Abuse-Resistant Location Tracking: Balancing Privacy and Safety in the Offline Finding Ecosystem

Harry Eldridge, Gabrielle Beck, and Matthew Green, *Johns Hopkins University;*
Nadia Heninger, *University of California, San Diego;*
Abhishek Jain, *Johns Hopkins University*

https://www.usenix.org/conference/usenixsecurity24/presentation/eldridge

This paper is included in the Proceedings of the
33rd USENIX Security Symposium.

August 14–16, 2024 • Philadelphia, PA, USA

978-1-939133-44-1

Open access to the Proceedings of the
33rd USENIX Security Symposium
is sponsored by USENIX.

Abuse-Resistant Location Tracking: Balancing Privacy and Safety in the Offline Finding Ecosystem

Harry Eldridge[*] *Gabrielle Beck*[*] *Matthew Green*[*] *Nadia Heninger*[†] *Abhishek Jain*[*]

Abstract

Location tracking accessories (or "tracking tags") such as those sold by Apple, Samsung, and Tile, allow owners to track the location of their property via offline finding networks. The tracking protocols were designed to ensure that no entity (including the vendor) can use a tag's broadcasts to surveil its owner. These privacy guarantees, however, seem to be at odds with the phenomenon of *tracker-based stalking*, where attackers use these very tags to monitor a target's movements. Numerous such criminal incidents have been reported, and in response, manufacturers have chosen to substantially weaken privacy guarantees in order to allow users to detect stalker tags. This compromise has been adopted in a recent IETF draft jointly proposed by Apple and Google.

We put forth the notion of *abuse-resistant offline finding protocols* that aim to achieve a better balance between user privacy and stalker detection. We present an efficient protocol that achieves stalker detection under realistic conditions without sacrificing honest user privacy. At the heart of our result, and of independent interest, is a new notion of *multi-dealer secret sharing* which strengthens standard secret sharing with novel privacy and correctness guarantees. We show that this primitive can be instantiated efficiently on edge devices using variants of Interleaved Reed-Solomon codes combined with new lattice-based decoding algorithms.

1 Introduction

Vendors such as Apple, Samsung, and Tile have recently begun to deploy large-scale *offline finding networks* to monitor network-disconnected devices. These systems employ short-distance communications networks such as Bluetooth Low Energy (BLE) or Ultra-Wideband (UWB) to transmit periodic advertisement messages to nearby receivers (such as smartphones). The receiving devices upload location reports to servers controlled by the service provider.

[*]Johns Hopkins University, {hme,becgabri,mgreen,abhishek}@cs.jhu.edu

[†]University of California San Diego, nadiah@cs.ucsd.edu

While offline finding networks can be used to locate relatively powerful devices like phones and laptops, the breakout product in this category is the *location tracking accessory* (LTA), colloquially known as a "tracking tag." Exemplified by Apple's AirTag and Tile Trackers, these tags embed a transceiver, microprocessor, and battery in a compact package that can be attached to physical objects. The popularity of tracking tags stems from their low cost (typically under $30 USD) as well as the availability of large volunteer-operated tracking networks that can detect them. As of this writing, the combined tag sales of Tile, Apple, and Samsung exceed 100 million units [33].

Privacy and stalking risks. The widespread deployment of offline finding networks exposes users to new privacy and safety risks. On the one hand, LTAs can undermine the privacy of individuals who carry them: if an LTA emits an unchanging identifier such as a static MAC address, then a *tracking adversary* such as the network operator or a third-party RF tracking network [38, 40] can easily monitor individuals' physical movements. In response to these privacy concerns, manufacturers deploy sophisticated countermeasures: for example, Apple's Find My system employs a cryptographic protocol that rotates pseudonymous identifiers and uses encrypted location reports to hide location information from third parties and from Apple itself [2].

On the other hand, the availability of inexpensive LTAs also enables *tracker-based stalking* [24], in which attackers surreptitiously place a tag on a targeted person or vehicle and then monitor the target's movements via the offline finding network. These devices have been used in hundreds of criminal incidents, including serious cases that culminated in physical assault and murder [5, 11, 19, 28, 42].

Privacy vs stalker detection. Apple, Tile, Samsung and Google have adopted stalker-detection countermeasures to alert users to the presence of an unrecognized tag that "moves with" a user for a pre-defined length of time [3, 13, 26, 27, 39]. Victims can typically trigger an audio alert from the tag, obtain the tag's serial number using Near-Field Communication

Protocol	Epoch duration	Broadcasts per epoch	Stalker detection?	Tracking privacy	Continuous Proximity	Stalker detection
Apple FindMy [2] / IETF [27]:						
Near-owner mode	15 min	450	✗	n/a	n/a	
Separated mode	24 hrs	43,200	●	n/a	✗	15-60 min†
This work (§4):						
4-second epochs / 1-hour window	4 sec	1	●	39 – 46 min*	●	60 min
1-minute epochs / 1-hour window	60 sec	15	●	41 – 47 min*	●	60 min

Figure 1: We compare the detection and privacy guarantees of our scheme with various parameters to existing tag protocols. Our goal in this work is to design a scheme that maximizes tracking privacy within the stalker detection window. "Tracking privacy" indicates the duration that a tracking adversary may receive broadcasts without de-anonymizing an LTA. "Stalking detection" indicates the minimum length of time that broadcasts must be collected before a stalking LTA can be detected. Epoch duration indicates the time between changes to an LTA identifier. Broadcasts/epoch indicates the number of *repeated* broadcasts of the same identifier that an LTA will issue in this time. "Continuous Proximity" means that a tracking adversary must be in the presence of an LTA for the entire unlinkability duration in order to de-anonymize it. *The former number uses current BLE payload sizes, while the second assumes BLE v5. †Apple does not publish their methodology, so this is an estimate.

(NFC), and then query the provider's servers to obtain partial account information.

In order to implement such countermeasures, manufacturers have adopted various compromises: for example, Apple AirTags rotate their identifier every 15 minutes when within range of their owner devices, but reduce this rate to once every 24 hours when out of range. This approach allows potential stalking victims to detect nearby AirTags, but at the cost of reducing privacy against tracking adversaries.[1] Apple and Google have jointly proposed an IETF draft [27] to standardize this approach.

As this solution illustrates, the goal of a stalker detection mechanism appears to be in direct conflict with the goal of preserving privacy against a tracking adversary. To defend against a tracking adversary, tags must routinely change identifiers: this ensures that a listener cannot link a series of broadcasts to a single emitting device. Yet, to detect stalkers, a potential victim must be able to determine that a series of broadcasts belongs to a single device. This raises the following question:

> *Is it possible to provide strong privacy protections against location tracking, while also enabling the detection of stalker abuse?*

Contributions. In this work, we answer the question in the affirmative, designing new protocols that offer strong privacy guarantees while ensuring that stalker tags can be reliably detected. Critically, our solutions *operate in the threat model of today's systems* and do not rely on the creation of new trusted parties or the placement of additional trust in the service provider itself.

1. **Abuse-resistant offline finding protocols:** We put forth the notion of *abuse-resistant offline finding protocols* that simultaneously achieve privacy against tracking adversaries and stalker detection by victim devices. We propose an efficient instantiation that can operate under the constraints of existing systems.
2. **Multi-dealer secret sharing.** To obtain the above result, we define and construct a new cryptographic notion: *multi-dealer secret sharing* (MDSS). MDSS extends standard secret sharing to admit multiple dealers with different secrets while achieving new properties of unlinkability and multi-dealer correctness. We provide two constructions of MDSS, including a practical solution (with heuristic correctness) where the decoder can operate on consumer-grade devices like smartphones based on lattice-based list decoding techniques [14]. Finally, we propose some optimizations, including a new notion of *collision-aware PRFs* that enables a small-state streaming algorithm for MDSS dealers.
3. **Implementation and Evaluation.** We explore parameters for our constructions and demonstrate that existing offline finding networks can be made substantially more private using our protocols. Concretely, we propose parameters that allow users to remain private for up to 40 minutes while still detecting stalkers within an hour (see Figure 1). Finally, we implement our algorithms and demonstrate that they are efficient in practice when running on modest hardware. When using our most efficient parameters, our proposed stalker detection algorithm runs in approximately two seconds on a Raspberry Pi 3 B, and a tenth of a second on a modern laptop.

Ethics and human-subjects research. In the course of this work we conducted experiments to determine the density of Apple LTA (AirTag) broadcasts in several public areas. The data we collected in these efforts includes public BLE advertisement data combined with the approximate time and GPS location noted by our detector. Advertisement data includes a rotating MAC address. We did not engage in any effort to link addresses to non-ephemeral device properties, or through rotations, and the results we present in this work

[1] In practice, our experience shows that many users routinely carry AirTags in "separated mode". This is due at least in part to the fact that users often share physical property with other individuals, and AirTags cannot easily be paired with multiple owner devices.

contain only aggregate statistics about tag broadcast rates. To eliminate the chance of future address linking, all addresses were anonymized[2], and all data was stored on an encrypted volume to be deleted subject to data retention policies.

Prior to conducting these experiments we presented our experiments to an IRB and received a determination that they do not constitute human subjects research.

1.1 Technical Overview

The core challenge we address in this work is that *the goal of a stalking victim is nearly identical to that of the tracking adversary*. As such, it seems intrinsically difficult to develop systems that allow the former to succeed without providing an advantage to the latter.

Separating stalking victims from tracking adversaries. Our work starts with the following insight: while tracking adversaries must solve a problem that is *similar* to that of a stalking victim, the two parties are *not* identical. Stalking victims are (by definition) guaranteed to be in close proximity to a stalking LTA for a relatively long period. In contrast, a tracking adversary may only have brief or intermittent access to a given tag's broadcasts (*e.g.,* as owners enter and leave the physical locations of tracking receivers). Figure 2 illustrates an example of the two different scenarios.

Our goal is to design a scheme that provides strong cryptographic privacy against a tracking adversary that sees a large subset of an LTA's broadcasts, and yet allows full linking (and thus stalker detection) when the receiver obtains a more complete series of broadcasts. Simultaneously, our scheme must handle broadcasts sent by unrelated tags in the vicinity, that is, be robust to substantial "noise." We refer to this goal as *abuse-resistant offline finding* (AROF).

Abuse-Resistant Offline Finding Protocols. Rather than broadcast a constant identifier, we propose to transmit frequently changing and unlinkable broadcast identifiers as a means to maximize privacy, where the rate of identifier turnover is a system configuration parameter. After a (potential) victim device has collected a list of broadcasts for some fixed amount of time, it can then run a detection algorithm to check if any stalkers are nearby and, if so, it can identify and *link* those broadcasts that come from the stalker. However, to a tracking adversary who has only seen some small number of broadcasts, not only will the detection algorithm fail to recover the user's tag, but it will also be unable to link any of the tag's broadcasts to one another. In fact, the distribution will be cryptographically indistinguishable from a series of broadcasts sent by many different tags.

A key desideratum in AROF is that detection should imply that an LTA has been in *continuous proximity* to a given target, not merely that they have seen a few related broadcasts in distant broadcast periods. More formally, we wish to ensure that a tracking adversary who receives an ε-fraction of broadcasts (for some choice of ε) during a detection window will have no advantage in linking those broadcasts, while a victim who receives only a slightly greater fraction can efficiently link and detect the stalker. We call the first property *tag indistinguishability* and the second *tag detectability*. The first property ensures that honest tag users are protected from a tracking adversary while the second ensures that stalkers are detected.

A Flawed Attempt. A simple but flawed approach to realizing AROF is to produce a cyclically repeating sequence of distinct broadcast identifiers. Such a construction provides privacy against tracking adversaries who receive broadcasts for a fixed duration, but is very fragile: a tracking adversary who receives only *two* broadcasts (*e.g.,* one broadcast from each cycle) would immediately link these broadcasts to the same LTA.

Our Approach: Multi-Dealer Secret Sharing. To obtain a better solution, we instead use secret sharing [41]. Recall that in classical t-out-of-n secret sharing, a dealer distributes a secret into n shares such that the original secret can be recovered from any subset of shares of size at least t. The privacy property of secret sharing guarantees that any subset of less than t shares (referred to as an *unauthorized* set) "hides" the secret.

Consider the following proposal for an AROF protocol: at periodic intervals, the LTA samples a fresh secret tag identifier. Then, at each epoch within this period, it outputs a pseudonym similar to the one used in Apple's scheme, *along with a secret share of the secret tag identifier.* By changing the pseudonym and using a different secret share at each epoch (and, critically, having volunteer devices keep these shares local without forwarding them to the service provider), we can hope to preserve the privacy of the tag against tracking adversaries who receive an unauthorized subset of the tag's emitted secret shares within the current period. At the same time, a stalking victim who receives a more complete set of shares can use the secret sharing reconstruction algorithm to recover the tag's identifier and communicate with the tag. The exact parameters to determine both privacy and the stalker-detection threshold can be determined by the chosen parameters of the secret sharing scheme, as well as estimates of likely broadcast traffic.

Unfortunately, this proposal runs into several challenges. Note that the broadcast traffic at any time may consist of broadcasts from *multiple* LTAs in the vicinity of each other. For example, the broadcasts emitted by a stalking LTA might be intermingled with broadcasts from both ephemeral LTAs that a victim is only briefly in range of, as well as additional LTAs that remain persistently near the victim. In this setting, the standard privacy and correctness properties of secret sharing no longer suffice; instead, new properties are necessary in order to realize the above proposal:

[2]MAC addresses were replaced by the HMAC-SHA256 of the address with a randomly sampled 32 byte key. The key was then deleted.

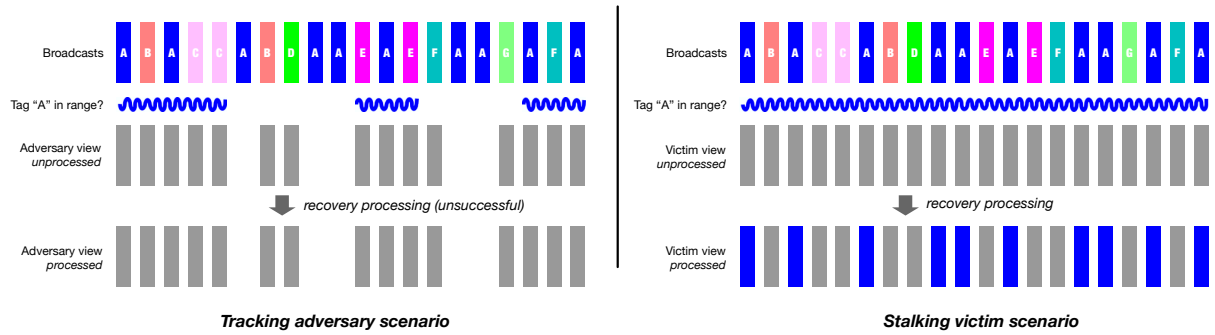

Figure 2: Illustration of two passive detection scenarios. **Left:** a static *tracking adversary* sees many broadcasts from different LTAs, including LTA "A" (blue). However, because "A" is mobile, the adversary is not in continuous range to receive its broadcasts and does not receive enough to enable tracking of the LTA. "A" broadcasts cannot be distinguished from any other broadcasts. **Right:** a *stalking victim* receives broadcasts from many LTAs including a *stalking LTA* "A". The victim is continuously within range of the stalking device and receives all of its broadcasts, which is enough to enable stalker detection.

1. **Unlinkability:** In order to achieve privacy against tracking adversaries, we require a new *unlinkability* property for secret sharing. Roughly speaking, this guarantees that given a mixed set of shares of different secrets broadcast by multiple dealers, an adversary cannot "link" any of the shares emitted by a dealer to one another as long as the shares comprise an unauthorized set. Note that this is *stronger* than the standard privacy property, which only guarantees that the adversary cannot recover the secret.
2. **MD-Correctness:** In order to detect stalker tags, we require a robust secret recovery mechanism in the presence of *multiple dealers*. In particular, we require that given a set of shares from multiple dealers, a receiver can reconstruct all secrets for which they have an authorized set (i.e., at least a threshold number) of shares. In the context of AROF, this means that a victim can identify potentially multiple stalker tags even in the vicinity of ephemeral tags.

We refer to a secret sharing scheme that satisfies these two properties as *multi-dealer secret sharing* (MDSS). While achieving either of the above two properties is non-trivial, it is especially challenging to achieve them simultaneously. Indeed, in prior uses of secret sharing (e.g., [6, 16, 18]), the problem of robust reconstruction in the presence of noisy shares is addressed by simply *identifying* (or labeling) all shares transmitted by a given dealer to identify which shares correspond to which set. This, however, is *not* an option in our setting since labeling of shares immediately breaks the unlinkability property.

Constructing MDSS. We present two constructions of MDSS. Our first construction is a variant of Shamir secret sharing where instead of using fixed evaluation points (which breaks unlinkability), we sample polynomial evaluation points uniformly at random from a super-polynomial sized field. In order to achieve MD-correctness, we leverage the connection between Shamir's secret sharing and Reed-Solomon (RS) codes [37] and utilize known list decoding algorithms [21] for RS codes. The main disadvantage of this construction is that the parameter restrictions of [21] cap the number of dealers in a way that renders the scheme impractical for our application to AROF.

To achieve better bounds, we employ a variant that uses multiple Shamir polynomials simultaneously, realizing an "interleaved" Reed-Solomon code (IRS) [8]. To achieve MD-correctness, we develop a novel lattice-based list-decoding algorithm building on the work of Cohn and Heninger [14]. This construction achieves provable unlinkability and heuristic correctness guarantees that we justify empirically. Crucially, this approach unlocks significantly better parameters that make efficient decoding achievable on highly-restricted devices. We remark that while efficient decoding algorithms for IRS codes exist in the literature [8, 15], they are defined for the random noise model which does not capture our multi-dealer setting.

Organization. We begin by formalizing the notion of abuse-resistant offline finding protocols in §2. Next, in §3, we give definitions and constructions for our main building block: multi-dealer secret sharing (MDSS). In §4, we show how to use MDSS to construct AROF schemes. In §5 we discuss how to tune MDSS parameters to optimize AROF deployments. In §6 we describe our implementation of MDSS schemes, and in §7, we provide benchmarks of our implementation using our suggested parameters. We discuss some limitations of our approach in §8, related work in §9, and conclude in §10.

2 Abuse-Resistant Offline Finding

An offline finding network, illustrated in Figure 3, is a crowd-sourced system designed to locate lost or stolen devices. In these systems, users purchase *location-tracking accessories* (LTAs) that run a tracking protocol with the network and service provider. Deployed networks typically work as follows: to enroll the device, the user pairs the LTA to a client device

(such as a smartphone or computer) and optionally registers the LTA with a service provider (SP) that controls the network. LTAs typically operate in two modes: in *near-owner mode* the LTA is in range of the owner's device and communicates directly with it. The LTA switches to *separated mode* when it is out of range of the owner device. In this work we focus primarily on the behavior of devices in separated mode, which is the typical setting for stalking attacks.

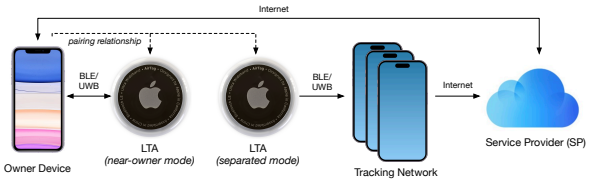

Figure 3: Components of a location-tracking network.

In separated mode, the LTA periodically emits RF-based broadcasts that can be detected by volunteer devices in the offline finding network. These volunteer devices construct *location reports* that combine the LTA broadcast data with the encrypted GPS coordinates of the volunteer device, then upload these reports to the service provider's servers. An owner device with the necessary identifiers (and other credentials) can query the service provider to obtain past and present location reports for a given LTA.

Abuse-Resistant Offline Finding Protocols. The devices in an offline finding network jointly conduct an offline finding protocol. We present algorithmic definitions and security notions for an *abuse-resistant offline finding protocol* that achieves privacy (formalized via a notion of tag indistinguishability) as well as detectability for malicious tags (formalized via a notion of tag detectability). Throughout this section, we use notation inspired by the work of Mayberry *et al.* [29].

While stalking is inherently a malicious activity, the LTA devices used in these attacks are typically *honest*, in the sense that they correctly execute the tracking network protocol. This is due to the fact that many attackers use legitimate (non-counterfeit) devices produced by the original manufacturer. In this work we will focus on the case where all LTAs honestly execute the protocol, and consider extensions to address counterfeit/malicious LTA devices, as well as other concerns, in the full version of the paper.

Definition 2.1. An *abuse-resistant offline-finding protocol* (AROF) is a tuple of algorithms $(\mathsf{KeyGen}, \mathsf{Beacon}, \mathsf{GetTagID}, \mathsf{GenReport}, \mathsf{Detect})$, a protocol $\mathsf{RetrieveReports}$ as specified in Figure 4, and two predicates P_s and P_h.

To use an offline-finding protocol, each LTA executes the KeyGen algorithm with a set of deployment-specific parameters, and shares the resulting key with an owner device. The LTA then initializes an *anonymity epoch* counter that increases monotonically whenever the identifier is to be rotated (we refer to this period as the *epoch duration*.) At the start of each epoch, the LTA executes the Beacon algorithm on the current epoch counter and transmits the output one or more times.[3] Periodically, after a *detectability period* of some number of anonymity epochs, the LTA may re-key so that very old beacons do not contribute to the tag being detectable. Volunteer devices collect the resulting broadcasts and use the $\mathsf{GenReport}$ algorithm to generate location reports for the service provider. An owner device executes the $\mathsf{RetrieveReports}$ protocol with the service provider to obtain reports collected by the network. Victim devices collect all received LTA broadcasts within a *detection window* and then use them in the Detect algorithm to detect the presence of nearby stalking LTAs.

Abuse-resistant Offline Finding Protocol

$\mathsf{KeyGen}(1^\lambda, \mathsf{cfg}) \to k_{\mathsf{tag}}$ given a set of implementation-specific scheme parameters cfg, generates a secret key k_{tag}

$\mathsf{Beacon}(k_{\mathsf{tag}}, i_{\mathsf{epoch}}, \mathsf{aux}) \to B$ on input the tag key, the index of the current *anonymity epoch*, and auxiliary data aux (*e.g.,* battery status), generates a broadcast message B

$\mathsf{GetTagID}(k_{\mathsf{tag}}, i_{\mathsf{epoch}}) \to \mathsf{id}_{\mathsf{tag}}$ is a helper algorithm used by the LTA and owner device to find the current identifier for the LTA.

$\mathsf{GenReport}(B, \mathsf{loc}) \to R$ on input a tag broadcast B and time/GPS coordinates loc, outputs a report R

$\mathsf{Detect}(\mathsf{cfg}, \{(B_1, \mathsf{loc}_1), \ldots, (B_n, \mathsf{loc}_n)\}) \to \{\mathsf{id}_{\mathsf{tag}_i}\}$ on input a set of broadcasts, outputs one or more identifiers $\mathsf{id}_{\mathsf{tag}}$

$\mathsf{RetrieveReports}(\mathsf{Owner}(k_{\mathsf{tag}}, i_{\mathsf{epoch}}), \mathsf{SP}(\mathcal{D}))$ a protocol executed between two parties. Owner provides a tag key k_{tag} and an epoch i_{epoch} and SP uses a database $\mathcal{D}$. The output to Owner is potentially a list of reports and SP's output is $\perp$.

Figure 4: Algorithms for abuse-resistant offline finding protocol.

An abuse-resistant tracking scheme must satisfy correctness and security properties. We first give informal descriptions of the desired properties, followed by the formal definitions. For the purposes of our definitions, we will use the predicates P_s and P_h to describe specific access patterns that determine whether or not a tag should be *detectable* as a potential stalker, or should remain unlinkable, respectively.

Correctness. An authorized Owner should be able to obtain location information on their LTA, provided a volunteer device sees at least one of its emitted broadcasts.

Detectability. A stalking LTA should be detectable by a victim. Even when selected adversarially, any set of received broadcasts satisfying P_s should both reveal the LTA as a stalker, and result in the victim device accurately learning the stalker's unique identifier.

[3] If the broadcast interval is longer than the epoch duration, then the LTA may transmit the same data multiple times.

Tag Indistinguishability. A tracking adversary who receives a broadcast pattern that satisfies P_h should *not* be able to distinguish a given LTA's broadcast from those of other LTAs. We model this as an indistinguishability game, where an adversary may see broadcasts from one of two LTAs, and then must determine which LTA a challenge broadcast belongs to. The advesary wins if it can successfully determine the correct LTA while only having seen broadcast patterns that satisfy P_h.

We can now proceed with the formal definitions.

Definition 2.2 (Correctness)**.** *A privacy preserving tracking protocol satisfies correctness if for all authorized owners* Owner *and compliant service providers* SP, $\forall \mathsf{loc}, \mathsf{aux}$ *and allowed anonymity epochs* i_{epoch}*, and* $\forall \mathcal{D}$ *provided by* SP*,*

$$\Pr\left[\begin{array}{l} k_{\mathsf{tag}} \leftarrow \mathsf{KeyGen}(1^\lambda, \mathsf{cfg}); \\ B \leftarrow \mathsf{Beacon}(\mathsf{cfg}, k_{\mathsf{tag}}, i_{\mathsf{epoch}}, \mathsf{aux}); \\ R \leftarrow \mathsf{GenReport}(B, \mathsf{loc}); \\ \mathcal{D}' := \mathcal{D} \cup \{R\}; \\ \mathsf{out} \leftarrow \mathsf{RetrieveReports}(\mathsf{Owner}(k_{\mathsf{tag}}), \mathsf{SP}(\mathcal{D}')) : \\ \exists m \in \mathsf{out}, m = (\mathsf{loc}, \mathsf{aux}) \end{array}\right] = 1$$

Tag Detectability

$\{\{(i_{\ell,j}, \mathsf{aux}_{\ell,j}, \mathsf{loc}_{\ell,j})\}_{j\in[z_\ell]}\}_{\ell\in[L]} \leftarrow \mathcal{A}(1^\lambda)$

$\forall \ell \in [L] : k^\ell_{\mathsf{tag}} \leftarrow \mathsf{KeyGen}(1^\lambda, \mathsf{cfg})$

$\forall \ell \in [L], j \in [z_\ell] : B_{\ell,j} \leftarrow \mathsf{Beacon}(k^\ell_{\mathsf{tag}}, i_{\ell,j}, \mathsf{aux}_{\ell,j})$

$\mathsf{out} \leftarrow \mathsf{Detect}(\{(B_{\ell,j}, \mathsf{loc}_{\ell,j})\})$

$Q \leftarrow \{i_{\ell,j}\}_{\ell\in[L], j\in[z_\ell]}$

$\mathcal{S} \leftarrow \{P_s(\mathsf{cfg}, k^\ell_{\mathsf{tag}}, Q, i_{\ell,j}) = 1 : \mathsf{GetTagID}(k_{\mathsf{tag}}, i_{\ell,j})\}$

Remove duplicates from $\mathcal{S}$

If $\mathcal{S} \subseteq \mathsf{out}$, output 1 else 0

Figure 5: Experiment $\mathsf{Exp}^{\mathsf{Det},P_s}_{\mathcal{A}}(\lambda, \mathsf{cfg})$.

Definition 2.3 (Detectability)**.** *A privacy preserving tracking protocol is detectable if* $\forall$ *valid* cfg *values,* $\forall n.u.p.p.t$ *algorithms* $\mathcal{A}$*,* $\exists$ *a negligible function* $negl(\lambda)$ *so that*

$$\Pr[\mathsf{Exp}^{\mathsf{Det},P_s}_{\mathcal{A}}(\lambda, \mathsf{cfg}) = 0] \leq negl(\lambda),$$

where $\mathsf{Exp}^{\mathsf{Det},P_s}_{\mathcal{A}}(\lambda, \mathsf{cfg})$ *appears in Figure 5.*

Definition 2.4 (Tag Indistinguishability)**.** *A privacy preserving offline finding protocol is tag indistinguishable if* $\forall$ *valid* cfg *values,* $\forall n.u.p.p.t$ *adversaries* $\mathcal{A}$*,* $\exists$ *a negligible function* $negl(\lambda)$ *so that*

$$|\Pr[\mathsf{Exp}^{\mathsf{Tag},P_h,0}_{\mathcal{A}}(\lambda, \mathsf{cfg}) = 1] - \Pr[\mathsf{Exp}^{\mathsf{Tag},P_h,1}_{\mathcal{A}}(\lambda, \mathsf{cfg}) = 1]| \leq negl(\lambda),$$

where $\mathsf{Exp}^{\mathsf{Tag},P_h,b}_{\mathcal{A}}(\lambda, \mathsf{cfg})$ *is given in Figure 6.*

Tag Indistinguishability

$O_{k^0_{\mathsf{tag}}, k^1_{\mathsf{tag}}, Q}(\mathsf{id}, i, \mathsf{aux})$**:** Record (id, i) in table Q.
Return $\mathsf{Beacon}(k^{\mathsf{id}}_{\mathsf{tag}}, i, \mathsf{aux})$

$k^i_{\mathsf{tag}} \leftarrow \mathsf{KeyGen}(1^\lambda, \mathsf{cfg}), \forall i \in \{0,1\}$

$(i^*_0, i^*_1, \mathsf{aux}) \leftarrow \mathcal{A}^{O_{k^0_{\mathsf{tag}}, k^1_{\mathsf{tag}}, Q}(\cdot,\cdot,\cdot)}(1^\lambda)$

$B^* \leftarrow \mathsf{Beacon}(k^b_{\mathsf{tag}}, i^*_b, \mathsf{aux})$

$\hat{b} \leftarrow \mathcal{A}(B^*)$

$Q \leftarrow Q \cup \{(0, i^*_0), (1, i^*_1)\}$

If $P_h(\mathsf{cfg}, Q) = 1$,

output $\hat{b}$, else 0

Figure 6: Experiment $\mathsf{Exp}^{\mathsf{Tag},P_h,b}_{\mathcal{A}}(\lambda, \mathsf{cfg})$.

Other Properties. Mayberry *et al.* [29] identify some additional properties for an offline finding system. For example, the service provider should not be able to determine the location of LTAs or volunteer devices from the encrypted location reports it receives. Our schemes in §4 achieve this property in much the same way as the constructions of [29]. Since the focus of our work is on the interaction between stalking tags and victims (and due to space limitations) we omit this analysis here.

3 Multi-Dealer Secret Sharing

In a secret sharing scheme [41], a dealer divides a secret into shares such that the secret remains hidden from anyone who holds an "unauthorized" set of shares, but can be recovered from any "authorized" set of shares. These properties are referred to as privacy and correctness, respectively.

In this work, we consider a setting where *multiple* dealers may simultaneously distribute shares of their respective secrets. We put forth a notion of *multi-dealer secret sharing* (MDSS) that achieves *stronger* security and correctness properties in this setting.

3.1 Definition

We study multi-dealer secret sharing for threshold access structures. Such a scheme is parameterized by four variables:

- t_{rec}, the number of shares required to recover a secret.
- t_{priv}, the number of shares one can emit before privacy and unlinkability are broken.
- d, the number of sufficient dealers the scheme is able to tolerate.
- max, the maximum number of shares that can be input to the reconstruction algorithm.

We consider the *ramp* setting of Blakley and Meadows [7], where we allow for a gap between t_{rec} and t_{priv} larger than 1. We now formally define multi-dealer secret sharing.

Definition 3.1 (Multi-dealer secret sharing scheme)**.** A $(t_{rec}, t_{priv}, d, \mathsf{max})$-multi-dealer secret sharing scheme (MDSS) is defined over a secret space $\mathcal{S}$ of values that can be shared and consists of the following algorithms, which are expected to run in polynomial time in a security parameter λ:

- $\mathsf{Share}(s, n; r) \to \{sh_i\}_{i\in[n]}$, takes as input a secret $s \in \mathcal{S}$, an integer n, and some randomness r, and outputs a set of n shares $\mathbf{sh} = \{sh_i\}_{i\in[n]}$.
- $\mathsf{Reconstruct}(\{sh_1, \ldots, sh_w\}) \to \{s_1, \ldots, s_m\}$, takes as input a set of shares $\{sh_1, \ldots, sh_w\}$ where $w \le \mathsf{max}$, and outputs a (potentially empty) set of secrets $\{s_1, \ldots, s_m\}$.

In the above, the tuple $(t_{rec}, t_{priv}, d, \mathsf{max})$ is an implicit input to both algorithms.

An MDSS scheme must satisfy two properties: *unlinkability*, which strengthens the standard notion of privacy (§3.1.1), and *MD-Correctness*, which strengthens the standard notion of correctness (§3.1.2).

In both definitions we will use some notational shorthand for computing a set of shares and then projecting them onto a set of indices. For a secret s, an integer n, and a set of indices $I \subseteq [n]$, define $\mathsf{Proj}(s, n, I)$ as follows:

1. Compute $\{sh_1, \ldots, sh_n\} \leftarrow \mathsf{Share}(s, n)$
2. Return $\{sh_i\}_{i\in I}$

When I is a singleton, we will denote it as $\hat{i}$.

3.1.1 Unlinkability

We propose a new security property for secret sharing that is stronger than the standard notion of privacy. Intuitively, this property requires that given a set of shares from multiple dealers, an adversary cannot associate (i.e., link) any unauthorized subset of shares to a common dealer.

We present two definitions: unlinkability and one-more unlinkability. The first intuitively matches the guarantee we would like to achieve, while the second is simpler and easier to use. We will show that the second definition implies the first (the first will obviously imply the second); therefore one can use the second definition without loss of generality.

Unlinkability. Our first definition is characterized by a security game $\mathsf{ULink}^b_{\mathcal{A}}(\lambda)$ where an adversary $\mathcal{A}$ can receive shares from an arbitrary polynomial number of dealers, each sharing a potentially different secret chosen by the adversary. For every dealer i, the adversary receives $|I_{i,b}|$ number of shares of the corresponding secret for adversarially chosen index sets $I_{i,0}$ and $I_{i,1}$. The adversary wins the game if it correctly predicts the bit b. To disallow trivial attacks, we model security against *admissible* adversaries defined as follows:

Experiment $\mathsf{ULink}^b_{\mathcal{A}}(\lambda)$

$\left\{\left(s_i, n_{i,0}, n_{i,1}, I_{i,0}, I_{i,1}\right)\right\}_{i\in[L]} \leftarrow \mathcal{A}(1^\lambda)$
$\forall i \in [L] : \mathbf{sh}_i \leftarrow \mathsf{Proj}(s_i, n_{i,b}, I_{i,b})$
$\hat{b} \leftarrow \mathcal{A}(\mathbf{sh}_1, \ldots, \mathbf{sh}_L)$
Output $\hat{b}$

Figure 7: Unlinkability

Experiment $\mathsf{OM\text{-}ULink}^b_{\mathcal{A}}(\lambda, t)$

$(s_0, s_1, n_0, n_1, I, \hat{i}) \leftarrow \mathcal{A}(1^\lambda, t)$
$\{sh_1, \ldots, sh_t\} \leftarrow \mathsf{Proj}(s_0, n_0, I)$
$\{sh^*\} \leftarrow \mathsf{Proj}(s_1, n_1, \hat{i})$
If $b = 1$, $sh_t \leftarrow sh^*$
$\hat{b} \leftarrow \mathcal{A}(sh_1, \ldots, sh_t)$
Output $\hat{b}$

Figure 8: One-More Unlinkability

Definition 3.2 (Admissible Adversary)**.** *An adversary $\mathcal{A}$ in experiment $\mathsf{ULink}^b_{\mathcal{A}}(\lambda)$ is said to be admissible iff:*

1. $L \in \mathsf{poly}(\lambda)$
2. $\forall i \in [L]$, $|I_{i,0}| \le t_{priv}, |I_{i,1}| \le t_{priv}$
3. $\sum_i |I_{i,0}| = \sum_i |I_{i,1}|$

The second requirement ensures that all share subsets are unauthorized, while the third requirement ensures that the total number of shares are equal in each experiment. The experiment $\mathsf{ULink}^b_{\mathcal{A}}(\lambda)$ is described in Figure 7.

Definition 3.3 (Unlinkability)**.** *A $(t_{rec}, t_{priv}, d, \mathsf{max})$-MDSS with secret space $\mathcal{S}$ is* ε-unlinkable *if there exists a function $\varepsilon(\cdot)$ such that for all* admissible *adversaries $\mathcal{A}$,*

$$|\Pr[\mathsf{ULink}^0_{\mathcal{A}}(\lambda) = 1] - \Pr[\mathsf{ULink}^1_{\mathcal{A}}(\lambda) = 1]| \le \varepsilon(\lambda)$$

When $\varepsilon(\lambda)$ is negligibly small (resp., zero), we say that the scheme is statistically (resp., perfectly) unlinkable. Computational security can be described as well in the standard way.

One-More Unlinkability. Our second definition considers only *two* dealers. The adversary sees up to t_{priv} shares – either all from the first dealer *or* all but one from the first dealer and one share from the second dealer. We formally model this in the security game $\mathsf{OM\text{-}ULink}^b_{\mathcal{A}}(\lambda, t)$ described in Figure 8.

Definition 3.4 (Admissible Adversary)**.** *An adversary $\mathcal{A}$ in experiment $\mathsf{OM\text{-}ULink}^b_{\mathcal{A}}(\lambda, t)$ is said to be admissible iff $|I| \le t$.*

Definition 3.5 (One-More Unlinkability)**.** *We say that a $(t_{rec}, t_{priv}, d, \mathsf{max})$-MDSS with secret space $\mathcal{S}$ is* ε-one-more

unlinkable *if for every* $t \leq t_{priv}$*, there exists a function* $\varepsilon(\cdot)$ *such that for all* admissable *adversaries* $\mathcal{A}$*,*

$$|\Pr[\mathsf{OM\text{-}ULink}^0_{\mathcal{A}}(\lambda,t)=1]-\Pr[\mathsf{OM\text{-}ULink}^1_{\mathcal{A}}(\lambda,t)=1]| \leq \varepsilon(\lambda)$$

We can define statistical, perfect and computational one-more unlinkability in a similar manner as above.

Relationships between definitions. Our proof that the two definitions of unlinkability are equivalent, along with a proof that unlinkability is stronger than the standard secret-sharing notion privacy, can be found in the full version of the paper.

3.1.2 MD-Correctness

Our correctness definition differs from standard secret sharing in that we must reconstruct secrets given a set of shares from *multiple* dealers. This set may include shares from both sufficient dealers (i.e. those outputting at least t_{rec} shares) and *insufficient* dealers (i.e. those outputting fewer than t_{rec} shares). We require that Reconstruct outputs exactly the set of secrets shared by the sufficient dealers, except with some failure probability ε. Similar to standard secret sharing, we only consider correctness for honestly generated (i.e. via the Share algorithm) sets of shares.

Definition 3.6 (MD-Correctness)**.** [4] *We say that a* $(t_{rec}, t_{priv}, d, \mathsf{max})$*-MDSS is* ε*-correct if for all sets* $\{(s_i, n_i, I_i)\}_{i\in[h]}$ *where* $\forall i,\ I_i \subseteq [n_i]$*, and* $\sum_{i\in h}|I_i| \leq \mathsf{max}$*, and* $\left|\{I_i \text{ s.t. } |I_i| \geq t_{rec}\}\right| \leq d$*:*

$$\Pr\left[\mathsf{Reconstruct}(S) \neq \{s_i \text{ s.t. } |I_i| \geq t_{rec}\} \,\middle|\, S := \bigcup_{i=1}^{h} \mathsf{Proj}(s_i, n_i, I_i)\right] \leq \varepsilon$$

3.2 Constructing MDSS

We present two constructions of MDSS based on variants of Shamir's secret sharing. Our first scheme (§3.2.1) achieves provable MD-correctness for $d < \frac{t_{rec}}{t_{priv}}$ number of dealers and is primarily of theoretical interest. Our second scheme (§3.2.2) achieves heuristic correctness but is able to support both more dealers and a concretely efficient reconstruction algorithm that can be executed on devices with limited capabilities. This forms the basis of our AROF construction presented in the next section.

[4] One could also consider a weakening of this definition where we allow the output of $\mathsf{Reconstruct}(S)$ to contain false positives, namely, some extraneous secrets along with the correct set of secrets. In applications where the extraneous secrets are easily discardable or non-impactful, such a relaxed requirement could be sufficient.

3.2.1 Construction I

Our first scheme is based on a variant of Shamir's secret sharing scheme [41]. Recall that in Shamir's secret sharing, a secret s is an element of a finite field $\mathbb{F}$: to share s, one samples a random polynomial $p \in \mathbb{F}[z]$ with degree t_{priv} with the constraint that $p(0) = s$. The i-th share is simply the pair $(i, p(i))$ where $i \in \mathbb{F} \setminus 0$ is a fixed evaluation point.

This scheme is *not* unlinkable: consider a set of shares where two of the shares (i, y) and (i', y') are such that $i = i'$ (but $y \neq y'$). Given such a set, an adversary can determine with probability 1 that these shares correspond to *different* secrets. To achieve unlinkability, we therefore we allow the share algorithm to sample each evaluation point uniformly at random from $\mathbb{F} \setminus 0$. If $\mathbb{F}$ has size super-polynomial in the security parameter λ, any two evaluation points collide with probability negligible in λ. Crucially, this probability is the same for any two shares, *irrespective of whether they correspond to the same or different secrets*.

To achieve MD-correctness, we start with the observation that a collection of Shamir secret shares can be viewed as a Reed-Solomon codeword [31]. Therefore, recovering a secret from a collection of Shamir secret shares in the presence of noise is equivalent to the problem of Reed-Solomon decoding, and a natural choice for Reconstruct is an algorithm based on a Reed-Solomon decoder. Suppose that a single dealer supplies s shares in a set of size $w \leq \mathsf{max}$, and we would like to know if recovering the secret input of this dealer is possible. From a coding theory perspective, we may consider the set of shares to be a word that is distance $w - s$ away from an equivalent length Reed-Solomon codeword corresponding to the dealer's input. As long as an RS decoder exists which can recover from a fraction of $1 - \frac{s}{w}$ errors, we can recover the dealer's secret.

This, however, is not an easy condition to satisfy for a large numbers of dealers; as one example, for any non-trivial scheme with $d > 1$, we cannot use a unique decoding algorithm. Nevertheless, our problem setting can be mapped onto that of list-decoding, which can tolerate higher error (in the ramp setting) and support recovery of multiple polynomials (with enough evaluations). A well-known list-decoding algorithm for Reed-Solomon codes is the one devised by Guruswami and Sudan (GS) [21], and indeed we can use their algorithm as the principal component of Reconstruct to produce a provably correct scheme. The full descriptions of the scheme can be seen in Figure 9. . We defer the theorems and proofs regarding this construction to the full version of the paper.

Limitations. The GS-decoder comes with a significant limitation: it can only recover from a $1 - \sqrt{R}$ fraction of errors, where R is the message rate. Phrased in terms of MDSS parameters, this means that the construction only guarantees correctness for parameters satisfying $t_{rec} > \sqrt{\mathsf{max} \cdot t_{priv}}$. When we consider that $\mathsf{max} \geq d \cdot t_{rec}$, as each dealer must contribute

Construction I

$\underline{\mathsf{Share}(s,n)}$:

Sample a polynomial p of degree t_{priv} where $s = p(0)$

Sample n field elements $x_1,\ldots,x_n \xleftarrow{\$} \mathbb{F}$

return $\{(x_i, p(x_i))\}_{i\in[n]}$

$\underline{\mathsf{Reconstruct}(\{sh_1,\ldots,sh_w\})}$:

$p_1,\ldots,p_m \leftarrow \mathsf{GSDecode}(t_{priv}, t_{rec}, \{sh_1,\ldots,sh_w\})$

return $\{p_1(0),\ldots,p_m(0)\}$

Figure 9: MDSS Construction I

MDSS Construction

$\underline{\mathsf{Share}(s,n)}$:

Sample c polynomials $p_1,\ldots,p_c$ each of degree t_{priv}, where $s = p_1(0)||\ldots||p_c(0)$

Sample n field elements $x_1,\ldots,x_n \xleftarrow{\$} \mathbb{F}$

return $\{(x_i, p_1(x_i),\ldots,p_c(x_i))\}_{i\in n}$

$\underline{\mathsf{Reconstruct}(\{sh_1,\ldots,sh_w\})}$:

return $\mathsf{CH}^*\text{-}\mathsf{MDSS}(\{sh_1,\ldots,sh_w\})$

Figure 10: MDSS Construction II.

at least t_{rec} shares, this gives us $\frac{t_{rec}}{d} > t_{priv}$. Therefore, the gap between t_{rec} and t_{priv} degrades linearly with the number of dealers we wish to tolerate. This is undesirable, and our next construction will focus on achieving a better relationship between the gap and the number of tolerable dealers.

Subsequent works have built list-decoders for *folded* Reed-Solomon codes (which contain multiple polynomial evaluation points in each symbol) [20] that can tolerate nearly $(1-R)$ (i.e., optimal) error. However, to achieve our desirable trade-off between t_{priv} and t_{rec} using these decoders requires a share-size that is impractical for our AROF application. Furthermore, to the best of our knowledge, these decoding algorithms are not known to be concretely efficient. We defer further discussion of these and other codes to the full version of the paper.

3.2.2 Construction II

Our second construction is inspired by interleaved Reed-Solomon (IRS) codes, which are known to achieve a good trade-off between message and error rate, and could feasibly result in a scheme with a smaller gap between t_{rec} and t_{priv} [8, 15]. For an IRS code, rather than sampling a single polynomial p, a dealer samples c polynomials $p_1,\ldots,p_c$ with the secret s subdivided across the constant term of each, and shares the values $(r, p_1(r),\ldots,p_c(r))$, where the evaluation points r are sampled uniformly at random as in the first construction.

There are numerous decoding algorithms for interleaved Reed-Solomon codes and other closely related noisy curve reconstruction problems in the literature [36, 44]. Unfortunately, all of these algorithms are only known to work in a random noise model - which would correspond to the single dealer setting for *MDSS* - and most are not efficient enough to operate on limited devices at the parameter sets we require. To address these issues, we devise a concretely efficient, novel decoder that can heuristically decode up to a $\frac{c}{c+1}(1-R)$ fraction of errors, where c is the number of polynomials, when the channel is "semi-honest" The decoder, which we call CH^*-MDSS, is based on a dual version of a lattice-based decoding algorithm proposed by Cohn and Heninger [14].

In their prior work, Cohn and Heninger give a general technique for list-decoding variants of Reed-Solomon codes and their siblings: first recover one (or more) multivariate polynomials Q_i which vanish on all polynomials of the appropriate degree that agree with a large fraction of input points, then recover the common roots of the Q_i. The first step is done by constructing a polynomial lattice and finding a reduced basis, while the second is done by computing a Groebner basis over the Q_i to find all common roots or, in the case where $c = 1$, by factoring a single Q_i.

We instead consider the *dual* of that lattice, as it directly contains our polynomials of interest and does not require additionally computing a Groebner basis which could be resource intensive. The full description of the algorithm, which we refer to as CH^*-MDSS, can be found in Appendix A.

Putting all the pieces together brings us to our main construction, which can be seen in Figure 10. We defer our theorems and proofs regarding Construction 10 to the full version of the paper.

3.2.3 Optimizations

We now discuss some efficiency optimizations.

Small Fields. A suitable choice of field is required to achieve practical unlinkability and efficiency. If a dealer randomly samples the same evaluation point twice and produces two identical shares, then unlinkability is compromised. While this will occur with negligible probability if fields are large enough, in practice small fields are often best to minimize bandwidth and decoding complexity. We therefore propose a stateful variant of our construction in which dealers keep track of previously sampled evaluation points and emit *noise shares* (comprising random elements in place of the polynomial evaluation) whenever an x-coordinate is repeated. This preserves unlinkability at the cost of decreasing the number of "useful" shares passed to the reconstruction algorithm, and thereby increasing the probability of reconstruction failure.

Collision-Aware PRF. In practice, MDSS users may want to output shares in a *streaming* fashion.This would require

keeping state (all previously sampled x-coordinates), which may be undesirable. As an optimization for this setting, we define a new primitive we call a *Collision-Aware PRF* (CA-PRF). Using a CA-PRF allows an MDSS dealer to sample x-coordinates pseudorandomly while being alerted to potential collisions.

Due to space concerns, we defer a formal treatment of both of the above to the full version of the paper.

4 Construction of AROF Protocols

We now describe our main contribution: an offline finding protocol that achieves strong privacy while also admitting stalker detection.

Protocol intuition. The core algorithms of our protocol are presented in Figure 11. At a basic level, this protocol can be viewed as an extension of the offline finding construction used by Apple's FindMy [1]. Like the Apple construction, our protocol employs a pseudorandom function to derive a fresh "pseudonym" for the LTA at the start of each epoch. Just as in the Apple system, each pseudonym also doubles as a public key for a secure encryption scheme: upon receiving a pseudonym from any nearby LTA, volunteer devices in the tracking network can encrypt their current GPS coordinates and transmit the resulting ciphertext to the service provider. Owner devices can later re-derive a sequence of missed pseudonyms and the corresponding decryption keys, and use the RetrieveReports protocol to query the provider for each missed time period.

Achieving stalker detection. To enable stalker detection, our protocol makes several additions to the basic protocol described above. First, each LTA maintains a *detectability period* consisting of L consecutive time epochs. At the start of each detectability period, the LTA generates a secret tag identifier $\mathsf{id}_{\mathsf{tag}}$. It then secret-shares $\mathsf{id}_{\mathsf{tag}}$ using an MDSS scheme configured with appropriate parameters. With each call to Beacon, the LTA generates the current pseudonym and appends one secret share to be broadcast by the LTA. The MDSS parameters are informed by how long it should take before a stalker is detected, the maximum number of stalkers that should be detected, and the maximum number of beacons that will be within any victim's detection window. We discuss more in depth how these parameters are set in §5.

Critically, volunteer devices in the offline finding network *do not transmit these secret shares to the service provider*: they are kept locally and used only to enable stalker detection. Each device maintains a set of all shares received from nearby LTAs during a time window specified in the deployment parameters. Periodically, the victim device executes the Detect algorithm to perform secret sharing recovery. If this collection contains at least t_{rec} shares emitted by one LTA (or a similar set from multiple LTAs), then the MDSS recovery algorithm will recover each $\mathsf{id}_{\mathsf{tag}}$ for the devices. Each LTA can be configured to respond to interactive connections containing $\mathsf{id}_{\mathsf{tag}}$, which enables the victim to contact and physically locate any stalking LTAs.

We build security for predicates that mostly follow the intuition from the technical overview. If t_{rec} beacons from a tag are seen within a detection window, then it is considered to be a stalking tag. Intuitively, t_{rec} is close to the number of broadcasts made in the detection window, so the victim sees almost all broadcasts emitted by the LTA. Privacy is maintained as long as no party sees more than t_{priv} beacons from any detectability period.

Security. We give here the predicates for which our construction satisfies the properties described in §2. Due to space concerns, we defer the theorems and proofs for our construction to the full version of the paper. Tag indistinguishability and detectability both follow naturally from the unlinkability and correctness properties of the underlying MDSS.

For detectability we will need assumptions on the input to the Detect algorithm that are enforced by the MDSS parameters. In terms of the AROF, these assumptions will correspond to there being less than or equal to the maximum number of stalkers in the vicinity, with small enough noise coming from other tags so that no more than max beacons will be given to the detection algorithm. If these assumptions hold, then any tag that has supplied at least t_{rec} beacons corresponding to the same detectability period will be identified. Below is the precise statement for the predicate.

$$P_s(\mathsf{cfg}, Q, \mathsf{id}) = \text{“}\ \exists\ e \text{ such that } |\{(\mathsf{id}, i) \in Q \mid e = \lfloor \tfrac{i}{L} \rfloor\}| \geq t_{rec},$$
$$|\{(\mathsf{id}', e) \text{ s.t. } |\{(\mathsf{id}', i) \in Q | e = \lfloor \tfrac{i}{L} \rfloor\}| \geq t_{rec}\}| \leq d,$$
$$|Q| \leq \mathsf{max}\text{"}$$

For tag indistinguishability, the situation is simpler. Indistinguishability holds as long as no party ever sees more than t_{priv} beacons for either tag within any detectability period.

$$P_h(\mathsf{cfg}, Q) = \text{“ for } \mathsf{id} \in \{0,1\},$$
$$\forall e \in \mathbb{Z}^+, |\{(\mathsf{id}, i) \in Q \mid e = \lfloor \tfrac{i}{L} \rfloor\}| \leq t_{priv} \text{ and } Q \text{ is not a multi-set"}$$

The condition that Q is not a multi-set is meant to rule out an attack where the adversary queries on the challenge indices i_0^* or i_1^* before the challenge phase: by doing so they can trivially win with an equality check on the challenge beacon.

4.1 Implementation Considerations

We now discuss several implementation considerations related to the AROF construction presented in §4.

De-duplication and filtering. The duration of the anonymity epoch (*i.e.,* the time between identifier changes) is an important deployment consideration in our scheme. A shorter interval is clearly desirable to improve privacy against tracking

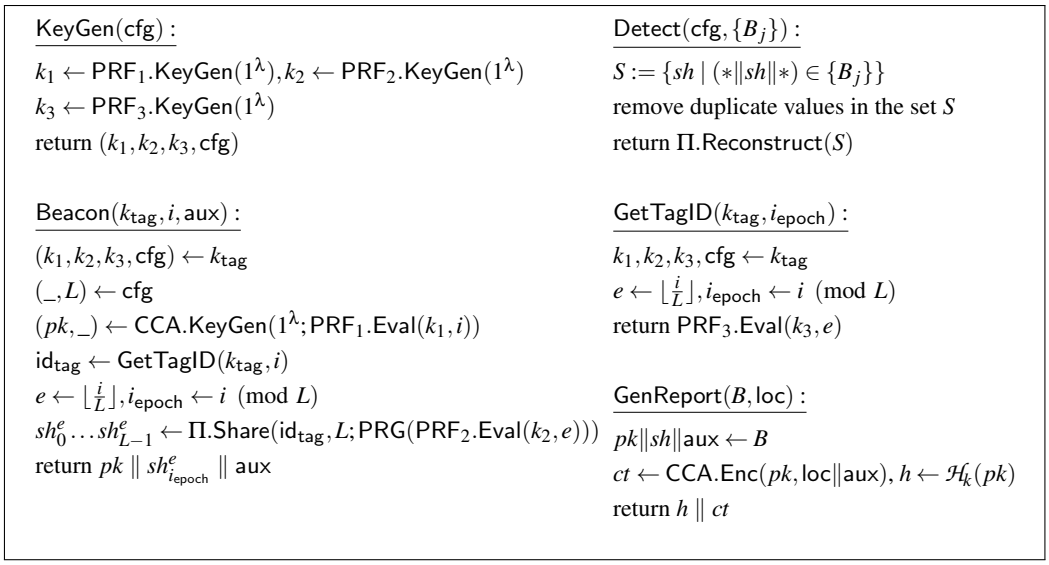

Figure 11: Main construction for abuse-resistant offline finding. This protocol assumes the existence of three pseudorandom functions PRF_i for $i \in [3]$ where the co-domain of $\mathsf{PRF}_1.\mathsf{Eval}$ and $\mathsf{PRF}_2.\mathsf{Eval}$ is $\{0,1\}^\lambda$ and the co-domain of $\mathsf{PRF}_3.\mathsf{Eval}$ is $\mathbb{F}_q$. PRG outputs a sufficient number of bits for the Share algorithm. CCA is a CCA-secure PKE scheme, $\{\mathcal{H}_k\}_{k \in \mathcal{K}}$ a family of collision resistant hash functions and Π a $(t_{rec}, t_{priv}, d, \mathsf{max})$-MDSS sharing scheme. $\|$ denotes concatenation. cfg contains the *MDSS* parameters $t_{priv}, t_{rec}, \mathsf{max}, d$ and a separate parameter L. The RetrieveReports protocol is equivalent to the one in Apple's FindMy [1].

adversaries. However, the duration of the anonymity epoch will also affect the efficiency of stalker detection. To illustrate these considerations we consider two candidate configurations.

Configuration 1: anonymity epoch ≈ broadcast interval. This is our recommended configuration, which maximizes privacy against tracking adversaries by minimizing the duration of the anonymity epoch. Under a 4-second anonymity epoch, this will result in a new pseudonym (via a call to Beacon) every broadcast that the LTA emits.[5] Our implementation transmits twice as much broadcast data at each interval, and so we propose to split each broadcast B into two separate transmissions, each sent at 4-second broadcast interval (see §6), producing 900 hourly unique secret shares from each LTA device.

Configuration 2: anonymity epoch ≫ broadcast rate. Current LTA deployments do not change the pseudonym with each broadcast.[6] This decision is likely motivated by computational costs and battery limitations. In these deployments, the LTA will re-broadcast each pseudonym many times. Applying the same logic to our protocol, these LTAs would also re-broadcast the same secret share.

Since duplicate secret shares do not aid in reconstruction, they are removed (de-duplicated) as part of the Detect algorithm. We note that this de-duplication procedure can produce some counter-intuitive effects on the noise rate. For example, LTAs that remain within range of a victim device for long periods of time (*e.g.,* stalking tags) will see proportionally more duplicate broadcasts removed, as compared to ephemeral LTAs that only briefly enter receive range of a victim device (see Figure 12.) On the positive side, these repeated broadcasts dramatically reduce the impact of erasures caused by RF-layer issues.

Pre-filtering. To compensate for the over-representation of non-stalking LTAs, systems with long-duration anonymity epochs can apply a *pre-filtering* heuristic prior to executing Detect. This heuristic takes advantage of the fact that ephemeral (*non-stalking devices*) are likely to exhibit broadcast behavior that can be recognized and filtered out prior to running Detect. For example, LTAs that remain in close proximity to the victim device for a period of time will transmit many duplicate broadcasts. Focusing detection exclusively on these broadcasts will reduce the number of non-stalker shares processed by the MDSS recovery algorithm.

Secret re-generation and detection window. In our construction, each LTA re-generates its tag identifier and secret-sharing polynomials every L epochs. This mechanism is intended to prevent tracking adversaries from correlating shares sent by the same LTA over longer periods. However, this periodic change of secrets poses a challenge for stalker detection: at boundary epochs where an LTA changes its secret, the victim may not receive a sufficient number of shares to detect a stalker. To address this, we propose two possible deployment options. A first solution is for a device to re-generate

[5] This is based on analysis of Apple's FindMy, where LTAs broadcast every 2 seconds [22, 23].

[6] For example, Apple's Find My rotates the pseudonym every 450 broadcasts (15 minutes) in near-owner mode.

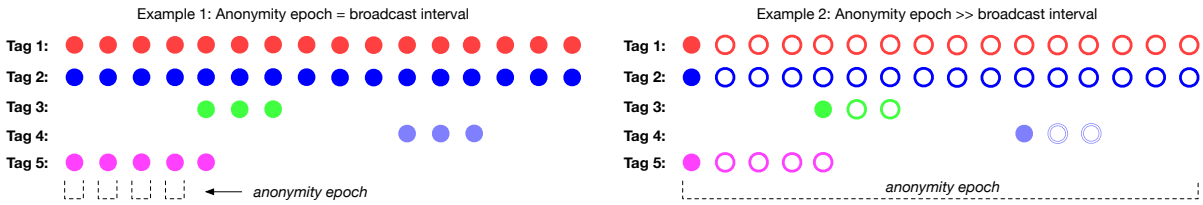

Figure 12: Illustration of the effect of de-duplicating repeated LTA broadcasts: both examples contain the same pattern of transmissions, but use different anonymity epoch durations. Filled circles represent fresh shares, hollow circles represent duplicate shares. **Left:** The anonymity epoch is short: persistent tags (Tag 1, Tag 2) produce the majority (32/43) of the unique broadcasts processed by the Detect algorithm. **Right:** The anonymity epoch is long: once duplicates are removed, the persistent tags (Tag 1, Tag 2) represent a minority (2/5) of the unique broadcasts processed by the decoding algorithm.

secrets relatively rarely. For example, re-generation every 24 hours ensures that such issues will only occur during a small fraction of each day. Second, a device could maintain two identifiers. The LTA would emit shares on both, and stagger their rotation to ensure that detection is always possible.

MAC Address Rotation To deploy with our suggested anonymity epochs would require rotating the LTA's MAC address either every 4 or 60 seconds, which is faster than the minimum rotation period of 15 minutes recommended by the BLE specification [9]. Care should therefore be taken during implementation to ensure that core BLE functionality is preserved. We note that cautious implementers could follow the recommendation and use an anonymity epoch of 15 minutes, which still represents a substantial privacy improvement over the state-of-the-art.

5 Selecting MDSS Parameters

Deploying our main construction of §4 requires us to identify reasonable system parameters for both the AROF and MDSS schemes. The challenge is to optimize privacy while also enabling *efficient* recovery in the face of reasonable noise rates. Our recommended parameters can be seen in Table 1. We now briefly describe the process we used to derive them.

High-level Considerations. We identify five high-level configuration parameters, and for the purposes of our analysis made exemplary choices for each. We note that these can be adjusted for specific deployments. Our identified parameters are:

1. The (minimum) number of minutes of continuous broadcasting after which a victim device must be able to detect a stalker LTA, which we set to 1 hour.
2. The duration of the anonymity epoch (*i.e.,* the time between pseudonym rotation), for which we consider 4 seconds and 1 minute.
3. The frequency of LTA secret/polynomial updates, which we set to 24 hours.
4. The maximum number of LTAs that will be in proximity to a victim device. We assume there will be at most three persistent ("stalker") LTAs at any given time, and at most a number of ephemeral LTAs equal to half the stalking LTA in the aggregate.
5. The bandwidth available for transmitting each secret share, which we derive to be 248 bits, based on available payload space in BLE, and 400 bits for BLEv5.

From these choices, we then optimize our MDSS scheme parameters to provide privacy for the longest period possible, while ensuring $> 99\%$ decoding success in the presence of missed broadcasts. We present two sets of parameter choices in Table 1: a recommended set given current deployment bandwidth limitations (based on widely-deployed versions of BLE), as well as a future recommendation for LTAs that support versions of BLEv5 with higher bandwidth limits. Due to space concerns we defer further description on how we derived these parameters to the full version of the paper.

6 Implementation

We now describe our implementation of our construction in §4, including optimizations and deployment choices.[7]

Implementing the Detect **algorithm.** We implemented the Detect algorithm using SageMath [43], a Python-based computer algebra system that includes fast C implementations for many algorithms. Our decoder for CH*-MDSS is around 400 lines of Python and is the bulk of our Detect algorithm implementation. For each parameter set in Table 1, *e.g.,* $c = 10$ with a field size of 22 bits, the decoding matrix requires less than 10 MB, even for the largest values of max. This is compatible with available application RAM on current smartphones.

Implementing the Beacon **algorithm.** For ease of comparison, we adopt the algorithmic choices of Apple's FindMy scheme: our implementation uses ECIES public keys [4] over the NIST P-224 curve. During each call to Beacon we define a subroutine GeneratePublicKey to pseudorandomly gener-

[7] Our code can be found at https://github.com/becgabri/abuse-resistant-private-lt

Epoch duration	Detect time	Max stalkers	L (= 24 hrs)	$\mathbb{F}_p$ (bits)	c	Share (bits)	Privacy time	t_{rec}	t_{priv}	max
Recommended parameters (compatible with current LTA bandwidth limits):										
4 sec	60 min	3	21,600	22	10	242	39 min	825	591	3150
1 min	60 min	3	1,440	24	9	240	41 min	59	41	210
Future parameters (compatible with BLE v5 bandwidth limits):										
4 sec	60 min	3	21,600	22	17	396	46 min	825	687	3150
1 min	60 min	3	1,440	26	14	390	47 min	59	47	210

Table 1: Recommended and possible future parameters for the MDSS construction of §4, instantiated with CH*-MDSS. Privacy times are rounded to the nearest minute.

Epoch duration	Avg. nearby LTAs	# Unique bxs received	Hardware	Detection runtime (no stalkers)	Detection runtime (stalkers present)	Polynomial recovery (all stalkers)
4 sec	1	1350	MacBook	0.60 sec	1.29 sec	1.37 sec
			RPi 3	8.07 sec	20.42 sec	20.99 sec
	2	2250	MacBook	1.88 sec	5.57 sec	8.74 sec
			RPi 3	25.05 sec	83.91 sec	126.18 sec
	3	3150	MacBook	3.86 sec	24.59 sec	32.96 sec
			RPi 3	51.83 sec	360.95 sec	484.21 sec
60 sec	1	90	MacBook	0.01 sec	0.01 sec	0.01 sec
			RPi 3	0.15 sec	0.27 sec	0.27 sec
	2	150	MacBook	0.02 sec	0.03 sec	0.04 sec
			RPi 3	0.31 sec	0.57 sec	0.73 sec
	3	210	MacBook	0.03 sec	0.08 sec	0.12 sec
			RPi 3	0.48 sec	1.27 sec	2.05 sec

Table 2: Benchmarks for our Detect algorithm (Figure 11) using the CH*-MDSS decoding algorithm with the recommended parameters of Table 1.

ate a new pseudonym *pk* from the master key. We implement this portion of the algorithm in SageMath.

To implement the secret sharing scheme we use the recommended parameters from Table 1, and split the Beacon algorithm into two stateful subroutines. At the beginning of each detection period (every L epochs), we execute GeneratePolynomials to sample polynomials $p_1, \ldots, p_c$, and cache these polynomials during the entire period. The tag password ($\mathsf{id}_{\mathsf{tag}}$) is the concatenation of the constant terms of each polynomial. During every call to Beacon we execute the subroutine GenerateSecretShare to pseudorandomly generate $r \in \mathbb{F}$ and compute a secret share. We implemented these functions in SageMath, as well as in a combination of JavaScript and C in order to run experiments on a low-powered hardware platform. For more information on the hardware used, see §7.

Detection in practice. Since we will be executing detection on resource-constrained devices, we propose that devices should continuously collect shares from the environment into a ring buffer containing the most recent N minutes of shares. The Detect algorithm can then be run every few minutes once the buffer is full. This strategy provides a relatively timely warning of stalking behavior and can be adjusted to account for computational resources. Additionally, once a stalker has been detected, their polynomial can be cached and the shares they emit quickly filtered out. This means that the expensive cases with many persistent LTAs should only occur rarely.

7 Experiments

In this section we evaluate the empirical runtime of our stalker-detection and encoding algorithms. We used three different pieces of hardware to conduct our experiments: a single core of a 2020 MacBook computer with an M1 chip and 16 GB of RAM running MacOS Ventura 13.3.1 (a), a Raspberry Pi 3 Model B running Ubuntu Server 23.04, and a Puck.js, a hardware platform based on the same SoC as the AirTag, running firmware version 2.15.

Stalker detection. To conduct this analysis we ran our CH*-MDSS-based Detect algorithm on simulated broadcasts. We split the experiments into two cases:

1. Decoding when *stalkers are present*, i.e. at least t_{rec} shares originate from the same LTA.
2. Decoding when *no stalkers are present*, the common case for most users, when no single (unknown) LTA contributes t_{rec} shares to the input.

We split these cases further to test two different anonymity epochs: 4 seconds and 60 seconds. More concretely, we performed the following experiment:

Algorithm	t_{priv} (epoch)	Runtime MacBook	Runtime Puck.js
GeneratePublicKey		484 μs	
GeneratePolynomials	591 (4 sec)	6233 μs	11.88 sec
GenerateSecretShare	591 (4 sec)	87 μs	0.26 sec
GeneratePolynomials	41 (60 sec)	451 μs	0.72 sec
GenerateSecretShare	41 (60 sec)	11 μs	0.04 sec

Table 3: Runtime average over 1,000 iterations for the GeneratePublicKey, GenerateSecretShare and GeneratePolynomials subroutines of the Beacon algorithm. Secret sharing uses the 22-bit field with $c = 10$ for the 4 second epoch and the 24-bit field with $c = 9$ for the 60 second epoch.

1. For each anonymity epoch (4 sec, 60 sec), and for each number of stalkers (1, 2, 3), we determine the maximum number of shares our decoder could receive (e.g. for a 4 second epoch and 2 stalkers, the decoder could receive a maximum of $(3600/4) * (2 + 0.5) = 2250$ shares).
2. We then run two experiments: first where that number of shares is generated by a combination of stalkers and ephemeral LTAs, and second where the shares are entirely ephemeral. For the first case we measure both the time to *detection* (i.e. determining the existence of a stalker), and the time to *recovery* (learning the id for all stalkers). We average the runtimes of each experiment over 500 iterations.

For each experiment, we generated the data so that all broadcasts have a unique x-coordinate in order to evaluate a fixed number of points passed to the decoding algorithm and produce an upper bound on the projected runtime. Our decoder was implemented as described in §6 and configured according to Table 1. Table 2 presents our running times, which for common cases take seconds.

Encoding. We benchmarked our encoding algorithms on both the laptop and Puck.js. Table 3 shows the running times of both.

8 Limitations

The AROF scheme proposed in this work comes with some limitations. First, the number of stalking LTAs that the scheme can tolerate must be fixed at the time of deployment. Should a user be stalked by more than the number of LTAs chosen at the outset, they will be unable to detect any of the malicious LTAs. We chose a maximum of three stalking LTAs based on a privacy/cost tradeoff. At the time of writing a four-pack of AirTags costs $99, which is comparable in price to a self-contained GPS tracker. Larger numbers of tolerable stalkers can be chosen at the outset, but would result in lower privacy times than those shown in Table 1. For example, for the 4 second anonymity epoch, tolerating up to 4 stalkers would result in 33 minutes of privacy, 5 stalkers 28 minutes, and 6 stalkers 22 minutes. We note that these numbers were computed assuming the BLE broadcast size is fixed, and as discussed in §5, privacy time can be increased by assuming a larger broadcast.

Additionally, our scheme is weak to large levels of collusion among trackers. We assume that the privacy adversary is not a global adversary. If they are, then the stalking victim no longer has an asymmetry, and passive stalker detection protocols are unlikely to be compatible with privacy.

9 Related Work

Offline finding. Several works have considered the privacy and integrity of the offline finding (tracker) ecosystem. Heinrich, Bittner and Hollick [22] evaluated the anti-stalking mechanism use in FindMy. Heinrich, Stute, Kornhuber and Hollick [23] also considered the privacy of Apple's FindMy protocol. Mayberry *et al.* [30] considered ways to bypass tracking alerts in AirTag devices, and in a separate work Mayberry, Blass and Fenske [29] devised protocols to protect against counterfeit tags. Depres *et al.* [17] develop a method to detect stalking devices with frequently rotating MAC addresses using physical-layer techniques such as RSSI.

Most of the previous works do not consider the attack we attempt to mitigate in this paper: tracking a user via the unchanging identifier of their own LTAs. [23] mentions such a concern, but dismisses it as the authors study only devices which rotate their identifier every 15 minutes. Similar to our work, [17] proposes a stalker-detection mechanism for LTAs with frequently changing identifiers. However, their work focuses on linking broadcasts via physical-layer attributes, whereas our work is entirely on the protocol level. Additionally, our work is the only proposal that prevents the tracking of an LTA until a signficant time in contact with it has passed.

Secret sharing. Many schemes use secret sharing for privacy applications: here we focus on some recent works closely related to ours. The Apple PSI system [6] defines the notion of a *detectable hash function* based on interleaved Reed-Solomon codes [8]. Similarly, the STAR protocol of Davidson *et al.* [16] emits shares of (multiple) secrets for private telemetry reporting: unlike our work, their protocol assumes that the decoder can recognize all shares from a dealer and so noise and unlinkability concerns are not considered. Finally, the literature on robust secret sharing (RSS) and cheater detection is itself robust, and considers many different models (see [10, 12, 34] and references therein). A key difference between the classical RSS setting and our setting is that in RSS there is exactly one dealer; crucially, RSS does not incorporate the notion of unlinkability.

10 Conclusion and Future Work

In this work we considered the problem of constructing privacy-preserving tracking protocols that enable efficient abuse detection. We demonstrate that the use of secret sharing enables privacy-preserving offline finding while also admitting efficient algorithms for detecting stalkers. This work leaves several open questions for future work, including the development of efficient protocols secure against malicious/counterfeit tags, as well as the development of improved multi-dealer secret sharing schemes.

References

[1] FindMy: One App to Find it All. https://www.apple.com/icloud/find-my/.

[2] Apple Platform Security: Find My. Available at https://support.apple.com/guide/security/find-my-security-sec6cbc80fd0/1/web/1, 2023.

[3] Apple. An update on AirTag and unwanted tracking. Available at https://support.apple.com/guide/security/find-my-security-sec6cbc80fd0/1/web/1, February 2022.

[4] Richard Barnes, Karthikeyan Bhargavan, Benjamin Lipp, and Christopher A. Wood. Hybrid Public Key Encryption. RFC 9180, February 2022.

[5] Lindsey Bever. She tracked her boyfriend using an AirTag — then killed him, police say. *Washington Post*, June 2022.

[6] Abhishek Bhowmick, Dan Boneh, Steve Myers, Kunal Talwar, and Karl Tarbe. The Apple PSI system. Available at https://www.apple.com/child-safety/pdf/Apple_PSI_System_Security_Protocol_and_Analysis.pdf, August 2021.

[7] G. R. Blakley and Catherine Meadows. Security of ramp schemes. pages 242–268, 1984.

[8] Daniel Bleichenbacher, Aggelos Kiayias, and Moti Yung. Decoding of Interleaved Reed Solomon Codes over noisy data. In *ICALP '03*, 2003.

[9] Bluetooth SIG. *Core Specification*, 12 2014. Version 4.2.

[10] E. F. Brickell and D. R. Stinson. The detection of cheaters in threshold schemes. In Shafi Goldwasser, editor, *Advances in Cryptology — CRYPTO' 88*, pages 564–577, New York, NY, 1990. Springer New York.

[11] CBS Chicago Team. Man killed girlfriend after she removed AirTag he'd secretly placed in her car, prosecutors say. *Channel 2 CBS News Chicago*, July 2023.

[12] Alfonso Cevallos, Serge Fehr, Rafail Ostrovsky, and Yuval Rabani. Unconditionally-secure robust secret sharing with compact shares. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology – EUROCRYPT 2012*, pages 195–208, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[13] Mitchell Clark. Samsung wants to make sure nobody's tracking you with its SmartTags. *The Verge*, April 2021.

[14] Henry Cohn and Nadia Heninger. Approximate common divisors via lattices. *The Open Book Series*, 1(1):271–293, 2013.

[15] Don Coppersmith and Madhu Sudan. Reconstructing curves in three (and higher) dimensional space from noisy data. In *Proceedings of the Thirty-Fifth Annual ACM Symposium on Theory of Computing*, STOC '03, page 136–142, New York, NY, USA, 2003. Association for Computing Machinery.

[16] Alex Davidson, Peter Snyder, E. B. Quirk, Joseph Genereux, Benjamin Livshits, and Hamed Haddadi. Star: Secret sharing for private threshold aggregation reporting. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, CCS '22, page 697–710, New York, NY, USA, 2022. Association for Computing Machinery.

[17] Tess Despres, Noelle Davis, Prabal Dutta, and David Wagner. Detagtive: Linking macs to protect against malicious ble trackers. In *Proceedings of the Second Workshop on Situating Network Infrastructure with People, Practices, and Beyond*, pages 1–7, 2023.

[18] Casey Devet, Ian Goldberg, and Nadia Heninger. Optimally robust private information retrieval. In *USENIX Security Symposium*, pages 269–283, 2012.

[19] FOX 7 News. Texas man uses Apple Airtag to track down stolen truck, then shoots, kills suspect: police. Available at https://www.fox7austin.com/news/san-antonio-texas-apple-airtag-stolen-truck-suspect-killed-police, April 2023.

[20] V. Guruswami and A. Rudra. Explicit codes achieving list decoding capacity: Error-correction with optimal redundancy. *IEEE Trans. Inf. Theor.*, 54(1):135–150, jan 2008.

[21] V. Guruswami and M. Sudan. Improved decoding of Reed-Solomon and algebraic-geometric codes. In *Proceedings 39th Annual Symposium on Foundations of Computer Science*, pages 28–37, 1998.

[22] Alexander Heinrich, Niklas Bittner, and Matthias Hollick. Airguard - protecting android users from stalking attacks by apple find my devices. In *WiSec '22*, 2022.

[23] 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. *Proc. Priv. Enhancing Technol.*, 2021(3):227–245, 2021.

[24] Johana Bhuiyan and agencies. Apple and Google submit plan to fight AirTag stalking. *The Guardian*, May 2023.

[25] Thomas Kailath. *Linear systems*, volume 156. Prentice-Hall Englewood Cliffs, NJ, 1980.

[26] Erik Kay. 3 ways unknown tracker alerts on Android help keep you safe. Available at https://blog.google/products/android/unknown-tracker-alert-google-android/, July 2023.

[27] Brent Ledvina, Zachary Eddinger, Ben Detwiler, and Siddika Parlak Polatkan. Detecting Unwanted Location Trackers. Available at https://datatracker.ietf.org/doc/draft-detecting-unwanted-location-trackers/00/, May 2023. Work in Progress.

[28] Adrienne Matei. 'I was just really scared': Apple AirTags lead to stalking complaints. *The Guardian*, January 2022.

[29] Travis Mayberry, Erik-Oliver Blass, and Ellis Fenske. Blind My: An improved cryptographic protocol to prevent stalking in Apple's Find My network. In *PoPETS '23*, volume 2023.

[30] Travis Mayberry, Ellis Fenske, Dane Brown, Jeremy Martin, Christine Fossaceca, Erik C. Rye, Sam Teplov, and Lucas Foppe. Who tracks the trackers? circumventing apple's anti-tracking alerts in the find my network. In *WPES '21*, page 181–186, 2021.

[31] R. J. McEliece and D. V. Sarwate. On sharing secrets and reed-solomon codes. *Commun. ACM*, 24(9):583–584, sep 1981.

[32] T. Mulders and A. Storjohann. On lattice reduction for polynomial matrices. *Journal of Symbolic Computation*, 35(4):377–401, 2003.

[33] Caleb Naysmith. Apple AirTags and Bluetooth Trackers Are Officially a Billion-Dollar Industry. *Yahoo News*, December 2022.

[34] Satoshi Obana. Almost optimum t-cheater identifiable secret sharing schemes. In Kenneth G. Paterson, editor, *Advances in Cryptology – EUROCRYPT 2011*, pages 284–302, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[35] Vasile Mihai Popov. Invariant description of linear, time-invariant controllable systems. *SIAM Journal on Control*, 10(2):252–264, 1972.

[36] Sven Puchinger and Johan Rosenkilde né Nielsen. Decoding of interleaved reed-solomon codes using improved power decoding. In *2017 IEEE International Symposium on Information Theory (ISIT)*, pages 356–360. IEEE, 2017.

[37] I. S. Reed and G. Solomon. Polynomial codes over certain finite fields. *Journal of the Society for Industrial and Applied Mathematics*, 8(2):300–304, 1960.

[38] RetailNext. RetailNext Announces General Availability of Aurora Sensor. Available at https://retailnext.net/press-release/retailnext-announces-general-availability-of-aurora-sensor, 2023.

[39] Nat Rubio-Licht. Tile adds anti-stalking feature after AirTag backlash. *Protocol*, March 2022.

[40] SensorMatic. ShopperTrak Traffic Insights. Available at https://www.sensormatic.com/shoppertrak-retail-traffic-insights, 2023.

[41] Adi Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, November 1979.

[42] Jonathan Stempel. Apple is sued by women who say AirTag lets stalkers track victims. *Reuters*, December 2022.

[43] The SageMath Developers. SageMath, June 2023.

[44] Jiun-Hung Yu and Hans-Andrea Loeliger. Simultaneous partial inverses and decoding interleaved reed–solomon codes. *IEEE Transactions on Information Theory*, 64(12):7511–7528, 2018.

A Full Description of CH*-MDSS

We begin this section by defining polynomial lattices. We then present the problem we need CH*-MDSS to solve in order for our second construction to satisfy MD-Correctness. We end by giving the construction and stating our experimentally-backed heuristic correctness assumption.

Polynomial lattice preliminaries. See [14] for a background on polynomial lattices and lattice reduction. Let $\mathbb{F}[z]$ be the ring of polynomials over variable z with coefficients in $\mathbb{F}$. Let $\mathbb{F}(z)$ be the field of rational functions $u(z)/d(z)$, $u(z), d(z) \in \mathbb{F}[z]$. For a polynomial $f \in \mathbb{F}[z]$, its degree is $\deg(f)$; for a rational function $f(z) = u(z)/d(z)$ in lowest terms, its degree is $\deg(u) - \deg(d)$. Consider a matrix $B \in \mathbb{F}(z)^{n \times n}$ with entries that are rational functions; let the rows of B be n-dimensional vectors $b_i \in \mathbb{F}(z)^n$. The *polynomial lattice* generated by basis B is the set of vectors

$L(B) = \{v \mid v = \sum_{i=1}^{n} a_i b_i, a_i \in \mathbb{F}[z]\}$. That is, the lattice consists of vectors over $\mathbb{F}(z)$ that are $\mathbb{F}[z]$ (polynomial) linear combinations of basis vectors. We define the *length* of a vector $v = (v_1, v_2, \ldots, v_n)$, $v_i \in \mathbb{F}(z)$ to be $|v| = \max_i \deg(v_i)$. Define the determinant $\det L(B) = \det B$. For a full-rank n-dimensional polynomial lattice $L(B)$, there is a polynomial-time algorithm to compute a so-called *reduced* basis B' for $L(B)$ given a basis B [32]. Such a reduced basis is guaranteed to contain a vector of length $|v| \leq (\deg \det L(B))/n$, and (unlike in the case of integer lattices) is guaranteed to contain a shortest vector of the lattice. We will also say that $\lambda_1(B)$ is the length of the shortest vector in $L(B)$. For two vectors $v, w \in \mathbb{F}(z)^n$ define the inner product $\langle v, w \rangle = \sum_i v_i \cdot w_i$. Finally, for any lattice $L(B)$ one can define the *dual* lattice $L^*(B) = \{w \in \mathbb{F}(z)^n \mid \langle w, v \rangle \in \mathbb{F}[z] \forall v \in L(B)\}$. Given a basis B for a full-rank lattice $L(B)$, $(B^{-1})^T$ is an explicit basis for the dual $L(B)^*$.

A note on reduced bases of lattices. While for many applications any reduced basis suffices, we are solely interested in those that are also in *popov form* [25, 35]. It is a direction of future work to determine why such bases are needed for the correctness of our algorithms. For the rest of this paper, we assume $\mathsf{LatticeReduce}(B)$ returns the popov form of B.

Below, we consider the following problem from coding theory which is related to the heuristic assumption that is needed to prove the correctness of our algorithm.

Definition A.1 (Simultaneous Curve Reconstruction Problem [8, 15])**.** *Given n, t, k, c, q, input points $\{(\alpha_i, (\beta_{i,1}, \ldots \beta_{i,c}))\}_{i \in [n]}$, where $\forall i \in [n], j \in [c]$, $\alpha_i, \beta_{i,j} \in \mathbb{F}_q$, each α_i distinct, recover all polynomial sets $P = (p_1, \ldots p_c) \in \mathbb{F}_q[x]^c$ s.t. $\forall i, \deg(p_i) \leq k$ and P agrees with $T \subset [n], |T| \geq t$ points. A polynomial set $P = (p_1 \ldots p_c)$* agrees *with a point $(\alpha, (\beta_1, \ldots \beta_c))$ if $\forall j \in [c], p_j(\alpha) = \beta_j$*

Given a set of input points $\{(\alpha_i, \beta_{i,1}, \ldots, \beta_{i,c})\}_{i=1}^{n}$, $(p_1 \ldots p_c)$ may be called a *solution* or a *solution set* if it agrees with at least t input points.

We briefly make some remarks about this problem from what is known in the literature. First, the difficulty of this problem depends heavily on the distribution of the points $\{(\alpha_i, (\beta_{i,1}, \ldots \beta_{i,c}))\}_{i \in [n]}$. When there is one solution, and all other points are randomly sampled (i.e. $\alpha_i, \beta_{i,j} \xleftarrow{\$} \mathbb{F}_q$) we can *theoretically* recover the solution P with high probability, provided that $t > (nk^c)^{\frac{1}{c+1}} + k + 1$ [15].We know of additional algorithms that are conjectured to achieve the more optimum bound of $t > (nk^c)^{\frac{1}{c+1}}$ [36].

To the best of our knowledge, there are no algorithms in the coding theory literature which solve this problem in a different setting from the random error model.

In our work, we handle a slightly more adversarial channel, where there are multiple solution sets but all solutions (and errors) are independent of one another. We formalize this below.

Input Distribution We begin by introducing two pieces of notational shorthand. First, we define the honest distribution of points that the input to the algorithm will draw from. This distribution is equivalent to sampling points along a collection of polynomials.

Definition A.2 (Honest Distribution)**.** *For a field $\mathbb{F}$, an integer m, and a set of polynomials $P = \{p_1, \ldots, p_c\}$, define the following distribution $D^{\mathbb{F}}_{m,P}$ on $(\mathbb{F}^{c+1})^m$:*

- *For $i \in [m]$, sample $x_i \xleftarrow{\$} \mathbb{F}$ and set $sh_i := (x_i, p_1(x_i), \ldots, p_c(x_i))$*
- *Return $\{sh_1, \ldots, sh_m\}$*

Definition A.3 (Channel Distribution)**.** *Let $\mathbb{F}$ be a field, and let $c, k, \mathsf{max}, m_1, \ldots, m_h$ be integers where $\mathsf{max} = \sum_{i=1}^{h} m_i$, and let $s_1, \ldots, s_h \in \mathbb{F}^c$ be vectors of field elements. Let $\zeta(\mathbb{F}, k, c, s)$ denote the set of all polynomial sets $P = \{p_1, \ldots, p_c\}$ over $\mathbb{F}$ where $\forall i \in [c]$, $\deg(p_i) \leq k$, and $s = (p_1(0), \ldots, p_c(0))$. Then define the following distribution $CD_{c,k,\mathsf{max},\vec{m},\vec{s}}$ on $(\mathbb{F}^{c+1})^{\mathsf{max}}$ where $\vec{m} = m_1, \ldots, m_h$ and $\vec{s} = s_1 \ldots s_h$:*

- *For $i \in [h]$, sample $P_i \xleftarrow{\$} \zeta(\mathbb{F}, k, c, s_i)$*
- *For $i \in [h]$, sample $S_i \leftarrow D^{\mathbb{F}}_{m_i, P_i}$*
- *Return $S := \bigcup_{i \in [h]} S_i$*

Looking ahead, we will need one additional requirement for our algorithm to work: all input points will need to have unique x-coordinates. We define $\mathsf{Unique}(S)$ for a set of points S to be true if all points have a unique x-coordinate, and false otherwise.

A.1 CH*-MDSS

Description of subroutines Briefly we describe all previously undefined subroutines that are needed to give our algorithm description. LagrInterpol simply does lagrangian interpolation of the input points. $\mathsf{Find}(M, d)$ takes as input a matrix M and a degree bound d, and returns all row vectors $\vec{r}$ with $|\vec{r}| \leq d$. $\mathsf{Translate}(\vec{v}, k)$ takes a vector $\vec{v} = (v_0, v_1, \ldots v_c)$ and transforms it into a potential solution polynomial set by unweighting v_0 by k, then dividing $v_1, \ldots v_c$ by v_0. $\mathsf{IsSol}(V, \mathsf{ws}, t, k)$ checks whether the set V agrees with at least t points in the working set ws and if all the entries are actually polynomials of degree less than or equal to k. Finally, $\mathsf{ProcessSol}(P, \mathsf{solns}, \mathsf{ws})$ adds the polynomial set P to solns and removes all agreeing points from ws.

We now give an overview of CH*-MDSS, which is given in Algorithm 1. Recall that this construction is based on the one presented by Cohn and Heninger [14], which creates a lattice for multivariate polynomials Q where a short lattice basis corresponds to low degree Q that have the polynomials of interest as common roots. In particular, we focus our attention on the "linear" version of this decoder, since it corresponds to a relatively small lattice (with dimension growing linearly

with c) and finding the common roots only requires solving a linear system of equations. While this algorithm is a good starting place, it is not ideal, since it still requires a two step procedure, where a lattice reduction is first done to get a short basis before potentially heavy post-processing. We would like to find solutions in one-shot, with minimal additional overhead.

Fortunately, such efficiency gains may be possible by using the dual of the lattice presented in [14]. The dual directly contains our solutions $(p_1 \dots p_c)$ as their dot product with any vector in the primal is a multiple of their syndrome polynomial by construction.[8] We also know these vectors are relatively short because of the degree bounds on the p_i. Thus, we may hope that these solutions show up in a reduced basis.

Constructing this dual lattice, reducing it, and searching for short vectors that correspond to polynomial sets that agree with our input constitutes the first stage of the algorithm (lines 3-7, Algorithm 1).

Experimentally, we found that this first part of the algorithm is capable of finding solutions not only in the random noise setting (which was explored in the past for the primal lattice) but even under our input distribution, with a few caveats:

- Often the reduced basis only contained one solution of interest, the solution agreeing with the most input points
- When there was no single solution agreeing with at least two more points than any other, the short basis contained no solution vector of interest

The former problem can be solved by an iterative solving procedure that removes an input point once a solution has been identified that agrees with it [9]. The second issue is trickier to deal with. To give a high level overview, in these cases we can identify smaller sublattices that contain our solution vectors and then tweak them to allow for a more efficient search of possible solution vectors over this space. A detailed explanation can be found in the full version of this paper. The procedure in lines 15-29 allow us to identify a single solution in the second case, which is enough to give us an iterative algorithm for finding all solutions.

We now state the assumption that must be fulfilled for correctness to hold.

Heuristic Assumption 1. *Let $CD_{c,k,\mathsf{max},\vec{m},\vec{s}}$ be defined as in Definition A.3. Then:*

$$\Pr\left[\mathsf{CH}^*\text{-}\mathsf{MDSS}(S) \neq \left\{ s_i \text{ s.t. } m_i \geq \frac{\mathsf{max}+ck}{c+1} \right\} \;\middle|\; \begin{matrix} S \leftarrow CD_{c,k,\mathsf{max},\vec{m},\vec{s}} \\ \mathsf{Unique}(S)=\mathsf{True} \end{matrix} \right] \leq negl(\log|\mathbb{F}|)$$

[8] To be precise, if there is a solution $(p_1 \dots p_c)$, the target vector in the dual will have the form $(z^k E(z), p_1(z)E(z), \dots p_c(z)E(z))$ where $E(z)$ is the error locator polynomial. This is due to scaling the entries in the lattice to be in $\mathbb{F}[z]$

[9] This is only possible because it is very unlikely, given our input distribution, that there will be an input point agreeing on more than one solution

Algorithm 1: CH*-MDSS

Input : $k, t, n, \{(\alpha_i, \beta_{i,1}, \dots, \beta_{i,c})\}_{i=1}^{n}$
Output : a list $\{(p_1^i \dots p_c^i)\}_{i=1}^{z}$ or $\perp$

1 $\mathsf{solns} := [], \mathsf{ws} := [n]$, $\mathsf{fail} := \mathit{False}$
2 **while** $|\mathsf{ws}| \geq t$ *and not* fail **do**
3 $\quad \forall j \in [c], f_j(z) = \mathsf{LagrInterpol}(\{(\alpha_i, \beta_{i,j})\}_{i \in \mathsf{ws}})$, $N(z) = \prod_{i \in \mathsf{ws}}(z - \alpha_i)$
4

$$M = \begin{bmatrix} z^k & f_1(z) & f_2(z) & \dots & f_c(z) \\ & N(z) & & & \\ & & N(z) & & \\ & & & \ddots & \\ & & & & N(z) \end{bmatrix}$$

5 $\quad M_{\mathsf{red}} \leftarrow \mathsf{LatticeReduce}(M)$
6 $\quad$ **if** $\lambda_1(M_{\mathsf{red}}) > k + (|\mathsf{ws}| - t)$ **then**
7 $\quad\quad$ set fail to *True*
8 $\quad$ **else**
9 $\quad\quad \vec{v_1}, \dots \vec{v_h} \leftarrow \mathsf{Find}(M_{\mathsf{red}}, \lambda_1(M_{\mathsf{red}}))$
10 $\quad\quad V \leftarrow \mathsf{Translate}(\sum_{i=1}^{h} \vec{v_i}, k)$
11 $\quad\quad$ **if** $\mathsf{IsSol}(V, \mathsf{ws}, t, k)$ **then**
12 $\quad\quad\quad \mathsf{ProcessSol}(V, \mathsf{solns}, \mathsf{ws})$
13 $\quad\quad$ **else**
14 $\quad\quad\quad$ **if** $\lambda_1(M_{\mathsf{red}}) == k + (|\mathsf{ws}| - t)$ **then**
15 $\quad\quad\quad\quad$ set fail to *True*
16 $\quad\quad\quad$ **else**
17 $\quad\quad\quad\quad$ choose $i \in \mathsf{ws}$, s.t. $(z - \alpha_i) \nmid v_0$
18 $\quad\quad\quad\quad \vec{b_1}, \dots, \vec{b_m} \leftarrow \mathsf{Find}(M_{\mathsf{red}}, k + (|\mathsf{ws}| - t))$
19 $\quad\quad\quad\quad (b_0^i, \dots b_c^i) \leftarrow \vec{b_i}, \forall i \in [m]$
20

$$S = \begin{bmatrix} z^{|ws|} \cdot b_0^1 & b_1^1 & \dots & b_c^1 & & & \\ \vdots & & & & & & I_m \\ z^{|ws|} \cdot b_0^m & b_1^m & \dots & b_c^m & & & \\ z^{|ws|} \cdot (z - \alpha_i) & 0 & \dots & \dots & \dots & \dots & 0 \end{bmatrix}$$

21 $\quad\quad\quad\quad S_{\mathsf{red}} \leftarrow \mathsf{LatticeReduce}(S)$
22 $\quad\quad\quad\quad \vec{s_1}, \dots \vec{s_h} \leftarrow \mathsf{Find}(S_{\mathsf{red}}, \lambda_1(S_{\mathsf{red}}))$
23 $\quad\quad\quad\quad \vec{r_i} \leftarrow \sum_{j=0}^{m-1} s_{c+1+j}^i \vec{b_j}, \forall i \in [h]$
24 $\quad\quad\quad\quad R \leftarrow \mathsf{Translate}(\sum_{i=1}^{h} \vec{r_i}, \mathsf{ws})$
25 $\quad\quad\quad\quad$ **if** $\mathsf{IsSol}(R, \mathsf{ws}, t, k)$ **then**
26 $\quad\quad\quad\quad\quad \mathsf{ProcessSol}(R, \mathsf{solns}, \mathsf{ws})$
27 $\quad\quad\quad\quad (q_1, \dots, q_c) \leftarrow R$, $M \leftarrow \{i \mid \forall j \in [c], q_j(\alpha_i) = \beta_{i,j}\}$, $I \leftarrow \{i \in \mathsf{ws} \mid (z - \alpha_i) \nmid v_0\}$, $L \leftarrow I \setminus M$
28 $\quad\quad\quad\quad \forall j \in [c], p_j \leftarrow \mathsf{LagrInterpol}(\{(\alpha_i, \beta_{i,j})\}_{i \in L})$
29 $\quad\quad\quad\quad$ **if** $\mathsf{IsSol}((p_1 \dots p_c), \mathsf{ws}, t, k)$ **then**
30 $\quad\quad\quad\quad\quad \mathsf{ProcessSol}((p_1, \dots p_c), \mathsf{solns}, \mathsf{ws})$
31 **return** solns