You're where? Prove it! – Towards trusted indoor location estimation of mobile devices

ABSTRACT

Location-enabled applications now permeate the mobile computing landscape. As technologies like Bluetooth Low Energy (BLE) and Apple's iBeacon protocols begin to see widespread adoption, we will no doubt see a proliferation of *indoor* location enabled application experiences. While not essential to each of these applications, many will require that the location of the device be true and verifiable. In this paper, we present LocAssure, a new framework for trusted indoor location estimation. The system leverages existing technologies like BLE and iBeacons, making the solution practical and compatible with technologies that are already in use today. In this work, we describe our system, situate it within a broad location assurance taxonomy, describe the protocols that enable trusted localization in our system, and provide an analysis of early deployment and use characteristics. Through developer APIs, LocAssure can provide critical security support for a broad range of indoor location applications.

Author Keywords

Indoor Location, Secure Location, Location Services, Location-Based Access Control.

ACM Classification Keywords

C.2.4 [**Computer-Communication Networks**]: Distributed Systems---*Distributed Applications*; K.6.5 [**Management of Computing and Information Systems**]: Security and Protection.

INTRODUCTION

A long-standing UbiComp research problem has been fast and accurate determination of indoor location. A variety of hardware and software techniques have been created and, especially in recent years, solutions have been proposed that are both reliable and easy to deploy [4, 5, 6]. Many of these technologies enable everyday devices like smartphones and tablets to determine their indoor position. In fact, technologies like Apple's iBeacon framework [2] are beginning to see large adoption, deployment, and use. As a result, a technology foundation is being laid for an explosion of indoor location-enabled applications.

Submitted to Ubicomp2015

For a broad class of applications, location provides context to assist or enhance the user's experience. These include points of interest applications, mapping tools, and social presence sharing applications. While an inaccurate or manipulated location would impact the usefulness of these applications, it would not be harmful from the perspective of the service provider. However, there are many other applications in which the accuracy and trustworthiness of the location is integral to the application itself. These include room access, inventory control, and document access solutions whose decisions to permit or deny access to physical or virtual objects have a contextual dimension that includes the user's physical location.

There has been a significant amount of past research that has focused on trusted localization. However, much of this work has explored point solutions. For instance, solutions have been developed to provide anonymous but verifiable location check-ins for location-based services [7,19], context-based proofs of co-presence that piggyback on existing infrastructure [16], and systems for creating and verifying secure proofs of (coarse) location in multi-stakeholder WiFi or cellular environments [15]. By contrast, our goal in this paper is to develop a holistic approach to enable immediate or retroactive proofs of indoor location supporting a (tunable) variety of assumptions regarding client-side adversarial behavior and client trust in the location-based service.

Towards this goal, we propose a system called LocAssure that provides a broad set of security and privacy affordances while still being compatible with existing, off the shelf devices. In doing so, we make the following technical contributions:

- *Taxonomy*. We identify several orthogonal dimensions along which location-based services can vary with corresponding security, privacy, or infrastructure implications. Specifically, we articulate several models of secure indoor localization by varying (i) whether the client or service reaps the benefit of location context, (ii) the

level of trust that location-based services place in their clients, (iii) the level of anonymity that clients expect when using location-based services, and (iv) whether proofs of location are disclosed immediately or retroactively.

- *Infrastructure*. To service the location models identified by our above exploration, we develop a novel beacon-based infrastructure that uses BLE signals to provide highly accurate, room-level localization with support for a range of security and privacy assurances. This framework is compatible with a wide variety of existing hardware (e.g., Android/iOS tablets and smartphones) and location/proximity technologies (e.g., iBeacon).

- *Protocols*. We develop a suite of protocols for supporting high-assurance proofs of location within our beaconing infrastructure. Our basic protocol requires neither client-side cryptography nor any modification to standard beaconing techniques. Higher assurance protocols providing, e.g., retroactive proofs of location or client traceability require only minimal client interactions and cryptographic assumptions, ensuring efficiency and ease of use.

- *Analysis*. We carry out sensitivity analysis experiments and security analyses of our protocols to demonstrate how our solution performs across a variety of deployment conditions and under a range of security and privacy assumptions.

RELATED WORK

We consider related work from several areas. Many researchers in the UbiComp space have studied techniques for providing a range of indoor localization services. Technologies like ultrasonic sound [10], infrared [23], RFID [12], and coded light [8] transmissions have been used. Unfortunately, client localization using these technologies requires special hardware not typically available on, e.g., smartphones or tablets. Our work differs from these prior efforts by enabling the production of unforgeable proofs of fine-grained, indoor location on unmodified, commodity devices.

There is also a wealth of work on indoor location and navigation based on Wi-Fi and Bluetooth received signal strength measurements. Our focus here is on location at a granularity of rooms rather than precise location. The starting point for our room-level classification system is the WiFi-based work presented in [5]. Here we deploy a system variant using BLE received signal strength (RSSI) information in combination. Many systems use BLE beacons to establish proximity [2]. The beacons we describe here can also be deployed this way although in our experience, we find using RSSI measurements from multiple beacons to establish the location of mobile devices to be more robust. Redpin [6] and RADAR [4] are two established WiFi-based methods for matching RSSI measurements to training data for indoor localization that can also be applied at room-level granularity. The architecture that we detail below can incorporate any location classification approach that provides either an absolute (x.y) or discrete room-level location. We have built upon the approach in [5] because of its high accuracy and efficiency that allow the option to deploy classification on mobile devices.

In [15], the authors also examined the design space and devised a set of goals for systems that provide location proofs while preserving user anonymity to end applications. They designed a Wi-Fi based system that used group signature schemes with symmetric-key encryption. It also allows for the proactive collection of location proofs. The approach relies on additional secure communication between the client and one or more access points to produce a location proof. Our approach does not require modifications to the BLE beacons' normal operation. [22] presents another system for providing location proofs using Wi-Fi APs using an interactive protocol. This scheme uses Wi-Fi to provide a proximity proof, but for cell towers, it could establish a location (via triangulation). This system trades off revealing user identity with preventing the sharing of a location proof between users. Again, the system presented in this paper uses a discrete set of locations rather than proximity and multiple BLE beacons to establish the location proof.

Other mobile device technologies have also been used. [16] presents a client-based system by which mobile devices passively exchange cryptographic keys with one another. Batches of keys are uploaded to a central service, and overlap between two sets of keys establishes a proof of co-location for two or more users (as opposed to proving presence at some physical location). [7] presents a private "check in" protocol for systems like Foursquare. Proximity is established at locations displaying frequently changing QR codes. Clients take pictures to extract cryptographic material used to produce a proof of proximity. [19] presents another secure location check-in service using NFC. These check in protocols provide coarse (i.e., building level) proofs location with various security and privacy guarantees. By contrast, our work provides the ability to produce unforgeable proofs of fine-grained (i.e., room level) location under a variety of threat models.

[18] present Zone-IT, which is a Bluetooth (classic) system that uses beacons to communicate policies to mobile devices that control their functionality. For example, it can disable phones' calling (and ringing) functionality in a specific location. It leverages established cryptographic methods to prevent attacks that can take control of users' mobile devices. [20] presents Loc-Auth, which is a similar system that used attribute-based encryption based on BLE beacons. Here, Bluetooth is used as the communication channel for an authentication (login) scheme that combines location, user identity, and application permission information. In contrast, we use BLE beacons for both location classification based on RSSI measurements and location proof generation based on modification of the beacon advertisement. Our approach

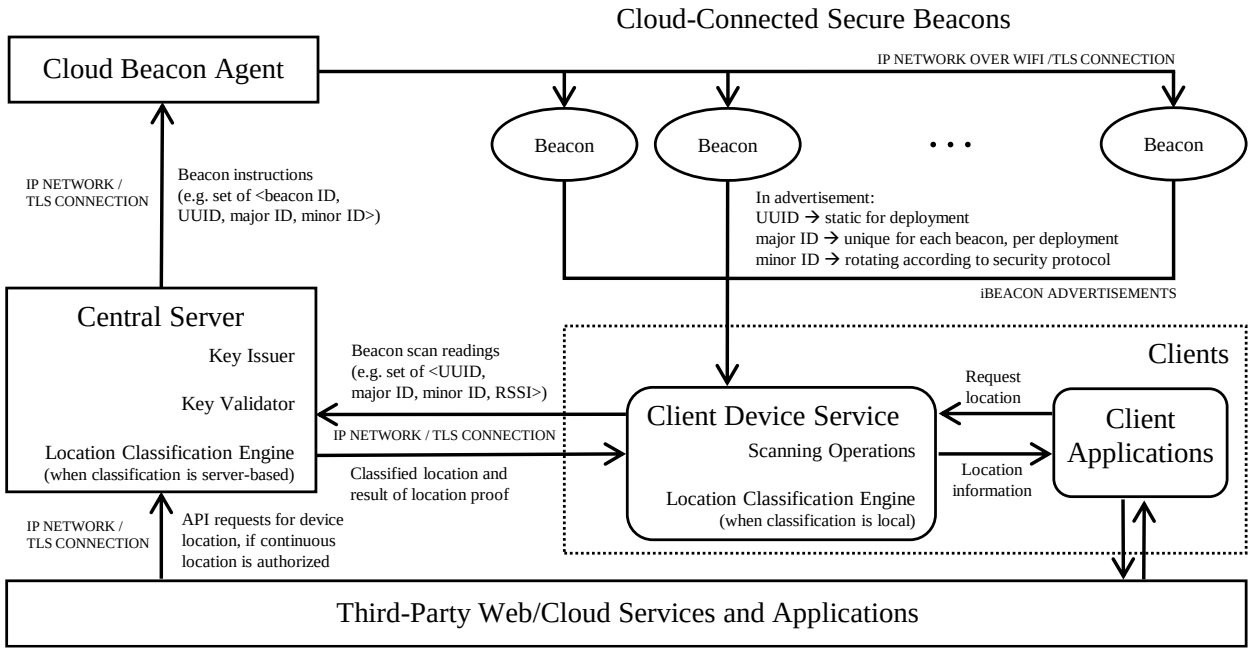

Figure 1: An overview of the LocAssure infrastructure.

requires neither auxiliary communication with the beacons nor any alteration of the beacons' standard operation.

INFRASTRUCTURE

The LocAssure system consists of five main components: a deployment of cloud-connected beacons, a client service for devices, a location classification engine (LCE), a central server, and a cloud beacon agent (CBA). Figure 1 provides an overview diagram of these components, and we discuss each in detail below.

Cloud-connected secure beacons

Like many other indoor location technologies, location estimates are determined through the collection of signal strength measurements from fixed beaconing devices. While measurements from a variety of radios can be used [3, 21], our current system relies on Bluetooth Low Energy devices to maintain compatibility across a wide-range of existing devices.

Shown in Figure 2, our secure beacons are comprised of two key pieces of hardware – a Bluetooth Low Energy radio (BlueGiga BLE112) and a WiFi enabled SoC called ElectricImp [1]. The ElectricImp communicates securely with a *Cloud Beacon agent* (CBA) that allows its software and functionality to be updated at regular intervals. The ElectricImp device also controls the state of the Bluetooth radio through an ICT bus connection. The device is powered by four AA 1.5V batteries that are power regulated to 3V output. Under normal use, this should power the beacon for at least a year.

The cloud beacon operates in two modes: beacon mode and update mode. In beacon mode the ElectricImp is placed into sleep state and the Bluetooth radio is programmed to wake every 2000ms to transmit an iBeacon [2] compatible advertisement. This advertisement contains three main segments: a UUID that associates the device to specific services or capabilities, major ID, and minor ID. The UUID is used to identify the beacon as a LocAssure beacon. The major ID is used to uniquely identify each beacon. This ID is used by the location classification engine along with the received signal strength of the beacon to determine a device's location. Finally, the minor ID is used to provide cryptographic material used by the location authority (discussed in further detail in the Secure Proofs of Location section).

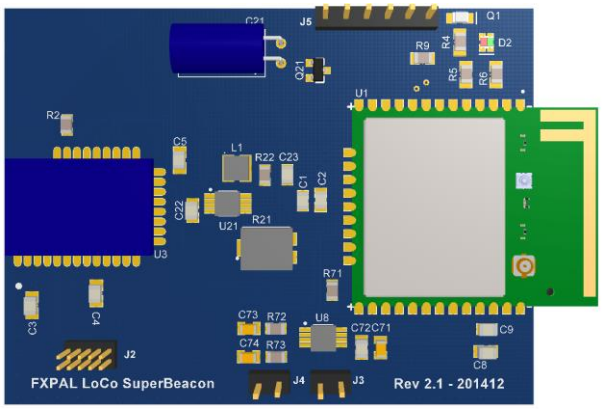

Figure 2. The LocAssure beacon. The device contains both a Bluetooth Low Energy Radio (left) and a WiFi SoC controller (right).

In update mode, which occurs at fixed, but configurable intervals, the ElectricImp connects to the CBA to download new beaconing assignments. In this state, all the parameters of the iBeacon advertisement can be updated. When a LocAssure beacon is deployed this operation normally configures a new minor ID which is a new unique key provided by the location authority.

Client service for devices

A simple service runs on client devices that collect BLE signal measurements. We have an implementation of this service for both Apple iOS 8 and Android 4.3+ operating systems. Each implementation varies in how it automates scan collection. However, each provides the basic functionality of periodically capturing and recording nearby location advertisements. These records include the UUID, major ID, and minor ID along with a measurement of the received signal strength (RSSI) of the advertisement.

iOS supports a limited set of use cases where background execution is permitted. The iBeacon proximity/ranging functionality is one such case. iOS 8 allows continuous monitoring for proximity to a known set of *beacon regions that* are defined as an individual or set of UUIDs. Once the iOS device is receiving iBeacon advertising packets over BLE an *in region* event is forwarded to the LocAssure client service, which has previously registered for such notifications. Once the device is in range of the beacons, the client service can initiate continuous ranging to the beacons – even when the application is in the background state. Ranging consists of a RSSI, UUID, major ID & minor ID values for each visible beacon.

The Android implementation is more straightforward. A background service is set to scan periodically for known UUIDs. If found, the service records the beacon's RSSI, UUID, major ID, and minor ID. The service can adjust the frequency in which it scans. This enables the application to adjust how often it provides location updates.

Once a set of scan records are collected, the client service provides this information to the *location classification engine (LCE)*. The LCE can perform classifications on the client device (local), or the client can send the scan information to a server-based instance of the LCE. As we discuss in the Secure Proofs of Location section, a client-only implementation provides the ability for the client's location to be determined without disclosing it to third-parties (maintain location privacy) while using a cloud-based LCE allows the computation to be offloaded and shared across multiple services.

Location classification engine (LCE)

Our location classification techniques are based on the ensemble learning method of boosting [9, 11]. In contrast to more common matching methods such as [6], boosting provides similar classification accuracy without requiring the storage of a search data structure that grows with the training set size. Further, boosting performs most of the computation in offline training, allowing for classification runtime performance to be orders of magnitude faster compared to matching approaches [5].

The classification engine seeks to provide location relative to a discrete spatial quantization, or more generally rooms (e.g., personal office) or locations of interest (e.g., near a landmark in a large space). For each discrete location, a binary classifier is constructed that outputs a score representing the probability that the RSSI scan vector S was observed in that room/location:

$$F_{room}(S) = \sum_m \alpha_m h_m(S)$$

Each per-room classifier combines "weak learners", h_m according to the scalar weights α_m. The weak learners are decision stumps that compare a scalar feature to a threshold θ_m:

$$h_m(s) = \begin{cases} 1 & X_m \geq \theta_m \\ 0 & \text{otherwise} \end{cases}$$

In training, the thresholds θ_m are tuned to minimize error. We define the feature vector comprised of elements X_m that is computed from each RSSI vector S below.

Given the observed RSSI vectors, the set of unique pairwise differences (margins) between the vectors' elements is computed. For an environment with B total beacons, the resulting margin feature vectors have size $0.5 \cdot B \cdot (B-1)$. Intuitively, these features express coarse order information for the pairs of beacons. The RSSI vector $S \in \mathbb{R}^B$ is transformed into a margin feature vector with elements:

$$X_m = S(a_m) - S(b_m),$$

for $a_m, b_m \in \{1, \cdots, B\}$. Missing RSSI values for specific beacons in the training set are set to a nominal value, R_{min} to indicate they were not visible to the mobile client. Thus, the fact that specific beacons are not visible at specific locations is incorporated into the features.

The margin features computed from the training scans form the input to classifier training. Each scalar margin feature corresponds to a weak learner (h_m) available for inclusion in any per-room classifier $F_{room}(S)$. The training procedure identifies a location-specific set of weak learners that best discriminates that location from all others. The weak learners and their relative weights (α_m in the equation above) for each per-location classifier are learned in a greedy iterative procedure that optimizes error using a per-sample weighting over the training data [11].

For location determination, a one versus all formulation is used. The estimated room is simply the maximum score among the set of per-room classifiers:

$$room^*(S_{test}) = \underset{room}{\operatorname{argmax}} F_{room}(S_{test})$$

Only the required set of RSSI differences that were selected in classifier training are computed. These differences are

compared to the thresholds (θ_m) and then combined linearly. The final room estimate is determined by comparing the scores $\{F_{room}(S_{test}\}\}$ and selecting the maximum.

Central server and cloud beacon agent

The central LocAssure server is a trusted entity that is responsible for coordinating the state of the beacons, performing location classification (when not performed on the client device), verifying proofs of location, and providing third-party applications an API to leverage location information.

As described above, each beacon's state (e.g. what it advertises to clients) can be changed. For each location deployment, the central server makes a periodic call to the *cloud beacon agent* (CBA) to set each beacon's UUID, major ID, and minor ID values. The central server ensures that a unique minor ID is set for each beacon, at each deployment location. As we discuss in more detail in the next section, a rotating minor ID is used as part of the location proof protocol. The central server generates and maintains a list of current minor/major ID pairs and, with a location determination, compares keys as part of the proof of location when scans are submitted for location verification.

A third-party API is also provided by the central server. With a unique API access key, external web applications can make REST-style requests for a device's location.

SECURE PROOFS OF LOCATION

We now describe several dimensions whose variation alters the requirements placed upon a secure localization service. To enable applications supporting many settings along these dimensions, we next describe the threat model assumed in this paper, and then specify the protocols that we have developed to build unforgeable location proofs in these settings.

Dimensions of Localization

By considering the needs of common location-aware applications, we have identified the following four dimensions of secure localization that guided the development of LocAssure.

- *Contextual Benefactor.* In some applications, the **client** is the only expected benefactor of the use of location context (e.g., awareness applications describing events near the user's present location). In others, the **service** may also benefit from the use of client context (e.g., physical/digital access control systems).
- *Client Threat Model.* There are several levels at which a location-based service can place trust in the clients using the system. Clients may be **trusted** to report their location faithfully; this is useful in, e.g., POI applications that require client location to process requests that have no security implications. It may be the case that clients are not fully trusted, and may try to **replay** contextual measurements to appear as if they are located in a prior location; e.g., a user may launch a replay attack from home to appear as if they are in their office in an attempt to access on-site resources. Finally, clients may be **untrusted** by the location-based service. These clients may collude in an attempt to carry out wormhole attacks [13] in which a device or user in one location relays contextual measurements to a device or user in another location to defeat location-based protections.
- *Client Expectation of Privacy.* In some cases, clients may agree to remain **identifiable** to a location-based service; e.g., this is a common assumption in physical access control systems where user accountability is required. In other cases, clients may wish to **mask** their identity from the service; e.g., a POI service has little reason to know the identity of its requesters.
- *Time of Proof Disclosure.* For most location-based applications, proofs of location will be disclosed **immediately**. For others, however, it may be the case that **retroactive** proofs of location are required.

In the remainder of this paper, we will use an abbreviated notation to describe combinations of choices from these dimensions. For example, a physical access control service for personal offices would likely operate within the S-R-I-I setting: the **service** is the primary contextual benefactor, the client may be expected to launch **replay** attacks against the service, the client agrees to remain **identifiable** to the service, and the proof of location is to be disclosed **immediately**.

		Trusted	Replay	Untrusted
Identifiable	**Immediate**	✔	✔	✔
	Retroactive	✔	✔	✔
Masked	**Immediate**	✔	✔	
	Retroactive	✔	✔	

Table 1: Our contributions. Green cells indicate settings of primary interest, while check marks indicate settings in which our proposed protocols can be used.

In this paper, we present several configurations of LocAssure that support indoor localization applications along several of the above dimensions. Table 1 shows a summary of our contributions within the space parameterized above, which will be elaborated upon throughout this section. Note that we focus primarily on situations in which the client is either *completely* trusted by the location service (*-T-*-*) or in which the client is untrusted but identifiable (S-R/U-I-*), as these parameterize a large number of interesting location-based applications for the workplace (e.g., controlling access to physical or virtual resources, location-based personalization, etc.). While the S-U-M-* settings are interesting, we leave these to future work, as the classes of applications supported within these settings (e.g., location-based, delayed-spend, anonymous coupons) are not directly

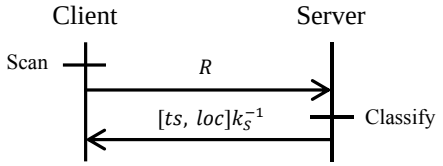

Figure 3: Basic protocol.

applicable to the workplace environments that are the focus of this paper.

Threat Model

We now describe the threat model assumed in the design of LocAssure, and provide details about the protocol extensions that we have developed to counter these threats.

Recall that LocAssure is a distributed system consisting of five key (classes of) entities: the central server, a network of cloud connected secure beacons, a cloud beacon agent that is used to control these beacons, a location classification engine that determines a device's location, and device client services. Additionally there are third-party location-enabled applications/services serviced by the LocAssure API. We now describe our assumptions regarding these entities.

The *central server* communicates regularly with both client device services (to collect scans, and in some cases carry out client localization), and the cloud beacon agent (to manage the network of smart beacons). All communication with the server takes place over TLS-protected connections to ensure that traffic is protected against replay, reorder, modification, and observation attacks. It is assumed that the central server correctly classifies client locations using the LCE described above. Clients may or may not be willing to disclose their identity to the central server.

The *cloud beacon agent* is responsible for asynchronously managing the network of smart beacons on behalf of the central server. Server-to-agent communications are TLS-protected, and the cloud agent will only accept beacon control messages from the central server.

Cloud-connected secure beacons periodically establish bidirectional communication with the cloud beacon agent to update their configurations, and also periodically broadcast iBeacon-compatible advertisements that are observed by client applications. All communications between the cloud agent and ElectricImp SoC running on each beacon are TLS protected, and beacons will only accept control messages from the agent. The beacon is trusted to adhere to any configuration changes mediated by the agent, although incorrect behavior is easily observable.

Client device services collect BLE advertisements in order to localize (either unilaterally, or with server assistance) and may use the determined location either on the client device or in conjunction with a location-enabled application/service. The degree to which the central server and location-enabled applications trust a client may vary as

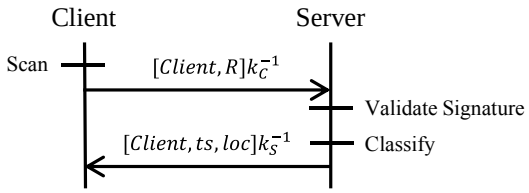

Figure 4:Basic protocol with client signatures for identifiability.

described above. We assume that each client device has a unique public/private key pair and that private keys are not shared between clients.

Location-enabled applications make use of client device locations classified by LocAssure to provide services to client applications. Clients may or may not wish to disclose their identity to these applications, and applications may have varying levels of trust in clients. The levels of trust assumed here parameterize the type of location proof that is to be produced by the client application in conjunction with the LocAssure central server.

Location-Based Services Supported by LocAssure

Recall from the Infrastructure section that localization within LocAssure is based upon RSSI values associated with BLE advertisements transmitted by a network of beacons that are tightly coupled to the LocAssure server. To determine their room-level location, a client application carries out a 15-second scan that generates a *report* containing a set of <UUID, major ID, minor ID, RSSI> tuples. These tuples are then fed into a classifier to determine the locations of the client.

In the event that the classifier is run on the client, LocAssure can provide location services to applications in any C-*-*-* setting: since the client classifies their own location, this suppresses the *Client Threat Model* and *Client Expectation of Privacy* dimensions, and clients can use locally-classified locations immediately or store them for retroactive reference.

In the event that the location classification engine is run on the server (c.f., Figure 3), LocAssure can provide location services to applications in any of the S-T-M-* settings. Since the client device is fully trusted by the central server in this setting, scan reports can be disclosed either immediately or retroactively to allow the server to classify the position of a masked client. The addition of a digital signature binding a scan report to a registered user identity (c.f., Figure 4) allows LocAssure to service applications in the S-T-I-* settings.

Additional Protections

The relatively static nature of scan reports implies that the baseline variant of LocAssure cannot provide location services to applications in the S-R-*-* or S-U-*-* settings, as clients can easily replay old reports or collude to carry out

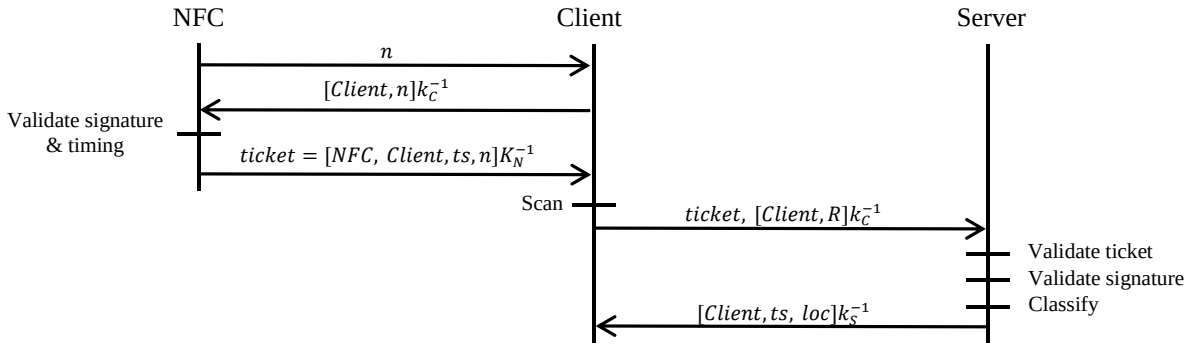

Figure 5: NFC-based protocol to protect against wormhole attacks.

wormhole attacks. We now describe two enhancements to LocAssure aimed at overcoming this limitation.

Preventing Replay Attacks

Replay attacks against LocAssure are made possible due to the relative stability of the BLE infrastructure visible at a given location over time. As a result, the <UUID, major ID, minor ID, RSSI> tuples comprising a location report are also relatively stable and easily replayed by miscreant users wishing to forge their current location. To combat these attacks, we must increase the entropy of these reports to make replay and context guessing attacks [17] more difficult to carry out.

We thus modified our system to randomly perturb the BLE advertisements transmitted by our beacons. The iBeacon-compatible advertisements transmitted by our beacons contain a fixed iBeacon prefix, a 128-bit UUID, a 16-bit major ID, and a 16-bit minor ID. Since altering the UUID transmitted by a given beacon would have the effect of confusing mobile apps attempting to leverage iBeacon functionality, and the location classification engine is already leveraging the minor ID, we instead perturb the 16-bit minor ID of the advertisement.

Recall from the Infrastructure Section above that the LocAssure server asynchronously updates beacons on a configurable basis (typically every two minutes in our deployment). As part of this update process, the LocAssure computes a randomized number to be placed in the minor ID field on a per-beacon basis according to the following formula:

$$HMAC - SHA1(mk, mac_i, ts)[1 \ldots 16]$$

The above extracts the first 16-bits of the output of the HMAC digest of a 160-bit master key known only to the LocAssure, the MAC address of a particular beacon, and the current timestamp. Each beacon thus receives a new, unique, and cryptographically randomized minor ID at each update cycle. Although the strength of protection afforded by this randomized minor is dependent on the frequency of updates and the number of beacons seen in a given location report, we will see in the Analysis section that this simple mechanism is a key building block in providing unforgeable proofs of location in the S-R-*-* and S-U-*-* settings.

Preventing Tunneling and Collusion

Note that even with the randomized minor IDs described above, it is possible for an off-site entity to collude with an on-site entity to create a forged location proof: the on-site entity can simply create a location report and forward it to the off-site entity, who can then forward it to the LocAssure server to forge a proof of location. To combat this, we must create a binding between the device requesting a proof of location, and the physical space containing the target (fine-grained) location.

Figure 5 describes a protocol similar to that described in [19] that creates this binding via the use of NFC. To enable the use of this protocol, the LocAssure infrastructure must contain an additional (coarse) deployment of NFC-equipped stations that can be used to localize a device to a region within a building. The NFC station first transmits a random nonce to the client, who signs and returns the nonce to the NFC station. The NFC station checks (i) the validity of the signature and (ii) that the time taken by the client to generate the signature is small enough to preclude wormhole attacks to an off-site signer (e.g., < 8ms). The NFC station then generates a coarse location *ticket* for the client that includes the identity of the NFC station, the client identity, the random nonce, and the timestamp at which the ticket was generated, which proves that the device was present at the coarse location of the NFC station at the specified timestamp.

The client device can send the location ticket and a location report to the LocAssure server, which can validate the ticket to ensure that the device whose coarse presence was proved via NFC matches the device that signed the location report that will be used to classify a fine-grained location. As we will see in the Analysis Section, the combination of this mechanism and the randomized addressing scheme described above enables the successful construction of unforgeable proofs of location in the S-U-I-* settings.

Figure 6: A map showing locations considered in our initial deployment. Blue dots indicate a BLE beacon. Each beacon location corresponds to a location considered for classification.

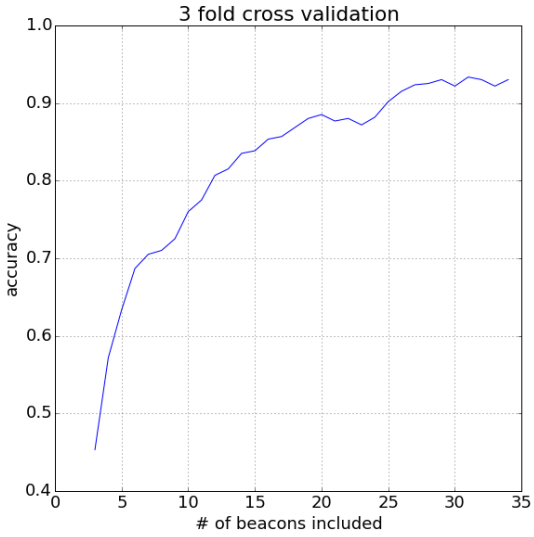

Figure 7: Location classification accuracy (y-axis) versus the number of deployed BLE beacons (x-axis).

ANALYSIS

We now explore the impact of beacon density on the entropy provided by our randomized advertisement approach, and assess the security guarantees of the protocols proposed in this paper.

Beacon Density Analysis

As detailed above, the strength of protection provided by our scheme depends in part on the number of beacons that are visible in a location. To better gauge this parameter, we've deployed 35 BLE beacons one each per location of interest in a portion of our office depicted in Figure 6. Each beacon's location is indicated by a blue dot. Note that the deployment includes sub-room locations in an open space that is divided in 3x3 grid with 2.5m spacing. We consider a total of 23 rooms, most of which are 9 m^2 square offices. The hallways are 1.5 meters wide, and the deployment also includes a conference room and kitchen (upper left).

We collected several sets of statistics to better ground parameters governing the strength of our security protections. The dataset we assembled was collected using an Android mobile war drive, and contains 20 scans per location for a total of 695 scans. Using all of the BLE beacons for classification, we achieved an accuracy of 93% in a three fold cross validation experiment. Simply using the map of Figure 6 and estimating the client device's location as the location corresponding to the BLE beacon with highest RSSI in each test scan produces a classification accuracy of 59% in our deployment.

On average, 29.81 beacons were visible in the scans taken from each location (s.d. = 3.17, min=23.2). These numbers provide a reference set of parameters for security analysis.

Despite the low cost of the beacons employed, we recognize that our deployment is dense, and thus simulated a lower spatial beacon density by pruning beacons from the data set and repeating the analysis. For this we greedily removed beacons based on their spatial locations so as to minimize the average distance between the centroid of each location and any remaining beacon. The simple idea is to maintain spatial coverage over a range of beacon densities. Figure 7 shows the 3-fold cross validation accuracy variation as the number of beacons available to the LCE is varied to illustrate this tradeoff.

Using the same simulation approach, we re-computed statistics for the number of visible beacons per location. Here again, we average the number of visible beacons per-scan, per-location. We then compute final statistics over the set of locations. The results as the number of available beacons is varied are shown in Figure 8 in which the mean and minimum values are depicted by green and blue bars, respectively.

Security Analysis

We now describe how the protocols proposed in this paper can protect against replay, context-guessing, and wormhole attacks, and address the combinations of protocols that can be used to produce unforgeable proofs of indoor location in the settings parameterized by the location taxonomy outlined earlier in this paper.

Replay and Context-Guessing Attacks

Cryptographically varying the minor IDs advertised by the network of smart beacons provides sufficient entropy to protect against replay and context-guessing attacks. In particular, the collision-resistance property of the underlying cryptographic hash function (SHA1) used by LocAssure ensures that all minor numbers *will* change at each update,

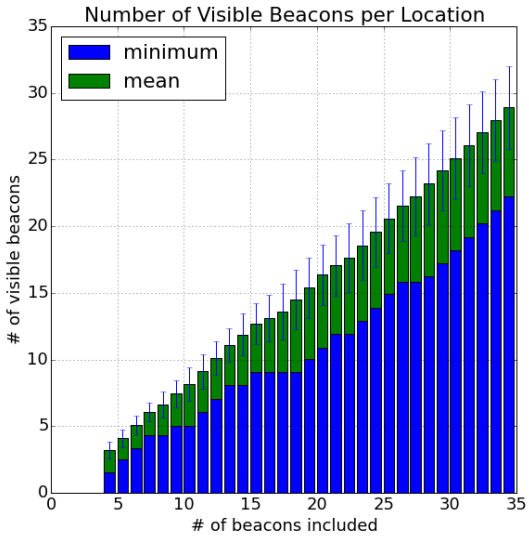

Figure 8: The graph depicts the average (green) and minimum (blue) number of beacons visible per scan as the number of BLE beacons deployed (x-axis) is varied.

which prevents naive attacks that simply replay old scan reports.

As shown in Figure 7, achieving > 90% accuracy in LocAssure within our building requires a network of at least 26 smart beacons. Under these circumstances, Figure 8 shows that a mean of 21 beacons (minimum of 15) are visible any given location. This means that a scan report will contain a mean of 336 (minimum of 240) bits that randomize at each beacon update. The pre-image resistance property of the underlying cryptographic hash function used by LocAssure ensures that the master key used to randomize the minor numbers cannot be discovered by observing the sequences of minor numbers advertised by LocAssure beacons. Without this master key, new minor numbers cannot be predicted in a systematic manner. As a result, context-guessing attacks will succeed with an average probability of $1/2^{336}$ (minimum of $1/2^{240}$).

Wormhole Attacks

The combination of randomized minor numbers and NFC device localization is sufficient to prevent wormhole attacks in which an individual on-site colludes with an adversary off-site to create a forged proof of location. Provided that clients do not share private keys—a reasonable assumption if these keys are used for purposes outside of LocAssure, e.g., signing purchase orders—the first two messages of the protocol described in Figure 5 create a binding between the client and a (coarse) physical space: the nonce *n* is unpredictable and the signature generated by the client must be returned within a time window that precludes collusion with an off-site adversary. The ticket returned by the NFC station (message 3) then binds the client identity to the NFC station's coarse location at the time *ts*. During the client's request for a proof of location (message 4), the LocAssure server can ensure that the client signing the location request matches the client identity in the NFC ticket. If the timestamp in the ticket is recent and the scan report generated by the client has accurate minor numbers, the client can be assured to be coarsely on-site and observing recent BLE advertisements, so a fine-grained location can be classified and returned by LocAssure (message 5).

		Trusted	Replay	Untrusted
Identifiable	**Immediate**	BS	RS	RNS
	Retroactive	BS	RS	RNS
Masked	**Immediate**	B	BR	
	Retroactive	B	BR	

Table 2: Protocol coverage

Scenario Coverage

Table 2 summarizes the scenarios in which LocAssure can create unforgeable proofs of fine-grained, indoor location using the protocols described in this paper. In this table, B denotes the base beaconing protocol (no randomization), R denotes the randomized beaconing protocol, N denotes the NFC protocol, and S denotes the variants of these protocols that include client signatures on localization requests. In the S-T-*-* settings, the base protocol can be used to localize clients, as clients are trusted not to replay, context guess, or collude. If client identifiablity is required, client signatures should be used (as in Figure 4), otherwise these can be omitted to allow clients to remain masked (as in Figure 3). In the S-R-*-* settings, the addition of randomized minors will ensure that replay and context guessing attacks can be prevented as discussed above; signatures can, again, be used if client identifiability is required. Finally, the S-U-I-* settings can be addressed by using randomized minors in conjunction with our NFC-based protocol, as discussed above. In all settings, location proofs contain a signed timestamp indicating their time of issuance by LocAssure; as such, they can be used either immediately or retroactively.

DISCUSSION

The LocAssure infrastructure and security protocols, when combined with our analysis of feasibility and performance, demonstrate a robust and practical system for assured indoor localization. While this work is not the first to address secure indoor localization, we believe our approach is the first that provides a holistic approach that enables immediate or retroactive proofs of location while supporting a deployment-tunable variety of assumptions, which include varying client-side adversarial behavior and client trust. In addition, with API access provided to third-party applications, we believe LocAssure is a major step forward in providing a platform to support a wide variety of applications that leverage secure indoor localization.

As illustrated by Tables 1 and 2, many different applications can be supported by LocAssure. In a workplace setting,

LocAssure could enable many new forms of access control to virtual assets like digital documents. Document viewing applications that leverage LocAssure could provide fine-grain controls over not only who has access to a document, but also where documents can be viewed. For instance, viewing of personnel documents could be restricted to individuals in Human Resources and only when they are in their assigned offices. Access from open spaces, like break rooms, could be prevented. Such controls could prove valuable as more and more documents are viewed and managed on mobile phones and tablets.

Beyond virtual access control, LocAssure could enable new applications for *physical* access control. For instance, when coupled with a network-enabled door lock, LocAssure could enable office owners to provide access to other colleague with the assurance that the colleague can only access the office when he or she is physically present. There are also similar applications of LocAssure to asset control and inventory management.

While these applications demonstrate new opportunities enabled by LocAssure, it is important to point out that limitations exist. One burdensome limitation comes from the underlying location classification technique. While it is high in accuracy, deployment requires that a fingerprint be collected from each room/location in which the system is expected to classify. The indoor location research community has proposed several methods for reducing this burden, including modeling the fingerprints [14] and crowd sourced, interactive labeling of ground truth [6].

Fingerprinting could be avoided altogether if location precision constraints are relaxed. Specifically, the location classification could simply be a function of receiving advertising packets from *n* required beacons. Thus, if a device is within range of the required beacons the location proof protocols above could still apply. However, location would be coarse, and the system could only restrict access to zones or portions of a location (e.g. a particular wing or floor of a building).

Tables 1 and 2 also illustrate that there are some use cases not supported by the LocAssure system. Particularly, situations where the client device is untrusted and needs to stay anonymous cannot be supported under the current set of protocols. While a few application cases may exist in this space (e.g. anonymous, but fair coupons, transit fare), we believe that the majority of useful applications across a variety of use contexts require some element of client identity. Beyond location assurance, identity is typically requisite for access lists, attribute identification, audit trails, and more. Thus, identity is often a constraint of the *application*, not just the location system used. However, our system could likely be extended using privacy-friendly cryptographic techniques (e.g., as in [7]), rather than the pervasively deployed public key cryptography techniques used in this paper.

Finally, we acknowledge that LocAssure has a large technology footprint. Specifically, a large number of beacons must be deployed to support a high degree of location certainty and to provide a high degree of cryptographic strength for some of the extension protocols proposed. Despite this, we believe the advantage of being able to work with off-the-shelf end-user devices (e.g. Android and iOS phones and tablets) provides a significant advantage over lighter, custom hardware solutions.

CONCLUSION AND FUTURE WORK

Location is becoming a driving component in many modern mobile computing applications. To be truly successful, a large number of these applications require that the location of the client device be trusted. In this work, we present LocAssure, a system that provides a broad set of security and privacy affordances for secure location estimation. Further, the system that we propose is compatible with existing and popular location technologies and protocols (e.g. Bluetooth low energy and Apple iBeacon).

In this paper, we have provided a multifaceted exploration of secure location, including providing a taxonomy that break down security, privacy, and infrastructure across several practical dimensions, we demonstrated a location infrastructure designed to support many of these different dimensions, we describe and prove effective security protocols on top of our proposed infrastructure, and finally we show how the protocols and infrastructure perform in a representative deployment.

Our work also serves as a starting point for several future directions. First, we intend to deploy LocAssure more broadly, exploring how the system performs across different physical configurations and beacon densities. We also look to explore deployments where location accuracy, and thereby location assurance, varies within the deployment. For instance, a deployment where some "secure" rooms have high location resolution and security affordances, while other "less secure" rooms provide relaxed location and security controls. Finally, we plan to explore the utility of LocAssure through the development of several applications, which include physical access control and document security.

ACKNOWLEDGMENTS

Removed for blind review.

REFERENCES

1. ElectricImp. http://electricimp.com/
2. iBeacon for Developers. https://developer.apple.com/ibeacon/
3. Aparicio, S., Perez, J., Bernardos, A., Casar, J.: A fusion method based on Bluetooth and wlan technologies for indoor location. In Proceedings of IEEE Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI 2008). pp. 487-491.

4. Bahl, P.; Padmanabhan, V.N., RADAR: an in-building RF-based user location and tracking system. *In Proceedings of the Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings (INFOCOM 2000).* pp.775-784.

5. Biehl, J.T., Cooper, M., Filby, G., Kratz, S. LoCo: A Ready-to-Deploy Framework for Efficient Room Localization using Wi-Fi. *In Proceedings of the ACM Conference on Ubiquitous Computing (Ubicomp 2014)*.

6. Bolliger, P. Redpin - adaptive, zero-configuration indoor localization through user collaboration. *In Proceedings of the ACM international workshop on Mobile entity localization and tracking in GPS-less environments.* (MELT 2008). pp. 55-60.

7. Carbunar, B., Sion, R., Potharaju, R., Ehsan, M. The shy mayor: Private badges in geosocial networks. *In Applied Cryptography and Network Security (ACNS 2012)*.

8. Fan, M., Liu, Q., Tang, H., Chiu, P. HiFi: Hide and Find Digital Contents Associated with Physical Objects via Coded Light. In Proceedings of the Workshop on Mobile Computing Systems and Applications (HotMobile 2014).

9. Freund, Y., Schapire, R.: A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences* 55(1), 119–139 (1997).

10. Harter, A., Hopper, A., Steggles, P., Ward, A., Webster, P. The anatomy of a contextaware application. *Wireless Networks* 8(2/3), 187-197 (2002).

11. Hastie, T., Tibshirani, R., Friedman, J., *The Elements of Statistical Learning*, Springer, New York, 2001.

12. Hightower, J., Want, R., Borriello, G.: Spoton: An indoor 3d location sensing technology based on RF signal strength. UW CSE 00-02-02, University of Washington, Department of Computer Science and Engineering, Seattle, WA 1 (2000).

13. Hu, Y.C., Perrig, A., Johnson D.B. Wormhole attacks in wireless networks. *In IEEE Journal on Selected Areas in Communications, 24.2* (2006): 370-380.

14. Li, L. Shen, G., Zhao, C., Moscibroda, T., Lin, J.H., Zhao, F. Experiencing and handling the diversity in data density and environmental locality in an indoor positioning service. In *Proceedings of the 20th annual international conference on Mobile computing and networking (MobiCom 2014)*. pp. 459-470.

15. Luo, W., Hengartner, U. Proving Your Location Without Giving Up Your Privacy. *In Proceedings of the Workshop on Mobile Computing Systems and Applications (HotMobile 2010)*.

16. Manweiler, J., Scudellari, R., Cox, L.P. Smile: Encounter-based trust for mobile social services. *In Proceedings of the 16th ACM Conference on Computer and Communications Security (CCS 2009)*.

17. Miettinen, M., Asokan, N., Koushanfar, F., Nguyen, T.C., Rios, J., Sadeghi, A.R., Sobhani, M., Yellapantula, S. I Know Where You Are: Proofs of Presence Resilient to Malicious Provers, *In Proceedings of the 10th ACM Symposium on Information, Computer and Communications Security (ASIACCS 2015)*.

18. Moors,T., Mei, M., Salim, A. Zone-IT: Using short range communication to control mobile device functionality. *Journal of Personal and Ubiquitous Computing, 2008*.

19. Polakis, J., Volanis, S., Athanasopoulos, E., Markatos, E.P. The man who was there: Validating check-ins in location-based services. *In Proceedings of the 29th Annual Computer Security Applications Conference (ACSAC 2013)*.

20. Portnoi, M., Shen, C.C., Loc-Auth: Location-Enabled Authentication Through Attribute-Based Encryption. *To appear, International Conference on Computing, Networking and Communications (ICNC 2015)*.

21. Rodrigues, M.L., Vieira, L.F.M., Campos, M.F.: Fingerprinting-based radio localization in indoor environments using multiple wireless technologies. *In Proceedings of the IEEE International Symposium on Personal Indoor and Mobile Radio Communications (PIMRC 2011)*, pp. 1203.

22. Saroiu, S., Wolman, A. Enabling new mobile applications with location proofs. *In Proceedings of the 10th Workshop on Mobile Computing Systems and Applications (HotMobile 2009)*.

23. Want, R., Hopper, A., Falc~ao, V., Gibbons, J.: The active badge location system. ACM Transactions on Information Systems (TOIS) 10(1), 91-102 (1992).