Distance Bounding Protocols: Authentication Logic Analysis and Collusion Attacks

Catherine Meadows[1], Radha Poovendran[2], Dusko Pavlovic[3], LiWu Chang[1], and Paul Syverson[1]

[1] Naval Research Laboratory, Code 5543, Washington, DC 20375 { meadows, lchang, syverson } @itd.nrl.navy.mil
[2] University of Washington, Department of Electrical Engineering, Seattle Washington, 98195 rp3@u.washington.edu
[3] Kestrel Institute, 3260 Hillview Avenue, Palo Alto, CA 94304 dusko@kestrel.edu

Summary. In this paper we consider the problem of securely measuring distance between two nodes in a wireless sensor network. The problem of measuring distance has fundamental applications in both localization and time synchronization, and thus would be a prime candidate for subversion by hostile attackers. We give a brief overview and history of protocols for secure distance bounding. We also give the first full-scale formal analysis of a distance bounding protocol, and we also show how this analysis helps us to reduce message and cryptographic complexity without reducing security. Finally, we address the important open problem of collusion. We analyze existing techniques for collusion prevention, and show how they are inadequate for addressing the collusion problems in sensor networks. We conclude with some suggestions for further research.

1 Introduction

Distance estimation, that is the estimate of the distance between two nodes, plays of a fundamental part in the setting up and maintenance of sensor networks. For example, a node trying to localize itself, can, if it learns its distance from three or more nodes with known locations, use multilateration to determine where it sits. This computation is a major part of many localization algorithms. Distance estimation can also be useful in synchronization: if node A knows its distance from node B, it can request a timestamp from node B and compute the clock skew by factoring in the round trip time of the request and the response.

One of the most accurate means of distance estimation is to use the time of flight of a signal. For example, one can send a signal to a seated node, have it respond, and then use the time of the round trip to measure the distance. For example, Multispectral Solutions [1] has recently developed an ultra wide band ranging radio based on such technology that measures round trip times of packets to provide range resolution of better than one foot.

Although such a technique can provide accurate measurements, it is not easy to figure out how to make use of it when a node (from now on referred to as the *verifier*) is attempting to find its distance form another node (from now on referred to as the *prover*) in the face of hostile attackers. If the prover is dishonest, it can pretend to be closer to or further away from the verifier than it actually is by either jumping the gun and sending a response before the request, or pretend to be further away than it is by delaying its response. Even if the prover is honest, a hostile attacker could attach its own identity to the prover's response, and pass off honest verifier's location as its own. Finally, dishonest provers can conspire to mislead the verifier, one prover lending the other prover its identity so that the second prover can make the first prover look closer than it is.

Probably the simplest secure distance measurement protocol is Sastry et al.'s Echo protocol [14], in which the verifier sends a nonce to the prover, and the prover returns it to the verifier. The use of a random nonce means that the prover can't respond until it has heard from the verifier, thus preventing the prover from jumping the gun. However, without any kind of authentication, it is possible for an attacker to usurp an honest prover's response and attach its own identity.

The obvious defense is to have the prover authenticate its response, and indeed, a variant of Echo protocol offers this capability. However, the time involved in computing the authentication function can be so large with respect to the travel time as to make it difficult to compute the distance except for relatively slow (and less accurate) sound frequencies.

An approach that gets around this problem is to have the prover send a *rapid*, unauthenticated, response and then send the *authenticated* response later. However, if this is not done carefully, it is again possible for an attacker to usurp an honest prover; he simply prevents the authenticated response from reaching the verifier, and substitutes his own authenticated response.

Fortunately, a solution to this problem already exists. This is the notion of a *secure distance bounding protocol*. This idea was first introduced by Brands and Chaum [2] to defend against Desmedt's Mafia attack [5] on zero knowledge protocols. The idea is that the prover first commits to a nonce using a one-way function, the verifier sends a challenge consisting of another nonce, the prover responds with the exclusive-or of its and the verifier's nonces, and then follows up with the authentication information. The verifier uses the time elapsed between sending its nonce and receiving the prover's rapid response to compute its distance from the prover, and then verifies the authenticated response when it receives it. In the Brands and Chaum protocol, the challenge and the response are done as a bit-by-bit exchange, and the time of flight is taken as the average of the time of flight of each pair of bits. Other protocols that take a similar approach, such as the Čapkun-Hubaux protocol [16], rely on a single exchange of packets. It is also possible to consider other variants, in which a single nonce is broken into k-bit chunks, and multiple packets are used.

Another, but related, approach is taken by Hancke and Kuhn in [6]. In this protocol the verifier sends the prover a nonce, and the prover computes the a collision-free one-way hash function over the nonce and a key shared between the prover and the

verifier. The principals then perform a rapid bit-by-bit exchange in which the verifier sends random challenges and the prover responds with a response based on the challenge and the hash.. In this case the authentication takes place previously to the rapid exchange.

Assuming that there is no collusion between provers, the Čapkun-Hubaux protocol, like the Brands-Chaum protocol, prevents hijacking because of the commitment step, and also prevents the prover from lying about being any closer to the verifier than it is, although it can lie about being farther away simply by delaying its response. Likewise, the authentication used in the Hancke-Kuhn protocol prevents hijacking, and the verifier's random challenge prevents a premature reply.

The problem of a delayed response can be dealt with in certain instances using multiple provers or verifiers. For example, in Čapkun and Hubaux's SPINE protocol [16] three verifiers forming a triangle around a prover use a distance bounding protocol to localize it. A prover who wants to lie about its location must pretend to be closer to one of the verifiers than it is; the distance bounding protocol makes this impossible.

Running all through this is the issue of guaranteeing correctness of distance bounding protocols. The presence of time as a factor puts an extra security requirement on the protocol: not only must messages have come from the indicated principal, but in the indicated amount of time. On the other hand, time may work for us as well. Certain types of message modification attacks will not be useful if a node is trying appear closer than it is, since intercepting and modifying the message will delay its arrival.

In spite of this, very little work exists in the formal and mathematical analysis of distance bounding protocols. Sastry et al. include a security proof for the Echo protocol, but, since no authentication is involved, the proof is limited to showing that a prover cannot respond before receiving the verifier's nonce. Brands and Chaum provide a proof that their protocol is zero-knowledge but do not provide any extended analysis of the timing properties. Thus there appear to be no analyzes of distance bounding protocols available that take into account the subtle interplay between authentication and timing.

In this paper we address all of these above issues. We first give an outline of requirements that distance bounding protocols should satisfy. We then describe a new distance bounding protocol, similar in structure to Brands-Chaum, and a generalized version of the protocol we presented in [12]. We then extend the authentication logic we used in [12] to so that it can be used to reason directly about distance bounding protocols, and use the logic to give a formal analysis of the protocol's security. This formal analysis allows us to simplify greatly the type of commitment used, and to omit one cryptographic operation.

We then address the issue of collusion. The problem of collusion in distance bounding, in which two dishonest verifiers pool information to make one of the verifiers look closer than it is, was first noticed by Desmedt [5] who dubbed it the "terrorist attack." The Brands-Chaum protocol is vulnerable to a collusion attack, in which one prover sends the rapid response and then passes the information in its commitment over to another, who sends the authenticated response. Brands and Chaum were

aware of this attack and left it as a an open problem. Since then, others have tackled it [3], but their solutions require colluders to share long term secrets. This approach, while possibly appropriate for the types of applications envisaged by Brands and Chaum, does not provide much help for sensor networks, in which colluding nodes are likely to be under control of the same attacker, and so would not be likely to have any objection to sharing any secret information. We study this problem in detail, showing how attacks are possible even on protocols such as ROPE [8] or SPINE that use multiple verifiers to detect cheating nodes, and make some recommendations.

2 Requirements for Distance Bounding Protocols

In [14] Sastry et al. give a set of requirements for distance bounding protocols. They are:

1. **Make few resource demands on the prover and verifier**. This means keeping the number of cryptographic operations and messages low.
2. **No previous setup required**. In particular, there should be no need for principals to share keys beforehand.
3. **Guarantees should be quantifiable.**

Although the use of authentication means that we must use some form of cryptography in the authenticated response, we can still keep costs down by minimizing its use in the rapid response. Note that hash functions and nonce generation will both count as cryptographic operations.

The second requirement seems completely at odds with any form of authenticated distance bounding protocol, but it can be partially satisfied by having the rapid exchange take place without the use of any cryptographic keys. A verifier could then request to have a key distributed to it and the prover, which the prover could then use to authenticate its authenticated response. This would be helpful, example, if a verifier was only interested in finding its nearest neighbor. This feature is present in the Brands-Chaum and Čapkun-Hubaux protocol, as well as the protocol that we present in this paper. However, it is not true of some other protocols, such as the Hanck-Kuhn protocol, which requires the prover's response to include a hash of its nonce with a key shared with the verifier.

As for quantitative guarantees, at present the Echo protocol is the only one of which we know that satisfies such quantitative guarantees, and even qualitative guarantees in the form of formal analyzes seem rare. However, we provide qualitative guarantees in this paper that we believe could ultimately be extended to quantitative guarantees in the manner of [14].

We now consider the main security requirements that have been identified in the literature.

1. A prover should be able to correctly determine its distance from an honest verifier, even when hostile attackers are present.

2. A prover should be able to determine an upper bound for its distance from even a dishonest verifier, as long as the verifier does not collude with other verifiers.
3. A prover should be able to determine an upper bound for its distance from a dishonest verifier even if it does collude.

In this paper, we prove the somewhat weaker goal that, if the prover is honest in the sense that it follows the rules of the protocol but may either delay its response (either due to dishonesty or processing time), or attempt to respond early, the verifier can compute an upper bound on the distance. Finally for (3) we argue that the known techniques for detecting or preventing fraud in distance bounding protocols are either insecure against collusion or are not applicable in sensor networks.

3 Distance Bounding Protocol and its Analysis

3.1 Assumptions

We assume that nodes have the ability to generate random or pseudorandom nonces and compute collision-free one-way hash functions. We also assume that provers have a means of authenticating themselves to verifiers, e.g. by shared keys or digital signatures. In this paper we use shared keys and message authentication codes (MACs), but the same analysis will work for digital signatures.

We also assume that principals have the ability to compute the time that an event occurs with respect to their local clocks. The unit of time may or may not be of a finer granularity then the sending or receipt of a message. If the time granularity is finer, we let the time of a message denote the beginning of a send or receive. We will be particularly interested in timed sends and receives of individual packets. In this case we will assume that it is possible to predict the time of any subevent of a send or receive (such as the end) from the time of the beginning, and vice versa. We will also assume that all subevents of the send of a given packet are engaged in by a unique principal. That is, A cannot send part of a packet and B send another, and have them both accepted as part of the same packet. Our reason for doing so is the belief that this would need a degree of synchronization that would require, at very least, cooperation between A and B, and in our analysis we are not trying to rule out collusion attacks.

3.2 The Protocol in Detail

We fix an interval I_0 that is the expected turnaround time between receiving a challenge and sending a response. Our protocol proceeds in five steps, four of which involve the sending of messages.

1. The prover P generates a nonce N_P. This, and any other computations that do not involve information from the verifier, can be done in advance of P's participating in the protocol.

2. The verifier V requests a distance measurement. This is mainly to warn P that a challenge is on the way, and to let P know V's identity.

   V **sends** $V, \mathrm{request}$

3. The verifier V sends a nonce as a challenge:

   V **sends** N_V

4. The prover P sends a response, of the application of a function F to N_P, P, and N_V. We refer to this message as the *rapid response*. The only condition that we put on F is that the verifier be able to verify that $F(N_V, P, N_P)$ was constructed using N_V, P, and N_P. Examples of such functions include N_V, P, N_P, where , denotes concatenation, $N_V, (P \oplus N_P)$, assuming that names are a distinct recognizable type, and $N_V \oplus h(P, N_P)$, where h is a collision-free hash function.

   P **sends** $F(N_V, P, N_P)$

   The verifier, on receiving this message, calculates the time elapsed between sending the challenge and receiving the rapid response.

5. The prover sends a message authenticated with a key shared between it and the verifier. We refer to this message as the *authenticated response*.

   P **sends** $P, Pos_P, N_P, N_V, MAC_{K_{PV}}(P, Pos_P, N_P, N_V)$

   where Pos_P is P's position. V, on receiving the message, verifies the MAC. It also computes F from the values it receives in the authenticated response, and compares it with the value it received in the rapid response. If the two are the same, and the MAC checks out, it accepts P's response as valid. V then subtracts I_0 from the time elapsed between sending the challenge and receiving the rapid response and uses the result to calculate its distance from the prover, that is, the distance is calculated to be $v \cdot (t_2 - t_1 - I_0)/2$.

An overview of the protocol is given in Figure 1.

4 Security Analysis

4.1 Overview

In this section we give a formal analysis of the distance bounding protocol using the combined authentication and secrecy logics of [4] and [13]. Although we are interested in authentication, not secrecy, we will use some of the concepts introduced in the secrecy logic, and so we will refer to that as well. We will use the logic to show what a verifier can conclude from interacting with an honest prover. We then show how the proof breaks down if the prover is dishonest, in particular if it is in collusion with another node.

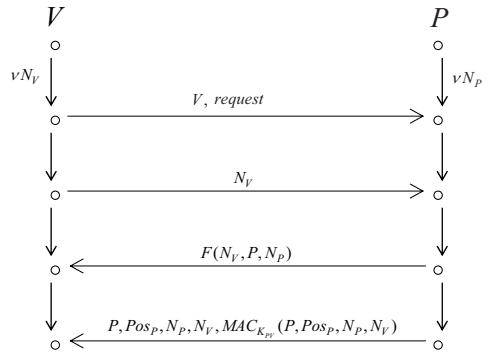


Fig. 1. Distance Bounding Protocol

4.2 The Authentication Logic

Basic Ideas and Notation

We begin by setting the stage for the logic, and introducing a little notation. The interested reader can find a more complete discussion in [4,13]. We consider a protocol as a partially ordered set of actions, as in Lamport [7], in which $a < b$ means that action a occurs before action b. We let $(t)_A$ denote t being received by A, $\langle t \rangle_A$ denote a message being received by A. We let $x \prec y$ denote the statement "if an action of the form y occurs, then an action of the form x must have occurred previously," and we let νn denote the generation of a fresh, unpredictable nonce, n.

For the purpose of the derivations in this paper, we will use a term algebra **T** consisting of constants, variables, and the following operations:available to principals: concatenation , denoted by ',' , deconcatenation, computation of message authentication codes, denoted by $MAC_{K_{XY}}(Z)$, collision-free hash functions, denoted by $h(Z)$, and exclusive-or, denoted by $\oplus$. **T** is provided with the following equational theory:

1. $MAC_{K_{XY}}(Z) = MAC_{K_{YX}}(Z)$
2. $x \oplus x = 0$
3. $0 \oplus x = x$
4. $x \oplus y = y \oplus x$
5. $(x \oplus y) \oplus z = x \oplus (y \oplus z)$

for a distinguished term 0.

We say that $g \equiv t$ if g and t can be made equal by applying the equational theory of **T**. We refer to rules (2) and (3) as the *cancellation rules*. We say that a term g is *irreducible* if no cancellation rules can be applied to g. We say that $s \sqsubseteq t$ if s is a subterm of t.

We supply **T** with a simple type theory. There is a general type "term" and one subtype,"name". A variable not of type name will often be referred to as untyped.

Free untyped variables are used to refer to terms about which the recipient knows nothing. We say that a map from variables from to terms is a *substitution* if it is the identity on all but a finite number of variables and preserves types. If σ is a substitution and t is a term, we let $t\sigma$ denote the image of t under σ.

The logic also includes a number of predicates describing states of principals. A : stands for "A knows", and $\mathsf{H}P$ means that P is an honest principal who follows the rules of the protocol.

Informally, a *role* is the set of actions that a principal performs to engage in a particular protocol. In the distance bounding protocol, we have two roles: the verifier and the prover. A *run* is the trace of a (possibly) partial execution of a protocol, i.e., the set of actions executed by the principals and their partial ordering. A *state* is a cut of the directed graph induced by the run; each action in the run should occur either before or after the state.

Stable Subterms

In this section we will formalize the notion that a subterm s of a term t must have been used in computing t. In our earlier work, where no cancellation was involved, this could be guaranteed by requiring that s was a subterm of t, since the term structure of t gave a unique history of the way in which t was built. However, when we allow cancellation rules, things become more complicated. For example, suppose that a principal receives a term $x \oplus s$ where x is a free untyped variable. If x were further instantiated to $y \oplus s$, then s would vanish after the cancellation rules were applied.

In order to avoid this we make the following definition.

Definition 1. *Let s be a subterm of t. We say that s is a stable subterm of t, denoted by $ss(s,t)$, if for all possible substitutions σ to the free variables in t and after all possible applications of the equations governing the term algebra* **T**, *we have $s\sigma \sqsubseteq t\sigma$. We say that a term t is simply stable, denoted by $ss(t)$, if every subterm of t is stable.*

The motivation behind the use of stable subterms is that it makes it possible to ascertain that, whatever values the free variables in t turn out to have, the stable subterm must have been used in the computation of t.

We use this insight to make the following definition:

Definition 2. *We use the notation $((s))_A$ (respectively, $\langle\langle s\rangle\rangle_A$) to denote A's receipt (respectively sending) of a message m containing s as a stable subterm.*

We note that our definition subsumes the definition used in [4, 11, 13], which required s merely to be a subterm of t. For the term algebras used in those papers, subterm implies stable subterm.

When we want to make it clear that A is receiving (respectively sending a term t with stable subterm s, we will use the notation $((s \sqsubseteq t))_A$ (respectively $\langle\langle s \sqsubseteq t\rangle\rangle_A$).

We can now formally describe the conditions on the term F used in our distance bounding protocol. We present this as an axiom that must be verified for particular choices of F.

$$ss((F(x_1, x_2, x_3)) \quad \text{(st)}$$

We now consider the sorts of functions F that can be proved to satisfy st. For example $(N_P \oplus N_V), P)$ where N_P is a free untyped variable and P is a variable of type name, and N_V is a nonce generated by V does *not* satisfy st. Since N_P is free and untyped, any substitution may be made to it. Thus, if $N_P\sigma = X \oplus N_V$, then $((N_P \oplus N_V), P) = (X \oplus N_V \oplus N_V), P) = (X, P)$ which can be computed without N_V. However, $(N_V \oplus P), N_P)$ does satisfy st. The term N_V a random value, so we cannot make arbitrary substitutions to it. The same goes for P. We can made arbitrary substitutions to N_P, but none of them will result in canceling out N_V, P, or N_P,

Lemma 1. *Let* **T** *be the term algebra described in section 4.2, and let m be an irreducible term from* **T**. *Every subterm of m is stable if for every irreducible substitution σ to the variables of m, $m\sigma$ is also irreducible.*

Proof. (Sketch) Let m be a term satisfying the hypothesis of the lemma. We want to show that after any possible substitution σ to the variables in m, $t\sigma$ is still a subterm of $m\sigma$ after all possible reductions have been made. For the case of an irreducible σ this follows directly from the hypothesis, since no reductions are possible. For the case of a reducible σ it follows from the fact that the rewrite theory associated with our term system is Church-Rosser, modulo the associative commutativity axioms for exclusve-or, which, in our case means, that when several applications of the cancellation rule are possible, it does not matter in what order they are taken. Thus, we can apply the cancellation rules to the cancellations induced on the variables by σ first. Once that is done, then σ becomes an irreducible substitution, and we are back to the first case.

We also give as a corollary the following procedure for stable subterms:

Corollary 1. *Suppose that t contains no subterm of the form $X \oplus Y$, where one of X or Y is a free untyped variable. Then t is simply stable.*

Proof. The proof follows from Proposition 1 and the fact that the only irreducible terms that do not necessarily remain irreducible after irreducible substitutions are those that contain $Z \oplus Y$, where either Z or Y is an untyped free variable.

The corollaries below follow directly from the fact that none of the terms in question contain subterms of the form $X \oplus Y$, where one of X or Y is an untyped free variable.

Corollary 2. *Suppose that P is a variable of type name, N_V is a nonce, and N_P is an untyped free variable. Then N_V, P, N_P, $N_V, (N_P \oplus P)$, and $N_V \oplus h(P, N_P)$ are simply stable.*

Corollary 3. *Any variable or constant is simply stable. $MAC_{XY}(Z)$ is simply stable as long as Z is. $X||Y$ is simply stable as long as X and Y are. $h(X)$ is simply stable as long as X is.*

Basic Axioms

We are now ready to describe the basic axioms of the logic as given in [4]. The logic describes what a principal can conclude from interacting via the protocol with another principal. Two basic axioms of the logic are the *receive* axiom rcv and the *freshness axiom* new, which we describe below.

The receive axiom says that everything that is received must have been originated by someone:

$$A: \ (\!(m)\!)_A \Rightarrow \ \exists X. \langle\!\langle m \rangle\!\rangle_{X<} < (\!(m)\!)_A \qquad \text{(rcv)}$$

The freshness axiom describes the behavior of the ν operator.

$$\begin{aligned}(\nu\, n)_B \wedge a_A \Rightarrow (n \in FV(a) \ &\Rightarrow (\nu\, n)_B < a_A \\ \wedge \ (A \neq B &\Rightarrow (\nu\, n)_B < \langle\!\langle n \rangle\!\rangle_B < (\!(n)\!)_A \leq a_A))\end{aligned} \qquad \text{(new)}$$

where $FV(a)$ denotes the free variables of a

The first part says that ν is a binder, that is, any event a mentioning n necessarily occurs *after* $(\nu\, n)$. The second line requires that if the agent B executing $(\nu\, n)$ and the principal A executing a are different, then B must have used a send action to transmit n and A must have acquired it by means of a receive action.

The fact that we can use ν as a binder means that it is possible to apply ν outside of a sequence of events S, e.g. as $(\nu n)_A(S)$. This will be convenient, since we often will not care exactly when νn occurs, as long as it occurs before n is sent in a message.

Axioms Governing Message Authentication Codes

The message authentication code has the property that it is possible to tell who created it. This property is formally derived in [4] for similar functions using their non-invertibility and assumptions about the secrecy of keys. Since we will not need the machinery of [4] for anything other than this result, we state it as an axiom here.

$$\langle\langle MAC_{K_{AB}} t \rangle\rangle_{X<} \implies X = A \vee X = B \qquad \text{(mac)}$$

Timestamps, Distance, and the Axioms Governing Them

Up to now we have considered only axioms that cover the ordering of messages. Now we will extend our logic to reasoning about distance. To do this will make use of the notion of a timestamp, which was already introduced in [13], although to reason different types of properties.

A *timestamp* represents an entity's recording of its local time, For this we use the expression τt, where $(\tau t)_A$ denotes A's reading its local time and storing it a local variable. We use $a_A^{[\tau t_1, \tau t_2]}$ to denote A's engaging in event a some time between times t_1 and t_2. Where appropriate, we can use the shorthand a_A^t for $a_A^{[t-\epsilon, t+\epsilon]}$.

We note that in some cases the granularity of time measurement may actually be less than the time it takes to engage in an event. Thus, the time it takes for a principal to receive or send a message may take more than one time interval. In that case, we take a_A^t to mean the time at which A begins to engage in the action. In this case, we will need to attach a stronger meaning to $\langle\langle x \rangle\rangle_A^t$ and $((x))_A^t$ as well. They will mean, not only that x must have been used in the construction of the message, but that either x or some term each of whose bits depends on x appears at the beginning of the message as well. Our analysis will hold for either definition of timed event.

For the purposes of reasoning about time and distance, we introduce the function $d(A, B)$ where A and B are two principals (we ignore the possibility of node mobility at this point). We define $d(A, B)$ as follows:

Definition 3. *Let A and B be two principals. We define the distance between A and B or $d(A, B)$ to be $v \cdot t$, where v is the velocity at which a signal travels, and t is the minimum of all possible $(t_1 - t_2 - I)/2$ such that the following occurs:*

$$(\nu n)_A\ (\nu m)_B\ (\langle\langle n \rangle\rangle_{A<}^{t_1} < ((n))_B < \langle\langle m \rangle\rangle_{B<} < ((m))_A^{t_2})$$

and I is the turnaround time at B.

The idea is, if that B receives a nonce created by A, or vice versa, either directly or indirectly, then the time it took must be bounded below by their distance times the velocity. If one pair of send and receive events occurs after another than the total time for the whole sequence of events to occur is bounded below by twice the distance times the velocity plus the turnaround time. The remainder of this section will be devoted to the construction and analysis of authentication techniques for proving that this sequence of events has taken place.

This leads us to the following simple proposition, whose proof follows directly from the above definition.

Proposition 1. *Suppose that $A : (\nu n)_A\ (\nu m)_B\ (\langle\langle n \rangle\rangle_{A<}^{t_1} < ((n))_B < \langle\langle m \rangle\rangle_{B<} < ((m))_A^{t_2})$. Then, the distance between A and B is less than or equal to $v(t_2 - t_1 - I_0)/2$, where v is the velocity at which a signal travels and I_0 is the minimum turnaround time at B.*

The point of our analysis will be to get a verifier to the point at which she can apply Proposition 1 to calculate her distance from a prover.

Challenge-Response and Distance Bounding Templates

A key feature of the logic is the *challenge response template*, which is as follows

$$A : \Phi' \wedge (\nu n)_A < \langle\!\langle c^{AX} n \rangle\!\rangle_{A<} < (\!(r^{AX} n)\!)_A$$
$$\Rightarrow (\nu n)_A < \langle\!\langle c^{AX} n \rangle\!\rangle_{A<} < (\!(c^{AX} n)\!)_X < \langle\!\langle r^{AX} n \rangle\!\rangle_{X<} < (\!(r^{AX} n)\!)_A$$

where c^{AX} is the challenge structure issued by A, r^{AX} is the corresponding response originated by X, and Φ' represents some additional precondition, such as an honesty assumption. For example, the challenge could be a nonce, and the response could be a MAC applied to the nonce using a key shared between A and X.

The challenge-response template is the basic building block of authentication protocols. Most authentication protocols can be built up by combining and extending various challenge-response protocols. However, the challenge-response template cannot be used in its basic form for distance bounding protocols. That is because the computational requirements on the response are so strict that much of the job of the challenge and response must be accomplished by auxiliary protocols occurring before the challenge and after the response. We refer to these auxiliary protocols as C_A and R_A, as below.

We describe the distance bounding template below:

$$A : \Phi' \wedge (\nu n)_A < C_A(n) < \langle\!\langle c^{AX} n \rangle\!\rangle^{t_1}_{A<} < (\!(r^{AX} n, m)\!)^{t_2}_A < R_A(n, m)$$
$$\Rightarrow (\nu n)_A < \langle\!\langle c^{AX} n \rangle\!\rangle^{t_1}_{A<} < (\!(c^{AX} n)\!)_X < \langle\!\langle r^{AX} n, m \rangle\!\rangle_{X<} < (\!(r^{AX} n, m)\!)^{t_2}_A$$

There are a number of ways of constructing $C_A(n)$ and $R_A(n, m)$. In Brands-Chaum and Čapkun-Hubeaux $C_A(n)$ is a commitment, and $R_A(n, m)$ is an authentication of the rapid response, plus an opening of the commitment. In our protocol, $C_A(n)$ is empty, and $R_A(n, m)$ is the authentication of the rapid response. In the Hancke-Kuhn protocol, the $C_A(n)$ is an exchange of nonrepeatable bitstrings, the rapid exchange is the exchange of a one-way collision-free hashes of the bitstrings with a shared key, while $R_A(n, m)$ is empty.

4.3 Analysis of the Distance Bounding Protocol

Proof of Security for Honest Prover

Our logic is designed to be used in for success refinement of a protocol. Normally, this involves either increasing the functionality of the principals involved, or making assertions about their behavior that is implemented in successive refinements. We have found in it this case it makes sense to do our refinements on the honesty of the prover. Refining our analysis on different assumptions about the prover's honesty allows us to see what the different kinds of guarantees are in the different cases. We use three types of prover, first one about which we make no assumptions, then a "semi-honest" prover who sends messages in the correct order and does not reveal secrets, but who does not necessarily reply with a nonce when expected to, and who may attempt to cheat by sending the response before getting the challenge. Finally, we specify the honest prover.

The semi-honest prover is specified as follows.

$$V : \mathsf{SH}P \implies \langle F(N_V, P, N_P)\rangle_P \prec \langle P, Pos_P, N_P, N_V, MAC_{K_{PV}}(P, Pos_P, N_P, N_V)\rangle_P \\ \wedge \langle F(X, P, Y)\rangle_P \implies X = P \qquad \text{(shpr)}$$

We can also specify the honest prover, who follows the protocol to the letter:

$$V : \mathsf{H}P \implies (\nu N_P)_P((V, \text{request})_P < (N_V)_P < \langle\langle N_P \sqsubseteq F(N_V, P, N_P)\rangle\rangle_{P<} \prec \\ \langle P, Pos_P, N_P, N_V, MAC_{K_{PV}}(P, Pos_P, N_P, N_V)\rangle_P) \\ \wedge \langle F(X, P, Y)\rangle_P \implies X = P \qquad \text{(hpr)}$$

Finally, we specify a necessary piece of information about the honest verifier's behavior.

$$V : \langle MAC_{K_{XV}}(Y, Pos, N, M)\rangle_V) \implies Y = V \qquad \text{(hv)}$$

This prevents V from concluding the a message sent by herself is from P.

Our proof will proceed incrementally, using stronger and stronger assumptions about P. We will start with proving what can the verifier can conclude when nothing at all is known about the principal or principals with which she is interacting. We will then progress to the case of the semi-honest prover, and conclude with the honest prover.

We start with what the verifier observes:

$$V\text{sees} : \langle V, \text{request}\rangle < (\nu N_V)_V < \langle N_V\rangle_V^{t_1} < (F(P, N_V, N_P))_V = ((N_V))_V = ((N_V))_V^{t_2} < \\ (P, Pos_P, N_P, N_V, MAC_{K_{PV}}(P, Pos_P, N_P, N_V))_V \qquad \text{(vfr)}$$

where P is a variable of type name.

By applying the rcv axiom twice, we obtain from it together with the st axiom governing $F(P, N_V, N_P)$ and the simple stability of the MAC expression that:

$$V : (F(P, N_V, N_P))_V^{t_2} \implies \langle F(P, N_V, N_P)\rangle_X < (F(P, N_V, N_P))_V^{t_2} \qquad \text{(a1)}$$

$$V : (MAC_{K_{PV}}(P, Pos_P, N_P, N_V))_V \implies \langle MAC_{K_{PV}}(P, Pos_P, N_P, N_V)\rangle_Y < \\ (MAC_{K_{PV}}(P, Pos_P, N_P, N_V))_V \qquad (1)\ \text{(a2)}$$

From the st axiom governing $F(P, N_V, N_P)$, the new axiom, and a1 we obtain

$$V : (\nu N_V)_V < \langle N_V\rangle_V^{t_1} < ((N_V))_X < \langle F(P, N_V, N_P)\rangle_X < (F(P, N_V, N_P))_V^{t_2} \qquad \text{(a3)}$$

This is as far as we can go without making some assumptions about honesty. Since we have two principals now, X and Y, we will need to make honesty assumptions about them both. The condition Φ that we assume will be $\mathsf{SH}X \vee \mathsf{SH}Y$.

Proposition 2. *Suppose that* a2 *and* a3 *hold. From* vfr *and* $\mathsf{SH}X \vee \mathsf{SH}Y$ *we can further conclude that*

$$\begin{aligned} V : (\nu N_V)_V < \langle N_V \rangle_V^{t_1} < ((N_V))_P < \langle F(P, N_V, N_P)\rangle_P < \\ \langle P, Pos_P, N_P, N_V, MAC_{K_{PV}}(P, Pos_P, N_P, N_V)\rangle_P < \\ (P, Pos_P, N_P, N_V, MAC_{K_{PV}}(P, Pos_P, N_P, N_V))_P \end{aligned} \quad \text{(b1)}$$

Proof. Suppose that X is semi-honest. From the shpr axiom and a3, we obtain that $X = P$. From the mac axiom, we get that $\langle P, Pos_P, N_P, N_V, MAC_{K_{PV}}(P, Pos_P, N_P, N_V)\rangle_P$ as well. The semihonesty of $X = P$ gives us $\langle F(P, N_V, N_P)\rangle_P < \langle P, Pos_P,\ N_P, N_V, MAC_{K_{PV}}(P, Pos_P, N_P, N_V)\rangle_P$, and the remainder follows from a2 and a3.

Suppose now that Y is semi-honest. Then, by the mac axiom, we get that $Y = P$. From the shpr axiom, we get that $\langle F(P, N_V, N_P)\rangle_P < \langle P, Pos_P, N_P, N_V, MAC_{K_{PV}}(P, Pos_P, N_P, N_V)\rangle_P$ as well, and from this and a2 and a3 we get the result.

We are now left with two things to prove, first that P was the first to send the rapid response, and secondly that V receives P's response after P sends it. The first is necessary in order for V to be able to conclude the second. We get both from the honest of P.

Proposition 3. *Suppose that* HP *and* vfr *hold. Then*

$$\begin{aligned} V : (\nu N_V)_V \wedge (\nu N_P)_P\ (\langle N_V \rangle_V^{t_1} < ((N_V))_P < \langle\langle N_P \sqsubseteq F(P, N_V, N_P)\rangle\rangle_{P<} \\ < ((N_V \sqsubseteq F(P, N_V, N_P)))_V^{t_2} < \langle MAC_{K_{PV}}(P, Pos_P, N_P, N_V)\rangle_P \\ < (P, Pos_P, N_P, N_V, MAC_{K_{PV}}(P, Pos_P, N_P, N_V))_V) \end{aligned} \quad \text{(c1)}$$

Proof. From HP we get SHP, and from that and vfr we get b1. From HP we also get that P was the first to sent a message constructed with N_P. From the new axiom, we get that $(F(P, N_V, N_P))_V$ must occur after $\langle\langle N_P \rangle\rangle_{P<}$. This, together with c2, gives us the result we need.

We are now able to conclude that V knows

$$\nu N_V)_V \wedge (\nu N_P)_P\ (\langle N_V \rangle_V^{t_1} < ((N_V))_P < \langle\langle N_P \rangle\rangle_P < ((N_P))_V$$

and we are thus able to conclude that $v((t_2 - t_1 - I_0)/2 \geq d(V, P)$.

Case of the Dishonest Prover

We first note that, although in our definition of an honest prover the prover responds after receiving the verifier's nonce, our result would hold even if it attempted to respond before receiving it, thanks to the new axiom. Thus our result holds for a prover who follows all the rules of the protocol but may attempt to respond early or late, as well as a completely honest prover.

However, we are not able to prove any such results about the semi-honest prover. The reason is that proving such result would require strong assumptions about the behavior of other dishonest nodes as well. Suppose that a dishonest (or badly implemented) prover P sends out a predictable value instead of generating a nonce. Then an attacker A who is closer to the verifier than P is, could if it is aware of this, anticipate P's rapid response before P does, thus making P looking closer than it is. In order to rule out this kind of attack, we would need to make the assumption that A could not anticipate P's response, which is so close to the assumption of the behavior of the honest P who sends an unpredictable nonce as to make no difference.

This problem is closely related to Desmedt's "terrorist attack" involving colluding verifiers. Consider the case in which both Q, the sender of the rapid response and P, the sender of the authenticated response, disobey pr. If Q and P share N_P, then Q could send $h(N_P, P) \oplus N_V$ in P's stead. If Q was closer to V than P, then P could use Q's response to pretend to be closer to V than it was. Of course, there is no reason for Q to cooperate with P in this way unless they are actively colluding, which is why we say that the protocol is vulnerable to collusion attacks.

We note that the Čapkun-Hubaux, Brands-Chaum, and Hancke-Kuhn protocols are vulnerable to the same type of collusion attacks, as are most other distance bounding protocols. Indeed, Brands and Chaum [2] pointed out in their original paper that their protocol was subject to this type of attack. Existing schemes for avoiding the terrorist attack rely either on tamper-proof hardware [15, 17] or on forcing the conspirators to reveal long-term keys to each other [3]. However, we would expect both of these types of solutions, although they may be useful for certain kinds of wireless networks, to find only limited applications in sensor network security. Forcing potential cheaters to share long-term secrets if they want to collude only makes sense when the parties are mutually distrusting. If, as in the case of a sensor network, they are more likely to have been compromised by the same attacker, it is not likely to provide much deterrence. Likewise, tamperproof hardware may not be the optimal solution in a sensor network in which one is highly motivated to keep hardware costs low because nodes may be lost, stolen, destroyed, or power-depleted. In the next section, we consider the problem of detecting, rather than preventing or deterring, collusion attacks. We show that colluding verifiers who are capable of implementing wormhole attacks can defeat even protocols such as SPINE that use triangulation to detect cheating.

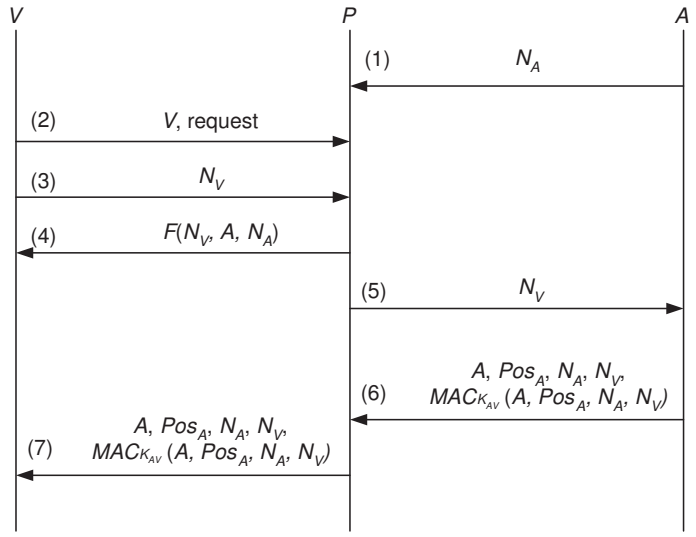
5 Analysis of Distance Bounding Under User Collusion

We now address the question of the impact of collusion on the distance bounding protocols that try to incorporate security. Distance bounding protocols are subject to collusion because they rely upon the keeping secrets and/or the delayed release of information to achieve security. If that information is shared, then collusion is possible. Using this shared information, the adversary then tries to make the verifier believe it is indeed executing all steps of the protocol. The desired outcome for the colluders is to make an adversary or cheater appear closer to the verifier than it really is. The end effect is that the relative location is artificially enlarged to include the colluding node that is far away. We will illustrate this using (a) standard collusion and also (b) wormhole in Figure 2. A wormhole attack is one in which a fast link is set up between the victims and an attacker who is outside of the normal range. The wormhole attack may appear to be an overkill for this problem since even without the resources to establish wormhole, adversaries can collude and create damage. However, wormholes can increase the range of the colluder who is farther away, thus increasing the amount of error that can be induced by collusion.

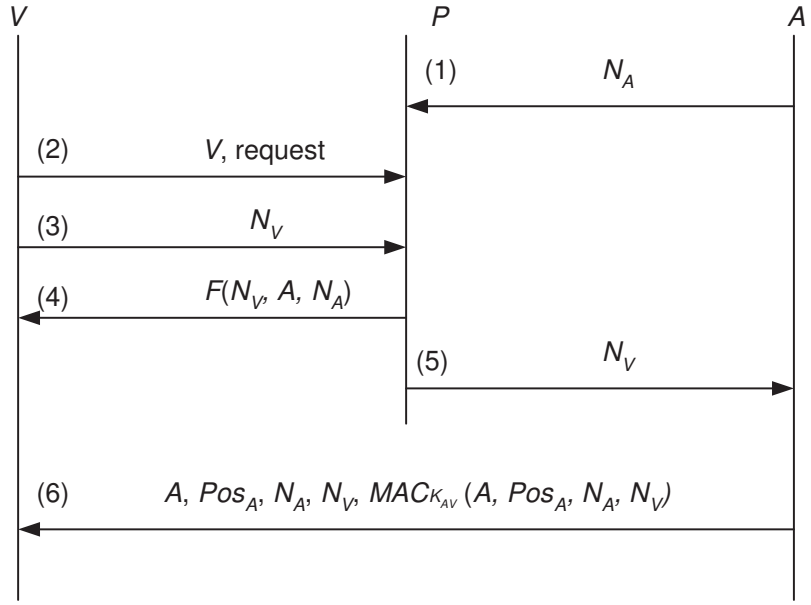
The attack for Figure 2 proceeds as follows: Colluding node P receives a nonce from colluder node A (This step can be removed if the reduction in communication is to be minimized, but the nodes P and A must know the nonce N_P at some point for executing the MAC. Node P would then declare its distance from node V to be that between nodes V and A, denoted d_{AV}. The third step in the protocol is now changed to $F(N_V, A, N_A)$. The node P then transmits the nonce N_V to node A, which computes $MAC_{K_{AV}}(A, Pos_A, N_A, N_V)$ and transmits it to node P. The final step is executed by node P by transmitting the message to verifier node V. Note that node A is assumed not to be able to communicate the information directly to the verifier node V in this version of the protocol. Thus the node P must execute the last step to complete the protocol.

In the case that there is a wormhole link such that the node A is able to transit the data without having a terminal node at both ends of the wormhole (say for example using a directional antenna) then the last step of the protocol termination does not have to involve node P and is modified as shown in figure2.

The next question then is: how does one recognize the existence of such collusion in the distance bounding protocol. We claim that if the nodes A, and P do form a collusion, and behave consistently with respect to relative distance measures and compute the MAC and terminate the last step of the protocol within a ”reasonable” time interval, there is no mechanism to detect the user collusion since it will create no inconsistency, and hence the protocol will exist with faulty measurements only. In making this claim, we have tried to stay away from showing how the detection can be made under certain assumptions about the existence of honest nodes since, even with that assumption, one can show that for infinitely many cases that the collusion cannot be detected. Our claims hold even if there are more than three independent verifiers as in the case of multilateration, as long as the power levels of the transmissions from nodes A and P are consistent. Figure3 illustrates a successful collusion with three



(a) Standard collusion.



(b) Collusion using wormhole.

Fig. 2. Protocols under Collusion of nodes A and P

verifiers. Hence our conclusion is that the secure distance bounding is vulnerable to collusion and in general the collusion cannot be detected.

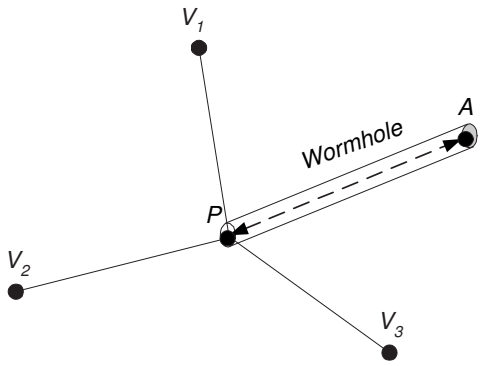


Fig. 3. Illustration of a successful collusion of nodes P and A through wormhole in presence of three verifiers V_1, V_2, and V_3.

6 Conclusion

We have presented a new protocol for distance bounding that requires less message and cryptographic overhead than similar protocols, while still possessing the property of delaying authentication, which can be desirable in a number of applications. More importantly, we have provided a qualitative logical analysis that makes the relationship between authentication and distance measurement clear. Moreover, in doing so we have extended our logic to cover exclusive-or in a way that we think will be applicable to many other equational theories as well. Furthermore, we provide a framework which can be used to in the evaluation of other distance bounding protocols as well. Finally, we point out some fundamental limitations in current distance bounding technologies; the use of cryptographic authentication means that even the ones that are designed to resist collusion are subject to attacks in which attacks in which one dishonest verifier shares its keying material with another. Moreover, these attacks are not detectable by protocols such as SPINE that use triangulation to detect dishonest non-colluding verifiers.

There is still of course, much to be done. Although our logic gives a framework for analyzing distance bounding protocols, it is still only a qualitative framework. What is really needs is a method for analyzing distance bounding protocols that combines both the logical method and the analytical approach of Sastry et al. This will also help us to derive tight bounds on the errors involved in communicating with an honest or isolated dishonest prover. Our logic is intended to be extensible to quantitative as well as qualitative theories, and we believe that the relatively simple distance bounding protocols would make a good test case, as will as providing a unified theory within which distance bounding can be analyzed.

Even if a fully worked our formal theory that covers both qualitative and quantitative aspects is developed, however, the problem of collusion remains. At this point, it seems to make sense to regard distance bounding as a tool which can be used to verify distance from an honest prover and provide a lower bound on the distance of an

isolated cheating prover. Incorrect locations calculated from colluding verifiers may show up as inconsistencies when compared against locations computed from honest verifiers. These inconsistencies could then be exploited to detect the dishonest verifiers, as long as certain assumptions about the distribution of the verifiers hold (e.g. that they are in the minority). We note that such techniques, e.g. SERLOC [9]and HIRLOC [10] have been developed to detect wormhole attacks on range-free location, and would expect a similar approach to work here.

References

1. MSSI Completes Phase II SBIR Contract for UWB-Based Urban Positioning System ('UPS') - Awarded $1M Phase II Plus, June 27 2006.
2. S. Brands and D. Chaum. Distance-bounding protocols. In *Advances in Cryptology - Eurocrypt '93*. LNCS 765, Springer-Verlag, 1995.
3. L. Bussard. *Trust Establishment Protocols for Communicating Devices*. PhD thesis, ESNT Paris, October 2004.
4. Iliano Cervesato, Catherine Meadows, and Dusko Pavlovic. An encapsulated authentication logic for reasoning about key distribution protocols. In Joshua Guttman, editor, *Proceedings of CSFW 2005*, pages 48–61. IEEE, 2005.
5. Y. Desmedt. Major security problems with the 'unforgeable' Feige-Shamir proofs of identity and how to overcome them. In *Proceedings of Securicom '88*, 1988.
6. G. Hancke and M. Kuhn. An RFID distance bounding protocol. In *Proc. of Securecomm 2005*, 2005.
7. Leslie Lamport. Time, clocks, and the ordering of events in a distributed system. *Commun. ACM*, 21(7):558–565, 1978.
8. L. Lazos, S. Čapkun, and R. Poovendran. ROPE: Robust position estimation in wireless sensor networs. In *The Fourth International Conference on Information Processing in Sensor Networks (ISPN '05)*, April 2005.
9. L. Lazos and R. Poovendran. SerLoc: Robust localization for wireless sensor network protocols. *ACM Transactions on Sensor Networks*, 2005.
10. Loukas Lazos and Radha Poovendran. HiRLoc: Hi-resolution robust localization for wireless sensor networks. *IEEE Journal on Selected Areas in Communication*, 24(2):993–999, February 2006.
11. Catherine Meadows and Dusko Pavlovic. Deriving, attacking and defending the GDOI protocol. In Peter Ryan, Pierangela Samarati, Dieter Gollmann, and Refik Molva, editors, *Proc. ESORICS 2004*, volume 3193 of *Lecture Notes in Computer Science*, pages 53–72. Springer Verlag, 2004.
12. Catherine Meadows, Paul Syverson, and LiWu Chang. Towards more efficient distance bounding protocols. In *SecureComm 2006*, August 2006.
13. Dusko Pavlovic and Catherine Meadows. Deriving secrecy properties in key establishment protocols. In Dieter Gollmann and Andrei Sabelfeld, editors, *Proceedings of ESORICS 2006*, Lecture Notes in Computer Science. Springer Verlag, 2006. to appear.
14. N. Sastry, U. Shankar, and D. Wagner. Secure verification of location claims. In *ACM Workshop on Wireless Security (WiSe 2003)*, pages 48–61. ACM, September 19 2003.
15. D. Singleé and B. Preneel. Location verification using secure distance bounding protocols. In *International Workshop on Wireless and Sensor Network Security*. IEEE Computer Society Press, 2005.

16. S. Čapkun and J. P. Hubaux. Secure positioning in wireless networks. *IEEE Journal on Selected Areas in Communication*, 24(2), February 2006.
17. B. Waters and E. Felten. Secure, private proofs of location. Technical Report TR-667-03, Princeton, 2003.