Detecting Unwanted Location Trackers
Internet-Draft
Intended status: Informational
Expires: 7 December 2025

C. Fossaceca
Microsoft
E. Rescorla
Independent

5 June 2025

Finding Tracking Tags
draft-ietf-dult-finding-01

Abstract

Lightweight location tracking tags are in wide use to allow users to locate items. These tags function as a component of a crowdsourced tracking network in which devices belonging to other network users (e.g., phones) report which tags they see and their location, thus allowing the owner of the tag to determine where their tag was most recently seen. This document defines the protocol by which devices report tags they have seen and by which owners look up their location.

About This Document

This note is to be removed before publishing as an RFC.

The latest revision of this draft can be found at https://ietf-wg-dult.github.io/draft-ietf-dult-finding/draft-ietf-dult-finding.html. Status information for this document may be found at https://datatracker.ietf.org/doc/draft-ietf-dult-finding/.

Discussion of this document takes place on the Detecting Unwanted Location Trackers Working Group mailing list (mailto:unwanted-trackers@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/unwanted-trackers/. Subscribe at https://www.ietf.org/mailman/listinfo/unwanted-trackers/.

Source for this draft and an issue tracker can be found at https://github.com/ietf-wg-dult/draft-ietf-dult-finding.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

This Internet-Draft will expire on 7 December 2025.

Copyright Notice

Table of Contents

1. Introduction

DISCLAIMER: This draft is work-in-progress and has not yet seen significant (or really any) security analysis. It should not be used as a basis for building production systems.

Lightweight location tracking tags are a mechanism by which users can track their personal items. These tags function as a component of a crowdsourced tracking network in which devices belonging to other network users (e.g., phones) report on the location of tags they have seen. At a high level, location tracking this works as follows:

* Tags ("Accessories") broadcast an advertisement payload containing accessory-specific information. The payload also indicates whether the accessory is separated from its owner and thus potentially lost.

* Devices belonging to other users ("Non-Owner Devices" or "Finder Devices") observe those payloads and if the payload is in a separated mode, reports its location to some central service ("Crowdsourced Network").

* The owner ("Owner Device") queries the central service ("Crowdsourced Network") for the location of their accessory.

A naive implementation of this design exposes users to considerable privacy risk. In particular:

* If accessories simply have a fixed identifier that is reported back to the tracking network, then the central server is able to track any accessory without the user's assistance, which is clearly undesirable.

* Any attacker who can guess or determine a tag ID can query the central server for its location.

* An attacker can surreptitiously plant an accessory on a target and thus track them by tracking their "own" accessory.

Section 6 provides a more detailed description of the desired security privacy properties, but briefly, we would like to have a system in which:

* Nobody other than the owner of an accessory would be able to learn anything about the location of a given accessory.

* It is possible to detect when an accessory is being used to track you.

* It is not possible for accessories that do not adhere to the protocol to use the crowdsourced network protocol.

* It is not possible for unverified accessories to use the crowdsourced network protocol.

A number of manufacturers have developed their own proprietary tracking protocols, including Apple (see [WhoTracks] and [Heinrich]), Samsung (see [Samsung]), and Tile, CUBE, Chipolo, Pebblebee and TrackR (see [GMCKV21]), with varying security and privacy properties.

This document defines a cryptographic reporting and finding protocol which is intended to minimize the above privacy risks. It is intended to work in concert with the requirements defined in [I-D.detecting-unwanted-location-trackers], which facilitate detection of unwanted tracking tags. This protocol design is based on existing academic research surrounding the security and privacy of bluetooth location tracking accessories on the market today, as described in [BlindMy] and [GMCKV21] and closely follows the design of [BlindMy].

2. Motivations

2.1. Stalking Prevention

This work has been inspired by the negative security and privacy implications that were introduced by lightweight location tracking tags, and defined in part by [I-D.detecting-unwanted-location-trackers]. The full threat model is described in detail in [DultDoc4], however, a significant element of the threat model lies in part with the security of the Crowdsourced Network, which will be discussed in detail here.

In addition to its designed uses, the Crowdsourced Network also provided stalkers with a means to track others by planting a tracking tag on them and then using the CN to locate the tracker. Thus, this document outlines the requirements and responsibilities of the Crowdsourced Network to verify the authenticity of the participants, while also preserving user privacy.

* First, the Crowdsourced Network should ensure that only authentic Finding Devices are sending reports to the Crowdsourced Network, and this should occur via an authenticated and encrypted channel. This will help prevent malicious actors from interfering with location reporting services.

* Second, the Crowdsourced Network should ensure that only authorized Owner Devices are able to download location reports, and this should occur via an authenticated and encrypted channel. This will prevent malicious actors from unauthorized access of location data.

* Third, the Crowdsourced Network must follow basic security principles, such as storing location reports in an encrypted manner

  (The benefits of this requirement are self explanatory.)

* Fourth, the Crowdsourced Network must validate that the accessory registered to an owner is valid. This wil prevent malicious actors from leveraging counterfeit accessories.

* Fifth, users should should be able to opt-out of their devices participating in the Crowdsourced Network.

2.2. Prior Research

There is substantial research into stalking via the FindMy protocol and overall crowdsourced network protocol deficiencies have been described in multiple bodies of work, such as:

* [GMCKV21]

* [Heinrich]

* [WhoTracks]

* [BlindMy]

* [Beck]

and others.

There are some suggested improvements, such as the security properties described in [GMCKV21] above. The authors of [GMCKV21] also suggested fusing a private key into the ACC to make it more difficult to spoof, and requiring that location updates be signed.

[Heinrich] and [WhoTracks] pointed out early deficiencies in the protocol, which [BlindMy] set out to solve. By introducing a Blind Signature scheme, the authors sought to overcome an attacker leveraging a large amount of keys to avoid triggering the anti-tracking framework. In this implementation, keys were predetermined for a set interval, and signed by the server, such that a specific, presigned key can only be used during a pre-determined interval. The drawback of this approach is that the authentication is left to the OD and the CN, and the CN does not do any authentication with the FD, so it still could store forged location reports. Additionally, the FD does not do any authentication with the ACC, which means that it could potentially interact with counterfeit ACC devices.

[Beck] introduces the idea of Multi-Dealer Secret Sharing (MDSS) as a privacy preserving protocol that should also be considered.

3. Conventions and Definitions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

Section 1.2 of [I-D.detecting-unwanted-location-trackers] provides definitions of the various system components.

Accessory (ACC): This is the device which will be tracked. It is assumed to lack direct internet access and GPS, but will possess Bluetooth Low Energy capabilities, which it uses to send advertisement messages. The accessory protocol is defined in [DultDoc3].

Advertisement: This is the message that is sent over the BLE Protocol from the Accessory

Crowdsourced Network (CN): This is the network that provides the location reporting upload and download services for Owner Devices and Finder Devices.

Finder Device (FD): This is a device that is a non-owner device that contributes information about an accessory to the crowdsourced network.

Owner Device (OD): This is the device which owns the accessory, and to which it is paired. There can be multiple owner devices, however, the security of that implementation is outside of the scope of this document.

4. Protocol Overview

Figure 1 provides an overall view of the protocol.

In this protocol, the Accessory communicates to Finder Devices or FDs(such as phones) solely via Bluetooth, and the FDs communicate to a centralized service on the Crowdsourced Network CN. Only during the setup phase is the Owner Device OD able to act as a relay between the Accessory ACC and the Crowdsourced Network CN. In this implementation, the CN is able to act as a verifier and signer by leveraging Blind Signatures, which allows the OD to obtain a signature from the signer CN without revealing the input to the CN.

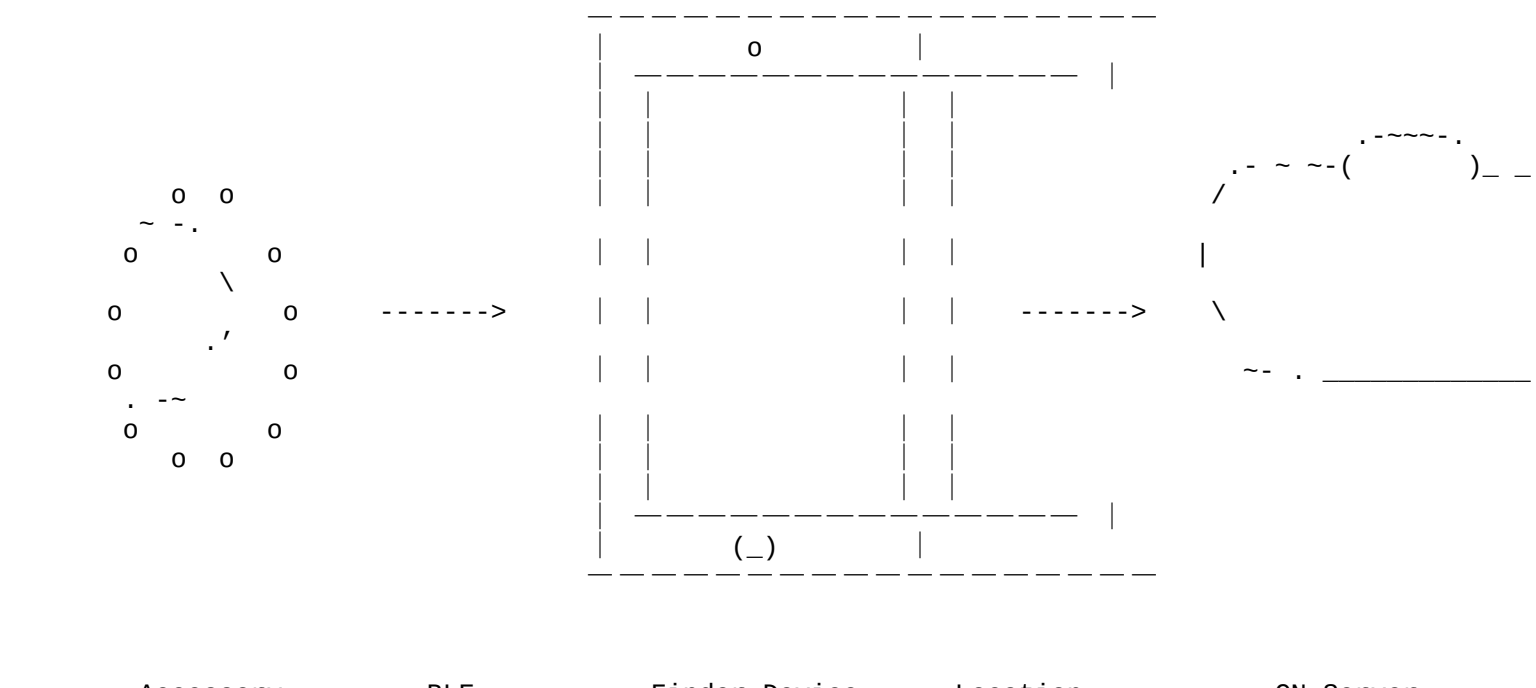

Figure 1: Protocol Overview

As part of the setup phase (Section 5.3) the accessory and owning device are paired, establishing a shared key SK which is known to both the accessory and the owning device. The rest of the protocol proceeds as follows.

* The accessory periodically sends out an advertisement which contains an ephemeral public key Y_i where i is the epoch the key is valid for (e.g., a one hour window). Y_i and its corresponding private key X_i are generated in a deterministic fashion from SK and the epoch i (conceptually as a X_i = PRF(SK, i)).

* In order to report an accessory's location at time i a non-owning device FD encrypts it under Y_i and transmits the pair (E(Y_i, location), Y_i) to the central service CN.

* In order to locate an accessory at time i, the owner uses SK to compute (X_i, Y_i) and then sends Y_i to the central service. The central service responds with all the reports it has for Y_i, and the owner decrypts them with X_i.

This design provides substantially improved privacy properties over a naive design:

1. Nobody but the owner can learn the reported location of an accessory because it is encrypted under Y_i. This includes the central service, which just sees encrypted reports.

2. It is not possible to correlate the public keys broadcast across multiple epochs without knowing the shared key SK, which is only know to the owner. However, an observer who sees multiple beacons within the same epoch can correlate them, as they will have the same Y_i. However, fast key rotation also makes it more difficult to detect unwanted tracking, which relies on multiple observations of the same identifier over time.

However, there are a number of residual privacy threats, as described below.

4.1. Reporting Device Leakage

If the central server is able to learn the identity of the device reporting an accessory or the identity of the owner requesting the location of an accessory, then it can infer information about that accessory's behavior. For instance:

* If device A reports accessories X and Y owned by different users and they both query for their devices, then the central server may learn that those users were in the same place, or at least their accessories were.

* If devices A and B both report tag X, then the server learns that A and B were in the same place.

* If the central server is able to learn where a reporting device is (e.g., by IP address) and then the user queries for that accessory, then the server can infer information about where the user was, or at least where they lost the accessory.

These issues can be mitigated by concealing the identity and/or IP address of network elements communicating with the central server using techniques such as Oblivious HTTP [RFC9458] or MASQUE [RFC9298].

4.2. Non-compliant Accessories

The detection mechanisms described in [I-D.detecting-unwanted-location-trackers] depend on correct behavior from the tracker. For instance, Section 3.5.1 of [I-D.detecting-unwanted-location-trackers] requires that accessories use a rotation period of 24 hours when in the "separated" state:

When in a separated state, the accessory SHALL rotate its address every 24 hours. This duration allows a platform’s unwanted tracking algorithms to detect that the same accessory is in proximity for some period of time, when the owner is not in physical proximity.

However, if an attacker were to make their own accessory that was generated the right beacon messages or modify an existing one, they could cause it to rotate the MAC address and public key Y_i more frequently, thus evading detection algorithms. The following section describes a mechanism which is intended to mitigate this attack.

4.2.1. Rate Limiting and Attestation

Because evading detection requires rapidly changing keys, evasion can be made more difficult by limiting the rate at which keys can change. This rate limiting works as follows:

1. Instead of allowing the accessory to publish an arbitrary key Y_i it instead must pre-generate a set of keys, one for each time window.

2. During the setup/pairing phase, the accessory and owning device interact with the central service, which signs each temporal key using a blind signature scheme. The owning device stores the signatures for each key Y_i.

3. When it wishes to retrieve the location for a given accessory the owning device provides the central service with the corresponding signature, thus proving that it is retrieving location for a pre-registered key; the central service will refuse to provide results for unsigned keys.

Note that this mechanism _does not_ prevent the accessory from broadcasting arbitrary keys, but it cannot retrieve location reports corresponding to those keys.

This is not a complete defense: it limits an attacker who owns a single accessory to a small number of keys per time window, but an attacker who purchases N devices can then use N times that many keys per window, potentially coordinating usage across spatially separated devices to reduce the per-device cost. [[OPEN ISSUE: Can we do better than this?]]

5. Protocol Definition

This section provides a detailed description of the DULT Finding Protocol.

5.1. System Stages

The there are 5 stages that will be outlined, taking into account elements from both [BlindMy] and [GMCKV21]. These stages are as follows:

1) *Initial Pairing / Accessory Setup*

In this phase, the Accessory ACC is paired with the Owner Device OD, and verified as authentic with the Crowdsourced Network CN

2) *Accessory in Nearby Owner Mode*

In this phase, the Accessory ACC is within Bluetooth range of the Owner Device OD. In this phase, Finder Devices FDs SHALL NOT generate location reports to send to the Crowdsourced Network CN. The Accessory SHALL behave as defined in [DultDoc3]. [[OPEN ISSUE: Need to make sure that walking around with an AirTag in Nearby Mode does not allow for stalking]]

3) *Accessory in Separated (Lost) Mode*

In this phase, the Accessory ACC is not within Bluetooth raange fo the Owner Device OD, therefore, the accessory must generate "lost" messages to be received by Finder Devices FD, as described in [DultDoc3].

4) *Finder Device creates a location report*

Finder Device FD receives a Bluetooth packet, and uploads a location report to the Crowdsourced Network CN if and only if it is confirmed to be a valid location report.

[[OPEN ISSUE: Should this be confirmed by the FD, or the CN? or Both?]]

[[OPEN ISSUE: Should there be auth between FD and ACC as well as FD and CN]]

5) *Owner Device queries the Crowdsourced Network*

Owner Device OD queries the Crowdsourced Network CN for the encrypted location report.

5.2. Partial Blind Signature Scheme

[[OPEN ISSUE: Which blind signature scheme to use.]]

In order to verify the parties involved in the protocol, we rely on a partially blind signature scheme. [RFC9474] describes a blind signature scheme as follows:

The RSA Blind Signature Protocol is a two-party protocol between a client and server where they interact to compute sig = Sign(sk, input_msg), where input_msg = Prepare(msg) is a prepared version of the private message msg provided by the client, and sk is the private signing key provided by the server. See Section 6.2 for details on how sk is generated and used in this protocol. Upon completion of this protocol, the server learns nothing, whereas the client learns sig. In particular, this means the server learns nothing of msg or input_msg and the client learns nothing of sk.

The Finding Protocol uses a partially blind signature scheme in which the signature also covers an additional info value which is not kept secret from the signing server.

5.3. Initial Pairing / Accessory Setup

During the pairing process, the Accessory ACC pairs with the Owner Device OD over Bluetooth. In this process, the ACC and OD must generate cryptographically secure keys that will allow for the OD to decrypt the ACC location reports.

5.3.1. Authenticity Verification

Upon the initial pairing of the the ACC and OD, before the key generation process, the OD must facilitate communication with the CN to verify the authenticity of the ACC.

The precise details of this communication are implementation-dependent, but at the end of this process the CN must be able to verify that:

1. The ACC is a legitimate (i.e., authorized) device.

2. The ACC has not already been registered.

For instance, each ACC might be provisioned with a unique serial number which is digitally signed by the manufacturer, thus allowing the CN to verify legitimacy. The CN could use a database of registered serial numbers to prevent multiple registrations. Once registration is complete, there must be some mechanism for the OD to maintain continuity of authentication; this too is implementation specific.

5.3.2. Key Generation and Signing with Partial Blind Signatures

The ACC must periodically be provisioned with new temporal keys which FDs can then use to encrypt reports. Each temporal key is associated with a given timestamp value,

Once the ACC has been authorized, the ACC (or OD on its behalf) needs to generate its temporal encryption keys Y_i. It then generates a signing request for the blinded version of each key.

contains two values:

blindedKey An opaque string representing the key to be signed, computed as below.

timestamp The time value for the first time when the key will be used in seconds since the UNIX epoch

blindedKey = Blind(pk, Y_i, info)

With the following inputs:

pk The public key for CN

Y_i The temporal key to be signed

info The timestamp value serialized as an unsigned 64-bit integer in network byte order.

Prior to signing the key, the CN must ensure the acceptability of the timestamp. While the details are implementation dependent, this generally involves enforcing rate limits on how many keys can be signed with timestamps within a given window. Once the CN is satisfied with the submission it constructs a blind signature as shown below and returns it to the OD.

[[OPEN ISSUE: Is it safe for ACC to hold all of the precomputed keys? Or does this create a privacy issue?]]

BlindSign(sk, blindedKey, info)

With the following inputs

sk The secret key for CN

blindedKey The raw bytes of the blinded key provided by CN

Upon receiving the signed blinded key, the OD unblinds the signature and stores it. If the OD generated Y_i, it must also transfer it to the ACC. Note that ACC does not need a copy of the signature.

5.3.3. Accessory in Nearby Owner Mode

After pairing, when the Accessory ACC is in Bluetooth range of OD, it will follow the protocol as decribed in [DultDoc3].

5.3.4. Accessory in Separated (Lost) Mode

After pairing, when the Accessory ACC no longer in the Bluetooth range of OD, it will follow the protocol as decribed below:, which should correspond to the behavior outlined in [DultDoc3]:

ACC periodically sends out an Advertisement which contains the then current ephemeral public key Y_i. The full payload format of the Advertisement is defined in [DultDoc3].

5.4. Finder Device creates a Location Report

The Finder Device FD receives the advertisement via Bluetooth. FD should have a mechanism by which to authenticate that this is a valid public key with CN. *

In order to report an accessory's location at time i, FD extracts the elliptic curve public key from the advertisement, and records it own location data, a timestamp, and a confidence value as described in [Heinrich].

FD performs ECDH with the public key Y_i and uses it to encrypt the location data using HPKE Seal [RFC9180]. It sends the result to the CN along with the hash of the current public key and the current time. [[OPEN ISSUE: Should we work in terms of hashes or the public keys. What we send has to be what's looked up.]]. CN stores the resulting values indexed under the hash of the public key.

5.5. Owner Device queries the Crowdsourced Network

ODs can retrieve the location of a paired ACC by querying the CN.

In order to query for a given time period i it presents:

* The public key Y_i [or hash of the public key]

* The CN's signature over Y_i as well as the associated info value.

The CN then proceeds as follows:

1. Verify the signature over the key [hash]

2. Verify that the timestamp in the info value is within an acceptable period of time (e.g., one week) from the current time [[OPEN ISSUE: Why do we need this step?]]

3. Retrieve all reports matching the provided Y_i

4. Remove all reports which have timestamps that are not within the acceptable time use window for the key, as indicated by the key's timestamp.

5. Return the remaining reports to OD.

Finally, OD uses HPKE Open to decrypt the resulting reports, thus recovering the location data for report.

6. Security Considerations

TODO Security - as described in [DultDoc4]?. This section still mostly needs to be written.

6.1. Effectiveness of Rate Limiting via Blind Signatures

The blind signature mechanism described here (adapted from [BlindMy]) helps to limit the damage of noncompliant devices.

Because the CN will only generate signatures when the request is associated with a valid device, an attacker cannot obtain a key directly for a noncompliant device. However, this does not necessarily mean that the attacker cannot provision noncompliant devices. Specifically, if the OD sees the public keys (it need not know the private keys, as described below) when they are sent to the CN for signature, then it can provision them to a noncompliant device.

Even an attacker who can provision invalid devices can only obtain one key per time window per valid device. Because key use windows overlap, it is possible to rotate keys more frequently than the window, but in order to rotate keys significantly more frequently than this, the attacker must purchase multiple devices. However, they may be able to provision the keys from multiple valid devices onto the same device, thus achieving a rotation rate increase at linear cost.

Note that enforcement of this rate limit happens only on the CN: the FD does not check. An attacker can generate advertisements with unsigned keys -- and thus at any rotation rate it chooses -- and the FD will duly send valid reports encrypted under those keys. The CN will store them but because the attacker will not be able to produce valid signatures, they will not be able to retrieve those reports.

As noted above, the ACC does not need to prove that it knows the corresponding private keys for a given public key. The ACC simply broadcasts the public keys; it is the OD which needs to know the private keys in order to decrypt the reports.

7. Privacy Considerations

TODO Privacy - as described in [DultDoc4]?

8. IANA Considerations

This document has no IANA actions.

9. References

9.1. Normative References

[I-D.detecting-unwanted-location-trackers]
Ledvina, B., Eddinger, Z., Detwiler, B., and S. P. Polatkan, "Detecting Unwanted Location Trackers", Work in Progress, Internet-Draft, draft-detecting-unwanted-location-trackers-01, 20 December 2023, <https://datatracker.ietf.org/doc/html/draft-detecting-unwanted-location-trackers-01>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/rfc/rfc2119>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/rfc/rfc8174>.

[RFC9180] Barnes, R., Bhargavan, K., Lipp, B., and C. Wood, "Hybrid Public Key Encryption", RFC 9180, DOI 10.17487/RFC9180, February 2022, <https://www.rfc-editor.org/rfc/rfc9180>.

9.2. Informative References

[Beck] Gabrielle Beck, Harry Eldridge, Matthew Green, Nadia Heninger, and Abishek Jain, "Abuse-Resistant Location Tracking: Balancing Privacy and Safety in the Offline Finding Ecosystem", 2023, <https://eprint.iacr.org/2023/1332.pdf>.

[BlindMy] Travis Mayberry, Erik-Oliver Blass, and Ellis Fenske, "Blind My — An Improved Cryptographic Protocol to Prevent Stalking in Apple’s Find My Network", 2023, <https://petsymposium.org/popets/2023/popets-2023-0006.pdf>.

[DultDoc3] Brent Ledvina, Lazarov, D., Detwiler, B., and S. P. Polatkan, "Detecting Unwanted Location Trackers Accessory Protocol", 2024, <https://www.ietf.org/archive/id/draft-ledvina-dult-accessory-protocol-00.html>.

[DultDoc4] Maggie Delano and Jessie Lowell, "DRAFT Dult Threat Model", 2024, <https://datatracker.ietf.org/doc/html/draft-delano-dult-threat-model>.

[GMCKV21] Chinmay Garg, Aravind Machiry, Andrea Continella, Christopher Kruegel, and Giovanni Vigna, "Toward a secure crowdsourced location tracking system", 2021, <https://dl.acm.org/doi/10.1145/3448300.3467821>.

[Heinrich] Alexander Heinrich, Milan Stute, Tim Kornhuber, and Matthias Hollick, "Who Can Find My Devices? Security and Privacy of Apple's Crowd-Sourced Bluetooth Location Tracking System", 2021, <https://petsymposium.org/popets/2021/popets-2021-0045.pdf>.

[RFC9298] Schinazi, D., "Proxying UDP in HTTP", RFC 9298, DOI 10.17487/RFC9298, August 2022, <https://www.rfc-editor.org/rfc/rfc9298>.

[RFC9458] Thomson, M. and C. A. Wood, "Oblivious HTTP", RFC 9458, DOI 10.17487/RFC9458, January 2024, <https://www.rfc-editor.org/rfc/rfc9458>.

[RFC9474] Denis, F., Jacobs, F., and C. A. Wood, "RSA Blind Signatures", RFC 9474, DOI 10.17487/RFC9474, October 2023, <https://www.rfc-editor.org/rfc/rfc9474>.

[Samsung] Tingfeng Yu, James Henderson, Alwen Tiu, and Thomas Haines, "Privacy Analysis of Samsung' s Crowd-Sourced Bluetooth Location Tracking System", 2023, <https://www.usenix.org/system/files/sec23winter-prepub-498-yu.pdf>.

[WhoTracks] Travis Mayberry, Ellis Fenske, Dane Brown, Christine Fossaceca, Sam Teplov, Lucas Foppe, Jeremey Martin, and Erik Rye, "Who Tracks the Trackers?", 2021, <https://dl.acm.org/doi/10.1145/3463676.3485616>.

Acknowledgments

TODO acknowledge.

Authors' Addresses

Christine Fossaceca
Microsoft
Email: cfossaceca@microsoft.com

Eric Rescorla
Independent
Email: ekr@rtfm.com