Paper **four**

LOCATION-ENHANCED AUTHENTICATION USING THE IOT

Because You Cannot Be in Two Places at Once

IOANNIS AGADAKOS, PER A. HALLGREN, DIMITRIOS DAMOPOULOS,
ANDREI SABELFELD AND GEORGIOS PORTOKALIDIS

User location can act as an additional factor of authentication in scenarios where physical presence is required, such as when making in-person purchases or unlocking a vehicle. This paper proposes a novel approach for estimating user location and modeling user movement using the Internet of Things (IoT). Our goal is to utilize its scale and diversity to estimate location more robustly, than solutions based on smartphones alone, and stop adversaries from using compromised user credentials (e.g., stolen keys, passwords, etc.), when sufficient evidence physically locates them elsewhere. To locate users, we leverage the increasing number of IoT devices carried and used by them and the smart environments that observe these devices. We also exploit the ability of many IoT devices to "sense" the user. To demonstrate our approach, we build a system, called Icelus. Our experiments with it show that it exhibits a smaller false-rejection rate than smartphone-based location-based authentication (LBA) and it rejects attackers with few errors (i.e., false acceptances).

PUBLISHED IN THE 32ND CONFERENCE ON COMPUTER SECURITY APPLICATIONS
ACSAC 2016, LOS ANGELES, CA, USA, DECEMBER 5-9, 2016

1 Introduction

Electronic user authentication is increasingly used in the physical world, where it is frequently employed to protect financial transactions and to control access to physical spaces and vehicles. Typical means to authenticate users entry include passwords and PIN codes, tokens (e.g., smartcards), and biometrics (e.g., fingerprints). Cards are frequently used to unlock doors, mainly in offices, either through swiping the card through a reader or by proximity of an RFID-based card to the reader. Smart locks (e.g., Kevo) enable user's to use their smartphone instead of a key, while an increasing number of vehicles use wireless key fobs to unlock their doors and start the engine. Credit and debit cards, and even smartphones today, also act as tokens that (usually) along with a PIN code enable users to authorize transactions.

While these advances have improved convenience and even security, they are not without problems. Fraudsters engage in various forms of deception for financial gain, like in Japan where $13m were stolen from ATMs [24]. The methods employed involve stealing, cloning, and counterfeiting credit and debit cards to perform transactions at POS [5, 43]. In the US such attacks are bolstered also by the limited deployment of PIN and chip technology [28]. Systems like Android Pay and Apple Pay, that enable users to pay with their smartphones or smart-wearables, can also be compromised if the PIN code [13, 52] or biometric [1, 14] used is bypassed. Door and car locks have also suffered various types of attacks including cloning RFID cards [17], relaying signals [6], and exploiting weakness in the authentication protocols [60, 63].

The state of the art in authentication mandates using multi-factor authentication, that is, combining a secret, a token, and a biometric. Interestingly, card-based financial transactions already use two factors, a PIN code and a card token. Security is compromised by eavesdropping the PIN (e.g., through tampered terminals) and creating a copy of the card. In other cases, multi-factor authentication is not used correctly [8] or not used at all because of usability issues [2]. Biometrics, such as fingerprints, have also been found to be vulnerable to attacks [14, 58] or reduce utility [40, 48].

Certain promising approaches employ user location as an additional factor of authentication for financial transactions [23, 41, 46]. They exploit the fact that users rarely get separated from their smartphones [29] and use them to either confirm the user's location or transparently provide location as an authentication factor. These approaches can fail if the smartphone is not present or not operational [20] (e.g., due to limited battery life), which may have stifled their broad deployment. Smartphones and other portable devices have been also used as proximity-based tokens [32].

In this paper, we propose using the Internet of Things (IoT) to model user location and movement for making user location continuously available as an additional factor of authentication, independently of whether a device is available (online) when the user authenticates. In contrast to prior works in location-based authentication (LBA) works, we argue that using the increasing number of smart things that users carry, wear, drive, or even have in their body, enables more robust methods for estimating user location. In other words, it allows us to estimate the location of users despite individual devices being offline or not with them.

IoT devices can help us locate their user by reporting their location (e.g., through GPS or WiFi) or by proximity to other other devices with known coordinates (e.g., wearables). Smart environments can also "observe" user things. For example, access points, smarthome hubs, etc. can report which devices are connected, financial institutions can report when and where a credit card is used, and smart traffic lights can report the location of vehicles. However, IoT devices can do much more, they can "sense" when they are being used. For instance, wrist wearables know when they are being worn because they sense walking and/or the user's heartbeat, and a smartphone that has just been unlocked with a PIN or fingerprint knows that the user is holding it. This is crucial in estimating whether a user is with a set of devices, as we no longer need to assume that users are de-facto with their smartphone, smartwatch, etc.

We couple location and activity data reported by devices to model users and their movement. Maintaining such a model enables us to estimate how likely it is that a user is at a particular location, without relying on any device being available and able to provide the user's location at the moment of authentication. Moreover, it enables us to use potentially-sparse data, as certain IoT devices may only report occasionally.

Another factor differentiating this work from previous ones is the way we use location data. Querying parties are *not* allowed to ask for the coordinates of any user device. Instead, they can place generic queries such as "Can the user be physically present at this location?" By only allowing such queries, we inhibit "curious" services from attempting to arbitrarily locate the user. More important, to respond to queries, we rely on evidence indicating that user is *not* at a given location. For example, we respond positively, only if we are confident that a user is not at a location. This strategy serves a twofold goal; first, to prevent falsely rejecting users that forget at home or have inoperable devices and, second, to prevent stolen devices from being misused to subvert the system.

To demonstrate our approach, we design and develop Icelus, a system that collects location and activity data from IoT devices to model user movement and location. Icelus can run as a service on a device of the user, such as a smarthome hub [30], or

it can be hosted in the cloud [42]. To collect data, it organizes the various devices in a hierarchy, so that the ones with Internet connectivity can relay the data of the ones without to the system. Third-party systems can also provide data by directly connecting to Icelus or indirectly by forwarding notifications of certain events (e.g., the use of a credit card at a location, an entry in the user's calendar, etc.). To alleviate privacy concerns, we also develop a privacy-preserving extension of the protocol used in Icelus that allows us to operate purely on distances, without revealing the actual locations of individual devices. At the core of the extension is a secure multi-party computation protocol that leverages additively homomorphic encryption and blinding. Finally, we evaluate Icelus by deploying it on set of devices readily available today.

Briefly, our contributions are the following:

- We propose a new approach that utilizes the IoT to estimate the location of a user and use it as an additional factor of authentication that is more robust that smartphone-only approaches .
- We develop a user movement model using the location data provided by IoT devices, which enables us to operate even when devices are not reachable.
- We define a method for determining the probability (referred to as confidence score) that the user actually is with a set of his devices, utilizing both the number of devices present and the user activities captured by device sensors.
- We develop Icelus, a prototype system that implements the proposed approach.
- We define a privacy-preserving protocol and formally establish privacy guarantees under an honest but curious attacker.
- We evaluate our approach by deploying Icelus on a set of devices readily available today and performing two field studies. Our results show that we can achieve a false-rejection rate of 4%-6%, which is lower than that of smartphone-based location-based authentication. At the same time, we are more resilient to attacks. We also evaluate the performance of our approach and find that it imposes negligible overhead on the devices tested of below 1%.

2 Threat Model

The attacks we aim to thwart in this paper include attempts to bypass user authentication with physical objects and terminals to gain unauthorized access to places, property, etc. of the user, as well as third-parties. Such attacks may include compromising passwords, security tokens (e.g., swipe cards and USB keys), or biometrics. The methods employed can involve stealing, cloning, and counterfeiting credit and

debit cards to perform transactions at POS [5, 43]. In the US such attacks are bolstered also by the limited deployment of PIN and chip technology [28]. Systems like Android Pay and Apple Pay, that enable users to pay with their smartphones or smartwearables, can also be compromised if the PIN code [13,52] or biometric [1,14] used is bypassed. Door and car locks have also suffered various types of attacks including cloning RFID cards [17], relaying signals [6], and exploiting weakness in the authentication protocols [60, 63].

We do not assume that the user's devices have not been compromised, instead our model considers that they could be physically stolen, tampered, or remotely compromised. We rely on the scale of the IoT for resistance to subversion. Our goal is to maintain correct operation, as long as the majority of devices have not been compromised.

3 Approach Overview

We propose using the IoT to estimate *where* a user can possibly be and use the *confidence* of our estimations to augment physical security decisions. For example, if someone enters the credentials of a user at a known physical location, such as a door keypad, we want to be able to answer the question "Can the user be in front of the door at this time?" Being able to answer this question will improve security, as the presentation of credentials on physical terminals without the legitimate user being present can indicate that a credential has been stolen or compromised. Our approach can enable policies that reject credentials or request additional identification, when it is determined that the user cannot physically be at the point of authentication (e.g., activate multi-factor authentication).

3.1 Viewing the Real World Through the IoT

An increasing number of objects contain computational and networking capabilities. The Internet of Things consists of objects that are carried by users or reside in their environment. They are able to sense each other (e.g., through Bluetooth) and frequently communicate with each other. They are also able to sense the environment and their user, e.g., wrist wearables can sense the heartbeat of the wearer.

We claim that through the IoT we can glimpse into the physical world to establish the location of their users. For example, according to a study 79% of people aged 18-44 have their smartphones with them 22 hours a day [29]. Smartphones can also establish their location by using information from network base stations, GPS, and WiFi, hence, they provide a strong indicator of their owner's location. Other objects,

like tablets or modern cars, also come with Internet connectivity and GPS, but may be shared among a few people, like the owner's family. They also provide hints, albeit weaker ones than smartphones, on the location of at least one of its usual users. These hints that can be significantly strengthened, when individual users can be authenticated. For example, both iOS and Android support multiple users, and certain BMW vehicles also support driver profiles.

Certain devices, like many wearables, do not feature GPS. However, they are able to connect to other devices through a wireless protocol, like Bluetooth, WiFi, etc. Such devices provide a different type of hint regarding the location of the user; because they are usually personal devices and the connection protocols have a limited range, being able to connect, or even just establishing the presence, of one indicates that the user is nearby. For example, paired Bluetooth devices can establish each other's presence, while the Bluetooth Low Energy (BLE) protocol enables the same without pairing. If we can establish the location of a single device in such a *cluster*, we are able to "ground" it and locate the user. Similarly, it is sufficient that a single device in a cluster is able to connect, directly or indirectly, to the Internet to make this information available to other parties.

Moreover, many IoT devices are not only able to sense each other but also the user. Smartphones and tablets provide PIN-based or even biometric-based authentication, fitness wearables can establish user movement and heart rate, while smart in-body health devices, such as insulin pumps, are always on the user. Thus, they can also help us detect when they are actually used by the user instead of being idle.

Hints on the location of a user are be also provided by third parties that observe one of the objects that the user owns. Observations are not limited to devices; tokens, like credit cards and passwords, are also "things" that can be observed. For instance, the bank observes that the user has used a debit card at an ATM or POS, and an employer notices that the user entered his credentials at a keypad-protected office door.

An example of devices and tokens owned by users and third parties that can observe them is depicted in Fig. 1. Collecting information from user-owned devices and third-parties provides us with the locations of his things. We assume that the data are collected under the purview of the user, for example, by a service hosted on the cloud with the user retaining ownership and control of the data. Of course, estimating the location of a user's things does not mandate that the user is necessarily with them, which raises the question: "How *confident* are we that the user is actually with a set of his things?", which is described in Sec. 5.

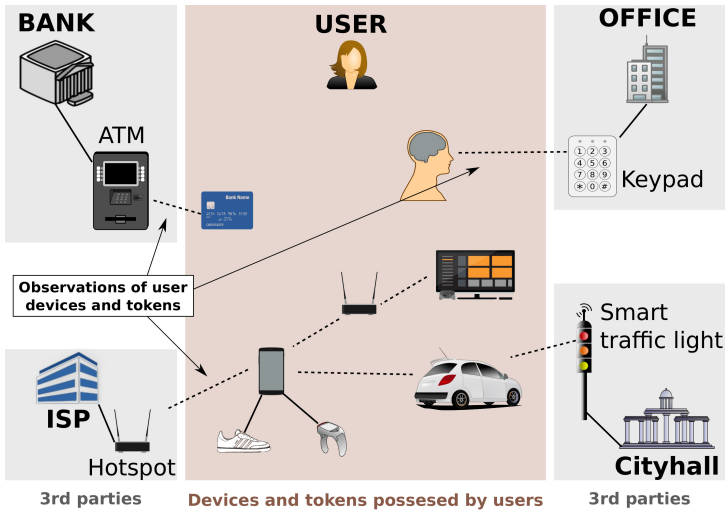

Fig. 1. "Things" owned by users and third-parties that can observe them. By collecting reports containing location information and proximity between things we can estimate user location.

3.2 Modeling User Location and Movement

We model users through *Avatars*, essentially their representations in the digital world, and multiple Avatars may concurrently exist for the same user. The location of an Avatar can be updated whenever information, including location information, is received from an IoT device. Since location reporting is not continuous, due to limited resources and connectivity, user movement must also be taken into account to enable meaningful location-possibility queries at any given time. User movement speed can be estimated using recent location reports and device sensor data [37]. However, we can also model certain vehicles or modes of transportation. Users walking, driving, or cycling can attain speeds within well established parameters. So by estimating an Avatar's speed, we can at any point establish the range of an Avatar, i.e., without making any assumptions on its direction remaining consistent, we can define the region where it is physically possible for the actual user to be. Additional physical world characteristics, like terrain and road networks, can also be incorporated to more accurately establish the range of an Avatar. For example, when on a car, the user is limited to driving on roads. However, we do not explore them in this paper.

The advantage of modeling the user is that we can remain operational even if there are inaccuracies in the reported data, location updates are sparse, and some devices have been compromised.

3.3 Using Location in Authentication

By obtaining a set of locations where the user may be, we enable pre-authorized locations to issue binary, "yes" or "no" type queries about whether the user can be present at the registered location. The reason for such queries is to avoid leaking the location of the user unnecessarily. If the user has just presented a security token at a location (e.g., a credit card), responding "yes" would confirm that the user is at the location, without leaking any additional information. If the response is "no", the service gains no additional information. A single third party with many pre-registered locations could attempt to maliciously narrow down the area where a user may be, by issuing multiple queries. However, such entities can be easily singled out and submitted to throttling or blocked entirely. As a result, location can be made part of a security decision without actually divulging the exact coordinates of the user.

Responses can be generated using a variety of policies. For example, we may want to simply check that there are no strong indicators placing the user at a different location, essentially, looking for paradoxes (the user cannot be in two places at once). Alternatively, we may actually require evidence that the user is actually at a particular location. In this paper, we adopt and evaluate the first strategy. Particularly, we respond negatively, only if there is an Avatar that cannot be at the querying location and its confidence is over a *rejection threshold*. Different services may require a different threshold, as not all actions have the same gravity. For example, the wrong person entering the gym is not as a serious problem as a fraudster withdrawing $10,000 from a bank account.

3.4 An Example

We illustrate how our approach through the example shown in Fig. 2. At 8:00 AM, the user is at home with all of his devices, including a tablet, a smartphone, sensors in his shoes, and his smartwatch, getting ready to walk to work. His WiFi access point also confirms that all of his devices are at home. He leaves home without his tablet and, at 8:15 AM, he stops at a coffee shop to buy a cup of coffee using his credit card. At this point, there are two Avatars. The first, is associated with the user's tablet. Since it is only a single device and it is idle, the confidence for that Avatar is low. However, because the tablet is powered on, it still sends regular reports, allowing us to limit the range of that Avatar around the house. The second Avatar is at the coffee shop, where an activity report from the user's bank has placed his credit card, and where the smartphone reported itself and the rest of the user's devices.

Finally, the user arrives at the office at 8:35 AM. He swipes his badge to enter, at which point the office queries the service. The moment the query is made, there are

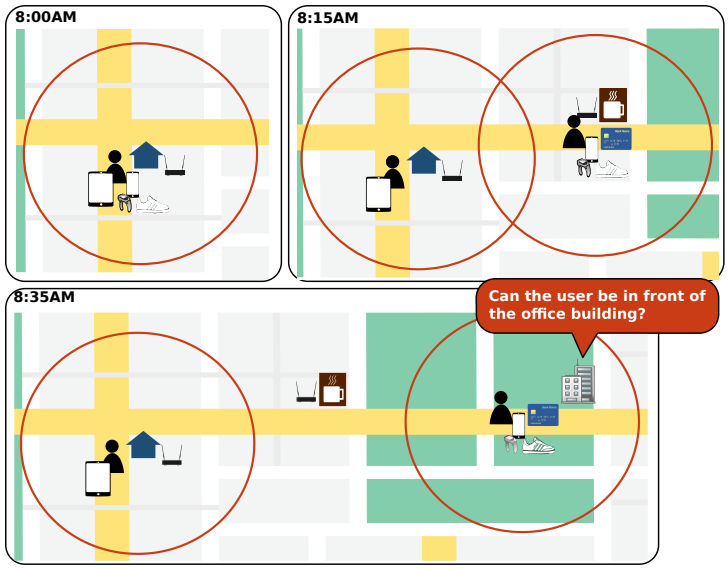

Fig. 2. Example scenario, where a user walks from his home to the office, stopping for a coffee.

still two Avatars, one at home and one on the street very close to the office location. Notice that the credit card previously used at the coffee shop lingers with the Avatar it was associated with when it was used. Our goal is to both identify that the Avatar located at home corresponds to idle devices and, as such, we are less confident it corresponds to the real user, as well as be more confident that the moving Avatar corresponds to the actual user. This way we can respond correctly to the query.

4 Icelus Architecture

We realize the approach presented in Sec. 3 with the Icelus system. Its architecture, which we present here, can incorporate the majority of IoT-devices that could be of use. Icelus organizes IoT devices into a hierarchy, where more powerful interconnected devices collect data from smaller devices, as depicted in Fig. 3. In turn, those devices send the information to a hub, which hosts the Icelus service.

4.1 The Icelus Hub

The brain of the system is a hub collecting information from various sources. We assume that the Hub is under the control of the user, so the data collected are never seen by third parties. We envision that it is hosted either in the cloud [42] or in a

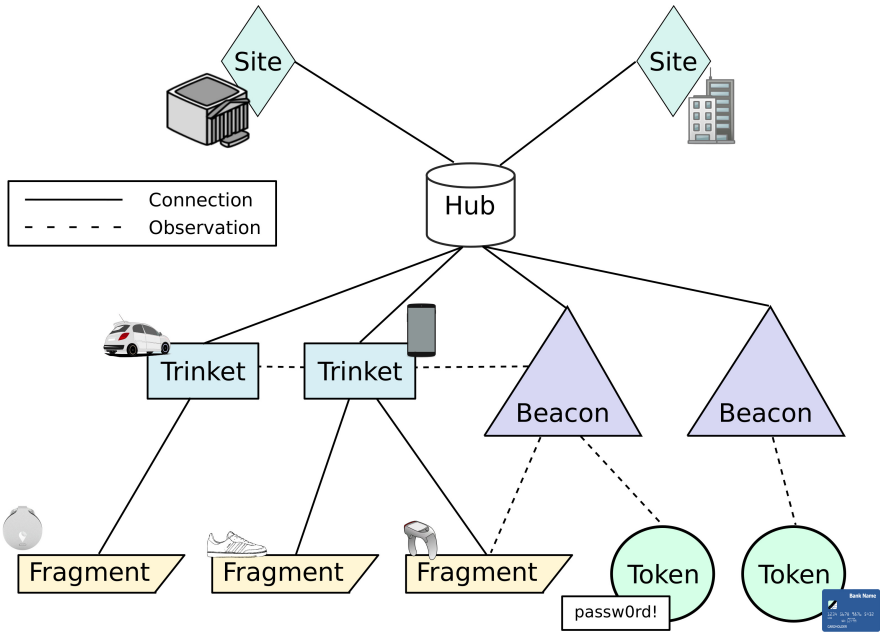

Fig. 3. Architecture overview.

smarthome hub device [30]. Hosting in the cloud provides us with all of its benefits and risks. We assume that the cloud provider is not malicious, however, it may be curious or compromised. In Sec. 6, we present a privacy-preserving extension that can alleviate such risks. Other approaches, such information-flow tracking [45] and SGX [54] are also applicable.

4.2 User-Owned Devices

Based on their intrinsic characteristics and communication capabilities, we classify the devices that can be part of Icelus into the classes described below:

– **Trinkets** are devices that can connect to the Internet directly (e.g., over WiFi, LAN, or 3G) or indirectly (e.g., by tethering through Bluetooth with a device that is connected). Such devices include smartphones, smartwatches, and even Internet-connected cars [16]. To join the system, Trinkets need to first register with the Hub, during which the two exchange their public keys. Thereafter, the Trinket uses its secret key to digitally sign all the information it reports and the Hub's public key to encrypt transmitted information.

– **Fragments** are similar to Trinkets, but cannot directly connect to the Internet. They are, however, able to connect to a Trinket directly (e.g., over Bluetooth). Such devices may include wrist wearables like a Fitbit, other smart wearables like shoes, and could even be in-body devices. A multitude of devices in the IoT are Fragments. Based on whether it is possible to load additional client software on them, they are more like Trinkets, in the sense that they can register with the Hub and sign their data, even though they still rely on a Trinket to relay their data to the Hub. If public-key cryptography cannot be supported, a shared secret key can be established with the Hub to use more lightweight data signing algorithms. On the other hand, other Fragments devices will rely on Trinkets for most operations. In the most restricted case, a Trinket may only be able to report that a Fragment is in the vicinity (e.g., BLE tags [4]).
– **Tokens** are devices that cannot actively connect to anything including things such as smartcards, magnetic identification cards (commonly referred to as swipe cards), RFID tags, etc. Tokens are passive and can only be observed through another device, commonly some short of a reader.

4.3 Beacons

Beacons are third-party devices, or even entire systems, able to report information about the whereabouts of a user or one of his devices. Reports can be generated after the user interacts with a Beacon or when it observes one of the user's devices. For example, the user *interacts* with a Beacon when entering valid authentication credentials at a physical terminal or when using a credit card at a POS. While, a Beacon *observes* the user's smartphone, when it authenticates with a wireless hotspot. In both cases, the location of Beacons is known. Services associated with Beacons must register with the Hub to be able to push observation information to it. However, it is also possible to extract information already available in other channels. For example, many banks transmit credit card usage reports through SMS or email, which can be used to locate Tokens like credit cards.

4.4 Avatars

An *Avatar* is a digital estimate of a user's location in the physical world. Each Trinket reporting geolocation information attempts to generate an Avatar at its location and attaches to it along with all its slave Fragments, so even the ones that are not with the user will create their own. Any devices in the same vicinity will be joined under one Avatar. We define the same vicinity to be as a circle with a center on the previous

Avatar location coordinates with a radius equal to 8 meters which is the *worst case* [61] accuracy of standard commercial GPS systems. When a Token appears, due to an observation from another device or a beacon, it is linked to the Avatar at the location of the report. Because Tokens appear only momentarily when they are used, they linger with the Avatar they are connected to, until a new report about them is received. Tokens appearing away from existing Avatars, create a new Avatar at the location where they were observed.

The **Confidence Score** of an Avatar represents the confidence of our system that it corresponds to the actual user. Our approach can support a variety of algorithms for calculating it. In Sec. 5, we present one such algorithm.

4.5 Servicing Location Queries - Sites

The entities that may query the system about whether it is possible for the user to be physically present at a location are referred to as *Sites*. A Site can be part of Icelus itself, because it is property of the user (e.g., the Trinkets corresponding to the user's home front door or car), or a third-party, such as the user's bank or employer. Sites need to be authorized by the user and registered with the Hub, before being allowed to issue queries, by *supplying the locations* they wish to place queries for.

The Hub listens for queries from Sites, which semantically follow the format: "Can the user be at location L?" When a query is received, it asks Trinkets to report with fresh data. The system can then wait for a bounded amount of time, collecting new reports and updating its model of possible user locations. Note that the model is continuously maintained independently of whether any requests have been made, by having devices report periodically or opportunistically. As a result, Icelus can always issue a response in a bounded amount of time, the only thing that changes is the freshness of the model used to make a decision, which could be milliseconds or few-minutes old.

The Hub then examines if an Avatar with confidence score higher, than the rejection threshold configured for the Site, exists in a location other than L. The threshold can also be a global setting. Our design is flexible; other factors can be introduced when responding to queries and a variety of policies can be implemented. For instance, instead of immediately responding negatively, Icelus can prompt users and ask them to authenticate on their devices that have such capabilities.

5 Avatar Confidence Score

An Avatar is a in reality a set of collocated devices in an area of a given radius, the Confidence Score of each Avatar is a quantity representing the probability that the user is physically "near" that set of devices.

In this section, we present an algorithm for calculating this likelihood, however, our design is not inherently bound with the presented Confidence formula.

Given its definition, an Avatars Confidence intuitively should be directly correlated with the number of devices in a given area where higher number (of colocated devices) should be positively correlated with a higher confidence score and a higher probability thus that a user is also present.

While our model is based on this intuition, we will see that there are additional factors in play, that significantly seem to affect the likelihood a user is present, other than simply the number of devices. The most prominent of which is user actions, that decisively identify idle or forgotten devices with devices that are in his presence. User *Activities* is a feature Icelus exploits and plays a central role to our Confidence calculation mechanism.

5.1 Device Credit

We call "Device Credit", the likelihood a user is colocated with a device at a given time and place.

In statistical terms, we are defining a probability where: *given that a device sends a report to Icelus, a user is also physically in the vicinity of that device*. More formally expressed, we are defining "Device Credit" to be the probability given by the following Bayesian formula:

$$P(U \mid D) = \frac{P(D \mid U) * P(U)}{P(D)} \text{ Where,} \tag{1}$$

- $P(U \mid D)$ Is what we are looking for, given that a device report is received by Icelus, what is the likelihood the user is collocated.
- $P(D \mid U)$ Formally, it is the probability that given a "sighting" of the user at any time, what is the expected probability he also has the device with him. In simpler terms: the probability the user is "carrying" the device.
- $P(U)$ The probability that if someone is operating the device it is the intended user.
- $P(D)$ How frequently the device is used and thus it is active and reporting. We used the survey to assign this value initially based on how often the user (believes)

	Sample Question
P(U)	Would you share your smartwatch with ...? [I do not share it][family][spouse]
	Would you share your smartphone with ..? [I do not share it][family][spouse]
P(D|U)	You carry your smartphone ..[I do not pay attention to its whereabouts][always]
	You wear your smartwatch ..[I do not pay attention to its whereabouts][always]
P(D)	You check or use your smartwatch ..[multiple times per hour] [rarely]
	You check or use your tablet .. [multiple times per hour] [rarely]

Table 1. Some of the survey questions, and the attributes they are associated with.

he checks or uses his device. However Icelus can calculate this value with higher accuracy over a period of time by simply counting the number of reports the device sends daily. For instance if the report window is 5 mins and the device is reporting in 144 out of 288 then $P(D) = 50\%$.

Since for novel devices such as the smart wearables and BLE sensors there are no studies we could find, that provide statistics such as what percentage of the time the user is near or using them (unlike the case for smartphone where it is studied extensively) we deployed a small user questionnaire in order to elicit the required elementary probabilities.

5.2 Questionnaire

We created an anonymous online questionnaire and disseminated it through various channels after obtaining IRB approval from our institution. Our goal was to obtain information about what kind of devices users own and how they use them in their day-to-day routine, we used responses to elicit elementary probabilities for our device weights.

Although our collected data to this point represent a population of 100 individuals, this survey does not have the statistical properties (demographic diversity or sufficient samples) to represent the general population. We used these weights as a starting point to evaluate the performance of our system. Our insight is that while these values are

adequate for our experiment, should our system be deployed they should be calibrated to each user during the device registration step. Since having a user complete a survey might be subject to both inaccuracies and potentially add to user frustration, Icelus could implicitly estimate device weights for each user by employing machine learning methods on the Hub; this however is beyond the scope of this work.

Max Credit We define as Max Credit the maximum credit a device may provide to the avatar it is attached to. Max Credit is equal to the value calculated from Equation 1.

Base and Activity Credit Since idle devices are generating false Avatars, erroneously leading Icelus to believe a user is in a different location, it is thus crucial that devices that actually follow the user to be able to out-weight those that are idle and at the same time idle devices should only be able to generate weak Avatars. In order to achieve both goals we separate its device's credit to two parts *Base* and *Activity* Credit.

Base Credit is a baseline value that the owner is colocated with this device just because it is powered on and reporting. A device is awarded Base Credit whenever it sends a report .

Activity Credit When the user performs activities on the device such as: unlocking the smartphone with a pin or the Fitbit detecting a heart-rate it is awarded extra credit.

Base and Activity credit together compose the Max Credit as defined previously. Activity Credit is assigned to each activity based on the evidence this activity provides that a user is present and it is the intended user, each software client is configured on a per device basis. For instance a smartphone that is able to perform fingerprint based authentication provides more credit than a simple pin authentication activity. Figure 4 attempts to illustrate composition of Credit clearly. `http://mathworld.wolfram.com/MovingAverage.html` *Moving Average [65]:* To smooth the erratic nature of user activities we introduce and evaluate for a variable number of past values an averaging window. For example for window of 1 if t is the current report window the new credit value is: $C_{New} = \frac{C(t-1)+C(t)}{2}$. A window size of 0 means we only consider the current estimated credit value.

5.3 Confidence Score Estimation

The aggregated credit of each device is the **Avatar's** confidence, and thus the likelihood that the owner is denoted by the cluster of devices composing this Avatar. We

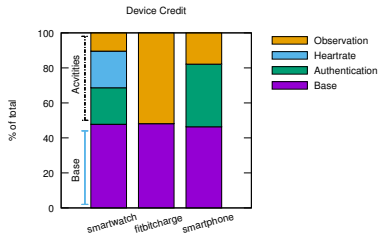

Fig. 4. Activity contribution example to device credit. Device credit is continuously calculated using Base credit plus credit from performed Activities.

also assume that the probability of each device being with the user is independent of others, and we mathematically model as an independent random variable. Based on this, the confidence score of an Avatar A with N attached devices is given by Equation 2. If there are more than one Avatars per user (e.g. the user has devices reporting from different locations) then each Avatar will have a Confidence score calculated from the devices under its area of influence.

$$A_{conf} = 1 - \prod_{i=1}^{N}(1 - C_i) \tag{2}$$

This can also be read as the *complementary probability of the event that the user is not near any of the devices attached to the Avatar*.

Rejection Threshold Since sets of devices in different locations lead to different Avatars, such as when idle devices are left home, we introduce a minimum confidence threshold to select only the Avatar that represents the devices "following" the user. We set this threshold to be equal to the *maximum* confidence an Avatar would produce if *all* registered devices are idle and attached to it. Since this is the maximum confidence possible achievable by any set of idle devices it will also satisfy the case of more than one idle Avatars.

Icelus blocks access to a Site *if* an Avatar is found with confidence strictly above this threshold. By following this decision policy we never falsely reject a transaction performed by the real user due to idle or powered off devices. As we will show in Section 8, our model is able to generate Avatars with sufficient confidence while filtering idle devices due to our activity mechanism.

Safe Zones Early results showed that the user is present at some locations while his devices are idle, such as being home during the night. This is reflected in our

confidence formula by adding extra credit when devices are detected to be in the users house during certain hours. This way there is a high confidence Avatar generated while the user is sleeping, protecting him from unauthorized accesses at different locations. In the current system version we annotate manually, the user specifies when he is at home. Our insight is that this can be learned automatically by Icelus but we do not evaluate it in the current work.

6 Privacy-Preserving 3rdParty Hub Hosting

Hosting a personal hub may be a challenge for many users. Alternatively, a third party can host a hub as a software-as-a-service. As highlighted previously, this may raise privacy concerns as this party may learn the positions of the user's devices at the time of protocol engagement.

We leverage *Secure Multi-party Computations (SMC)* to mitigate these privacy concerns, with the goal of allowing the hub to learn only distances between reported positions, and not the actual positions. For simplicity yet without loss of generality, we focus on the case of a single site (referred to as "the site").

6.1 Additively Homomorphic Encryption and Additive Blinding

SMC is an active area of research in cryptography including tracks on secret sharing [56], garbled circuits [66], or homomorphic encryption [51, 62]. Homomorphic encryption is suitable for arithmetic computations, which makes it our choice for dealing with Euclidean distances [22, 26]. For efficiency, we require an additively homomorphic encryption scheme such as the one provided by Paillier [44].

Additively homomorphic encryption schemes provides features as described by equations 3–5, which shows the three primary operations of addition, negation, and multiplication (with a known plaintext):

$$[\![m_1]\!] \oplus [\![m_2]\!] = [\![m_1 + m_2]\!] \tag{3}$$

$$\neg[\![m_1]\!] = \quad [\![-m_1]\!] \tag{4}$$

$$[\![m_1]\!] \odot m_2 = \; [\![m_1 \cdot m_2]\!] \tag{5}$$

For our purposes, let the plaintext space $\mathcal{M}$ be isomorphic to the group $(\mathbb{Z}_n, +)$ for some number n, and the public and private key be K and k respectively. For the scope of this paper there is only one such key-pair, for which only the site holds k but where K is known by all parties. For readability, the operations $\oplus, \odot, \neg$ described below do not have an explicit key associated to them, we assume they all use the usual

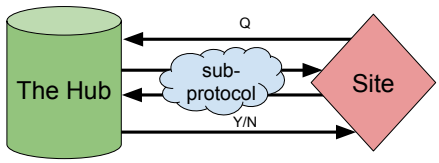

Fig. 5. Depiction of communication when a site queries a third-party hub

k, K pair. The $\ominus$ symbol is used in the following to represent addition by a negated term. That is, $c_1 \oplus \neg c_2$ is written as $c_1 \ominus c_2$. For brevity, the encryption of a plaintext p using the key K is denoted as $[\![p]\!]$.

As a building block in our protocol, we will use the technique of *blinding*. A party A can blind a variable x by addition with a uniformly random value $b \in \mathcal{M}$ as $x' = x + b$. B cannot distinguish x' from a random sample in $\mathcal{M}$ but can return to A a value $x' + y$, from which A can compute $x' - b = x + y$.

6.2 Protocol outline

A user-owned hub can receive location information in the clear, and continuously update avatars. For a privacy-preserving third-party hosted hub, all location reports will arrive at the hub encrypted using K. When a query is made by the site, the hub will initiate a sub-protocol run together with the site. Through this sub-protocol, detailed in the following section, the hub is able to compute distances between any pair of locations. Holding the pairwise distance between three points, it is possible to calculate their relative positions. Thus, using the sub-protocol three times per location, the hub can calculate a full relative coordinate system for all locations. The setup is visualized in Figure 5.

If the hub needs several historical locations for a trinket, fragment, or token (e.g. to compensate for movement), it can cache them and calculate the relative positions retroactively.

6.3 Privacy-preserving distance calculations

There are several existing works on Euclidean distances using additively homomorphic encryption (e.g., [26, 67]). In most cases however, one of the two parties knows the coordinates of one of the two positions. In our case, neither the site nor the hub should learn any positions.

The hub needs to initiate the sub-protocol multiple times. For each invocation, the hub chooses two encrypted positions $([\![x_1]\!], [\![y_1]\!])$ and $([\![x_2]\!], [\![y_2]\!])$ and then runs the protocol for them.

The goal is to compute the squared distance together with the site as $(x_1 - x_2)^2 + (y_1 - y_2)^2$. This requires two roundtrips. The first one is due to the fact that the hub cannot compute a squaring in the ciphertexts. This will be done by requesting that the squaring is done by the site, in a blinded fashion. After the first roundtrip, the hub holds the encrypted squared distance, and will ask the site to decrypt it, again using blinding. Finally, the hub can compute the square root and arrive at the distance between the two points. The protocol follows as:

1. The hub computes $[\![x]\!] = [\![x_1]\!] \ominus [\![x_2]\!]$ and $[\![y]\!] = [\![y_1]\!] \ominus [\![y_2]\!]$ and creates a blinded version of each as $[\![x']\!] = [\![x]\!] \oplus [\![b_x]\!]$ and $[\![y']\!] = [\![y]\!] \oplus [\![b_y]\!]$. The hub caches b_x and b_y and sends $[\![x']\!]$ and $[\![y']\!]$ to the site.
2. The site decrypts $[\![x']\!]$ and $[\![y']\!]$, computes their squares, and sends $[\![x'^2]\!]$ and $[\![y'^2]\!]$ to the hub.
3. The hub derives $[\![x^2]\!] = [\![x'^2]\!] \ominus [\![2xb_x + b_x^2]\!]$, and $[\![y^2]\!] = [\![y'^2]\!] \ominus [\![2yb_y + b_y^2]\!]$
4. The hub then computes the encrypted squared distance $[\![d^2]\!]$ and sends a blinded ciphertext $[\![d']\!]$ to the site and caches the blinding as b_d.
5. The site decrypts d' and sends it to the hub.
6. The hub computes $d = \sqrt{d' - b_d}$.

We establish privacy guarantees by proving that only the distance between the devices can be learned by the hub and nothing else about the positions of the devices. The formal concepts and proofs are detailed in Appendix 1.

7 Implementation

We developed a prototype of Icelus including the Hub service and client software for Android smartphones and wearables. The prototype implementation is henceforth referred to as Icelus. The Hub service in Icelus is implemented as a web application deployed under JBoss AS 6.3. The Hub uses RESTfull services to receive reports from devices encrypted using public- or shared-key cryptography over HTTP connections. Icelus performs location modeling without using the privacy-preserving protocol presented in Sec. 6. Instead, we developed a separate proof-of-concept implementation that utilizes the privacy-preserving protocol to calculate distances between devices to evaluate its performance, which we plan to integrate in future implementations of the Hub.

Currently Icelus client software was created for smartphone and wearable devices running Android. We implemented two versions of the client, one for Trinkets and one for Fragments. The Trinket and Fragment clients were developed using the Android SDK v17 and v20, respectively, and communicate over Bluetooth using the Android-recommended messaging framework for hand held-to-wearable communication. The Trinket client is also able to monitor Fragments that do not feature client software, such as Fitbit devices, by passively monitoring the devices paired with the Trinket over Bluetooth .

Messages from The Hub to clients are sent over Google Cloud Messaging (GCM), while HTTP is used in the opposite direction except during the initial registration step that takes place over HTTPS. To register a device the user logs in to Icelus UI over HTTPS, and performs a standard two way registration step with the Icelus client, server-client keys are generated and exchanged. After this step all communication happens over HTTP.

The messages exchanged can be broken down to three parts: a header, which is composed by the device ID and a flag indicating whether the message body is encrypted, the body which includes timestamped sensor data, and a tail, where the digital signature of the header and body, produced using the private key of the sender, is placed. Sensor data can include information such as GPS data (coordinates, speed, bearing), step count (indicates activity), the SSID of the currently connected WiFi access point, list of paired Bluetooth devices, etc.. *Optimizations:* Clients can choose to omit certain data from reports when they determine that no significant change to their state has occurred. For example, when the device has not moved and is idle. The client will then send the equivalent of a heartbeat message that simply notifies the Hub that the device is active and at its previous location and state.

8 Evaluation

In this section, we present the evaluation of Icelus in terms of effectiveness in making authentication decisions and efficiency.

8.1 Effectiveness

To evaluate our approach, we performed two field studies having one of the paper's authors use Icelus. We hosted a Hub on Amazon's EC2 cloud and registered the following user devices: an Asus Zenfone 2 smartphone as a *Trinket*, a Samsung Gear Live smartwatch as a *Trinket*, a Fitbit Charge wrist wearable as *Fragment*, and a TrackR Bravo BLE tag attached on the user's keychain as a *Token*. Trinkets were set

to periodically report to the Hub every five minutes. We could not place code to control the Fitbit Fragment, and the TrackR Token is passive and can be observed by the Smartphone and the TrackR's crowdGPS service. We did not include any third-party services as Beacons.

In the first study, $S1$, we deployed Icelus using the smartphone, the Fitbit, and the TrackR and collected data over the course of one month. In the second study, $S2$, we deployed Icelus using the smartphone, the smartwatch, and the TrackR and collected data over the course of one workday.

Accuracy To test the accuracy of the decisions made by Icelus, we emulated query requests coming from the institute, which acts as a Site, whenever the magnetic id swipe card of the user was used to enter any of the access-controlled spaces in the institute. For example, this included the door to the user's office, the gym, etc. We obtained this data through the institute's IT department. In total, this included 49 accesses in 5 different card-protected doors in the first study, and 5 accesses in 3 doors in the second one. In all cases, the user was the one actual using the access card, so these data also correspond to the ground truth. We use the access-card and study $S1$ data to calculate the following authentication metrics:

– **False Reject Rate (FRR).** FRR is the rate of falsely denying access to the real user. It occurs when an Avatar above the rejection threshold is estimated to be *at a different location* from the one the Site is querying about, an access-controlled door in our case
– **Potential False Acceptance Rate (PFAR).** Since during our experiments there were no attempts to gain illegal access, PFAR represents the potential False Acceptance Rate (FAR) of Icelus. For calculating PFAR, we assume that an attacker continuously attempts to enter the institute. This means that the attacker has obtained the user's swipe card and attempts to enter the institute every five minutes. Since we are using a five minute period to update the Confidence Score a higher attempt frequency would not change anything. Hence, PFAR is the rate of falsely accepting such an ideal malicious user, because the Confidence Score of existing Avatars is below the rejection threshold.

Table 2 presents the results, when we employ a different window size in the moving average calculation of the Confidence Score. The FRR and PFAR are equal for a window size of one. Note that because we were not able to receive live queries from the swipe-card system, we relied strictly on the data periodically received by the Hub, that is, we could not request for fresh data from Trinkets.

Window size	**FRR** (# FR)	**PFAR**
0	6.12% (3)	8.16%
1	6.12% (3)	6.12%
2	4.08% (2)	8.16%
3	4.08% (2)	8.16%
4	4.08% (2)	10.20%
5	4.08% (2)	12.24%

Table 2. Accuracy of Icelus in field study $S1$ conducted over the period of a month. We report FRR and PFAR when using different window sizes in the moving average function in the calculation of Avatar Confidence Score. The total number of authentication requests was 49.

We also investigated the three false rejections of the system. Two of them occurred because of invalid data received from the user's smartphone. In detail, the user was driving to the office, a short drive of about 5 minutes. The smartphone reported once during the drive to the Hub, but failed to read its updated GPS location. That triggered a bug in our implementation that transmits the previous coordinates, if new coordinates cannot be read from the GPS, which also led to an invalid estimation of the user's speed. As a result, the Avatar remained at the previous location and its range did not increase. In a full deployment of Icelus, we would be able to contact the user's Trinkets to update location at the point of authentication and prevent such false rejections.

The third rejection occurred when the user forgot his smartphone when going to the gym, which is only a few minutes away from the office. As a result, the now forgotten smartphone, only reported that it is idle and no longer finds the user's Fitbit in the next 5-minute time window. We should note that such cases may not cause a huge inconvenience to the user, who only needs to walk a few minutes to retrieve the smartphone. We could argue that it is similar to forgetting one's keys.

Lessons Learned Besides correcting the buggy behavior in Icelus Trinket software, other actions that we are considering to address such issues is to enable devices to asynchronously report to the Hub, when a significant change in acceleration occurs. Immediately reporting Fragments that disappear is another option. In future work, we also plan to explore using learning to identify user habits for the same purpose. For instance, knowing that the user goes to the gym every afternoon could prevent errors.

Comparison with Smartphone-only LBA A smartphone-only location-based authentication (LBA) system [41] would accept the user correctly as long as the smartphone is with him, it is on, and it has Internet connectivity. Moreover, it may require

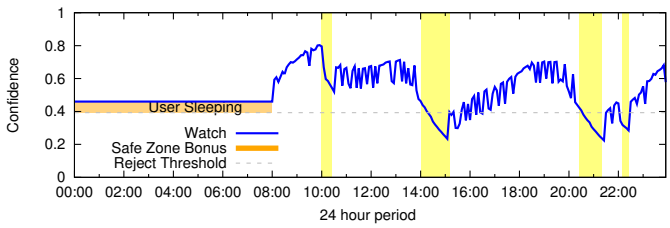

Fig. 6. Confidence plot when user carries a smartwatch and a Bravo tracker. Yellow rectangles denote no WiFi access.

interaction with the user. Using the smartphone data obtained from our field studies, we estimate the FRR for such a system to 8.1% (4 rejections). These occurred when the user forgot/left his phone at the office when going to the gym.

Assuming the user's smartphone has not been compromised, the smartphone-only approach does not have false acceptances. However, compromising the smartphone leads to an 100% FAR. *In contrast*, the true power of Icelus lies in numbers. For example, in study $S2$ the user has two Trinkets; if the smartphone is compromised by an adversary, the smartwatch and the remaining devices enable us to still reject the attacker.

Let us consider the following attack scenario. The adversary steals the user's smartphone at a coffee shop, which is also a wallet containing they access card. Alternatively, the adversary may have cloned the user's contactless access card earlier [17]. The user still carries TrackR's Bravo attached in his key chain and wearing his smartwatch, which is connected to the shop's WiFi. The adversary then attempts to enter the user's office, while the user is still enjoying his coffee.

In the case of smartphone-only LBA, the adversary has obtained all the tokens required to gain access. With Icelus, on the other hand, the adversary can only attempt to reduce confidence score of the user's Avatar to gain access. In Fig. 6, we plot the confidence of the user's Avatar in study $S2$ after removing the smartphone from him, that is, ignoring all data obtained through it. We notice that the Confidence Score of the user's Avatar remains over the rejection threshold for 80% of the day even without the smartphone, which would reject prospective attackers.

Simulation of Larger Scale To evaluate how our solution will fare in the future, when more devices are included, we conducted a simulation with a larger number of Trinkets, Fragments, and Tokens. Our goal is to compare the confidence score of Avatars corresponding to the legitimate user, and a set of devices that have been left

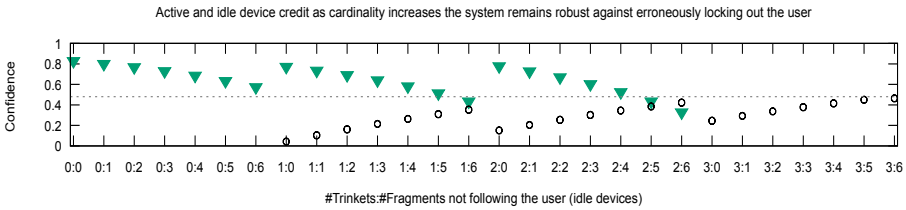

Fig. 7. Comparing the Confidence Score that can be achieved by Avatars corresponding to the legitimate user, and an unattended cluster of devices. Our system's robustness increases with higher device cardinality. Forcing the attacker to compromise multiple devices. The X axis shows the number of devices not carried by the user, formatted as *Trinkets #*:*Fragments #*.

unattended or compromised (e.g., forgotten at home). As before, the adversary may have compromised devices physically or virtually to reduce the Confidence Score of the user's Avatar. *As the number of devices of a certain type increases*, we also normalize the Max Credit of each device splitting it equally amongst all devices of the same type that the user registers.

We present the results of these simulations in Fig. 7 The y-axis in the plots measures Confidence Score, while the x-axis corresponds to the set or number of devices that are *not* carried by the user. For example, *2:5* corresponds to the scenario where the user is missing two of his Trinkets and 5 of his Fragments. As it is evident even when the user has under his control the minority of the devices our proposed mechanism produces an Avatar strong enough to differentiate him and answer inquiries about him with confidence while at the same time we vastly benefit from device number since as it is shown the attacker would have to compromise an ever increasing number of devices to achieve the same result.

8.2 Efficiency

We evaluate Icelus in terms of efficiency by measuring the response times and the impact of running the client on mobile devices in terms of battery consumption. We tested the following Android devices, which were not modified in any way other than adding our client app: (a) **3 smartphones**: an Asus Zenfone 2 with an Intel Atom Z3560 Quad-core CPU at 1.8GHz, a Samsung Galaxy S5 SM 900H with a Cortex-A15 Quad-core CPU at 1.9GHz, and a Samsung Galaxy S4 GT-I9500 with a Cortex-A15 Quad-core CPU at 1.6GHz, and (b) **1 smartwatch**: a Samsung Gear Live E42F with a Snapdragon 400 CPU at 1.2GHz.

Device	**Time and stddev.**	
	WiFi	3G
Devices with direct connection to the Hub		
Zenfone 2	170.8ms ±53.8	645.8ms ±280.8
Galaxy S5	173.7ms ±47.7	683.1ms ±300.1
Galaxy S4	172.9ms ±52.8	710.2ms ±290.9
Bluetooth devices tethered through Zenfone2		
Gear (as a Trinket)	280.8ms ±70.8	680.8ms ±283.8
Gear (as a Fragment)	310.8ms ±83.8	745.8ms ±330.8

Table 3. Device response time.

Device Response Time To evaluate the amount of time it takes for devices to report to the Hub, we conducted an experiment were devices submit 255-byte long reports to the Hub and timed the operations. We issued over 20,000 reports, where each report included data from available device sensors, their list of connected devices, and movement speed. Messages are padded to 255 bytes and encrypted with a 4096-bit RSA key before being sent.

Table 3 shows the average time and standard deviation in milliseconds to complete the operation when connected on our campus WiFi and over 3G. Note that the total time required for the Hub to request and receive a report include the time required to notify the devices, which in Android's case happens through the GCM service. So while previous works [41] have demonstrated low connection times, others have reported that they fluctuate when the notified device is offline [3]. The experiment shows that the process takes less than a second. Hence, even though not required, requesting new information from devices will not impose significant delays on authentication.

Reporting Period How frequently devices report to the Hub, affects battery consumption and the accuracy of the location information. In Table 4, we show the battery consumption imposed by our prototype client over a 10-hour period, when using different report windows. The last row shows the radius of the area where a user may have moved during the report window, assuming he is walking at 4Km/hr. Since energy consumption may change non-linearly depending on the battery's charge, we initiated all tests with fully charged devices. Battery levels were collected using system calls and utilities available on Android. Continuous messaging, appearing on the first column, pertains to sending requests to the server as fast as receiving the previous response. The results show that the effect on battery is small even when reporting

Period	Zenfone 2	Gear Live (as Trinket)	Gear Live (as Fragment)	Accuracy*
0s	7.1%	22.4%	10%	$\leqslant$ 5m
15s	1.5%	4.5%	3%	17m
30s	$\leqslant$ 0.8%	2.3%	1.3%	35m
60s	$\leqslant$ 0.4%	1.5%	$\leqslant$ 1%	70m
300s	$\leqslant$ 0.1%	1%	$\leqslant$ 1%	330m

*How accurately can we estimate location, assuming the user is walking

Table 4. Effects of reporting period in battery consumption and location accuracy.

frequently. As such, we expect that the major obstacle in submitting frequently will be lack of connectivity in certain environments.

Privacy-preservation We evaluated the performance of a privacy-preserving setup using our proof-of-concept implementation. As the main functionality of Icelus is independent on how the (relative) positions are provided, these benchmarks are carried out separately from the main server. It would not require significant engineering effort, apart from storing encrypted locations, to create a combined implementation.

For fragments, trinkets and beacons, the computational load is very similar to the encryption benchmarks presented in Section 8. For the site and the hub, this setup will incur a noticeable overhead. The experiments were conducted with requests corresponding to an example user with 8 devices, requiring 18 distance computations.

The experiments used the DGK encryption scheme [18] with a key-size of 2048 bits. The machines used was a normal workstation with an Intel i7-4790 CPU running at 3.60GHz with 16 GB RAM, and a MacBook pro with an 3.1 GHz Intel Core i7 CPU and 16 GB RAM. The experiments indicate that the time needed to complete the protocol is is about 2.2 seconds when executed between the university campus and a home network in the geographic vicinity, or about 0.81 seconds with minimalistic network delay where only the workstation was used. The rest of the time needed by the hub to compute the confidence score is identical to that of a non-privacy-preserving solution.

9 Related Work

Utilizing a user's location for authentication and augmenting security decisions, like deciding whether to permit raising a user's privileges, is conceptually described in

a position paper by Denning et al. [19]. Unlike this proposal, they propose an approach based on geodetic signatures, that is, signatures that tie a user or terminal to a particular physical location. The increasing popularity of mobile phones has lead to approaches that use them to establish user location and perform fraud detection. Park et al. [46] propose a mechanism where the bank sends a message to the user's phone when he performs a transaction, including the details of the transaction and the location of the POS.

More recently, Marforio et al. [41] use the trusted platform module (TPM) found on smartphones to sign GPS coordinates, preventing a compromised device from supplying forged location data. In contrast, this proposal is more robust as it uses the entire interconnected world instead of a single device to establish user location and augment authentication.

Location-based authentication has also been explored in the context of indoor "smart spaces" [7, 10, 27, 47, 64]. The common denominator of such systems is attempting to identify that a particular user is physically present in a room, usually through the use of proximity sensors, to protect an asset from unauthorized access and, less related to this proposal, customizing services (e.g., displays) according to user preferences. In the same area, the work of Al-Muhtadi et al. [7], which defines a context-aware security scheme for smart spaces, does include the notion of confidence that is affected by the various sensors being present in a space and the type of authentication a user performs. Unlike this proposal, research in this area focuses on indoor spaces and relies on infrastructure that can immediately detect a user entering one.

Also relevant to this paper are publications on predicting the user's location [9, 11, 15, 21, 36, 38, 49, 53], relying primarily on a single device and GPS, and implicit authentication based on learning the user's behavior patterns [33, 34, 50, 55, 57]. It should be noted that these methods focus on a *single device*, typically a smartphone or a portable computer.

Related to our location privacy-preserving protocol, there is large body work on location privacy [35, 59] and biometric authentication [22] where similar problems are investigated. Steps 1–3 are closest to the work of Erkin et al. [22] who perform blinded outsourced multiplication for privacy-preserving face recognition. Steps 4–6 are closest to a building block of the Louis protocol by Zhong et al. [67] to privately decrypt the result of a distance computation.

10 Conclusion

We have presented a new approach that uses the IoT to establish user location and use it as an additional factor of authentication. Location-based authentication has been explored before [31, 41], however, this work is the first to propose using something as pervasive as the IoT to locate users and model their movement. Undeniably, smartphones will still play an important part, as their proximity to their owner is high most of the time, but using the numerous devices that are part of the IoT, we *are* able to locate users more robustly. A major advantage of this approach is that, in the future, it can operate even more effectively, as more IoT devices become broadly available and integrated in the daily life of users. Today, our evaluation with readily available devices shows that our approach exhibits a low error rate and has negligible impact on the performance of the tested devices. Finally, collecting location information in a central location, even under the ownership of the user, must have undoubtedly raised concerns. We described a privacy-preserving protocol that can alleviate this concerns, and argue that progress in the field of privacy-preserving computation, as well as, information-flow tracking [45] and SGX [54] will further diminish the risks.

Acknowledgments This work was partly funded by the European Community under the ProSecuToR project and the Swedish research agencies SSF and VR.

References

1. Fingerprint hack. "`http://www.instructables.com/id/How-To-Fool-a-Fingerprint-Security-System-As-Easy-/`".
2. User perceptions of security, convenience and usability for ebanking authentication tokens. *Computers & Security*, 28(1–2):47 – 62, 2009.
3. Google cloud messaging (GCM): An evaluation. Metadata Blogspot, November 2014. `http://muratbuffalo.blogspot.com/2014/11/google-cloud-messaging-gcm-evaluation.html`.
4. Find your phone, keys, anything. Tile, September 30 2016. `https://www.thetileapp.com`.
5. 2015 LexisNexis Risk Solutions. Merchants contend with increasing fraud losses as remote channels prove especially challenging. True Cost of Fraud(SM) Study, September 2015. `https://www.lexisnexis.com/risk/downloads/assets/true-cost-of-fraud-2015-study.pdf`.
6. F. A, B. Danev, and S. Capkun. Relay attacks on passive keyless entry and start systems in modern cars. In *Proceedings of the Network and Distributed System Security Symposium (NDSS)*, 2011.

7. J. Al-Muhtadi, A. Ranganathan, R. Campbell, and M. D. Mickunas. Cerberus: a context-aware security scheme for smart spaces. In *Proceedings of the IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pages 489–496, 2003.
8. C. R. Alexandra Dmitrienko, Christopher Liebchen and A.-R. Sadeghi. When more becomes less: On the (in)security of mobile two-factor authentication. In *Proceedings of Financcial Cryptography and Data Security (FC)*, March 2014.
9. D. Ashbrook and T. Starner. Using GPS to learn significant locations and predict movement across multiple users. *Personal and Ubiquitous Computing*, 7(5):275–286, 2003.
10. P. Bahl and V. N. Padmanabhan. RADAR: an in-building RF-based user location and tracking system. In *Proceedings of IEEE INFOCOM*, March 2000.
11. P. Baumann, W. Kleiminger, and S. Santini. The influence of temporal and spatial features on the performance of next-place prediction algorithms. In *Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, UbiComp '13, pages 449–458, New York, NY, USA, 2013. ACM.
12. M. Bellare, A. Boldyreva, and S. Micali. Public-key encryption in a multi-user setting: Security proofs and improvements. In B. Preneel, editor, *Advances in Cryptology - EUROCRYPT 2000, International Conference on the Theory and Application of Cryptographic Techniques, Bruges, Belgium, May 14-18, 2000, Proceeding*, volume 1807 of *Lecture Notes in Computer Science*, pages 259–274. Springer, 2000.
13. J. Bonneau, S. Preibusch, and R. Anderson. A birthday present every eleven wallets? the security of customer-chosen banking PINs. In *Proceedings of the International Conference on Financial Cryptography and Data Security (FC)*, pages 25–40, 2012.
14. Chaos Computer Club (CCC). Chaos Computer Club breaks Apple TouchID. `http://ccc.de/en/updates/2013/ccc-breaks-apple-touchid`, September 2013.
15. Y. Chon, H. Shin, E. Talipov, and H. Cha. Evaluating mobility models for temporal prediction with high-granularity mobility data. In *IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pages 206–212, 2012.
16. B. Cooley. Cadillac rolls out in-car Internet access. c|net, 2009. `http://www.cnet.com/news/cadillac-rolls-out-in-car-internet-access/`.
17. N. T. Courtois. The dark side of security by obscurity and cloning MiFare Classic rail and building passes anywhere, anytime. In *Proceedings of the International Conference on Security and Cryptography (SECRYPT)*, July 2009.
18. I. Damgård, M. Geisler, and M. Krøigaard. Efficient and secure comparison for online auctions. In J. Pieprzyk, H. Ghodosi, and E. Dawson, editors, *Information Security and Privacy, 12th Australasian Conference, ACISP 2007, Townsville, Australia, July 2-4, 2007, Proceedings*, volume 4586 of *Lecture Notes in Computer Science*, pages 416–430. Springer, 2007.
19. D. E. Denning and P. F. MacDoran. Location-based authentication: Grounding cyberspace for better security. *Computer Fraud & Security*, 1996(2):12 – 16, 1996.
20. A. K. Dey, K. Wac, D. Ferreira, K. Tassini, J.-H. Hong, and J. Ramos. Getting closer: An empirical investigation of the proximity of user to their smart phones. In *Proceedings of*

the 13th International Conference on Ubiquitous Computing (UbiComp), pages 163–172, 2011.

21. T. M. T. Do and D. Gatica-Perez. Where and what: Using smartphones to predict next locations and applications in daily life. *Pervasive and Mobile Computing*, 12:79–91, 2014.
22. Z. Erkin, M. Franz, J. Guajardo, S. Katzenbeisser, I. Lagendijk, and T. Toft. Privacy-preserving face recognition. In I. Goldberg and M. J. Atallah, editors, *Privacy Enhancing Technologies, 9th International Symposium, PETS 2009, Seattle, WA, USA, August 5-7, 2009. Proceedings*, volume 5672 of *Lecture Notes in Computer Science*, pages 235–253. Springer, 2009.
23. P. Fourez and Mastercard International Inc. Location controls on payment card transactions, patent no. WO/2011/022062. `http://patentscope.wipo.int/search/en/WO2011022062`, 2011.
24. A. Ghosh. Fraudsters steal $13m from over 1,400 ATMs in Japan in less than three hours. International Business Times, May 2016. `http://www.ibtimes.co.uk/hacker-group-steals-13m-over-1400-atms-japan-less-three-hours-1561435`.
25. O. Goldreich. *The Foundations of Cryptography - Volume 2, Basic Applications*. Cambridge University Press, 2004.
26. P. A. Hallgren, M. Ochoa, and A. Sabelfeld. Innercircle: A parallelizable decentralized privacy-preserving location proximity protocol. In A. A. Ghorbani, V. Torra, H. Hisil, A. Miri, A. Koltuksuz, J. Zhang, M. Sensoy, J. García-Alfaro, and I. Zincir, editors, *13th Annual Conference on Privacy, Security and Trust, PST 2015, Izmir, Turkey, July 21-23, 2015*, pages 1–6. IEEE, 2015.
27. J. Hightower and G. Borriello. Location systems for ubiquitous computing. *Computer*, 34(8):57–66, August 2001.
28. J. Huang. 60% still have old credit cards as oct. 1 EMV card deadline looms. USA TODAY, September 30 2015. `http://krebsonsecurity.com/2014/10/replay-attacks-spoof-chip-card-charges/`.
29. IDC. Always connected - how smartphones and social keep us engaged. IDC Research Report, Sponsored By Facebook. `http://www.nu.nl/files/IDC-Facebook%20Always%20Connected%20%281%29.pdf`.
30. Insteon. Insteon hub. `http://www.insteon.com/insteon-hub/`.
31. M. Jakobsson, E. Shi, P. Golle, and R. Chow. Implicit authentication for mobile devices. In *Proceedings of the 4th USENIX Conference on Hot Topics in Security (HotSec)*, 2009.
32. A. Kalamandeen, A. Scannell, E. de Lara, A. Sheth, and A. LaMarca. Ensemble: cooperative proximity-based authentication. In *Proceedings of the 8th international conference on Mobile systems, applications, and services*, pages 331–344. ACM, 2010.
33. H. Khan, A. Atwater, and U. Hengartner. *17th International Symposium Research in Attacks, Intrusions and Defenses (RAID)*, chapter A Comparative Evaluation of Implicit Authentication Schemes, pages 255–275. Springer International Publishing, Cham, 2014.

34. H. Khan, A. Atwater, and U. Hengartner. Itus: An implicit authentication framework for android. In *Proceedings of the 20th Annual International Conference on Mobile Computing and Networking*, MobiCom '14, pages 507–518, New York, NY, USA, 2014. ACM.
35. J. Krumm. A survey of computational location privacy. *Personal and Ubiquitous Computing*, 13(6):391–399, 2009.
36. J. Krumm and E. Horvitz. Predestination: Inferring destinations from partial trajectories. In *UbiComp 2006: Ubiquitous Computing*, pages 243–260. Springer, 2006.
37. S. L. Lau and K. David. Movement recognition using the accelerometer in smartphones. In *Future Network and Mobile Summit*, pages 1–9, June 2010.
38. L. Liao, D. J. Patterson, D. Fox, and H. Kautz. Learning and inferring transportation routines. *Artificial Intelligence*, 171(5):311–331, 2007.
39. Y. Lindell and B. Pinkas. Secure multiparty computation for privacy-preserving data mining. *IACR Cryptology ePrint Archive*, 2008:197, 2008.
40. S. Mare, A. Molina-Markham, C. Cornelius, R. Peterson, and D. Kotz. ZEBRA: Zero-effort bilateral recurring authentication. In *Proceedings of IEEE Symposium on Security and Privacy*, May 2012.
41. C. Marforio, N. Karapanos, C. Soriente, K. Kostiainen, and S. Capkun. Smartphones as practical and secure location verification tokens for payments. In *Proceedings of the Network and Distributed System Security Symposium (NDSS)*, 2014.
42. Microsoft. Event hubs. Microsoft Azure. `http://azure.microsoft.com/en-us/services/event-hubs/`.
43. D. Naccache, R. Géraud, H. Ferradi, and A. Tria. When organized crime applies academic results: a forensic analysis of an in-card listening device . *Journal of Cryptographic Engineering*, pages 1–11, Oct 2015.
44. P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In J. Stern, editor, *Advances in Cryptology - EUROCRYPT '99, International Conference on the Theory and Application of Cryptographic Techniques, Prague, Czech Republic, May 2-6, 1999, Proceeding*, volume 1592 of *Lecture Notes in Computer Science*, pages 223–238. Springer, 1999.
45. V. Pappas, V. P. Kemerlis, A. Zavou, M. Polychronakis, and A. D. Keromytis. CloudFence: Data flow tracking as a cloud service. In *Proceedings of the International Symposium on Research in Attacks, Intrusions and Defenses (RAID)*, October 2013.
46. F. Park, C. Gangakhedkar, and P. Traynor. Leveraging cellular infrastructure to improve fraud prevention. In *Proceedings of the Annual Computer Security Applications Conference (ACSAC)*, pages 350–359, December 2009.
47. N. B. Priyantha. *The Cricket Indoor Location System*. PhD thesis, MIT, 2005.
48. K. B. Rasmussen, M. Roeschlin, I. Martinovic, and G. Tsudik. Authentication using pulse-response biometrics. In *Proceedings of NDSS*, February 2014.
49. I. Rhee, M. Shin, S. Hong, K. Lee, S. J. Kim, and S. Chong. On the levy-walk nature of human mobility. *IEEE/ACM transactions on networking (TON)*, 19(3):630–643, 2011.

50. O. Riva, C. Qin, K. Strauss, and D. Lymberopoulos. Progressive authentication: Deciding when to authenticate on mobile phones. In *USENIX Security Symposium*, pages 301–316, Bellevue, WA, 2012. USENIX.
51. R. L. Rivest, L. Adleman, and M. L. Dertouzos. On data banks and privacy homomorphisms. *Foundations of secure computation*, 32(4):169–178, 1978.
52. V. Roth, K. Richter, and R. Freidinger. A PIN-entry method resilient against shoulder surfing. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, pages 236–245, 2004.
53. S. Scellato, M. Musolesi, C. Mascolo, V. Latora, and A. T. Campbell. Nextplace: a spatio-temporal prediction framework for pervasive systems. In *Pervasive computing*, pages 152–169. Springer, 2011.
54. F. Schuster, M. Costa, C. Fournet, C. M. Peinado, G. Mainar-Ruiz, and M. Russinovich. VC3: Trustworthy data analytics in the cloud using SGX. In *Proceedings of the IEEE Symposium on Security and Privacy*, May 2015.
55. S. F. Shahandashti, R. Safavi-Naini, and N. A. Safa. Reconciling user privacy and implicit authentication for mobile devices. *Comput. Secur.*, 53(C):215–233, Sept. 2015.
56. A. Shamir. How to share a secret. *Commun. ACM*, 22(11):612–613, 1979.
57. E. Shi, Y. Niu, M. Jakobsson, and R. Chow. Implicit authentication through learning user behavior. In *Proceedings of the International Conference on Information Security (ISC)*, pages 99–113, 2011.
58. starbug. Fingerprint biometrics hacked again. Chaos Communication Congress (31C3), December 2014. `http://www.ccc.de/en/updates/2014/ursel`.
59. M. Terrovitis. Privacy preservation in the dissemination of location data. *SIGKDD Explorations*, 13(1):6–18, 2011.
60. The Telegraph – UK. Three quarters of cars stolen in France 'electronically hacked'. `http://www.telegraph.co.uk/news/worldnews/europe/france/11964140/Three-quarters-of-cars-stolen-in-France-electronically-hacked.html`, October 2015.
61. US AirForce. GPS Accuracy. "`http://www.gps.gov/systems/gps/performance/accuracy/`".
62. M. van Dijk, C. Gentry, S. Halevi, and V. Vaikuntanathan. Fully homomorphic encryption over the integers. *IACR Cryptology ePrint Archive*, 2009:616, 2009.
63. R. Verdult, F. D. Garcia, and B. Ege. Dismantling megamos crypto: Wirelessly lockpicking a vehicle immobilizer. In *Supplement to the 22nd USENIX Security Symposium (USENIX Security 13)*, pages 703–718, Washington, D.C., 2015.
64. R. Want, A. Hopper, V. Falcão, and J. Gibbons. The active badge location system. *ACM Trans. Inf. Syst.*, 10(1):91–102, January 1992.
65. E. W. Weisstein. Moving Average from mathworld–a wolfram web resource.
66. A. C. Yao. How to generate and exchange secrets (extended abstract). In *27th Annual Symposium on Foundations of Computer Science, Toronto, Canada, 27-29 October 1986*, pages 162–167. IEEE Computer Society, 1986.

67. G. Zhong, I. Goldberg, and U. Hengartner. Louis, lester and pierre: Three protocols for location privacy. In N. Borisov and P. Golle, editors, *Privacy Enhancing Technologies, 7th International Symposium, PET 2007 Ottawa, Canada, June 20-22, 2007, Revised Selected Papers*, volume 4776 of *Lecture Notes in Computer Science*, pages 62–76. Springer, 2007.

APPENDIX

1 Proof of Privacy-preservation

1.1 Background

In the following we recall briefly some fundamental concepts.

Definition 1 (Negligible functions). *A function* $\epsilon : \mathbb{N} \to \mathbb{R}$ *is said to be* negligible *if*

$$\forall\, c \in \mathbb{N}.\ \exists\, n_c \in \mathbb{N}.\ \forall_{n \geq n_c}\ |\epsilon(n)| \leq n^{-c}$$

That is, ϵ *decreases faster than the inverse of any polynomial.*

Definition 2 (Indistinguishability). *The two random variables* $X(n, a)$ *and* $Y(n, a)$ *(where* n *is a security parameter and* a *represents the inputs to the protocol) are called* computationally indistinguishable *and denoted* $X \stackrel{c}{\equiv} Y$ *if for a probabilistic polynomial time (PPT) adversary* $\mathcal{A}$ *the function* $\delta(n)$ *is negligible:*

$$\delta(n) = |\Pr[\mathcal{A}(X(n, a)) = 1] - \Pr[\mathcal{A}(Y(n, a)) = 1]|$$

Definition 3 (Semantic Security). *A public key encryption scheme* E *is*semantically secure *or IND-CPA secure if the advantage of any PPT adversary of winning the game IND-CPA in Figure 8 is negligible. The game is won if an attacker can construct the procedures* $\mathcal{A}_1$ *and* $\mathcal{A}_2$ *such that* $b = b'$ *with non-negligible probability. If a cryptosystem is IND-CPA secure, it is also* multiple message *IND-CPA [12].* Multiple message *IND-CPA is formalized by the game MM-IND-CPA in Figure 8, where* k *is polynomially bounded by the security parameter. Essentially, this means that any any ciphertext or order of ciphertexts is computationally indistinguishable from their plaintexts.*

1.2 Privacy against semi-honest adversaries

Our privacy definition follows the standard definitions of secure multi-party computation in the semi-honest adversarial model [25,39], but is here simplified for the case

Game IND-CPA :	**Game** MM-IND-CPA :
$(m_0, m_1) \leftarrow \mathcal{A}_1$;	$((m_0^0, ..., m_k^0), (m_0^1, ..., m_k^1)) \leftarrow \mathcal{A}_1$;
$b \xleftarrow{\$} \{0,1\}$;	$b \xleftarrow{\$} \{0,1\}$;
$b' \leftarrow \mathcal{A}_2(E_K(m_b))$;	$b' \leftarrow \mathcal{A}_2(E_K(m_0^b), ..., E_K(m_k^b))$;
return $b = b'$	return $b = b'$

Fig. 8. IND-CPA and MM-IND-CPA

with two parties. For two parties A and B, where A has inputs $\overrightarrow{x}$ and B inputs $\overrightarrow{y}$, the framework formalizes the output of a protocol as $f(\overrightarrow{x}, \overrightarrow{y}) = (g(\overrightarrow{x}, \overrightarrow{y}), h(\overrightarrow{x}, \overrightarrow{y}))$. The function f is called the *functionality* of the protocol,

The functions g and h are functions describing all outputs presented to A and B from the execution of the protocol, respectively. f, giving the tuple of the parties' joint output, is called the *functionality* of the protocol.

Definition 4 (privacy). Privacy *within the standard framework (for deterministic functionalities) holds when the overall knowledge of each party after the execution of the protocol, called the party's* view*, can be computed from the inputs and outputs of that party. This is called that the view can be* simulated*. That is, for the two-party case with A and B as described above, one must show that:*

$$\{S_A(\overrightarrow{x}, g(\overrightarrow{x}, \overrightarrow{y}))\} \stackrel{c}{\equiv} \{view_A(\overrightarrow{x}, \overrightarrow{y})\}$$
$$\{S_B(\overrightarrow{y}, h(\overrightarrow{x}, \overrightarrow{y}))\} \stackrel{c}{\equiv} \{view_B(\overrightarrow{x}, \overrightarrow{y})\}$$

where S_A and S_B are the the simulators for A and B, respectively.

1.3 Instantiation for the proposed solution

For our purposes, let A be the hub, B be the site, and f be the functionality of the sub-protocol described in Section 6, g returns only the distance between the two points, and that h returns nothing. The explicit hub inputs for the hub are $p = (b_x, b_y, b_d, [\![x_1]\!], [\![y_1]\!], [\![x_2]\!], [\![y_2]\!])$ and the site has no explicit inputs. Both parties also have randomness as inputs for each encryption, but this is omitted here as it is trivial to simulate.

The protocol has two round-trips. During the first round-trip the site learns the blinded value $x' = x_1 - x_2 + b_x$ and $y' = y_1 - x_2 + b_y$, and the hub learns two ciphertexts. During the second round-trip the site learns the blinded $d' = d^2 + b_d$, and the hub learns d^2. The view of the two parties can be given as:

$$view_{hub} = (p, [\![x'^2]\!], [\![y'^2]\!], d')$$
$$view_{site} = (x', y', d')$$

Theorem 1. *The protocol privately discloses the distance for the hub according to Definition 4.*

Proof. To prove that Theorem 1 holds, we need to show that there exist two functions S_{hub} and S_{site} such that:

$$\{S_{hub}(p, g(p, \{\}))\} \stackrel{c}{\equiv} \{(p, [\![x'^2]\!], [\![y'^2]\!], d')\}$$
$$\{S_{site}(\{\}, h(p, \{\}))\} \stackrel{c}{\equiv} \{(x', y', d')\}$$

Blinded values are indistinguishable from a random sample in $\mathcal{M}$ when the blinding used is unknown. Thus, we define

$$S_{site} = (\alpha, \beta, \gamma)$$

where α, β, and γ are independent and uniformly random variables in $\mathcal{M}$.

Ciphertexts are easy to simulate for any semantically secure cryptographic system for any principal not holding the private key. In fact, for any list of plaintexts an arbitrary list of ciphertexts of the same length can be used, as per MM-IND-CPA in Definition 3. Since $g(p, \{\}) = d$, to create a simulation of d', one can simply use $d^2 + b_d$. The simulator for the hub is thus easily defined as

$$S_{hub}(p, g(p, \{\})) = (p, E(0), E(0), d^2 + b_d)$$