Location location

Leveraging the crowdsourcing network in emergencies does not necessarily require the system to process or store user locations. For instance, if Person X in Area A is looking for an autoinjector and sends a request to the service without any location information, the system can simply send contact info for Person X to all other users, or contact info for all other users to Person X, and allow the location sharing to happen via non-system communication methods. However, even with a very small number of users this is highly impractical and provides very little benefit.

On the other hand, if the system can process user locations, then the service can become more streamlined and useful. Suppose the service does not persistently store any user location information. If Person X in Area A sends a request to find an autoinjector with their current location information, then all other users on the network can be notified of that location and opt to respond if they are able. This method is much better than sharing no location information with the service at all, yet, if there are n users on the network, $n-1$ users will always be notified for every emergency, regardless of their location or proximity. This protocol creates many useless notifications and still requires $n-1$ users to manually check if they are nearby the emergency. The number of notifications involved and work required disincentivizes users from taking notifications seriously.

Persistent system knowledge of all users' locations, however, can improve the service drastically: when Person X requests an autoinjector, the system can only send notifications to users within a reasonable proximity to Person X. Users who receive a notification will immediately know they are within reasonable distance of someone in need of an autoinjector and can opt to respond. This protocol will be taken more seriously be users and does not require any unreasonable communication of users' locations. However, storing users' locations comes at a price: privacy. Users might be reluctant or unwilling to have their locations stored in or processed by the system. If the system did store user locations and became compromised, location information would be exposed. In the next section, we discuss different methods to store user locations while providing some degree of anonymity in the event of a breach and still retaining the usefulness of the location data.

Location Privacy

In the context of protecting privacy for the public or private release of any data set for purposes of research, Sweeney sweeney presents a model known as k-anonymity, where any data set released "provides k-anonymity protection if the information for each person contained in the release cannot be distinguished from at least $k-1$ other individuals in the data set" sweeney. This model can be adopted to the storage and processing of user information in network databases. If a database is k-anonymized and later breached, then the users whose information was compromised still retain some form of privacy: namely, anonymity amongst k other users.

Gedik and Liu gedik describe a k-anonymity model for protecting user privacy in mobile systems that allows for each user to indicate the level of k-anonymity desired for each message sent to a particular service. The k-anonymity is achieved by either decreasing location accuracy until $k-1$ other users share the same spatial location or delaying message processing until $k-1$ other messages have been received from the same spatial region.

Cheng et. al Cheng2006 present a different scheme that uses a more probabilistic approach. They argue that k-anonymity may not be used if there are fewer than k users in the system, and that even if there are more than k users, "they may span in a large area over an extended time period, in which case the cloaked location can be very large and cause a severe degradation of service quality." They instead provide a framework that stores cloaked geometric areas for each user, where the users' true location is somewhere in the area stored. The framework lets each user decide on the size and boundaries of the cloaked location stored, and the larger the area, the more anonymity. k-anonymity may be achieved as a side effect in this framework if cloaked locations overlap, but it does not dictate how locations are anonymized. Location-based search queries find cloaked areas that overlap with a given query range, and queries have a runtime in the worst case of $O(e^2 \log e)$ and best case of $O(m+e)$, where e is the number of sides of the geometric shapes stored and m is the number of sides of any polygon used for the query computation.

These frameworks (and others) have their advantages, but neither is the best fit for this particular application. k-anonymity is not a good option for reasons described by Cheng et. al Cheng2006: if there are too few users within a given region, then the size of the cloaked location stored would be too large to provide

useful data to the service. Similarly, letting each user specify the size of the cloaked region might cause the service to run into problems as discussed previously if they specify a large area. Instead, we present the following location privacy contract to the user of the application and an implementation differing from those above to achieve it.

Data Privacy Contract

enumerate

A user's exact location will never be stored in any system database.

F or any user $u_i \in U$, let the stored location of that user in the system be $L(u_i)$. The granularity of $L(u_i)$ will be no less than 1.4 km^2, where $Area(L(u_i)) \geq 1.4\ km^2\ \ \forall\ i$, and the true location of user u_i will be uniformly distributed across $Area(L(u_i))$.

O nly the most recently received location will be stored for each user and location history will never be stored.

U ser locations will be stored separately from other user information, including user IDs. The IDs for each user location entry will be hashed using a secure hash function. An adversary would need the user ID, hash function, and entire location table to find a specific user's entry.