Group Distance Bounding Protocols
Short Paper

Srdjan Capkun[1], Karim El Defrawy[2*], and Gene Tsudik[2]

[1] ETH Zurich capkuns@inf.ethz.ch
[2] UC Irvine {keldefra,gts}@ics.uci.edu

Abstract. Distance bounding (DB) protocols allow one entity, the verifier, to securely obtain an upper-bound on the distance to another entity, the prover. Thus far, DB was considered mostly in the context of a single prover and a single verifier. There has been no substantial prior work on secure DB in group settings, where a set of provers interact with a set of verifiers. The need for group distance bounding (GDB) is motivated by many practical scenarios, including: group device pairing, location-based access control and secure distributed localization. This paper addresses, for the first time, one-way GDB protocols by utilizing a new *passive DB* primitive. We show how passive DB can be used to construct secure and efficient GDB protocols for various one-way GDB settings. We analyze the security and performance of proposed protocols and compare them with existing DB techniques extended to group settings.

1 Introduction

Enabled by pervasive availability of location information, new wireless communication scenarios have emerged where accurate proximity information is essential for both applications and basic networking functions. Such scenarios require secure, reliable and efficient verification of distances between nodes. Distance Bounding (DB) addresses such scenarios by allowing one entity (verifier) to obtain an upper bound on the distance to another entity (prover) and, optionally, authenticate the latter. DB was introduced by Brands and Chaum [3] as a means of preventing so-called “mafia fraud” attacks on bank ATMs. Such an attack occurs if the adversary identifies itself to the verifier using the identity of the prover, without the latter being aware, i.e., man-in-the-middle attack. In [3], a user’s smart-card (verifier) checks its proximity to the ATM (prover). DB has been recently implemented [17] using commercial off-the-shelf electronics with 15cm accuracy. It was also suggested as means of securely determining node locations in wireless networks [14, 5, 8, 20]. In most prior work, DB was considered in the context of a single prover and a single verifier. Group Distance Bounding (GDB) is the natural extension of distance bounding to group settings with multiple provers and verifiers.

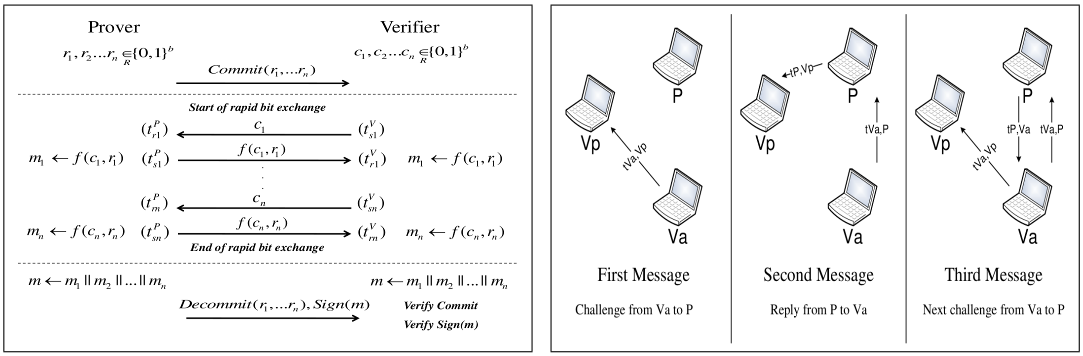
GDB is motivated by some emerging wireless applications. First is group device pairing, a procedure for setting up an initial secure channel among a group of previously unassociated wireless devices; e.g., several users establishing keys among their devices [7] or a single user with multiple devices in a home-area network [4]. In either case, GDB can help verify that the entire group of devices is clustered within a particular area. GDB is also useful in critical (e.g., military) mobile ad hoc network (MANET) settings where all nodes must track locations of, and authenticate, other friendly nodes [2]. Critical MANETs can operate in hostile environments where node compromise is quite realistic. GDB can be used for location based-access control, node tracking and location-based group key management.

We begin by showing that straightforward extensions of current single prover-verifier DB techniques to GDB is inefficient and insecure for localization, without synchronization between verifiers[3]. We then continue by exploring and constructing more efficient and secure GDB techniques. This work makes four contributions: (1) definition of Group Distance Bounding (GDB), (2) a novel one-way *passive DB* primitive, (3) a secure and efficient GDB protocol for several group settings and (4) security and performance analyses of the proposed protocol.

This paper is organized as follows: prior DB protocols, formulation of the GDB problem and our system and adversary models are discussed in Section 2. Passive DB, including its security analysis, is presented in Section 3. Applications of passive DB to the construction of GDB protocols are discussed in Section 4. Performance and security aspects of proposed GDB protocols are considered in Section 5. Related work is overviewed in Section 6, followed by future work summarized in Section 7.

[*] This work was conducted as an academic guest at the System Security Group at ETH Zurich.

[3] This was also pointed out in [8]


Prover
Verifier
$r_1, r_2 \ldots r_n \underset{R}{\in} \{0,1\}^b$
$c_1, c_2 \ldots c_n \underset{R}{\in} \{0,1\}^b$
$Commit(r_1, \ldots r_n)$
Start of rapid bit exchange
(t^P_{r1})
c_1
(t^V_{s1})
$m_1 \leftarrow f(c_1, r_1)$
(t^P_{s1})
$f(c_1, r_1)$
(t^V_{r1})
$m_1 \leftarrow f(c_1, r_1)$
(t^P_{rn})
c_n
(t^V_{sn})
$m_n \leftarrow f(c_n, r_n)$
(t^P_{sn})
$f(c_n, r_n)$
(t^V_{rn})
$m_n \leftarrow f(c_n, r_n)$
End of rapid bit exchange
$m \leftarrow m_1 \| m_2 \| \ldots \| m_n$
$m \leftarrow m_1 \| m_2 \| \ldots \| m_n$
$Decommit(r_1, \ldots r_n), Sign(m)$
Verify Commit
Verify Sign(m)
Vp
P
Va
tVa,Vp
First Message
Challenge from Va to P
Vp
P
Va
tP,Vp
tVa,P
Second Message
Reply from P to Va
Vp
P
Va
tVa,Vp
tP,Va
tVa,P
Third Message
Next challenge from Va to P

(a) Basic DB Operation

(b) Messages Observed by Passive Verifier.

Fig. 1. DB Operation and Messages Observed by a Passive Verifier

2 Preliminaries

This section overviews DB protocols, formulates the GDB problem and presents our environmental assumptions.

2.1 Overview of Distance Bounding (DB)

Figure 1(a) shows a generic (Brands-Chaum-based) *one-way DB* protocol. The core of any distance bounding protocol is the distance measurement phase, in which the verifier measures a round-trip time between sending its challenge and receiving the prover's reply. The verifier's challenge is unpredictable and each reply needs to be computed as a function of the received challenge. Thus, the prover cannot reply before receiving a challenge. Consequently, it cannot pretend to be closer to the verifier than it really is (only further). First, the verifier and the prover each generate n b-bit nonces c_i and r_i ($1 \leq i \leq n$), respectively. In the Brands-Chaum DB protocol [3], the prover also commits to its nonces using any secure commitment scheme. The verifier sends all c_i to the prover, one at a time. Once each c_i is received, the prover computes, and responds with a function of both nonces, $f(c_i, r_i)$. The verifier checks the reply and measures elapsed time between each challenge and response. The process is repeated n times and the protocol completes successfully only if *all* n rounds complete correctly. Prover's processing time: $\alpha = t^P_s - t^P_r$ must be negligible compared to time-of-flight; otherwise, a computationally powerful prover could claim a false bound. This time might be tolerably small, depending on the underlying technology, the distance measured and required security guarantees: less than $1nsec$ processing time yields $0.15m$ accuracy [17].

Security of DB protocols relies on two assumptions: (1) verifier's challenges must be random and unpredictable, and (2) challenges traverse the distance between the two parties at maximum possible speed, i.e., the speed of electromagnetic waves. After running a DB protocol, the verifier knows that the distance to the prover is at most $\frac{t^V_r - t^V_s - \alpha}{2} \cdot c$, where α is prover's processing time and c is the speed of light [3]. DB protocols typically require $(2n + \mathcal{C})$ messages, where $\mathcal{C}$ is the number of messages exchanged in the pre- and post-processing protocol phases. Typically, $\mathcal{C} << n$ and can thus be ignored.

In some applications, e.g., distributed localization, there is a need for mutual DB between two parties: P_1 and P_2. This can be achieved by modifying the one-way DB protocol such that each response from P_2 to a challenge by P_1 also includes a challenge from P_2 to P_1 (more details in [22]). If prover authentication is required, public key signatures can be used to sign challenges and responses. The verifier validates the signature in the last step, as shown in Figure 1(a). The protocol succeeds *only* if the signature is valid. Public key identification schemes, e.g., Schnorr or Fiat-Shamir can also be used, as described in [3].

2.2 Problem Statement and System Model

We first present the general GDB problem statement and its variants, then describe our system and adversary models.

$DB(s)$	Distance Bound(s)
P	Prover
$V \quad (V_a, V_p)$	Active and Passive Verifier, respectively
$DB_{x,y}$	DB established by verifier x to prover y
$t_{x,y}$	Time of flight between x and y
$d_{x,y}$	Distance between x and y ($d_{x,y} = d_{y,x}$)
$n \quad (n_a, n_p)$	Number of active and passive DB rounds, respectively
d_a	Fraction of verifiers performing n_a active rounds
$H()$	Cryptographic hash function
$Pr_{ch}(X)$	Fraction of DB rounds where node X cheats

Table 1. Notation.

GDB Problem Statement: GDB involves *one or more provers* interacting with *one or more verifiers*. Verifiers want to accurately and securely establish DBs to provers and, optionally, authenticate them. Provers are generally untrusted; they may behave maliciously by reporting false distances and identities. In the most general case, any device can be a prover, a verifier or both, corresponding to one-way or mutual DB. We consider three GDB cases: (1) N verifiers establish DBs to M provers (*MPNV*), (2) N verifiers establish DBs to a single prover (*1PNV*), and (3) a single verifier establishes DBs to M provers (*MP1V*). In mutual GDB, two special cases ($1PNV$ and $MP1V$) are equivalent; we refer to them as 1-to-M. There is an additional case where N peer nodes need to establish mutual DBs, that we refer to as *mutual multi-party* GDB. This paper focuses on the one-way GDB settings.

System Model: Our assumption are as follows: (1) *Coverage:* all devices are within each others' transmission range. [4] (2) *Accuracy:* each device is capable of fast and accurate processing, on the order of nanoseconds. This is possible using off-the-shelf components [17] or UWB ranging platforms e.g.[1]. (3) *Interaction between Verifiers:* Verifiers know each others' locations or distances separating them. Assumptions (4) and (5) are needed only if authentication is performed. (4)*Keys:* each device has a public/private key-pair and a certificate binding the public key to its identity. (5) *Collusion:* colluding provers do not reveal their private keys to each other.

Adversary Model: We assume that the adversary is computationally bounded and can not prevent nodes within its radio range from receiving transmissions, i.e., is not using directional antennas. The adversary can compromise only provers. Verifiers trust each other. Our adversary model covers the following attacks (based on DB [3]):

1– Distance Fraud Attack: A dishonest prover claims to be closer than it really is, aiming to shorten the distance to one or more verifiers. Note that a prover can always claim to be further by delaying responses.

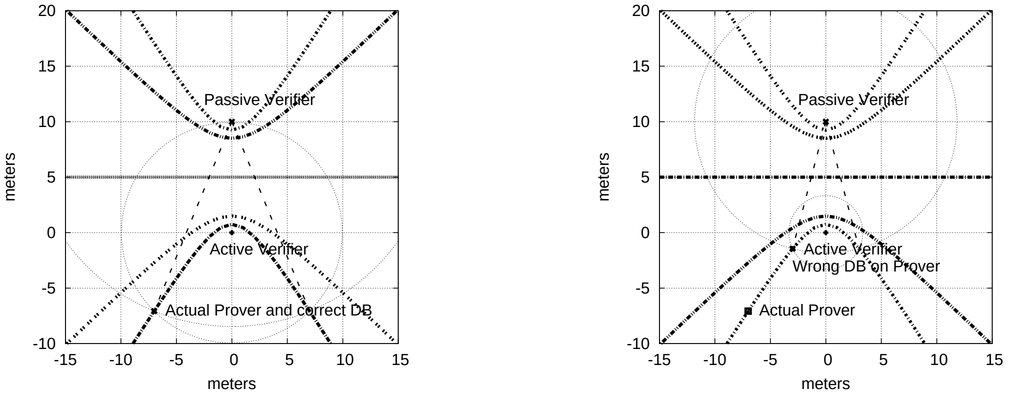
2– Mafia Fraud Attack: The adversary, posing as prover, interacts with the verifier. In parallel, it interacts with the actual prover posing as the verifier. The goal is to fool the verifier into believing that the adversary is the prover located closer to the verifier than the actual prover. We consider a GDB version of this attack where the adversary places one or more nodes between the prover(s) and one or more verifiers. The adversary aims to convince verifiers that these malicious nodes are real provers.

We assume that the DB protocols used in the construction of our GDB protocols are designed such that they prevent Distance Fraud and Mafia Fraud Attacks and further prevent Distance Hijacking attacks [9]. We do not consider Terrorist Fraud Attacks.

3 Passive DB

Whenever a prover and a verifier engage in a DB protocol, some information about their locations and mutual distance is leaked [13]. We use this observation in the presence of multiple verifiers and show that it is unnecessary for *every verifier* to directly interact with the prover (P) to establish a DB to that prover. If *at least one* verifier (V_a) interacts with P, any other (passive) verifier (V_p) can deduce the DB between itself and P by observing messages between P and V_a. We assume that V_p and V_a trust each other, know the distance separating them (or each other's locations) and both need to establish a DB to P. We address passive DB with untrusted verifiers in Section 5.3.

[4] This is a common assumption in all DB literature, e.g., [3, 14, 20, 19, 13, 21].



(a) Correct Passive DB

(b) Incorrect Passive DB

Fig. 2. Establishing a Correct and Incorrect Passive DB.

Figure 1(b) shows V_p observing the timings (T_i) of messages exchanged between P and V_a. V_p constructs three equations:

$$T_1 = t_0 + t_{V_a,V_p} \tag{1}$$

$$T_2 = t_0 + t_{V_a,P} + \alpha_p + t_{P,V_p} \tag{2}$$

$$T_3 = t_0 + 2 \cdot t_{V_a,P} + \alpha_P + \alpha_{V_a} + t_{V_a,V_p} \tag{3}$$

where α_P and α_{V_a} are processing times of P and V_a, respectively (ideally $\alpha_P = 0$) and t_0 is the protocol starting time. V_p can determine time of flight for signals between P and V_a thus computing the distance between them (where c denotes speed of light):

$$d_{V_a,P} = c \cdot t_{V_a,P} = c \cdot \frac{(T_3 - T_1) - \alpha_P - \alpha_{V_a}}{2} \tag{4}$$

For V_a (and V_p) to measure the distance between itself and P, α_P must be negligible (or constant and known)[5].

Overview: V_p uses time differences of arrival (TDoA) of three messages, its own and V_a's locations to construct the locus of P's possible locations (a hyperbola, similar to other TDoA techniques [11]). V_p then determines the distance between V_a and P (as shown in Equation 4) and constructs a circle with the radius of that distance. This circle intersects with P's location locus at two points (s_1 and s_2). V_p computes DB to P as the distance between itself and s_1 (or s_2)[6].

Passive DB Details: We show, among other things, that if P fools V_p with a shorter-than-real passive DB, then the active DB established by V_a *must* also be shortened. However, since V_a's DB can not be shortened, a passive DB is as secure as its active counterpart. Suppose V_a and V_p are located at (x_a, y_a) and (x_p, y_p), respectively. V_p knows both of these locations and the distance d_{V_a,V_p}). Without the loss of generality, we assume $(x_a, y_a) = (0, 0)$ to be the origin of the coordinate system. It follows that:

$$d_{V_a,V_p} = \sqrt{(x_p - x_a)^2 + (y_p - y_a)^2} = \sqrt{(x_p)^2 + (y_p)^2} \tag{5}$$

We further assume that P is at (x, y). V_p also knows that:

$$d_{V_p,P} = \sqrt{(x - x_p)^2 + (y - y_p)^2} = \sqrt{(x)^2 + (d_{V_a,V_p} - y)^2} \tag{6}$$

[5] This is a common assumption in the literature [3, 14, 20, 19, 13, 21].

[6] If V_p does not know V_a's exact location, but only its distance to V_a, then, instead of a sector of a circle, V_p obtains an area between two circles with radii corresponding to furthest and closest points to V_p on the hyperbola. In that case, the larger radius will be used as a DB to P.

$$d_{V_a,P} = \sqrt{(x - x_a)^2 + (y - y_a)^2} = \sqrt{(x)^2 + (y)^2} \tag{7}$$

If three messages in Figure 1(b) are received at times: T_1, T_2 and T_3, V_p computes $d_{V_a,P}$ as shown in Equation 4. V_p also computes:

$$c \cdot (T_2 - T_1) = c \cdot \delta_1 = d_{V_a,P} + c \cdot \alpha_P + d_{P,V_p} - d_{V_a,V_p} \tag{8}$$

Where c is the speed of light. However, since $d_{V_a,P}$ (Equation 4) and d_{V_a,V_p} (verifiers know distances between them) are known, V_p obtains:

$$\begin{aligned} \mathit{\Gamma} = c \cdot (\delta_1 - \alpha_P) + d_{V_a,V_p} = d_{V_a,P} + d_{V_p,P} = \\ \sqrt{(x)^2 + (y)^2} + \sqrt{(x)^2 + (d_{V_a,V_p} - y)^2} \end{aligned} \tag{9}$$

This yields the following formula for the locus of P's possible location (on a hyperbola, due to TDoA [11]):

$$y = \frac{d_{V_a,V_p}\sqrt{(d^2_{V_a,V_p} - \mathit{\Gamma}^2)} \pm \mathit{\Gamma} \cdot \sqrt{(4x^2 + d^2_{V_a,V_p} - \mathit{\Gamma}^2)}}{2\sqrt{(d^2_{V_a,V_p} - \mathit{\Gamma}^2)}} \tag{10}$$

Note that $DB_{V_a,P} = d_{V_a,P}$ is an upper bound for the distance between P and V_a. Using $d_{V_a,P}$, V_p can construct another equation for the locus of P's possible location (a circle around V_a with radius $d_{V_a,P}$):

$$(x - x_a)^2 + (y - y_a)^2 = (d_{V_a,P})^2 \tag{11}$$

V_p can now establish a passive DB using the intersection of both loci (i.e., solving equations 11 and 10). The computed DB is the distance between V_p's own location (x_p, y_p) and the intersection of P's loci described by equations 11 and 10. The end point of this DB ($DB_{V_p,P} = d_{V_p,P}$) lies within a sector of a circle, not within the entire circle, as in the case of active DB. Substituting $x = x_a + \sqrt{(d_{V_a,P})^2 - (y - y_a)^2}$ (from equation 11) into equation 10, the y-coordinate of P's location becomes: $y \propto (d_{V_a,P})$ (same for P's x-coordinate). For V_p to compute a wrong (shorter) DB to P, it has to have computed a shorter $d_{V_a,P}$. A shorter $d_{V_a,P}$ requires $DB_{V_a,P}$ to have been computed shorter than the actual distance between P and V_a (which is impossible, as shown in Section 2.1).

To better illustrate this, Figures 2(a) and 2(b) show an example scenario. P (labeled Actual Prover in the Figures) at $(-7, -7)$ is on one of several possible hyperbolas. The DB from V_a at $(0, 0)$ to P is shown as a circle around V_a (labeled as Active Verifier) If P cheats, the circle drawn around V_p at $(0, 10)$ would intersect the hyperbola at a point ($(-3, -1.5)$ in Figure 2(b)), close to V_p. This point would be inside the circle established by V_a. However, the DB computed by V_a would be shorter than the actual distance to P, which is impossible. If V_p engages in a DB protocol with V_a, it would obtain the circle shown around it in Figure 2(a). However, in this passive case, it obtains a sector of that circle defined by the arc connecting two points ($(-7, -7)$ and $(7, -7)$) where the computed hyperbola intersects the circle around V_a. Section 2.1 shows that active DB prevents the distance fraud attack. Since passive DB is as secure as active DB, it also prevents the distance fraud attack. Adding authentication to passive DB prevents the mafia-fraud attack on passive DB since an attacker would be unable to authenticate itself to V_p unless it also does so to V_a. Both types of verifiers can use the same authentication mechanism, e.g., public key signatures (or public key identification schemes), as described in Section 2.1. All information (commitments, challenges, responses and signatures) required to authenticate provers also reaches passive verifiers. The only disadvantage is that V_p does not generate its own challenges. Therefore, passive DB remains secure as long as active verifiers are trusted. We assume that the active DB protocol also prevents distance hijacking attacks.

4 One-Way DB extended to Group Settings

We focus on the most general one-way GDB case: M provers and N verifiers (MPNV). All other scenarios correspond to special cases. For comparison, we consider a GDB protocol where nodes sequentially engage in a naíve single prover single verifier DB. In each case, we propose an alternative approach based on passive DB. We assume that n rounds of DB are needed in all cases.

Setting	Base Case # Messages	Our Protocol # Messages	Base Case Time	Our Protocol Time
MPNV	$2n \cdot N \cdot M$	$(2n_a + 1) \cdot (N \cdot d_a) \cdot M$	$2n \cdot \sum_{i=1}^{N} \sum_{j=1}^{M} t_{V_i,P_j}$	$(2n_a + 1) \cdot \sum_{j=1}^{d_a \cdot N} \sum_{k=1}^{M} t_{P_k,V_j}$
1PNV	$(2n + 1) \cdot N$	$(2n_a + 1) \cdot N \cdot d_a$	$2n \cdot \sum_{i=1}^{N} t_{P,V_i}$	$(2n_a + 1) \cdot \sum_{j=1}^{d_a \cdot N} t_{P,V_j}$
MP1V	$(2n + 1) \cdot M$	$2n + \sum_{j=1}^{M-1} (j + 1) \cdot (n - ((M - 1) - j))$	$2n \cdot \sum_{i=1}^{j} t_{V,P_i}$, $j \in 1, M$	$(n \cdot max(t_{V,P_i})) + \sum_{i=1}^{M-1} t_{V,P_i}$

Table 2. Number of Messages (# Messages) and Execution Time of One-Way GDB Protocols.

In a one-way MPNV setting, nodes act either as provers or as verifiers. In a naive MPNV protocol, each prover sequentially interacts with each verifier. This is repeated until *all* provers have interacted with all verifiers. The total number of messages is: $(2n \cdot N \cdot M)$. In this context, there are two parameters to consider: (1) number of active and passive DB rounds performed by each verifier and (2) selection of active verifiers (i.e., deterministic or probabilistic). The second parameter does not affect how nodes act, however, it impacts security if verifiers are compromised (see Section 5). Active verifiers can be selected at random or by any leader election protocol, e.g., [15]. The numbers of active verifiers and active rounds influence the number of messages required, the time to complete the protocol, the security of the DB. If all verifiers are treated equally, two parameters can describe a general protocol: (1) the number of active verifiers and (2) the number of active rounds by each verifier. If each verifier performs n rounds, let n_a and n_p ($n_a + n_p = n$) be the number of active and passive rounds, respectively. Let d_a be the fraction of verifiers which perform n_a active rounds. Each verifier has $(d_a \cdot (N - 1) \cdot n_a)$ opportunities to execute passive DB with each prover. Two interesting special cases are where (1) $d_a = 1/N$ and $n_a = n$ – one verifier interacts with all provers, and (2) $d_a = 1$ and $n_a = n/N$ – all verifiers interact with *all* provers. By varying n_a and d_a, we can obtain a protocol with the required security level and fewer messages than in the sequential pairwise interaction.

5 GDB Performance and Security Analysis

5.1 Performance

Table 2 compares number of messages in one-way GDB protocols based on passive DB to the base case of pairwise DB. Table 2 also shows total time required to compute all DBs. We compare with this base case since there are no prior GDB proposals. Our protocols require fewer messages and depend on the fraction of active verifiers and active rounds. MPNV requires $(n_a \cdot d_a)$ messages. For example, with 60 nodes (30 provers and 30 verifiers) and $d_a = 0.8$, 33% fewer messages are needed. Decreasing d_a to 0.6 saves more than 55% of messages. Similar savings are also attainable for lower and higher numbers of provers/verifiers.

5.2 Security with Trusted Verifiers

The probability of a single prover successfully cheating a single verifier decreases exponentially with the number of DB rounds. For n rounds, a prover has 2^{-n} chance of successfully guessing all challenge bits and sending responses ahead of time. This tricks the verifier into measuring a shorter round trip time of flight[7]. An active verifier (V_a) does not need to trust any other entity. In a group setting, where each prover-verifier pair engages in an active DB protocol, these security guarantees still hold. However, active DB in a group setting is insecure if used for localization. When a prover interacts with each verifier separately, it can selectively lengthen its distance by delaying messages. Verifiers would then incorrectly localize the prover. Secure localization schemes must therefore require *at least three* verifiers that interact with the prover *simultaneously*.

5.3 Security Untrusted Active Verifiers

Passive DB with untrusted verifiers is mainly useful in MANETs, where nodes continuously encounter new peers. Passive DB is secure if each V_a is trusted, as shown in Section 3. This is be the case in a fixed (or mobile) verification

[7] 2^{-n} is the probability for Brands-Chaum protocol [3], whereas in other protocols (e.g., Hancke-Kuhn [12]), this probability is $(3/4)^{-n}$.

infrastructure with prior security association, or under control of one administrative entity. A malicious V_a could undermine security of passive DB as follows:

(1) Report Fake Location (or Distance): V_p requires the exact location of, or the distance to, V_a in order to construct the DB, as shown in Section 3. V_p will compute the hyperbola incorrectly (via Equation 10), if V_a reports an incorrect location or distance.

(2) Sending Early Challenges: Even if V_a reports its location or distance correctly, it can send new challenges prematurely. This would convince V_p that the prover is closer than it actually is, as shown in Figure 1(b). V_p would then compute $d_{V_a,P}$, intersecting incorrectly with the hyperbola, as shown in Figure 2(a), thus yielding a wrong DB.

We introduce a metric, *DB Correctness* (DBC), to illustrate the effectiveness of passive DB in the presence of such attacks:

$$DBC = 1 - 2^{n \cdot (Pr_{ch}(V_a)-1)} \tag{12}$$

where $Pr_{ch}(V_a)$ is the fraction of rounds where V_a cheats. If V_a does not cheat at all, $Pr_{ch}(V_a) = 0$ and $DBC = 1 - 2^{-n}$. If V_a cheats in all rounds, $Pr_{ch}(V_a) = 1$ and $DBC = 0$. Average correctness (DBC_{avg}) For a DB obtained by a V_p to a specific prover (in case of N active verifiers: $V_{a_1}, V_{a_2}...V_{a_N}$, each engaging in $n_1, n_2...n_N$ rounds) average correctness (DBC_{avg}) can be computed as the average of individual DBC for each verifier:

$$DBC_{avg} = \frac{N - \sum_{i=1}^{N} 2^{n_i \cdot (Pr_{ch}(V_a(i))-1)}}{N} \tag{13}$$

For example, for 10 verifiers and 10 DB rounds, even if half of all V_a-s cheat in half of their rounds, DB would be established correctly $> 98\%$ of the time. Also, as long as less than half of V_a-s cheat in $< 90\%$ of rounds, DB would be correct $> 70\%$ of the time.

5.4 Combined Passive/Active DB Security

When a verifier performs n_a active rounds and n_p passive rounds both can be combined to obtain a more stable DB. We estimate the correctness in such a combined DB using a metric ($DBC_{a/p}$) as follows (note that both passive and active rounds have to result in the same DB):

$$DBC_{a/p} = 1 - (2^{-n_a} \cdot \frac{\sum_{i=1}^{N} 2^{n_{p(i)} \cdot (Pr_{ch}(V_a(i))-1)}}{N}) \tag{14}$$

If all other active verifiers cheat in all their DB rounds, $DBC_{a/p}$ becomes that of the active rounds performed by a verifier only, i.e., $1 - (2^{-n_a})$. Otherwise the correctness of the established DB increases with any additional passive round. As an example, consider the case of 10 verifiers. Even if only two rounds of active (n_a) DB are performed and as long as the fraction of rounds being cheated in is less than 1 correctness of the DB captured by $DBC_{a/p}$ increases. Even if the probability of cheating in passive DB rounds is as high as 0.5, $DBC_{a/p}$ will increase to over 0.95 if there are four or more opportunities to do passive DB.

6 Related Work

DB was introduced in [3], as mentioned in Section 2.1. Several DB optimizations and studies were done subsequently. In particular, [13] studied information leakage in DB protocols as a privacy problem. [22] proposed a mutual DB protocol by interleaving challenges and responses; between a single prover and a single verifier. [20], [19] and [5] investigated DB protocols in location verification and secure localization with three verifiers. [18] investigated so-called "in-region verification" and claimed that, for certain applications (such as sensor networks and location-based access control) in-region verification is better than location determination. [8] and [6] considered collusion attacks on DB location verification protocols. Other work, such as [23] looked at using time difference of arrival (TDoA) to determine location of transmitters. [23] proposed using TDoA in the context of Ultra-Wideband (UWB). DB was implemented using commercial off-the-shelf electronic components [17] and commercial off-the-shelf UWB ranging devices [21, 14]. DB was also studied in the context of ad-hoc networks [22], sensor networks [16, 5] and RFIDs [10, 12].

7 Discussion and Conclusion

This paper presents the initial foray into group distance bounding (GDB). GDB is a fundamental mechanism for secure operation in wireless networks where verifying distances between, or locations of, groups of nodes is required. We investigated one-way GDB settings and constructed secure and efficient one-way GDB protocols. In doing so, we made minimal assumptions. However, there remain some open issues for future work, such as: (1) Can a passive verifier establish a DB without knowing the location of (or distance to) an active verifier, while perhaps knowing other information about distances to other nodes? (2) can passive DB be used to obtain mutual GDB protocols? (3) What can be done to address denial-of-service attacks in group settings (i.e., noisy environments)?

References

1. Multispectral Solutions Inc., Urban Positioning System (UPS). `http://www.multispectral.com`.
2. RFC1677-Tactical Radio Frequency Communication Requirements for IPng. `http://www.faqs.org/rfcs/rfc1677.html`.
3. S. Brands and D. Chaum. Distance-bounding protocols. In *EUROCRYPT*, 1994.
4. E. Callaway and P. Gorday. Home networking with ieee 802.15.4: a developing standard for low-rate wireless personal area networks. In *IEEE Communications Magazine*, 2002.
5. S. Capkun and J. Hubaux. Secure positioning of wireless devices with application to sensor networks. In *IEEE INFOCOM*, 2005.
6. N. Chandran, V. Goyal, R. Moriarty, and R. Ostrovsky. Position based cryptography. In *CRYPTO*, 2009.
7. C. Chen, C. Chen, C. Kuo, Y. Lai, J. McCune, A. Studer, A. Perrig, B. Yang, and T. Wu. Gangs: gather, authenticate 'n group securely. In *ACM MobiCom*, 2008.
8. J. Chiang, J. Haas, and Y. Hu. Secure and precise location verification using distance bounding and simultaneous multilateration. In *ACM WiSec*, 2009.
9. C. Cremers, K. Rasmussen, and S. Capkun. Distance hijacking attacks on distance bounding protocols. In *Cryptology ePrint Archive: Report 2011/129*, 2011.
10. S. Drimer and S. Murdoch. Keep your enemies close: distance bounding against smartcard relay attacks. In *USENIX Security Symposium*, 2007.
11. F. Gunnarsson. Positioning using time-difference of arrival measurements. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2003.
12. G. Hancke and M. Kuhn. An rfid distance bounding protocol. In *IEEE SECURECOMM*, 2005.
13. Rasmussen K and S. Čapkun. Location privacy of distance bounding protocols. In *ACM CCS*, 2008.
14. H. Luecken M. Kuhn and N. Tippenhauer. UWB impulse radio based distance bounding. In *Workshop on Positioning, Navigation and Communication (WPNC)*, 2010.
15. N. Malpani, J. Welch, and N. Vaidya. Leader election algorithms for mobile ad hoc networks. In *ACM DIALM*, 2000.
16. C. Meadows, P. Syverson, and L. Chang. Towards more efficient distance bounding protocols for use in sensor networks. In *IEEE Securecomm*, 2006.
17. K. Rasmussen and S. Čapkun. Realization of rf distance bounding. In *USENIX Security Symposium*, 2010.
18. N. Sastry, U. Shankar, and D. Wagner. Secure verification of location claims. In *ACM WiSe*, 2003.
19. V. Shmatikov and M. Wang. Secure verification of location claims with simultaneous distance modification. In *ASIAN*, 2007.
20. D. Singelee and B. Preneel. Location verification using secure distance bounding protocols. In *IEEE International Conference on Mobile Adhoc and Sensor Systems Conference*, 2005.
21. N. Tippenhauer and S. Čapkun. Id-based secure distance bounding and localization. In *ESORICS*, 2009.
22. S. Čapkun. L. Buttyán and J. Hubaux. Sector: secure tracking of node encounters in multi-hop wireless networks. In *ACM SASN*, 2003.
23. D. Young, C. Keller, D. Bliss, and K. Forsythe. Ultra-wideband (uwb) transmitter location using time difference of arrival (tdoa) techniques. In *Conference Record of the Thirty-Seventh Asilomar Conference on Signals, Systems and Computers*, 2003.