MutualAid.world
Secure Location Intersection

Goal

Tell if two people were near the same places and times… without knowing where else they were.

Background

In some areas, you can prove that your age is over a certain number, without letting the other person know how old you actually are. This is a key piece of cryptographic technology (T), that can tell if one number (a) is greater-than-or-equal-to another number (b), using only the encrypted (g) version of the numbers (g(a) and g(b)). It is mathematically summarized as:

$$(a >= b) = T(g(a), g(b))$$

Summary of Solution

We will store multiple location dimensions, each one expressed as ordered min,max bounds, each value encrypted, and already including both the intersection radius and a noise factor:

$$shared = [\ (\ min = g(x - r * s_1),\ \ max = g(x + r * s_2)\ )\ ]$$

$$near(a,b) = \prod_i T(a_i.max, b_i.min)\ \&\ T(b_i.max, a_i.min)$$

$$|a - b| \le r \quad \Rightarrow \quad near(a,b) = true$$

$$|a - b| > 2r \quad \Rightarrow \quad near(a,b) = false$$

Where an individual dimension (x) is encrypted (g) with intersection radius (r) and random scale (s, between 1 and $\sqrt{2}$), into a vector (shared) of encrypted dimensions with min and max values. The intersection (near) is then computable using only greater-than-or-equal-to operators over encrypted bounds, and combined using traditional AND operators thereafter.

Evolution of Solution

1. No privacy: Share the centers of users movement (x), and calculate if the centers get near each other by a radius (r).
2. Basic privacy in 1D: encrypt the bounds of their movement, and calculate if the encrypted bounds intersect:

   $a = \{ min = g(x - r), max = g(x + r) \}$

   $test = T(a.max, b.min)\ \&\ T(b.max,\ a.min)$

3. Better privacy in 1D: add some random noise to avoid sharing the same number encoded with two different offsets. It would significantly easier to reconstruct x if we two samples of it encrypted with and without r:
   $a = \{ min = g(x - r * s_1), max = g(x + r * s_1) \}$
   $s = random(1, \sqrt{2})$
   We use random number (s) between 1 and $\sqrt{2} \approx 1.414$ to add noise, ensure that intersections are caught, but limit the scope of false-positives. Why $\sqrt{2}$? read on...
4. Expanding to 2D (lat, lon), 3D (lat, lon, time), etc.: the above intersection only works in one dimension, but our users GPS is in 2D, and over time that's in 3D. So we need to do the same in multiple dimensions:
   $a.X = \{ min = g(X - r), max = g(X + r) \}, \; a.Y = \{ min = g(Y - r), max = g(Y + r) \}$
   $test = (T(a.X.max, b.X.min) \;\&\; T(b.X.max, \; a.X.min)) \;\&\; (T(a.Y.max, b.Y.min) \;\&\; T(b.Y.max, \; a.Y.min))$
   I.e. one can simply test for intersection in each dimension and then combine the results. Because this is a square intersection rather than a round one ("Manhattan distance" rather than "as the crow flies"), it will on average add $\sqrt{2}$ to the radius. We combine this with a random offset of 1 to $\sqrt{2}$, giving us a total multiplied error of at most 2*r.
5. Levels of Scale: because the above encrypting mechanics mostly work on positive-integers, we will use cascading levels of detail to express higher precision value.
6. Expanding to Semantic Tags: to support semantics tags, without giving out which tags are being used by the user, we will provide a standard dictionary of string semantic tag ("home", "store", etc.) to small integer ranges, so that a min-max pair can be constructed from different numbers, and there enough space between ranges to avoid false positives. For example:
   $semantic_i = 3 + 5i$
   $semantic(\text{"}home\text{"}) \Rightarrow 3 \; (range \; 1...5)$
   $semantic(\text{"}store\text{"}) \Rightarrow 8 \; (range \; 6...10)$
   This way, each semantic can be stored as a random range, with a radius (r) of 1, and max error of 2, meaning the encoded ranges would be between 1 and 5 for home, and 6 and 12 or store, which may overlap

Credentials and Third Party Validation

The larger social solution, without the spatial intersection above, is expressed in the W3C's DID (Decentralized Identity Document) format:
https://www.w3.org/TR/did-core/
In particular it covers the aspect of Verifiable Credentials, that solve a key problem in the above which is that we are assuming all parties encrypt valid information in the first place.
https://www.w3.org/TR/vc-data-model/
In short the longer term solution is that user A, and have known valid source B (such as the transit authority), signed a "verifiable credential" that user A was indeed at a particular location at a certain time. User A and can then present this credential to user C, who can verify:

- That it came from authority B, is about A, is well formed, etc.
- That it keep math criterion (like the nearby calculation etc.)
- Without being able to read the unencrypted version of the credential (i.e. know all the other places that A was)

-Lewey Geselowitz
Lead Software Engineer @ Unity3D
Director of Spatial Standards @ The Spatial Web Foundation