RUSSIA
4712
CHINA
770
TOTAL NUMBER OF IDENTIFIED COMMAND & CONTROL HOSTS
PER REGION, TOP 20 COUNTRIES

Proof of concept

Due to the inherent protocol behavior creating the attack vector there is no need to provide a specific PoC implementation.

A simple addition of the two identifier data points to a central store would suffice. In-the-wild device fleets possibly already include these data points in time series databases as they are often used for network logging.

In both cases the location history for a device can simply be queried by hopping between the two identifier data points whenever a rotation occurred, and following through the fleet on device jumps (because the user moved).

Possible rectification

The authors have not verified the other protocol layers in too much depth, but it would seem feasible to replace the rolling proximity identifier with the random MAC address (or an encrypted derivative) for devices supporting Bluetooth Smart Privacy.

Since the API is backwards compatible with very old devices it is possible that the current rolling proximity identifier approach is also required to be kept for non-privacy enabled Bluetooth devices to ensure the integrity as proposed in the specification.

References

[1] https://www.bluetooth.com/blog/bluetooth-technology-protecting-your-privacy/

[2] https://covid19-static.cdn-apple.com/applications/covid19/current/static/contact-tracing/pdf/ExposureNotification-BluetoothSpecificationv1.2.pdf

[3] https://www.bluetooth.com/specifications/bluetooth-core-specification/