NTNU | Norwegian University of Science and Technology

ABUSE-RESISTANT LOCATION TRACKING: BALANCING PRIVACY AND SAFETY IN THE OFFLINE FINDING ECOSYSTEM

Presented by Caroline Sandsbråten

Authors: Grabielle Beck, Harry Eldridge, Matthew Green, Nadia Heninger, Abhishek Jain

January 29, 2024

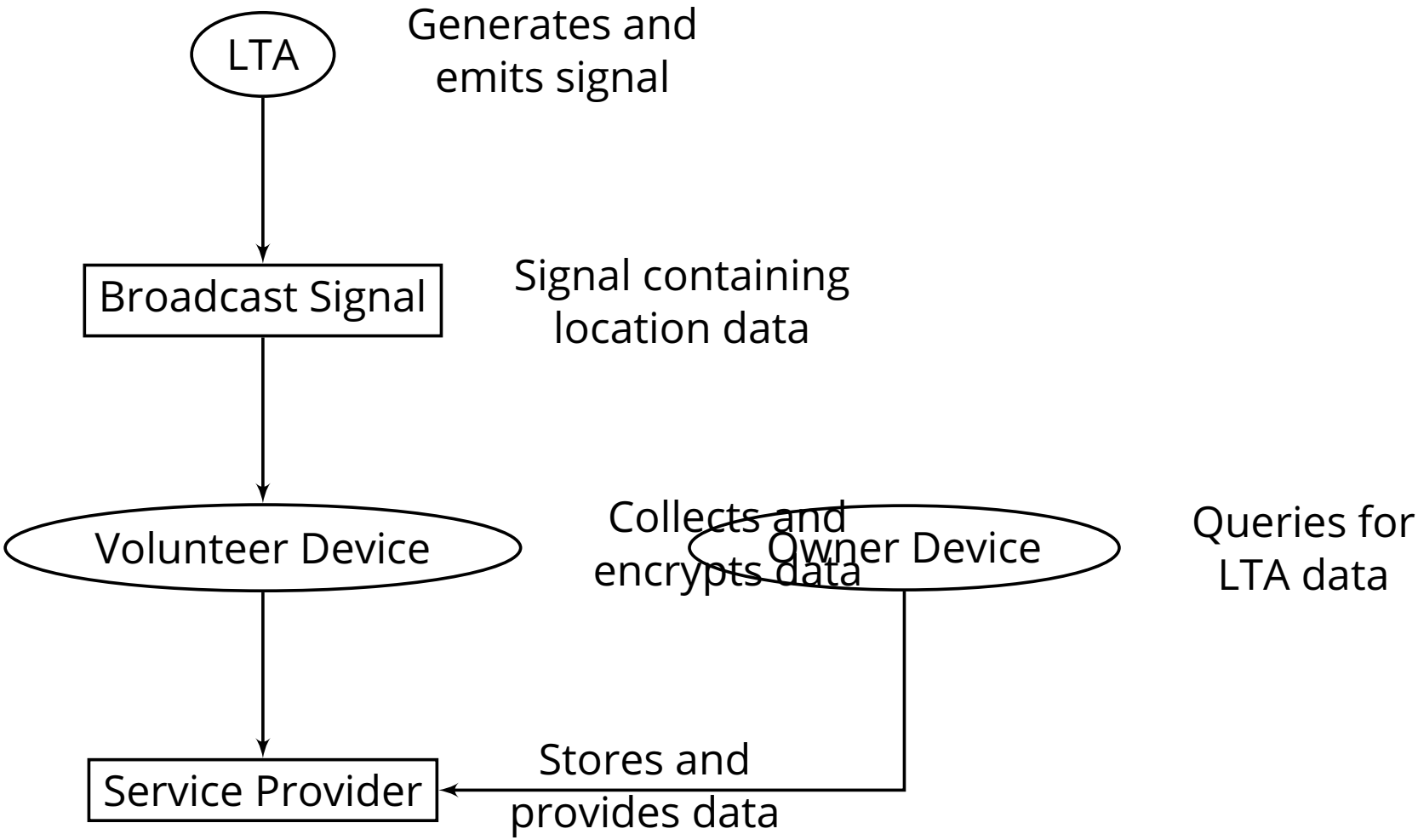
Contents

Contents

Privacy and Security in Location Tracking Accessories (LTAs)

Picture from: https://www.forbes.com/sites/forbes-personal-shopper/2023/11/30/best-luggage-trackers/

LTAs


LTA
Generates and emits signal
Broadcast Signal
Signal containing location data
Volunteer Device
Collects and encrypts data
Owner Device
Queries for LTA data
Service Provider
Stores and provides data

What can they be used for?

- Tracking personal items

What can they be used for?

- Tracking personal items
- Tracking your kids and pets

What can they be used for?

- Tracking personal items
- Tracking your kids and pets
- Tracking your vehicle

What can they be used for?

- Tracking personal items
- Tracking your kids and pets
- Tracking your vehicle
- Tracking PEOPLE! :(

Privacy and Stalking Risks associated with LTAs

- With tracking comes stalking (unfortunately)

Privacy and Stalking Risks associated with LTAs

- With tracking comes stalking (unfortunately)
- The challenge is therefore to design a system that both tracks and can be detected if stalking.

Privacy and Stalking Risks associated with LTAs

- With tracking comes stalking (unfortunately)
- The challenge is therefore to design a system that both tracks and can be detected if stalking.
- This is hard to do because you don’t want other people tracking your things either.

Contributions

- Development of Abuse-Resistant Private Offline Finding Protocols

Contributions

- Development of Abuse-Resistant Private Offline Finding Protocols
- Introduction of Multi-Dealer Secret Sharing (MDSS)

Contributions

- Development of Abuse-Resistant Private Offline Finding Protocols
- Introduction of Multi-Dealer Secret Sharing (MDSS)
- Exploration and Demonstration of Enhanced Privacy in Existing Tracking Protocols

Contributions

- Development of Abuse-Resistant Private Offline Finding Protocols
- Introduction of Multi-Dealer Secret Sharing (MDSS)
- Exploration and Demonstration of Enhanced Privacy in Existing Tracking Protocols
- Experimental Data Collection for Parameter Selection

Contributions

- Development of Abuse-Resistant Private Offline Finding Protocols
- Introduction of Multi-Dealer Secret Sharing (MDSS)
- Exploration and Demonstration of Enhanced Privacy in Existing Tracking Protocols
- Experimental Data Collection for Parameter Selection
- Proposing Protocol Optimizations

Contributions

- Development of Abuse-Resistant Private Offline Finding Protocols
- Introduction of Multi-Dealer Secret Sharing (MDSS)
- Exploration and Demonstration of Enhanced Privacy in Existing Tracking Protocols
- Experimental Data Collection for Parameter Selection
- Proposing Protocol Optimizations
- Implementation and Efficiency Demonstration

Contents

Current State of LTAs

- Rotate identifiers for privacy

Current State of LTAs

- Rotate identifiers for privacy
- Use secret keys to generate sequences of identifiers

Current State of LTAs

- Rotate identifiers for privacy
- Use secret keys to generate sequences of identifiers
- Stalker detection becomes harder because of rotating identifiers

Current State of LTAs

- Rotate identifiers for privacy
- Use secret keys to generate sequences of identifiers
- Stalker detection becomes harder because of rotating identifiers
- Attempts to balance privacy and stalker detection

Current State of LTAs

- Rotate identifiers for privacy
- Use secret keys to generate sequences of identifiers
- Stalker detection becomes harder because of rotating identifiers
- Attempts to balance privacy and stalker detection
- Limited against sophisticated tracking

Balancing User Privacy and Stalker Detection

Apple FindMy

Balancing User Privacy and Stalker Detection

Apple FindMy

- Pseudonym identifiers

Balancing User Privacy and Stalker Detection

Apple FindMy

- Pseudonym identifiers
- Secret key mechanism generating IDs

Balancing User Privacy and Stalker Detection

The Paper

Balancing User Privacy and Stalker Detection

The Paper

- Using MDDS and list-decodable error-correcting codes to enhance privacy

Balancing User Privacy and Stalker Detection

The Paper

- Using MDDS and list-decodable error-correcting codes to enhance privacy
- Better stalking detection

Balancing User Privacy and Stalker Detection

The Paper

- Using MDDS and list-decodable error-correcting codes to enhance privacy
- Better stalking detection
- Requires no more trust than Apple FindMy

Contents

Definitions

Definition (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 $\forall D$ provided by SP.

Definitions

Definition (Detectability)

A privacy preserving tracking protocol is detectable for predicate P' if $\forall$ valid cfg values, $\forall n.u.p.p.t$ algorithms $\mathcal{A}$, $\exists$ a negligible function $negl(\lambda)$ so that $\Pr[\mathsf{Exp}_{\mathcal{A}}^{\mathsf{Det},\mathsf{P}'}(\lambda, \mathsf{cfg}) = 0] \leq negl(\lambda)$.

Tag Detectability Experiment

$Q := \emptyset; L := []; B_{\mathsf{list}} := []; \mathsf{st} = \perp$
$\forall j \in [1, o = poly(\lambda)]:$
  $(\mathsf{id}, i, \mathsf{aux}, \mathsf{loc}, \mathsf{st}) \leftarrow \mathcal{A}(\mathsf{st}, \text{“query”})$
  If $(\mathsf{id}, *) \notin L:$
    $k_{\mathsf{tag}} \leftarrow \mathsf{KeyGen}(1^{\lambda}, \mathsf{cfg})$
    $L := L \cup \{(\mathsf{id}, k_{\mathsf{tag}})\}$
  find k_{tag} so that $(\mathsf{id}, k_{\mathsf{tag}}) \in L$
  $B \leftarrow \mathsf{Beacon}(k_{\mathsf{tag}}, i, \mathsf{aux})$
  $B_{\mathsf{list}} := B_{\mathsf{list}} \cup \{(B, \mathsf{loc})\}$
  $Q := Q \cup \{(\mathsf{id}, i)\}$
$\mathsf{out} \leftarrow \mathsf{Detect}(B_{\mathsf{list}})$
$b = 1$
$\forall (\mathsf{id}, k_{\mathsf{tag}}) \in L,$
  If $P'(\mathsf{cfg}, Q, \mathsf{id}) = 1,$
    $b' = (\exists i, (\mathsf{id}, i) \in Q \wedge$
      $\mathsf{GetTagID}(k_{\mathsf{tag}}, i) \in \mathsf{out})$
    $b = b \wedge b'$
return b

Figure 6: Experiment $\mathsf{Exp}_{\mathcal{A}}^{\mathsf{Det},P'}(\lambda, \mathsf{cfg})$.

Protocol Overview

- Extension of traditional secret sharing

Protocol Overview

- Extension of traditional secret sharing
- MDSS extends secret sharing by allowing multiple dealers, each with their own secrets, to distribute shares independently.

Protocol Overview

- Extension of traditional secret sharing
- MDSS extends secret sharing by allowing multiple dealers, each with their own secrets, to distribute shares independently.
- Unlinkability - It’s hard to determine if two shares comes from the same dealer or not.

Protocol Overview

- Extension of traditional secret sharing
- MDSS extends secret sharing by allowing multiple dealers, each with their own secrets, to distribute shares independently.
- Unlinkability - It’s hard to determine if two shares comes from the same dealer or not.
- Multi-Dealer correctness - Secrets can be accurately reconstructed even with multiple dealers

List-Decodable Error-Correcting Codes

List-Decodable Error-Correcting Codes

Challenge: Stalker-detection algorithms must determine if an LTA is persistently nearby.

List-Decodable Error-Correcting Codes

Challenge: Stalker-detection algorithms must determine if an LTA is persistently nearby.

Problem: When LTAs change their identifier, linking a sequence of broadcasts to a single device is difficult.

List-Decodable Error-Correcting Codes

Challenge: Stalker-detection algorithms must determine if an LTA is persistently nearby.

Problem: When LTAs change their identifier, linking a sequence of broadcasts to a single device is difficult.

Solution: List-Decodable Error-Correcting Codes

List-Decodable Error-Correcting Codes

Purpose: To link sequences of broadcasts from a single device, even when identifiers change.

List-Decodable Error-Correcting Codes

Purpose: To link sequences of broadcasts from a single device, even when identifiers change.

Function: These codes can decode a larger number of errors than standard error-correcting codes.

List-Decodable Error-Correcting Codes

Purpose: To link sequences of broadcasts from a single device, even when identifiers change.

Function: These codes can decode a larger number of errors than standard error-correcting codes.

Application in LTAs: Enables stalker-detection algorithms to function effectively despite identifier rotation.

List-Decodable Error-Correcting Codes

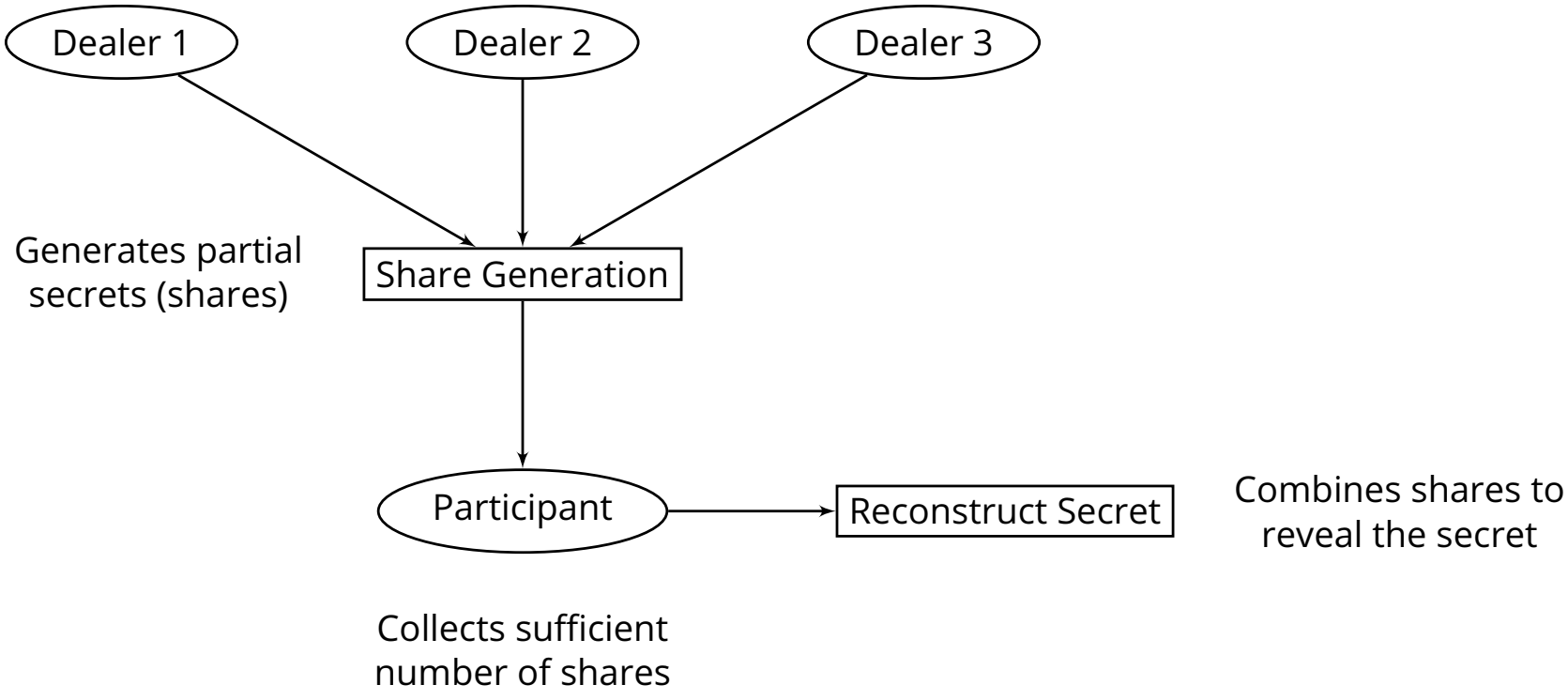
Purpose: To link sequences of broadcasts from a single device, even when identifiers change.

Function: These codes can decode a larger number of errors than standard error-correcting codes.

Application in LTAs: Enables stalker-detection algorithms to function effectively despite identifier rotation.

Benefit: Improves the balance between user privacy (through identifier rotation) and effective stalker detection.

Multi Dealer Secret Sharing (MDSS)



MDSS Construction

Share(s, I')

MDSS Construction

Share(s, I')

1. Sample c polynomials $p_1, \ldots, p_c$, where $s = p_1(0)|| \ldots ||p_c(0)$.

MDSS Construction

Share(s, I')

1. Sample c polynomials $p_1, \ldots, p_c$, where $s = p_1(0)|| \ldots ||p_c(0)$.
2. Sample I' field elements $x_1, \ldots, x_{|I'|} \leftarrow \mathbb{F}$.

MDSS Construction

Share(s, I')

1. Sample c polynomials $p_1, \ldots, p_c$, where $s = p_1(0)|| \ldots ||p_c(0)$.
2. Sample I' field elements $x_1, \ldots, x_{|I'|} \leftarrow \mathbb{F}$.
3. Return $\{(x_i, p_1(x_i), \ldots, p_c(x_i))\}_{i \in I'}$.

MDSS Construction

Share(s, I')

1. Sample c polynomials $p_1, \ldots, p_c$, where $s = p_1(0)|| \ldots ||p_c(0)$.
2. Sample I' field elements $x_1, \ldots, x_{|I'|} \leftarrow \mathbb{F}$.
3. Return $\{(x_i, p_1(x_i), \ldots, p_c(x_i))\}_{i \in I'}$.

Reconstruct($\{sh_1, \ldots, sh_{max}\}$)

MDSS Construction

Share(s, I')

1. Sample c polynomials $p_1, \ldots, p_c$, where $s = p_1(0)|| \ldots ||p_c(0)$.
2. Sample I' field elements $x_1, \ldots, x_{|I'|} \leftarrow \mathbb{F}$.
3. Return $\{(x_i, p_1(x_i), \ldots, p_c(x_i))\}_{i \in I'}$.

Reconstruct($\{sh_1, \ldots, sh_{max}\}$)

1. Return $CH^* - MDSS(\{sh_1, \ldots, sh_w\})$

CH* Construction

- What does it do? - Lattice based polynomial recovery

CH* Construction

- What does it do? - Lattice based polynomial recovery
- Input: Set of evaluation points

CH* Construction

- What does it do? - Lattice based polynomial recovery
- Input: Set of evaluation points
- Output: Returns solution or error

CH* Construction

- What does it do? - Lattice based polynomial recovery
- Input: Set of evaluation points
- Output: Returns solution or error
- How?

CH* Construction

- What does it do? - Lattice based polynomial recovery
- Input: Set of evaluation points
- Output: Returns solution or error
- How?
  1. Uses interpolation to construct polynomials from input

CH* Construction

- What does it do? - Lattice based polynomial recovery
- Input: Set of evaluation points
- Output: Returns solution or error
- How?
  1. Uses interpolation to construct polynomials from input
  2. Creates a lattice basis

CH* Construction

- What does it do? - Lattice based polynomial recovery
- Input: Set of evaluation points
- Output: Returns solution or error
- How?
  1. Uses interpolation to construct polynomials from input
  2. Creates a lattice basis
  3. Reduces the basis

CH* Construction

- What does it do? - Lattice based polynomial recovery
- Input: Set of evaluation points
- Output: Returns solution or error
- How?
  1. Uses interpolation to construct polynomials from input
  2. Creates a lattice basis
  3. Reduces the basis
  4. Solution is based on the shortest vector

Algorithm 1: CH*

Input : $\lambda, k, n, \{(\alpha_i, \beta_{i,1}, \ldots, \beta_{i,c})\}_{i=1}^{n}$
Output: None, Multiple, or a single solution $(p_1 \ldots p_c)$

forall $j \in [c]$ **do**
  $f_j(z) = \mathsf{LagrInterpol}(\{(\alpha_1, \beta_{1,j}), \ldots, (\alpha_n, \beta_{n,j})\})$
end
$N(z) = \prod_{i=1}^{n}(z - \alpha_i)$
construct the matrix $M \in \mathbb{F}[z]^{(c+1)\times(c+1)}$:

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

$M_{\mathsf{red}} \leftarrow \mathsf{LatticeReduce}(M)$
if *there is one shortest vector* $\vec{v} = (v_0 \ldots v_c)$ *that has a length* $\leq \lambda$ **then**
  return $(v_1 \cdot z^k/v_0, \ldots, v_c \cdot z^k/v_0)$
end
if *there are multiple vectors with the shortest length and the shortest length* $\leq \lambda$ **then**
  return Multiple
else
  return None
end

CH*-MDSS Construction

- What does it do? - Extends CH* for MDSS (multiple valid solutions)

CH*-MDSS Construction

- What does it do? - Extends CH* for MDSS (multiple valid solutions)
- Output: List of solutions or error

CH*-MDSS Construction

- What does it do? - Extends CH* for MDSS (multiple valid solutions)
- Output: List of solutions or error
- How?

CH*-MDSS Construction

- What does it do? - Extends CH* for MDSS (multiple valid solutions)
- Output: List of solutions or error
- How?
  1. Applies CH* iteratively

CH*-MDSS Construction

- What does it do? - Extends CH* for MDSS (multiple valid solutions)
- Output: List of solutions or error
- How?
  1. Applies CH* iteratively
  2. If multiple solutions are found, modify ws to isolate a single solution

CH*-MDSS Construction

- What does it do? - Extends CH* for MDSS (multiple valid solutions)
- Output: List of solutions or error
- How?
  1. Applies CH* iteratively
  2. If multiple solutions are found, modify ws to isolate a single solution
  3. Does this with random deletion of coordinates

Algorithm 2: CH*-MDSS Construction

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

```
1  solns := [], ws := [n], fail := False
2  while |ws| ≥ t and not fail do
3      res ← CH*(k + (|ws| − t), k, n, {(α_i, β_{i,1} ... β_{i,c})}_{i∈ws}).
4      if res = None then
5          return solns
6      else if res = Multiple then
7          s ← ⌊|ws| / t⌋, found ← False
8          while not found: do
               // randomly remove points until a single solution is found
9              pts ← ws.pop(2(s − 1))
10             res ← CH*(k + (|ws| − t), k, n, {(α_i, β_{i,1} ... β_{i,c})}_{i∈ws})
11             ws ← ws ∪ pts
12             if res can be parsed as (p_1 ... p_c) then
13                 found ← True
14                 add (p_1 ... p_c) to solns, remove agreeing points from ws
15             end
16         endw
17     else
18         parse res as (p_1 ... p_c), add (p_1 ... p_c) to solns, remove agreeing points from ws
19     end
20 endw
21 return solns
```

Abuse-resistant Offline Finding Protocol

$\mathsf{KeyGen}(1^\lambda, \mathsf{cfg}) \to k_{\mathsf{tag}}$ In: required params, out: a secret key.

Abuse-resistant Offline Finding Protocol

$\mathsf{KeyGen}(1^\lambda, \mathsf{cfg}) \to k_{\mathsf{tag}}$ In: required params, out: a secret key.

$\mathsf{Beacon}(i_{\mathsf{tag}}, i_{\mathsf{epoch}}, \mathsf{aux}) \to B$ In: tag key, epoch and aux. Out: a broadcast message

Abuse-resistant Offline Finding Protocol

$\mathsf{KeyGen}(1^\lambda, \mathsf{cfg}) \to k_{\mathsf{tag}}$ In: required params, out: a secret key.

$\mathsf{Beacon}(i_{\mathsf{tag}}, i_{\mathsf{epoch}}, \mathsf{aux}) \to B$ In: tag key, epoch and aux. Out: a broadcast message

$\mathsf{GetTagID}(i_{\mathsf{tag}}, i_{\mathsf{epoch}}) \to \mathsf{id}_{\mathsf{tag}}$ Helper algorithm used by the LTA and owner device to find the current identifier of the LTA

Abuse-resistant Offline Finding Protocol

$\mathsf{KeyGen}(1^\lambda, \mathsf{cfg}) \to k_{\mathsf{tag}}$ In: required params, out: a secret key.

$\mathsf{Beacon}(i_{\mathsf{tag}}, i_{\mathsf{epoch}}, \mathsf{aux}) \to B$ In: tag key, epoch and aux. Out: a broadcast message

$\mathsf{GetTagID}(i_{\mathsf{tag}}, i_{\mathsf{epoch}}) \to \mathsf{id}_{\mathsf{tag}}$ Helper algorithm used by the LTA and owner device to find the current identifier of the LTA

$\mathsf{GenReport}(B, \mathsf{loc}) \to R$ In: A broadcast and location, out: a report.

Abuse-resistant Offline Finding Protocol

$\mathsf{KeyGen}(1^\lambda, \mathsf{cfg}) \to k_{\mathsf{tag}}$ In: required params, out: a secret key.

$\mathsf{Beacon}(i_{\mathsf{tag}}, i_{\mathsf{epoch}}, \mathsf{aux}) \to B$ In: tag key, epoch and aux. Out: a broadcast message

$\mathsf{GetTagID}(i_{\mathsf{tag}}, i_{\mathsf{epoch}}) \to \mathsf{id}_{\mathsf{tag}}$ Helper algorithm used by the LTA and owner device to find the current identifier of the LTA

$\mathsf{GenReport}(B, \mathsf{loc}) \to R$ In: A broadcast and location, out: a report.

$\mathsf{Detect}(\mathsf{cfg}, \{(B_1, \mathsf{loc}_1), \ldots, (B_n, \mathsf{loc}_n)\}) \to \{\mathsf{id}_{\mathsf{tag}_i}\}$ In: set of broadcasts, out: one or more identifiers.

Abuse-resistant Offline Finding Protocol

$\mathsf{KeyGen}(1^\lambda, \mathsf{cfg}) \to k_{\mathsf{tag}}$ In: required params, out: a secret key.

$\mathsf{Beacon}(i_{\mathsf{tag}}, i_{\mathsf{epoch}}, \mathsf{aux}) \to B$ In: tag key, epoch and aux. Out: a broadcast message

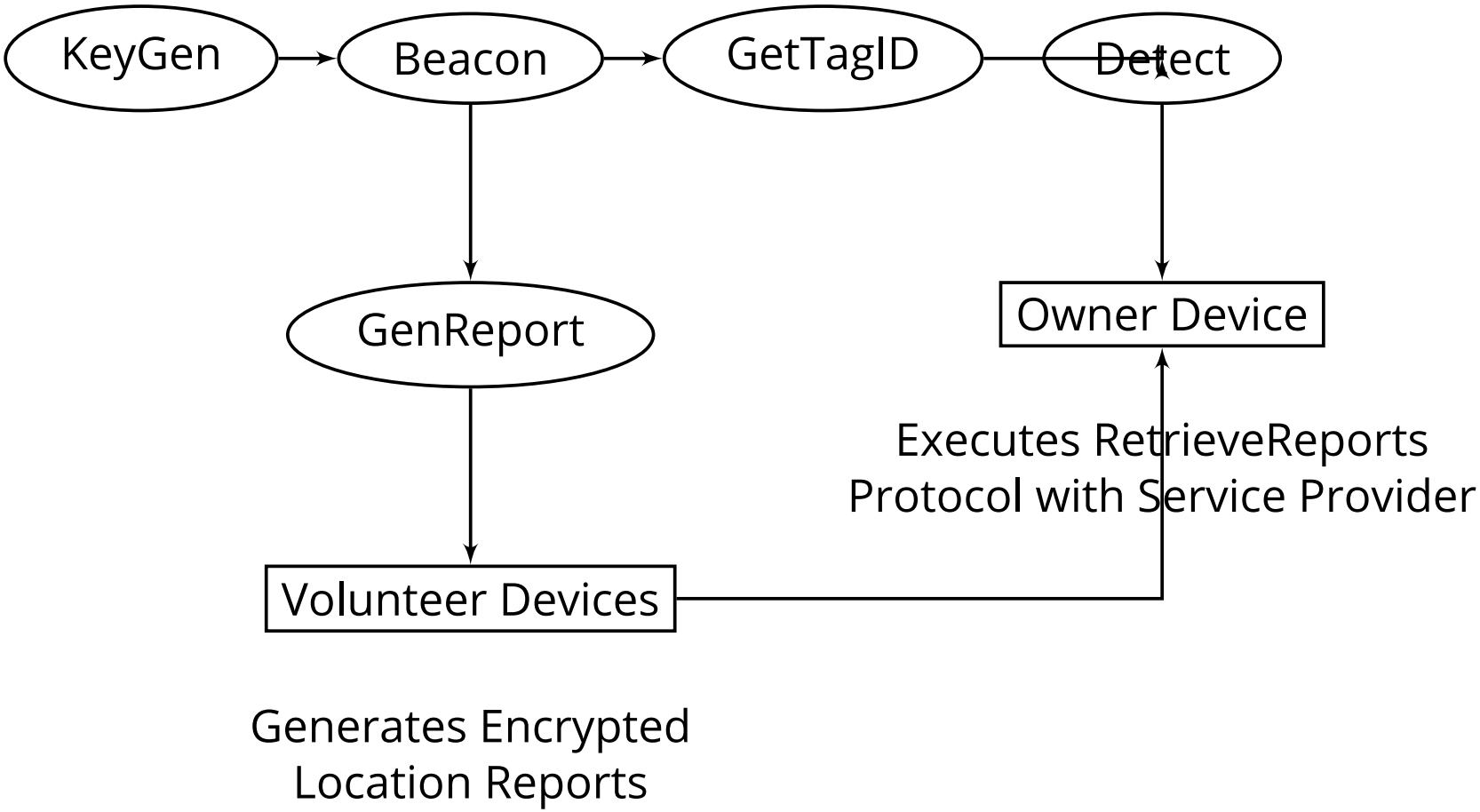
$\mathsf{GetTagID}(i_{\mathsf{tag}}, i_{\mathsf{epoch}}) \to \mathsf{id}_{\mathsf{tag}}$ Helper algorithm used by the LTA and owner device to find the current identifier of the LTA

$\mathsf{GenReport}(B, \mathsf{loc}) \to R$ In: A broadcast and location, out: a report.

$\mathsf{Detect}(\mathsf{cfg}, \{(B_1, \mathsf{loc}_1), \ldots, (B_n, \mathsf{loc}_n)\}) \to \{\mathsf{id}_{\mathsf{tag}_i}\}$ In: set of broadcasts, out: one or more identifiers.

$\mathsf{RetrieveReports}(\mathsf{Owner}(k_{\mathsf{tag}}, i_{\mathsf{epoch}}), \mathsf{SP}(\mathcal{D}))$ Executed between 2 parties, Owner provides a tag key and epoch, SP uses a database $\mathcal{D}$, ouputting a list of reports.

Offline Finding Protocol Flow



Main Construction: KeyGen(cfg)

1. $k_1 \leftarrow \mathsf{PRF}_1.\mathsf{KeyGen}(1^\lambda)$

Main Construction: KeyGen(cfg)

1. $k_1 \leftarrow \mathsf{PRF}_1.\mathsf{KeyGen}(1^\lambda)$
2. $k_2 \leftarrow \mathsf{PRF}_2.\mathsf{KeyGen}(1^\lambda)$

Main Construction: KeyGen(cfg)

1. $k_1 \leftarrow \mathsf{PRF}_1.\mathsf{KeyGen}(1^\lambda)$
2. $k_2 \leftarrow \mathsf{PRF}_2.\mathsf{KeyGen}(1^\lambda)$
3. $k_3 \leftarrow \mathsf{PRF}_3.\mathsf{KeyGen}(1^\lambda)$

Main Construction: KeyGen(cfg)

1. $k_1 \leftarrow \mathsf{PRF}_1.\mathsf{KeyGen}(1^\lambda)$
2. $k_2 \leftarrow \mathsf{PRF}_2.\mathsf{KeyGen}(1^\lambda)$
3. $k_3 \leftarrow \mathsf{PRF}_3.\mathsf{KeyGen}(1^\lambda)$
4. return $(k_1, k_2, k_3, \mathsf{cfg})$

Main Construction: $\mathsf{Beacon}(k_{\mathsf{tag}}, i_{\mathsf{epoch}}, \mathsf{aux})$

1. $(k_1, k_2, k_3, \mathsf{cfg}) \leftarrow k_{\mathsf{tag}}$

Main Construction: Beacon$(k_{\mathsf{tag}}, i_{\mathsf{epoch}}, \mathsf{aux})$

1. $(k_1, k_2, k_3, \mathsf{cfg}) \leftarrow k_{\mathsf{tag}}$

2. $(E, L) \leftarrow \mathsf{cfg}$

Main Construction: Beacon$(k_{\mathsf{tag}}, i_{\mathsf{epoch}}, \mathsf{aux})$

1. $(k_1, k_2, k_3, \mathsf{cfg}) \leftarrow k_{\mathsf{tag}}$
2. $(E, L) \leftarrow \mathsf{cfg}$
3. $(\mathsf{pk}, \_) \leftarrow \mathsf{CCA.KeyGen}(1^\lambda; \mathsf{PRF.Eval}(k_1, i))$

Main Construction: Beacon$(k_{\mathsf{tag}}, i_{\mathsf{epoch}}, \mathsf{aux})$

1. $(k_1, k_2, k_3, \mathsf{cfg}) \leftarrow k_{\mathsf{tag}}$
2. $(E, L) \leftarrow \mathsf{cfg}$
3. $(\mathsf{pk}, \_) \leftarrow \mathsf{CCA.KeyGen}(1^\lambda; \mathsf{PRF.Eval}(k_1, i))$
4. $id_{\mathsf{tag}} \leftarrow \mathsf{GetTagID}(k_{\mathsf{tag}}, i_{\mathsf{epoch}})$

Main Construction: Beacon$(k_{\mathsf{tag}}, i_{\mathsf{epoch}}, \mathsf{aux})$

1. $(k_1, k_2, k_3, \mathsf{cfg}) \leftarrow k_{\mathsf{tag}}$
2. $(E, L) \leftarrow \mathsf{cfg}$
3. $(\mathsf{pk}, \_) \leftarrow \mathsf{CCA.KeyGen}(1^\lambda; \mathsf{PRF.Eval}(k_1, i))$
4. $id_{\mathsf{tag}} \leftarrow \mathsf{GetTagID}(k_{\mathsf{tag}}, i_{\mathsf{epoch}})$
5. $e \leftarrow \left\lfloor \frac{i_{\mathsf{epoch}}}{L} \right\rfloor, i \leftarrow i_{\mathsf{epoch}}(\mod L)$

Main Construction: $\mathsf{Beacon}(k_{\mathsf{tag}}, i_{\mathsf{epoch}}, \mathsf{aux})$

1. $(k_1, k_2, k_3, \mathsf{cfg}) \leftarrow k_{\mathsf{tag}}$
2. $(E, L) \leftarrow \mathsf{cfg}$
3. $(\mathsf{pk}, \_) \leftarrow \mathsf{CCA.KeyGen}(1^\lambda; \mathsf{PRF.Eval}(k_1, i))$
4. $id_{\mathsf{tag}} \leftarrow \mathsf{GetTagID}(k_{\mathsf{tag}}, i_{\mathsf{epoch}})$
5. $e \leftarrow \left\lfloor \frac{i_{\mathsf{epoch}}}{L} \right\rfloor, i \leftarrow i_{\mathsf{epoch}}(\mod L)$
6. $I := \{0, \ldots, L-1\}$

Main Construction: $\mathsf{Beacon}(k_{\mathsf{tag}}, i_{\mathsf{epoch}}, \mathsf{aux})$

1. $(k_1, k_2, k_3, \mathsf{cfg}) \leftarrow k_{\mathsf{tag}}$
2. $(E, L) \leftarrow \mathsf{cfg}$
3. $(\mathsf{pk}, \_) \leftarrow \mathsf{CCA.KeyGen}(1^\lambda; \mathsf{PRF.Eval}(k_1, i))$
4. $id_{\mathsf{tag}} \leftarrow \mathsf{GetTagID}(k_{\mathsf{tag}}, i_{\mathsf{epoch}})$
5. $e \leftarrow \left\lfloor \frac{i_{\mathsf{epoch}}}{L} \right\rfloor, i \leftarrow i_{\mathsf{epoch}}(\mod L)$
6. $I := \{0, \ldots, L-1\}$
7. $\mathsf{sh}_0^e \ldots \mathsf{sh}_{L-1}^e \leftarrow \mathsf{Share}(\mathsf{id}_{\mathsf{tag}}, I; \mathsf{PRG}(\mathsf{PRF}_2.\mathsf{Eval}(k_2, e)))$

Main Construction: Beacon$(k_{\mathsf{tag}}, i_{\mathsf{epoch}}, \mathsf{aux})$

1. $(k_1, k_2, k_3, \mathsf{cfg}) \leftarrow k_{\mathsf{tag}}$
2. $(E, L) \leftarrow \mathsf{cfg}$
3. $(\mathsf{pk}, \_) \leftarrow \mathsf{CCA.KeyGen}(1^\lambda; \mathsf{PRF.Eval}(k_1, i))$
4. $id_{\mathsf{tag}} \leftarrow \mathsf{GetTagID}(k_{\mathsf{tag}}, i_{\mathsf{epoch}})$
5. $e \leftarrow \left\lfloor \frac{i_{\mathsf{epoch}}}{L} \right\rfloor, i \leftarrow i_{\mathsf{epoch}}(\mod L)$
6. $I := \{0, \ldots, L-1\}$
7. $\mathsf{sh}_0^e \ldots \mathsf{sh}_{L-1}^e \leftarrow \mathsf{Share}(\mathsf{id}_{\mathsf{tag}}, I; \mathsf{PRG}(\mathsf{PRF}_2.\mathsf{Eval}(k_2, e)))$
8. Return $\mathsf{pk}||\mathsf{sh}_i^e||\mathsf{aux}$

Main Construction: Detect$(\mathsf{cfg}, \{B_j\})$

1. $S := \{\mathsf{sh} | (* || \mathsf{sh} || *) \in \{B_j\}\}$

Main Construction: Detect$(\mathsf{cfg}, \{B_j\})$

1. $S := \{\mathsf{sh} | (* || \mathsf{sh} || *) \in \{B_j\}\}$
2. Return $\Pi.\mathsf{Reconstruct}(S)$

Contents

Achieving Privacy Goals

- Unlinkability and threshold privacy from MDSS

Achieving Privacy Goals

- Unlinkability and threshold privacy from MDSS
- Ensure identifiers change periodically

Achieving Privacy Goals

- Unlinkability and threshold privacy from MDSS
- Ensure identifiers change periodically
- Should need a copy of the secret key to re-derive the correct sequence to obtain location reports

Stalker Detection

- Each LTA maintains a detection period consisting of L consecutive time epochs.

Stalker Detection

- Each LTA maintains a detection period consisting of L consecutive time epochs.
- At the beginning of each detection period LTA generates $\mathrm{id}_{\mathrm{tag}}$. Then it secret-shares this using an MDSS scheme

Stalker Detection

- Each LTA maintains a detection period consisting of L consecutive time epochs.
- At the beginning of each detection period LTA generates $\mathrm{id}_{\mathrm{tag}}$. Then it secret-shares this using an MDSS scheme
- LTA generates pseudonym for each Beacon call and appends one secret share to be broadcast by LTA.

Stalker Detection

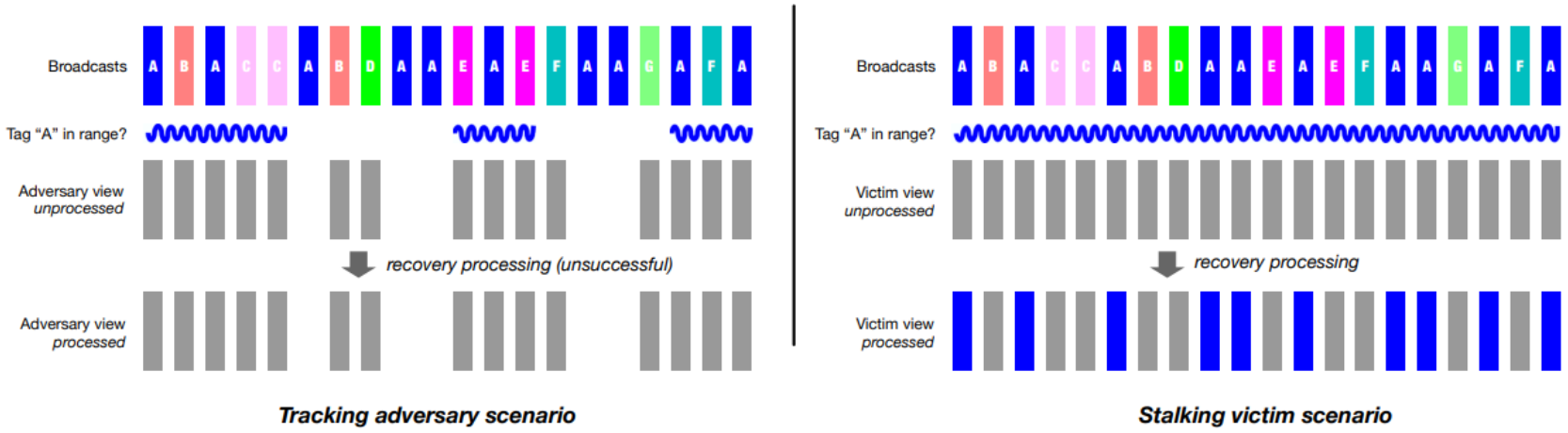
- Each LTA maintains a detection period consisting of L consecutive time epochs.
- At the beginning of each detection period LTA generates $\mathrm{id}_{\mathrm{tag}}$. Then it secret-shares this using an MDSS scheme
- LTA generates pseudonym for each Beacon call and appends one secret share to be broadcast by LTA.
- These secret shares are only used locally for stalker detection.

Contents

FindMy vs. This Construction

Protocol	Epoch duration	Broadcasts per epoch	Stalker detection?	Tracking privacy	Continuous Proximity	Stalker detection
Apple FindMy [2] / IETF [32]:						
Near-owner mode	15 min	450	✗	n/a	n/a	
Separated mode	24 hrs	43,200	●	n/a	✗	15-60 min†
This work (§4):						
2-second epochs / 1-hour window	2 sec	1	●	40 – 46 min*	●	60 min
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

2 Detection Scenarios



Questions?