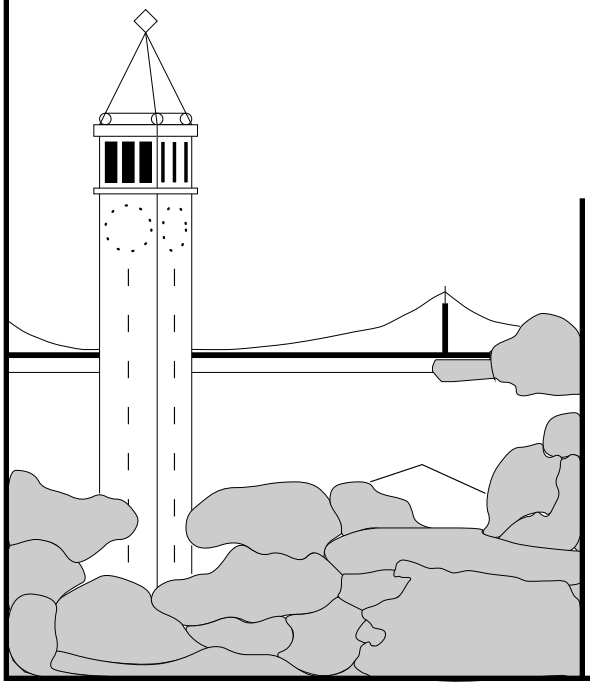
Secure Verification of Location Claims

Naveen Sastry *Umesh Shankar* *David Wagner*



Report No. UCB//CSD-03-1245

Computer Science Division (EECS)
University of California
Berkeley, California 94720

Secure Verification of Location Claims

Naveen Sastry Umesh Shankar David Wagner*
University of California, Berkeley
{nks, ushankar, daw}@cs.berkeley.edu

Abstract

With the growing prevalence of sensor and wireless networks comes a new demand for location-based access control mechanisms. We introduce the concept of secure location verification, and we show how it can be used for location-based access control. Then, we present the Echo protocol, a simple method for secure location verification. The Echo protocol is extremely lightweight: it does not require time synchronization, cryptography, or highly accurate clocks. Hence, we believe that it is well suited for use in small, cheap, mobile devices.

1 Introduction

Computer scientists are used to studying access control mechanisms where one's identity determines what one is authorized to do. However, in the physical world, identity is not the only thing that matters: often, the physical location of the requester may also play an important role in determining access rights. This suggests studying location-based access control.

Location-based access control in the physical world is easy, natural, and familiar. For example, being able to turn on or off the lights in a particular room *requires* having a physical presence in the room. The very design of the light switch is what enforces the security policy. In contrast, achieving the same kind of guarantee with information systems, such as wireless networks, is less straightforward; it is not simply a matter of putting a switch in the right place. To enforce location-based access control policies on information resources, we need a way to perform *location verification*, where a principal's location is securely verified to meet certain criteria: e.g., being inside a building.

Location verification enables location-based access control. Once a principal's location has been verified using a protocol for location verification, the principal can be granted access to a particular resource according to the desired policy. This approach is naturally combined with physical security; guards or locks might be used to determine who is allowed to enter a building, then location verification employed to allow wireless access to all those inside. In this way, the location verification problem is the key technical challenge that must be surmounted to implement location-based access control.

Location-based access control has several benefits. Most importantly, it is natural for many applications. One simple policy might allow wireless control of only the lights for the room you are in, or might insist that a company server cease operating if it is taken outside the building. In addition, using location for access control obviates the need to establish shared secrets in advance. Visitors to a building need not obtain wireless encryption keys prior to their visit; instead, the keys could be granted automatically to all physical occupants of the building. Likewise, at the ballpark, fans at a baseball game could receive live scorecards on their wireless devices, while stadium owners could restrict this service to only those actually present in the stadium. It would be quite cumbersome to distribute new keys to all fans attending each game, but location-based access control allows bootstrapping off the existing physical security measures controlling entry to the premises.

In this paper, we study the location verification problem. First, we introduce and define the location verification problem in a careful way (Section 2). Then, we propose a new protocol for location verification, called *the Echo protocol* (Section 3), and we prove its security (Section 4). This work provides a foundation for securely using location in wireless information systems.

2 Goals and Assumptions

2.1 Problem Statement

There are many natural variants of the secure location problem. We focus on solving the *in-region verification* problem: a set of *verifiers* V wish to verify whether a

*This work was supported in part by DARPA NEST contract F33615-01-C-1895, NSF CCR-0113941, and an equipment donation from Intel.

prover p is in a region R of interest. R may be a room, a building, a stadium, or other physical area. The region typically has some sort of physical control to restrict people's entry into it; the purpose, then, is to control access to resources that are not intrinsically constrained by physical security, such as wireless networks. The verifier infrastructure V may, in some cases, be a distributed system consisting of multiple nodes.

The protocol must run correctly in the face of adversaries. Thus, when p does not in fact have a physical presence inside R, the verifier must be careful not to accept p's claim to be in R. Furthermore, if p does have a presence in R, the verifier should accept p's claim; otherwise the protocol would not be useful in practice. We therefore require the following two properties to ensure that the protocol is useful and secure:

- *Completeness:* If p and V both behave according to the protocol, and p is in R, then V will accept that p is in R.

- *Security:* If V behaves according to the protocol and accepts p's claim, then p, or a party colluding with p, has a physical presence in R.

It is important to distinguish between the problem we are addressing, the in-region verification problem, and the *secure location determination problem*. In the latter problem, V attempts to securely discover the physical location of p. In contrast, in the in-region verification problem, p claims to be in a particular region, and V accepts or rejects the claim. Framing the problem in terms of secure in-region verification, not secure location determination, simplifies the problem and allows different location determination algorithms to be used.

In fact, it is possible to compose an in-region verification protocol with any location determination algorithm, even a potentially insecure one, without compromising the security of the ultimate guarantee that a prover is in the region. The in-region verification algorithm verifies whether the claimed location is in R or not; thus, p can use an insecure localization algorithm to generate a claimed location that will be securely tested for accuracy by V. At worst, p's claim will be rejected; in no case will V believe something about p's location that has not been securely checked. p thus has the flexibility to choose any appropriate location determination algorithm, even if it has not been proven secure. After running the determination algorithm, p will know which claims it can plausibly make.

2.2 Assumptions

It is worth considering in more detail what our particular protocol is and is not attempting to do. We list below some of our assumptions:

- **Regions, not points.** We are not attempting to verify the exact location of the prover. In other words, the locations claims we verify are not claims of particular *point* locations (plus or minus some error distance), but rather just presence in a particular region of interest R. This model accords well with our anticipated applications. We assume that, before the verification protocol begins, both the prover and verifier know the definition of the region R.

- **Only "local" regions.** It is not a requirement to verify *all* location claims; and indeed, there are some location claims we do not even attempt to verify. More specifically, we only attempt to verify location claims for regions R that are "near" V. We will explore more precisely what this means in Sections 3 and 4. The restriction makes sense in light of the proposed application domains: if you have sensors scattered through a building, you are typically not interested in regions that are outside the building.

- **RF and sound capability.** The verifier and prover must both be able to communicate using both radio frequency (RF) and sound (typically ultrasound frequencies). We will use both transmission media in our protocol.

- **Bounded processing delay.** The prover must be able to bound its processing delay. We will describe the effects that a loose bound will have on the protocol in Section 4.

2.3 Threat Model

In order to verify the security property, we must consider the protocol with respect to a particular threat model. We assume the verifier nodes are all trusted, and they can communicate securely amongst themselves. In contrast, the prover p might behave maliciously, and we will consider an adversarial prover consisting of multiple colluding nodes, arbitrary computing power, and secure RF (speed of light) communication amongst its own nodes as well as sound generation and detection capability on each of its nodes.

Lastly, by definition, the adversary must not actually have any presence in the region R. Otherwise, it would be able to make a legitimate claim and would not need to attack the protocol.

2.4 Design Principles

We designed our protocol according to the following design principles:

- **Make few resource demands on the prover and verifier.** We would like to limit the computation power and hardware resources necessary to participate in the protocol to an absolute minimum. The real goal is to enable location proofs for a large class of devices.
- **No prearranged setup.** It should not be necessary for the prover to have previously engaged in a setup or registration step with the verifier. This excludes many cryptographic solutions; even public-key cryptography requires prearranged trust relationships, and thus is not suitable for our purposes. By eliminating the setup step, we are enabling access to resources to be granted based on physical presence alone.
- **Quantitative guarantees.** We would like to provide precise bounds on the uncertainty in the protocol.

2.5 Design Setting

We initiated this work primarily in the context of nodes such as those found in sensor networks. This choice imposes certain design constraints. Briefly, sensor networks are composed of many small, cheap nodes equipped with a variety of environmental sensors. Examples include accelerometers, microphones, and thermometers. The nodes contain a general purpose CPU, though it is often useful only for minimal computation. Finally, the nodes communicate using a wireless network over distances of tens of meters. Thus, the sensing capabilities of sensor networks can be used to help bridge the physical-computational gap.

One consequence of considering this domain is that many techniques, such as public-key cryptography, are infeasible. The Berkeley Mica sensor nodes, for example, have 4MHz 8-bit processors with 4KB of RAM [11]. What we need, then, is a lightweight way to perform location verification given many sensor-class nodes. The principal trying to prove its location need not be a sensor-class node, though we do not assume that it is more powerful. The fact that our protocol can work within these tight constraints makes our results are all the more meaningful, and we expect that the Echo protocol will be broadly applicable to sensor networks, networked embedded systems, ubiquitous computing, wireless networks, and many other similar application settings.

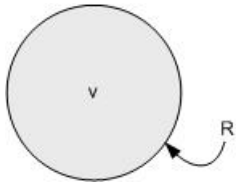


Figure 1: An illustration of our first simplification of the problem. The prover (not shown here) will try to convince the single verifier node v that it is inside the region R (depicted as a shadowed circle, which in this first scenario is assumed to be centered at v).

2.6 Our Contributions

We present the problem description for a secure in-region verification protocol and argue that it is a better model than trying to solve the secure location determination problem directly. We also present a provably secure protocol for performing in-region verification that can be run on minimal hardware.

3 Our Design: The Echo Protocol

Next, we describe the design of our proposal for location verification, which we dub the Echo protocol. For expository purposes, we start by considering a simplified toy scenario and developing a simple protocol for this scenario (Section 3.1); then, we extend it repeatedly (Section 3.2) until we obtain the full protocol (Section 3.3).

Notation. We define s to be the speed of sound, or 331 m/s. Likewise, we will take c to be the speed of light, or 3×10^8 m/s. Define $d(x, y)$ to be the distance between x and y. We define R to be the area in which we would like to verify the location of a prover p. The set of all verifier nodes is denoted by V.

3.1 Protocol Intuition

Consider first a simplified case, where we have only a single verifier node v, where the region R is a circle[1], and where this circle is centered at v. This scenario is shown pictorially in Figure 1. Now, suppose that the prover claims to be at some location $\ell \in R$ inside the region.

We present a simple protocol for validating the location claim in this restricted case. First note that if

[1]In practice, the region will likely be a sphere, instead of circle. This simplification will make the protocol easier to understand and does not affect the validity of our results.

1. $p \xrightarrow{\text{radio}} v : \ell$
2. $v \xrightarrow{\text{radio}} p : N$
3. $p \xrightarrow{\text{sound}} v : N$

v accepts iff $\ell \in R$ and
elapsed time $\leq d(v, \ell) \cdot (c^{-1} + s^{-1})$.

Figure 2: A protocol that solves our first simplification of the problem.

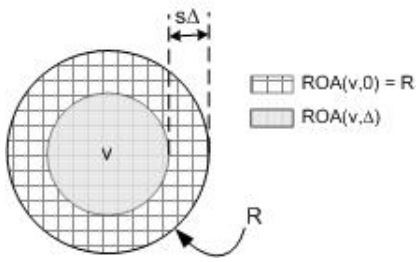


Figure 3: Diagram illustrating a single verifier at the center of a circular region R where the prover has an upper bound of Δ_p on its processing delay. The diagram illustrates the relationship between $\mathsf{ROA}(v, \Delta_p)$ and $\mathsf{ROA}(v, 0)$, which is equal to R in this case.

the claimed location ℓ is not inside R, then the verifier can reject the claim immediately. Thus, we may safely assume that the prover claims to be inside R. The protocol begins when the verifier node v sends a packet containing a nonce to the prover using RF; the prover immediately echoes the packet back to the verifier using ultrasound. The verifier node v can then calculate how long it should take to hear the echo, namely, the sum of the time it takes to reach ℓ using RF, plus the time it takes for a return packet to go from ℓ to v using ultrasound. Thus, the total elapsed time for the prover to hear the echoed nonce should be about $d(v, \ell)/c + d(v, \ell)/s$. The only thing v has to do is time this process: If the elapsed time from the initial transmission to reception of the echo packet is more than this amount, the verifier node v rejects the prover's claim; otherwise, if the elapsed time is at most this expected amount, v accepts. This protocol is summarized in Figure 2.

Why does this work? If the prover is able to return the packet in sufficient time, then the verifier is assured that the prover is within $d(v, \ell)$ units of v. This means that ℓ is known to be inside a circle of radius $d(v, \ell)$ centered at v. Call this circle C; then we know $\ell \in C$. Since R is defined to be a circle of radius at least $d(v, \ell)$ centered at v, we have $C \subseteq R$, and hence $\ell \in R$. In short, we know that the prover must be inside R.

If the prover cannot return the nonce in sufficient time, it may be for one of two reasons. Either the prover is more than $d(v, \ell)$ away from v, or the prover has some processing delay between receiving the RF packet and returning the ultrasound packet. We will explore this latter issue in the following section.

What if the prover tries to cheat by delaying his response? This attack only increases the total elapsed time of the process, thereby making the verifier reject. Intuitively, the longer it takes to complete the protocol, the farther away the prover appears to be. It is not in the prover's interest to appear to be farther from v, because this will put the prover's apparent location outside of R, hence making v reject the prover's claim.

Can the prover cheat by starting the transmission of the response early? No, this attack is not possible. The nonce in the packet prevents the prover from sending a reply before it has received the outgoing RF packet. Hence, the speed of light and sound prevents the prover from pretending to be closer to v than he really is.

3.2 Processing Delay & Nonuniform Regions

In this section, we present a slightly more advanced protocol that addresses two additional issues: the fact that the prover has a non-zero processing delay Δ_p and that R may not be a circle. We base this protocol on the simple protocol presented in the previous section.

Processing delay. Let us first address the processing delay. We will start with the configuration mentioned in Section 3.1 with a single verifier located at the center of a circular region R. In the ideal case, the prover can receive the RF packet from the verifier node and send out the response over ultrasound instantly; practically, this is not possible, as the prover will require some time to process the incoming packet. Suppose the prover can bound its processing delay to be at most Δ_p and makes the verifier node aware of this maximum delay. Then, if the prover claims to be at ℓ, the verifier node can compute the time for a prover actually at ℓ to get the packet back: the time for the RF signal to travel from v to ℓ, a processing delay of at most Δ_p, and finally the time for the sound to travel from ℓ back to v.

Now we have a problem: A malicious prover could submit a location claim ℓ at the border of R and grossly overstate its true processing delay to be some very large Δ_m. If, however, the prover's true processing delay were zero, then it could fool the verifier node into thinking that it was inside R when in fact it wasn't.

Since the verifier allows up to Δ_m processing delay while the adversary has no delay, the adversary could be $\Delta_m \cdot (c^{-1} + s^{-1})^{-1} \approx \Delta_m \cdot s$ units outside of R and the verifier would still accept the claimed location, violating our security condition.

The solution to such a problem is for the verifier node to shrink the allowable region in which location claims are accepted. If the prover claims a maximum processing delay of 0, then the protocol presented earlier in Section 3.1 is sufficient. If, however, the prover claims a processing delay of $\Delta_p > 0$, the verifier should not engage in the protocol if the claimed location ℓ is within $\Delta_p \cdot s$ of the outside border. Thus, we define the term *Region of Acceptance* (ROA) to be the area in which the verifier node v is sure that it can correctly verify claims for a prover. Note that this region depends on Δ_p. We write $\mathsf{ROA}(v, \Delta_p)$ to indicate the region where location claims are permitted by v, if the claimed processing delay is Δ_p. See Figure 3 for an illustration.

As stated above, $\mathsf{ROA}(v, \Delta_p)$ is a circle centered at v and fully contained within R. Its radius is $\Delta_p \cdot s$ less than R's radius (since R was assumed to be a circle). Amending our prior protocol, the verifier should engage in the protocol only if the location claim ℓ is within $\mathsf{ROA}(v, \Delta_p)$. For $\Delta_p = 0$, we have $\mathsf{ROA}(v, 0) = R$, and so the simple protocol presented earlier is a special case of the amended protocol.

We note at this point that if the prover has a processing delay of Δ_p, the protocol is not *complete*. Recall that the completeness condition from Section 2.1 requires that the verifier always accept if the prover is inside R and behaving properly. Yet, for a processing delay of Δ_p, our verifier will not accept location claims that are in the annulus $R \backslash \mathsf{ROA}(v, \Delta_P)$, so our protocol cannot be fully complete.

This suggests an alternate way to view $\mathsf{ROA}(v, \Delta_p)$: it is the region for which the protocol is complete. In other words, $\mathsf{ROA}(v, \Delta_p)$ is the region where a verifier will accept location claims from a correctly functioning prover with processing delay less than Δ_p. We will define the *coverage* of the ROA as the ratio between the area of the ROA and the area of R. A coverage of 100% indicates that the protocol is complete; a coverage of less than 100% indicates only partial completeness.

Non-circular regions. Up until now, we have been assuming that R is a circle centered at v. However, that is not always a realistic assumption: perhaps we are interested in verifying location claims in a square room, for instance. We will now relax that assumption and assume that the verifier node is contained somewhere within an arbitrarily shaped region R. This causes a larger area to be *incomplete*, or non-verifiable, as shown in Figure 4. We will address incompleteness in the next section with our final iteration of the protocol.

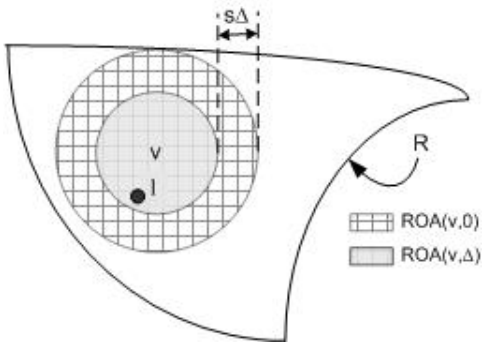


Figure 4: A single verifier v, inside a irregular region R. We are interested in proving that the prover is within R. The larger circle represents $\mathsf{ROA}(v, 0)$, the area in which v is useful for location verification proofs. This is the largest circle centered at v and wholly contained within R. The inner circle represents $\mathsf{ROA}(v, \Delta)$, the region in which v will accept location claims from a device that is able to bound its processing delay by Δ.

Previously, $\mathsf{ROA}(v, 0)$ had been equivalent to R. But this will not work when R is not a circle centered at v. Since we are assuming that radios are omni-directional, the ROA must be a circle. Furthermore, the ROA must be wholly contained within R. By definition, the ROA is the region where the verifier will accept a correctly functioning prover; if the ROA were not fully contained within R, the prover could accept a location claim for a prover outside of R. Furthermore, we would like to maximize the area of the ROA since a larger ROA leads to a larger coverage. Thus, $\mathsf{ROA}(v, 0)$ should be the largest circle that fits within R; in other words, it should be the largest circle that is tangent to R and still contained within it.

We now extend the protocol to handle non-circular regions R where the verifier can bound its processing delay to be at most Δ_p. Recall that both the prover node and verifier node know R *a priori*. Using this, the verifier node can compute ahead of time the region $\mathsf{ROA}(v, 0)$.

The protocol then proceeds as follows: the prover first broadcasts its claimed location ℓ and processing delay Δ_p to the verifier. If $\ell \notin \mathsf{ROA}(v, \Delta_p)$, the verifier should immediately reject the location claim since it will not be able to definitively validate the claim. Otherwise, the verifier node broadcasts a nonce to the prover; the prover echoes the nonce back over ultrasound. The verifier can again time the communication: if it is no greater than the time for the signal to travel out and back and allowing for processing delay, the

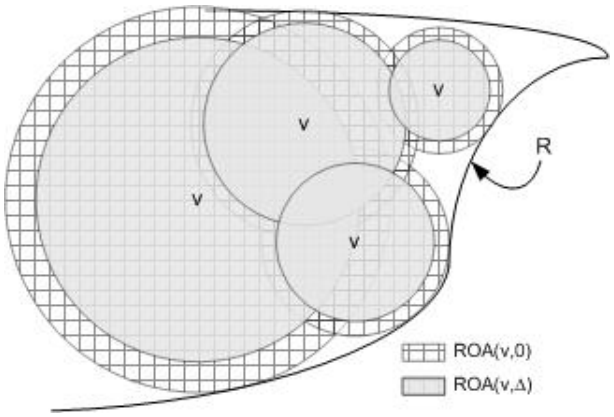


Figure 5: The relationship between $\mathsf{ROA}(v)$ (for a single verifier v) and the aggregate ROA. Each gray circle represents $\mathsf{ROA}(v, \Delta)$ for a particular verifier v. Taken collectively, the gray region represents $\mathsf{ROA}(\Delta)$, the aggregate region in which the set of verifiers can successfully verify the location of a prover that features a processing delay less than Δ. Note that $\mathsf{ROA}(\Delta)$ is wholly contained within R.

verifier accepts the location claim.

3.3 Full Protocol Description: The Echo Protocol

In the final iteration of the protocol, we introduce multiple verifier nodes in an attempt to increase the coverage of R. Recall that if R is not a circle, no single node can provide 100% coverage. Consequently, multiple verifiers are needed. Intuitively, we will run the protocol presented in Section 3.2 after selecting one verifier from among the set of verifiers V.

The protocol is quite simple. See Figure 6 for the complete definition. First, a verifier is chosen so that the claimed location ℓ lies within that verifier's ROA. If no such verifier exists, execution is aborted, since the claim can not be verified. After choosing a verifier v to participate, v sends a packet to p using RF, which is echoed back to it using ultrasound. v can calculate how long it should take to hear the echo, namely, the sum of the time it takes to reach ℓ using RF, plus Δ_p, plus the time it takes for a return packet to go from ℓ to v using ultrasound. If the measured elapsed time exceeds this anticipated time, v rejects the location claim. The nonce in the packet prevents the prover from sending a reply before it has received the outgoing RF packet.

The extra verifier nodes serve to expand the region of acceptance within R. Thus, while $\mathsf{ROA}(v, \Delta_p)$ refers to the region that one particular verifier node can accept, we define $\mathsf{ROA}(\Delta_p)$ to be the region where at least one

COMMUNICATION PHASE:

1. $p \xrightarrow{\text{radio}}$ broadcast : (ℓ, Δ_p).
   The prover broadcasts its claimed location ℓ and processing delay Δ_p.
2. $v \xrightarrow{\text{radio}} p : N$.
   A single verifier v responds with a random nonce. We require $\ell \in \mathsf{ROA}(v, \Delta_p)$.
   If no such verifier exists, **abort**.
3. $t_s \leftarrow$ **time** ().
   the verifier starts its timer.
4. $p \xrightarrow{\text{sound}} v : N$.
   The prover echoes the nonce over ultrasound.

VERIFIER COMPUTATION PHASE:

5. $t_f \leftarrow$ **time** ().
   The verifier records the finish time.
6. **if** sent nonce differs from received nonce
   return false
7. **if** $t_f - t_s > \frac{d(v,\ell)}{c} + \frac{d(v,\ell)}{s} + \Delta_p$
   return false
8. Otherwise, **return true**

Figure 6: Formal description of the Echo protocol, which can perform location verification in an arbitrary region R with multiple verifier nodes. We represent the prover node as p and the verifier node that runs the protocol as v.

verifier node can prove location claims. It is then clear that

$$\mathsf{ROA}(\Delta_p) \equiv \bigcup_{v \in V} \mathsf{ROA}(v, \Delta_p)$$

since the set of verifiers can accept a location proof if the claimed location is inside at least one verifier's region of acceptance.

In the Echo protocol, the infrastructure chooses a single verifier node to participate in the protocol. A verifier v may participate if $\ell \in \mathsf{ROA}(v, \Delta_p)$, since by definition that is the region that it can perform secure location verification proofs. Note that the claimed location ℓ may be inside $\mathsf{ROA}(v, \Delta_p)$ for many different verifier nodes v, hence more than one verifier node might be eligible for participation in the protocol. We only require one to be chosen, and we allow the verifiers to use some mechanism to choose which particular verifier node will run the protocol. They may have a purely deterministic mechanism for electing verifiers, or they may use a dynamic algorithm in an attempt to conserve power, for example.

4 Analysis

4.1 Security Analysis

As explained in Section 3, the Echo protocol relies on timing: the amount of time it takes to get a response from the prover bounds how far the prover can be from the verifier. We will now show that it is impossible for an adversary outside R to convince the verifier that it is in R.

Proof of security. The heart of the argument is that an attacker would not be able to get the sound signal to the verifier in time. In order to confirm that the prover is at ℓ, all a particular verifier node v must do is verify that the incoming sound signal, which includes the outgoing nonce, is received within

$$t_{\max} \leq \frac{d(v,\ell)}{c} + \frac{d(v,\ell)}{s} + \Delta_p \quad \text{seconds},$$

where $d(v,\ell)$ is the distance from the verifier to the claimed location, c is the speed of radio propagation (approximately the speed of light), s is the speed of sound, and Δ_p is the prover's processing delay. Recall that v agrees to run the protocol only if $\ell \in \mathsf{ROA}(v, \Delta_p)$, i.e., if the circle of radius $d(v,\ell) + \Delta_p \cdot s$ lies wholly within R. By definition, the attacker A is outside R; thus we have

$$d(v, A) > d(v,\ell) + \Delta_p \cdot s.$$

Since the attacker needs to receive the initial signal and respond, the minimum time required for the attacker to get a response to v is

$$\begin{aligned} t^{\mathrm{A}}_{\min} &= \frac{d(v,A)}{c} + \frac{d(v,A)}{s} \\ &> \frac{d(v,\ell) + \Delta_p \cdot s}{c} + \frac{d(v,\ell) + \Delta_p \cdot s}{s} \\ &\geq \frac{d(v,\ell)}{c} + \frac{d(v,\ell)}{s} + \frac{\Delta_p \cdot s}{c} + \frac{\Delta_p \cdot s}{s} \\ &\geq \frac{d(v,\ell)}{c} + \frac{d(v,\ell)}{s} + \Delta_p. \end{aligned}$$

Consequently, the attacker's signal cannot reach the verifier before the deadline. Note that nowhere in our analysis did we rely on *which* verifier node was used. The only difference would be in the magnitude of the error terms and, therefore, in the chance that the location claim would even be accepted for verification. The attacker does not gain any advantage by selecting a different verifier from the one elected to participate.

Attacks. One possible attack could exploit the difference in propagation speed of sound in different media. If the verifier's estimation of s is slower than the actual one, then the inequality above does not hold. If this is a valid threat model—say there is a lot of metal near the verification region that is capable of transmitting sound from the outside—then the verifier's estimation of s should be adjusted. This can be done once on a site-specific basis. An alternate defense would be to have other verifier nodes confirm the estimate of s based on when the sound signals are received.

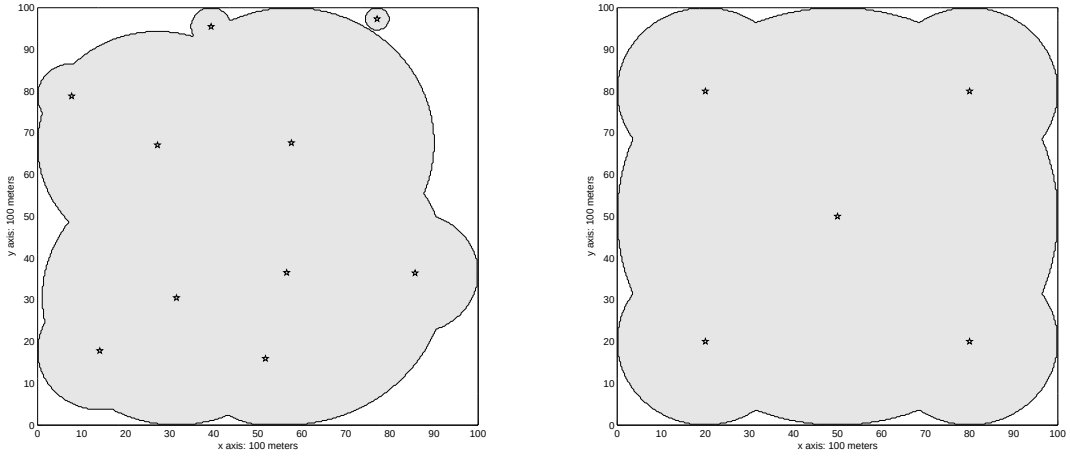
Variants. One might also consider the implications of other variants of the protocol, where the use of sound and radio for the outgoing and incoming signals is changed from (radio, sound) to (radio, radio), (radio, sound), or (sound, sound). If radio communication is used in both directions, then the error term $\Delta \cdot c$ would be very large (10^5 to 10^6 times as large as the sound case), and it is quite likely that the verifier would not accept location claims at all, since the error might exceed the size of R itself! If sound is used in the outgoing direction, one attack could be to use a laser-based "bugging" attack, where sound-induced vibrations inside R could be picked up optically from outside R, thus effectively speeding up the transmission speed of the sound wave and invalidating our proof above.

4.2 Coverage

The Echo protocol requires that all the verifiers must be inside the region of interest, R. Furthermore, the region of acceptance is a subset of the region of interest and is determined by the placement of the verifiers within R. Thus, a natural question to ask is how well the verifiers "cover" a given region R. We define coverage to be the fraction of the region in which successful location claims can be validated. Obviously, the ideal scenario would allow full coverage with only a few verifiers. In that case, the verifiers would be *complete*, according to our definition in Section 2.1. Recall that the completeness condition requires that the verifiers accept if the prover has a presence inside R. When the coverage is less than 100%, completeness is only partial.

To measure this effect, we ran simulations to measure the coverage in two scenarios with only a few nodes. In Figure 7(a), we placed 10 nodes in a 100m by 100m room. The nodes were randomly placed with the constraint that each node had to be 20m away from every other node. This condition shows the effects of sprinkling the nodes over the entire room. In this trial, the placement achieved a 81% coverage; averaging 5 trials yielded a mean of 78% coverage.

By manually placing 5 nodes, we were able to achieve a coverage of 93% in a similar room (see Figure 7(b)). Thus, with very few verifier nodes, the ROA covers a



(a) 10 randomly placed nodes; the nodes were constrained so that they were at least 20m away from all other nodes. This corresponds to a reasonable, but not precise, dispersement of nodes. These 10 verifiers covered 81.5% of the room.

(b) 5 manually placed nodes to maximize the coverage area (93.3%)

Figure 7: Simulation results showing the effective ROA coverage area that a few verifier nodes can achieve in a 100m by 100m square room. The simulations show that the Echo protocol is quite effective, even with only a small number of verifier nodes.

significant fraction of the region that we are interested in. The regions with the least amount of coverage are as expected: at the edges and corners of the region.

Hence, there are a number of policies that can help with verifier placement in the room; using these simple policies can cover significant fractions of R using only a few verifier nodes.

4.3 Evaluation

The Echo protocol is explicitly designed to make few demands on the prover and verifier.

First, we do not use cryptography. By avoiding the use of both public- and private-key cryptography, we achieve two goals. We lower the CPU and memory requirements on both the prover and verifier, and perhaps more significantly, we remove the need for any prior agreement between the prover and verifier with respect to keys or certificates. This means that if R is, say, a baseball stadium, then any fan attending the game with a suitable small device can act as a prover.

The approach taken by Waters and Felten [19] requires processing speeds fast enough that distance errors are small even with speed-of-light communication. In contrast, error in our scheme is correlated to the speed of sound, which is 10^5 to 10^6 times slower than radio communications. That means that a correspondingly greater processing delay can be tolerated, which is crucial, if low-cost devices are to participate.

Lastly, our protocol does not require time synchronization between any two nodes. It only requires that each node have a clock that can measure real time with some precision. For example, taking the speed of sound to be 331 m/s, each 1ms of clock skew in the receiver would increase uncertainty by about 1/3 of a meter. If the prover and verifier are 50m apart, the protocol runtime is about 150ms; clock skew is unlikely to be more than a few microseconds during this interval [8], so the uncertainty added would be on the order of millimeters, which is acceptable for our application domain.

5 Related Work

A number of authors have proposed using time-of-flight measurements and the speed of light to securely gain location information about untrusted parties. Brands and Chaum proposed a time-bounded challenge-response protocol [4] as a defense against man-in-the-middle attacks on cryptographic identification schemes. Hu, et al., proposed using temporal packet leashes for wireless networks to defend against

similar attacks [12]. However, a major limitation of these schemes is that both the prover and verifier send RF signals, requiring the access to a much more accurate timing system at the verifier as well as tight real-time processing guarantees on both the prover and verifier for accurate readings. For these reasons, we believe our algorithm is more suited to mobile devices.

Waters and Felten present a scheme that uses round-trip time-of-flight of RF signals to achieve goals similar to ours [19]. Their architecture is similar to ours, in that they, too, suggest focusing on secure location verification rather than on secure location determination. However, their use of RF seems likely to limit deployment, like the previous proposals mentioned above. Also, their system only proves the location of tamper-resistant, trusted devices.

Coarse-grained location authentication has been used in the television industry to prevent cloning of set-top boxes [9]. Gabber and Wool propose four coarse-grained techniques, relying on extensive telecommunications infrastructure such as satellites, paging and cellular networks. Their techniques rely on tamper-resistant hardware.

Location-limited channels provide a communication mechanism that is restricted to a short range and provides both endpoints a mechanism to guarantee the authenticity of each participant [16]. Balfanz, et al., have proposed using location-limited channels for location-based access control [3], and many others have also proposed use of limited-range radio broadcasts as a way to verify proximity [13, 6, 5]. However, there are no strong security guarantees that the communication range will always be limited as desired: an adversary with more powerful equipment may be able to participate in the protocols even if they are substantially further away than non-malicious parties.

Finally, there are many techniques to help localize devices [2, 14, 15, 10, 18, 1], GPS being one of the most widely deployed. However, none of those works addressed security, and in fact, GPS signals can be spoofed [17, §3.2.2]. Nonetheless, we have noted that combining a localization mechanism with our secure location verification technique yields a secure localization algorithm. Thus, insecure localization protocols should be seen as complementary to our work on secure location verification.

Many authors have commented on the value of location-based access control [7, 5, 3, 13, 6].

6 Future Work

One area for future work is performing more precise region verification using intersection. The idea is that if verifier $v1$ can verify $\mathsf{ROA}(v1, \Delta)$ and $v2$ can verify $\mathsf{ROA}(v2, \Delta)$, then together they should be able to verify $\mathsf{ROA}(v1, \Delta) \cap \mathsf{ROA}(v2, \Delta)$. The intuition is simple: if a prover is verified to be present in both regions, then it must be in their intersection. The allure of this approach is considerable: under our current scheme, a verifier node is required inside R, whereas in principle nodes at the edges of, or even outside, R could be used. There are significant challenges, though. First, the verifications would have to be done simultaneously; if not, the prover could move from location to another between the two runs. One way to do this is to have the prover receive both nonces and return a hash of their concatenation to both verifiers, who would have exchanged nonce information in advance. Another problem is that this approach is not robust against collusion attacks by attackers with multiple nodes: such an adversary may legitimately have one node in $\mathsf{ROA}(v1, \Delta)$ and one node in $\mathsf{ROA}(v2, \Delta)$ independently without having any presence in $\mathsf{ROA}(v1, \Delta) \cap \mathsf{ROA}(v2, \Delta)$.

7 Conclusion

We introduced the in-region verification problem. Then, we designed provably secure, lightweight protocol to address it, named the Echo protocol. The Echo protocol does not require cryptography, time synchronization, or any prior agreement between the prover and verifier, making it suitable for low-cost devices such as those in sensor networks. It is robust against a malicious adversary with unbounded computing power; the security rests on physical properties of sound and RF signal propagation. We showed that for a reasonable scenario, coverage of 80–90% could be easily achieved, i.e., the Echo protocol could guarantee in-region verification for 80–90% of legitimate location claims. Consequently, we expect the Echo protocol to be a useful contribution in contexts where physical presence is used for access control.

References

[1] GPS Documentation. `https://www.peterson.af.mil/GPS_Support/gps_documentation.htm`.

[2] Paramvir Bahl and Venkata N. Padmanabhan. RADAR: An In-Building RF-Based User Location and Tracking System. In *INFOCOM (2)*, pages 775–784, 2000.

[3] Dirk Balfanz, D.K. Smetters, Paul Stewart, and H. Chi Wong. Talking to Strangers: Authentication in Ad-Hoc Wireless Networks. In *Network*

and Distributed System Security Symposium Conference Proceedings, 2002.

[4] Stefan Brands and David Chaum. Distance-Bounding Protocols. In *EUROCRYPT '93*, volume 765 of *LNCS*.

[5] Deborah Caswell and Philippe Debaty. Creating Web Representations for Places. In *2nd International Symposium on Handheld and Ubiquitous Computing*, pages 114–126, 2000.

[6] Mark D. Corner and Brian D. Noble. Zero-Interaction Authentication. In *MOBICOM '02*. ACM Press, 2002.

[7] Dorothy E. Denning and Peter F. MacDoran. Location-Based Authentication: Grounding Cyberspace for Better Security. In *Computer Fraud & Security*. Elsevier Science Ltd., February 1996.

[8] Jeremy Elson, Lewis Girod, and Deborah Estrin. Fine-Grained Network Time Synchronization using Reference Broadcasts. In *Proceedings of the Fifth Symposium on Operating Systems Design and Implementation (OSDI 2002)*, 2002.

[9] Eran Gabber and Avishai Wool. How to Prove Where You Are: Tracking the Location of Customer Equipment. In *Proceedings of the 5th ACM conference on Computer and Communications Security*, pages 142–149, 1998.

[10] Lewis Girod, Vladimir Bychkovskiy, Jeremy Elson, and Deborah Estrin. Locating Tiny Sensors in Time and Space: A Case Study. In *ICCD*, 2002.

[11] Jason Hill, Robert Szewczyk, Alec Woo, Seth Hollar, David Culler, and Kristofer Pister. System architecture directions for network sensors. In *ASPLOS*, 2002.

[12] Yih-Chun Hu, Adrian Perrig, and David B. Johnson. Packet Leashes: A Defense against Wormhole Attacks in Wireless Ad Hoc Networks. In *INFOCOM*, 2003.

[13] Tim Kindberg, Kan Zhang, and Narendar Shankar. Context Authentication Using Constrained Channels. In *Fourth IEEE Workshop on Mobile Computing Systems and Applications*, 2002.

[14] A.M. Ladd, K.E. Bekris, G. Marceau, A. Rudys, D.S. Wallach, and L.E. Kavraki. Robotics-Based Location Sensing for Wireless Ethernet. In *Eigth Annual International Conference on Mobile Computing and Networks (MobiCOM 2002)*, 2002.

[15] Nissanka B. Priyantha, Allen K. L. Miu, Hari Balakrishnan, and Seth J. Teller. The cricket compass for context-aware mobile applications. In *Mobile Computing and Networking*, pages 1–14, 2001.

[16] Frank Stajano and Ross Anderson. The Resurrecting Duckling: Security Issues for Ad-Hoc Wireless Networks. In *7th Security Protocols Workshop*, volume 1796 of *Lecture Notes in Computer Science*, pages 172–92, 1999.

[17] John A. Volpe. Vulnerability Assessment of the Transportation Infrastructure Relying on the Global Positioning System, August 2001.

[18] A. Ward, A. Jones, and A. Hopper. A New Location Technique for the Active Office. *IEEE Personal Communications*, 4(5):42–47, October 1997.

[19] Brent Waters and Ed Felten. Proving the Location of Tamper Resistent Devices. `http://www.cs.princeton.edu/~bwaters/research/location_proving.ps`.