

 Open access • Proceedings Article • DOI:10.1109/MAHSS.2005.1542879

Location verification using secure distance bounding protocols — [Source link](#)

Dave Singelée, Bart Preneel





Institutions: Katholieke Universiteit Leuven

Published on: 12 Dec 2005 - Mobile Adhoc and Sensor Systems

Topics: Distance-bounding protocol, Password and Authentication

Related papers:

- [Distance-bounding protocols](#)
- [Secure verification of location claims](#)
- [Secure positioning of wireless devices with application to sensor networks](#)
- [An RFID Distance Bounding Protocol](#)
- [Position Based Cryptography](#)

Share this paper:    

View more about this paper here: <https://typeset.io/papers/location-verification-using-secure-distance-bounding-1jml1vycz>

Location Verification using Secure Distance Bounding Protocols

Dave Singelee, Bart Preneel
ESAT-COSIC, K.U. Leuven, Belgium
Email: Dave.Singelee@esat.kuleuven.be

Abstract—Authentication in conventional networks (like the Internet) is usually based upon something you know (e.g., a password), something you have (e.g., a smartcard) or something you are (biometrics). In mobile ad-hoc networks, location information can also be used to authenticate devices and users. We will focus on how a prover can securely show that (s)he is within a certain distance to a verifier. Brands and Chaum proposed the distance bounding protocol as a secure solution for this problem. However, this protocol is vulnerable to a so-called “terrorist fraud attack”. In this paper, we will explain how to modify the distance bounding protocol to make it resistant to this kind of attacks. Recently, two other secure distance bounding protocols were published. We will discuss the properties of these protocols and show how to use it as a building block in a location verification scheme.

I. INTRODUCTION

A. Location based authentication

A prover convincing a verifier of some assertion is a frequently recurring element in many applications. The assertion is often the identity of the prover, but it can also be more general. Successful authentication provides privileges (e.g., access to a network). In conventional networks, authentication is often based upon something you know (e.g., a password or a secret key), something you have (e.g., a smartcard) or something you are (biometrics). In daily interactions, other assertions, like the location of the proving entity, occur commonly. For instance, one has to be present in a room to be able to use the light switch. To enter a building, one has to open the (closed) door. This is only possible when one stands before the door and has the correct key. These examples show that location based authentication is used very commonly in our daily interactions. It is also sometimes useful in mobile (ad-hoc) networks. E.g., a node in a sensor network would only like to talk to its neighbors. Another example is a user who wants to print confidential documents. How does (s)he know that (s)he is talking to the trusted printer in front of him and not to a malicious one? Some services are only accessible for users inside (or outside) a certain area. One can easily think of other scenarios in which one wants to verify location claims of a prover.

A lot of solutions can be found in the literature [1], [2]. One could for example use GPS coordinates in a location verification scheme [3]. There are however some drawbacks to this method. E.g., it can not be used indoor. Location information does not always have to be so detailed, sometimes we are interested in the orientation (e.g., is the prover to the right

or to the left), the distance, the environment, ... Combining these pieces of information will enable to determine the exact position of the other party. In the rest of this paper, we will focus on a prover claiming to be within a certain distance. The protocols that enable the verifying party to determine an upper-bound on this distance are called *distance bounding protocols*. We will extend this protocol and use it in a secure location verification scheme. In contrast to other solutions, we will not assume that the environment can be trusted and only consider techniques which are resistant to active attackers.

B. Organization of the paper

This paper is organized as follows. In the introduction, we briefly discussed the general idea of authentication based upon location. In the rest of the paper, we will focus on how a prover can show that (s)he is within a certain distance to a verifier. This can be accomplished by using distance bounding protocols. To be secure, such protocols have to prevent distance fraud attacks, mafia fraud attacks and terrorist fraud attacks. This will be comprehensively described in section 2, 3 and 4. Finally, in section 5, we will show how to extend the protocol and use it as a building block in location verification schemes. Because this paper is written from a security point of view, we will only consider protocols that prevent one (or more) of the attacks described below. Most of the (commercial) distance bounding protocols and verification schemes [2] can not be used in security-critical applications.

II. PREVENTING DISTANCE FRAUD ATTACKS

We are interested in solving the following scenario: a verifier wants to check if a prover is within a certain distance, as (s)he claims to be. All entities which are outside this range, will be ignored by the verifier. Just asking the location will not be sufficient because the verifier does not trust the prover. One wants to prevent a dishonest prover claiming to be closer than (s)he really is. This *distance fraud attack* is conceptually shown in Fig. 1. An overview of location mechanisms that prevent this attack (sometimes only partially) can be found in [1]. In the rest of this section, we will focus on two important categories of solutions.

A. Measuring the signal strength

One could control the transmitting range of a wireless signal. It is not difficult to design such a protocol. E.g., the verifier can send out a nonce. If the prover is close, then (s)he

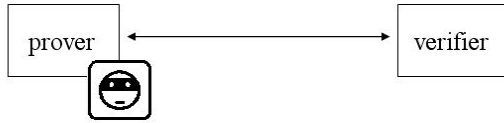


Fig. 1. Distance fraud attack

can prove this by sending a reply including this nonce. One could allow a device access to a wireless ad-hoc network if it is able to detect this network. Another example of a solution based on network visibility is described in [4]. Unfortunately, there is one major security problem. All these mechanisms implicitly assume that the prover has a standard, unmodified device. It is however not so difficult to build a directional antenna to largely increase the sending or receiving range. To illustrate this, let's have a look to the Bluetooth wireless technology. The maximum range of a Bluetooth device is about 10 m [5]. In 2004, the BlueSniper rifle was presented at DEFCON '04 [6]. The device can pick up signals from more than 1.5 km away. This already illustrates that measuring the signal strength does not prevent distance fraud attacks. The same problem arises when the verifier measures the signal strength of messages transmitted by the prover (an example of such a system can be found in [7]).

B. Measuring the round trip time

Another class of solutions measures the round trip time while exchanging messages to determine the distance between the prover and the verifier. The basis principle is as follows: the prover and the verifier perform a challenge-response protocol. This protocol is the major building block of the entire scheme. How does the protocol work? The verifier sends out a challenge and starts a timer. After receiving the challenge, the prover does some very elementary computations to construct the response. In some schemes, these computations are even omitted. The response is sent back to the verifier and the timer is stopped. Multiplying this time with the propagation speed of the signal gives the distance.

One should however take into account some important details. It should be impossible for the prover to send the response before receiving the challenge [8]. This implies that the response should be dependent on the challenge. A second remark is that a challenge-response protocol is not enough. After execution of this protocol, the verifier only knows that some party is close. But how does one know that this entity is the prover? This problem arises for example in the Echo protocol [9]. That is why the prover has to use his private (or secret) key somewhere in the scheme (not necessarily in the challenge-response protocol itself). Finally, one should notice that the round trip time is not equal to the propagation time. It takes some time to calculate and transmit the response. This processing delay should be as small as possible because we are only interested in the propagation time. In almost every practical scenario, it won't be possible to determine the processing delay exactly. The solution for this problem will

be dependent on the propagation speed. Let's examine two techniques: (ultra-)sound and electromagnetic signals.

1) *Ultra-sound*: (Ultra-)sound is interesting to measure distances because it is slow. The processing delay can be neglected compared to the propagation time and the accuracy of the measurements is not very critical. An example of a protocol using this technique can be found in [10]. There are however some security problems. (Ultra-)sound is not resistant to physically present attackers. Such an attacker can modify the medium (e.g., sound travels faster through metal than through the air) or use wormholes (e.g., by retransmitting the signal using electromagnetic waves) to claim that (s)he is closer than (s)he really is. By delaying the response, (s)he can also claim to be further away. Because it is impossible to exclude physically present attackers in a mobile environment, the use of (ultra-)sound signals has to be avoided.

2) *Electromagnetic signals*: If we don't take into account quantum cryptography, an active attacker can not use a wormhole. The signals travel with the speed of light and nothing propagates faster. This means that an attacker can only claim to be further away than (s)he really is (by delaying the response). This can graphically be depicted by a circle around the verifier with radius d (according to the propagation delay) and the prover being somewhere in this circle, as shown in Fig. 2. The attacker can not be outside this circle. There are however some practical issues. The verifier has to be able to measure the round trip time with very high precision. A similar problem is estimating the processing delay. A small deviation of time influences the measured distance a lot (because the speed of light is a very large constant). This implies that measuring the time of flight of electromagnetic signals is only useful when the processing delay will be negligible compared to the propagation delay. In the rest of the paper, we will (implicitly) assume that electromagnetic signals are used.

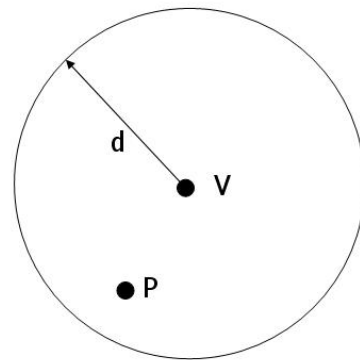


Fig. 2. Distance bounding protocol with 1 verifier

III. PREVENTING MAFIA FRAUD ATTACKS

As discussed in section II-B, measuring the time of flight of an electromagnetic signal in the distance bounding protocol assures that an attacker can not be further away than (s)he pretends to be. Using this principle in a clever way does

not only prevent distance fraud attacks, but also mafia fraud attacks. We will first explain this advanced man-in-the-middle attack and then present the distance bounding protocol of Brands and Chaum.

A. Mafia fraud attack

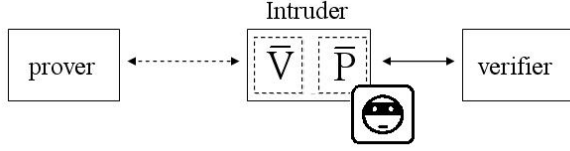


Fig. 3. Mafia fraud attack

Mafia fraud attacks were first described in [11]. It is a man-in-the-middle attack where the intruder I is modeled as a malicious prover \bar{P} and verifier \bar{V} cooperating together, as shown in Fig. 3. The malicious verifier \bar{V} interacts with the honest prover P and the malicious prover \bar{P} interacts with the honest verifier V . The physical distance between I and V is small. This attack enables I to identify himself to V as being P being close to V , without any of P and V noticing the attack.

Mafia fraud attacks are particularly useful for the intruder I in scenarios (1) where authentication is successful when a specific entity is close to the verifier and (2) where the result of a successful authentication is access to a service offered by the verifier. A classical example is a server that only allows access to a local database after a successful execution of a challenge-response protocol. Other examples are described in [8]. To perform the mafia fraud attack, I only has to forward the challenges and responses. The attack is shown in Fig. 4.

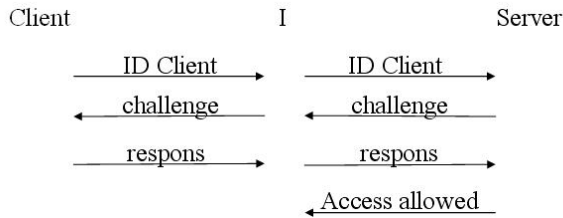


Fig. 4. Classical example of mafia fraud attack

B. Distance bounding protocol (S. Brands, D. Chaum)

In 1993, S. Brands and D. Chaum presented their distance bounding protocol [8]. This clever protocol prevents mafia fraud attacks and consists of a single bit challenge and a rapid single bit response. In practice, a series of these rapid bit exchanges is used. The number of challenge-response interactions is being determined by a chosen security parameter k . The delay time for receiving the responses enables the verifier to compute an upper-bound on the distance. After **correct** execution of the distance bounding protocol, the verifier knows

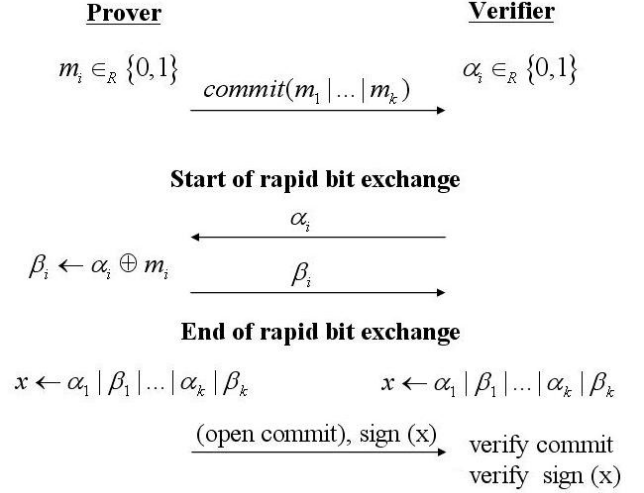


Fig. 5. Distance bounding protocol of Brands and Chaum [8]

that an entity in possession of a certain secret is in the vicinity. The protocol is shown in Fig. 5. It can easily be integrated into common identification protocols. Čapkun et al. extended the protocol to SECTOR, a mutual authentication protocol using distance bounding [12].

Note that the prover has to calculate the *XOR* of two bits. This can be done very efficiently in hardware. It is important that the prover has to perform as few operations as possible during the fast bit exchange. This is important to keep the processing delay very small (see also section II-B). The bottleneck won't be the calculation of the *XOR*, but sending out and receiving of the 1 bit messages. The message has to be read out (or written into) memory, it has to go through some protocol layers to take care of signal-modulation, among other things, ... For this reason, the prover will need dedicated hardware for the distance bounding protocol. The verifier also needs special hardware, namely to measure the round trip time with very high precision (which will be very small).

IV. PREVENTING TERRORIST FRAUD ATTACKS

Designing a distance bounding protocol that is resistant to distance fraud and mafia fraud attacks is not enough. A powerful attacker can perform a *terrorist fraud attack*. We will briefly explain this attack, show that the distance bounding protocol of Brands and Chaum is vulnerable to it and present some countermeasures. There are at least three (modified) distance bounding protocols which are resistant to all the attacks described in this paper.

A. Terrorist fraud attack

The terrorist fraud attack [11] is an interesting extension of the mafia fraud attack. The intruder and the prover will collaborate in this attack, whereas in the mafia fraud attack, only the intruder performs the fraud. This implies that a protocol which is resistant to terrorist fraud attacks also prevents mafia fraud attacks. The terrorist fraud attack is shown in Fig. 6.

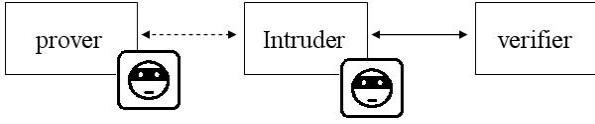


Fig. 6. Terrorist fraud attack

The dishonest prover uses the intruder to convince the honest verifier that (s)he is close.

The intruder does not know the private key (or the secret key if symmetric cryptography is used) of the prover. This certainly has to be emphasized! If the intruder would know this private key, then it is impossible to make a distinction between the intruder and the prover. They would be the same party from cryptographic point of view (distance bounding protocols only check if a party which knows the private key is close to the verifier). It is not unrealistic to assume that the intruder has no knowledge of the private key because in practice, it will be a small device with some computation power and memory (e.g., a sensor). It will be placed nearby the verifier (or nearby the location where the verifier will be present somewhere in the future). The prover does not have full control over the intruder (e.g., the sensor can be stolen or discovered by another party), and therefore won't be willing to give the private key to the intruder.

We will now demonstrate how this terrorist fraud attack can be applied to the distance bounding protocol of Brands and Chaum. The authors themselves already noted that their protocol is vulnerable to this attack. Roughly, one could assert that their distance bounding protocol can be divided in three parts: the commitment phase, the fast bit exchange phase and finally the signing phase (in which also the commitment is opened). There is however no strong (cryptographic) relation between these 3 phases. The verifier has no way of checking if the party that executes the commitment phase is the same as the one that executes the fast bit exchange phase or the signing phase. One is only certain of the fact that the party that executed the fast bit exchange phase is nearby the verifier and that the party that executed the signing phase knows the private key. But how do you know that those 2 parties are in fact the same party (the prover)? The terrorist fraud attack exploits this "uncertainty". An example of such an attack is given in Fig. 7. Another protocol that is vulnerable to this attack can be found in [13]. Every distance bounding protocol in which one is not sure if the party that executes the timed phase (if there is one), knows the private key, is vulnerable to a terrorist fraud attack!

B. Modified distance bounding protocols

Let's generalize the ideas of the previous section. To the best of our knowledge, all the interesting distance bounding protocols that measure the round trip time of the exchanged messages (to determine the upper bound of the distance between prover and verifier), can be divided in at least two phases: a timed phase (in which the time of flight of the

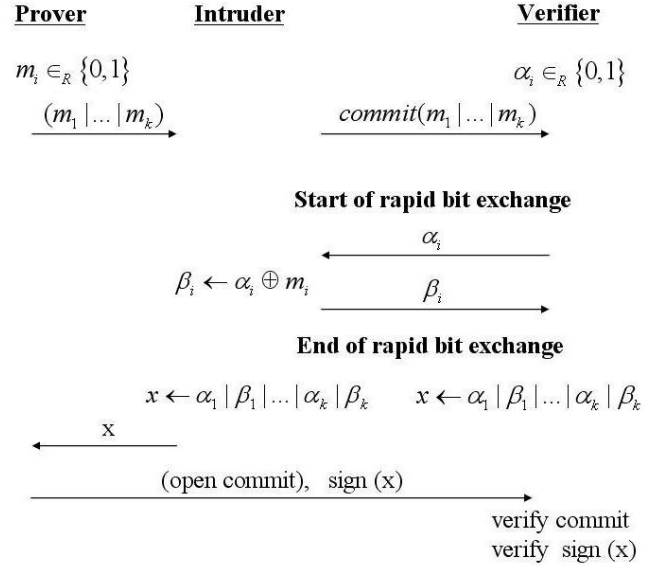


Fig. 7. Example of terrorist fraud attack

electromagnetic signal is measured) and a phase in which one proves the knowledge of a private key. To avoid terrorist fraud attacks, one has to make sure that both phases are intermingled in a cryptographic way. It has to be impossible to divide the distance bounding protocol in the 2 phases mentioned above. There are at least two possibilities to accomplish this. Or one uses the private (or symmetric) key in the fast bit exchange, or one uses trusted hardware.

1) *Using the secret key in the fast bit exchange:* One could replace the XOR in the protocol of Brands and Chaum by the calculation of a signature on some data sent by the verifier. This prevents terrorist fraud attacks, but makes the distance bounding protocol unusable. Calculating a digital signature takes a lot of time [14], and one wants the processing delay as small as possible. Even a symmetric signature algorithm would take too much time. However, this does not imply that it is impossible to use the secret key in the fast bit exchange phase in a clever way. L. Bussard proposed a very nice distance bounding protocol that is resistant to terrorist fraud attacks [1]. It consists of 3 phases. In the first phase, commitments are sent out. In the second phase, there is a series of single bit challenge-response interactions. Accordingly to the challenge of the verifier, the prover responds with a bit of a temporary secret key k or e . Those 2 temporary secret keys are not randomly chosen. There is a relation between them: $e = ux - k$, in which u is a publicly known value and x the (permanent) secret key of the prover. In round i , the verifier learns the i^{th} bit of k or e . Because the verifier never learns both bit values, (s)he can never learn something about the i^{th} bit of the secret key x . In the third phase, the prover uses a zero-knowledge proof to convince the verifier that (s)he knows the secret key x . It is not possible to give an intruder the values of k and e , because it then also would know x . Because of this, the protocol is not vulnerable to a terrorist fraud attack.

The processing delay during the fast bit exchange phase is also very small, the prover only has to send out single bit values. There is however also a major drawback. The protocol uses a lot of modular exponentiations and this is an expensive operation for mobile ad-hoc networks [15]. A more energy-efficient version of the protocol is needed.

2) *Using trusted hardware:* There is another way to enforce a relation between the fast bit exchange phase and the signing phase, namely by using trusted hardware. The trusted hardware has the following properties: it is impossible for an attacker to extract values out of the trusted hardware or to change the protocol that it has to perform. It can only be used as a black-box by the attacker.

Using such hardware protects the distance bounding protocol of Brands and Chaum against terrorist fraud attacks. The values m_i (see section III-B) are kept inside the secure device. A malicious prover can not give them anymore to a collaborating intruder. It is possible to extract these values by executing the protocol in advance (and hence using the trusted hardware as a black-box). Fortunately, the attacker does not know yet the correct values of β_i at that moment. The probability of a correct guess is 2^{-k} with k a security parameter of the system (again, see section III-B). Of course, it is also not possible to extract the private key which is used for signing messages.

Another secure distance bounding protocol that makes use of trusted hardware, is the (improved) protocol of Waters and Felten [4]. Instead of a series of single bit challenge-response interactions, nonces are sent out during the timed phase. Their protocol is very efficient: the response in the challenge-response protocol is only one message. It exists of two nonces: one chosen by the prover and one which was included in the challenge of the verifier. Sending out one large message is always more efficient than sending out k single bit messages because of the communication overhead (which will always be present, even if dedicated hardware is used). This distance bounding protocol has also another nice feature: the “proof of distance” can afterwards be shown to other parties, on the condition that the verifier is trusted. Because of all these properties, this protocol is probably the best one to choose for mobile ad-hoc environments. More details can be found in [4].

V. LOCATION VERIFICATION

In the previous section, we described three secure distance bounding protocols which are resistant to distance fraud attacks, mafia fraud attacks and terrorist fraud attacks. It is very easy to extend such a distance bounding protocol with 1 verifier to a protocol with more verifiers. These “extended protocols” can be used as a building block for location verification schemes. Using some special properties, these schemes can be made very efficient.

A. Basic building block

Let’s recall the concept of distance bounding with 1 verifier. After measuring the round trip time of the electromagnetic

signals, the verifier determines an upper bound d on the distance to the prover. The prover can be closer, but can not be further away. To know the exact location of the prover in a plane (we leave the 3-dimensional case as an exercise for the reader), we need 3 collaborating verifiers. The prover performs a distance bounding protocol with each of the 3 verifiers. By combining the results of these 3 distance bounding protocols, the exact location of the prover can be found by performing some basic triangulation. In case the prover does not delay his responses, one has the situation as shown in Fig. 8. In case there is a delay (deliberate or not), one gets an area instead of a point (the radii of the circles increase). The larger the area, the less one knows about the location of the prover. It is however unclear how a dishonest prover can take advantage of this uncertainty. In a common scenario, an attacker wants to pretend to be somewhere else. The verifiers will not accept a large area and as a consequence, reject the location claim of the attacker. Fortunately, one can use the geometrical properties of broadcast mode (which will be discussed in the next subsection) to avoid this uncertainty, even in the case delay is added.

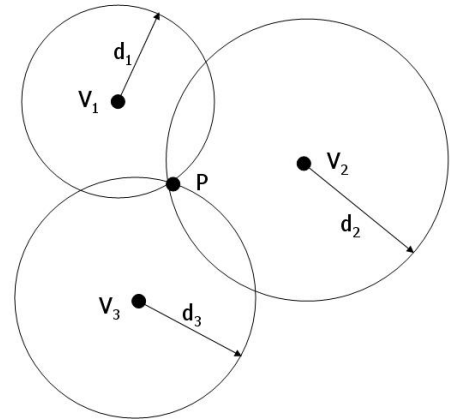


Fig. 8. Distance bounding protocol with 3 verifiers

B. Broadcast mode

If there are no restrictions on the location of the prover, then distance bounding with 3 verifiers is not secure enough. An attacker can pretend to be at every location that is further away from all 3 verifiers. On Fig. 8, this corresponds with the area outside the 3 circles. By carefully selecting the delay in every distance bounding protocol, the verifier will believe that the prover is at a certain point, different from the real location of the prover. This attack is possible because the 3 distance bounding protocols are executed independently from each other. The solution is *broadcast mode*. In this mode, only 1 distance bounding protocol is executed with the 3 verifiers simultaneously. 1 verifier sends a challenge and the prover broadcasts the response. The 3 verifiers will receive the response at a different moment of time. When a dishonest prover would insert a delay, this delay would be equal for all 3 the verifiers. This property assures that delaying the response

will always result in an area, and never in a point. This area has some nice geometrical properties which can be used to reduce it to a point.

Suppose one would construct a connection line between the position of one of the verifiers and the prover. If the distance between both points is r , then the distance between the position of the prover and the contour of the area (following the prolongation of this connection line) would be d . This geometrical property is valid for all 3 the contours (and the corresponding verifiers) and is partially shown in Fig. 9. The shaded area is the result of the prover delaying the response and the point in the center of this area is the exact position of the prover. The length of the 3 arrows is d . This geometrical property can be expressed analytically and yields a multinomial of degree 4. By solving this equation, one knows the delay d and hence the exact position of the prover. Note that broadcast mode is only possible when the 3 verifiers are well synchronized (all verifiers have to know exactly when the challenge was sent to the prover).

There is however one situation in which a prover can delay messages and still enforce the result of the triangulation to be one point. Suppose the 3 verifiers are on a hyperbola and the prover is located exactly on one of the foci of the hyperbola. The distance between the prover and the 3 verifiers is respectively r_1 , r_2 and r_3 . Then the distance between the other focus of the hyperbola and the 3 verifiers is $r_1 + d$, $r_2 + d$ and $r_3 + d$. So the only thing the prover has to do to perform this “hyperbola-attack” is carefully selecting the delay to correspond to this distance d (which is a specific distance for every hyperbola). To avoid this attack, one should only allow the prover to be inside the triangle $\{V_1, V_2, V_3\}$.

C. Practical recommendations

First, use one of the three secure distance bounding protocols described in this paper as a building block for the location verification scheme. All the problems of the previous sections can be solved by placing restrictions on the location of the prover. By delaying the messages, the attacker can only pretend to be somewhere outside the triangle $\{V_1, V_2, V_3\}$. So if we only allow locations inside this triangle, then our location verification scheme is secure against a prover that

delays responses. If it is possible to synchronize the 3 verifiers, then broadcast mode is certainly recommended. This because of the geometrical properties explained in the previous section. Using this property allows to recover the exact position and learn more about the delay of the responses. The larger the area, the more the message was delayed. However, such a delay does not always mean that the prover is cheating. It can also be the result of a large processing delay. This implies that the restrictions on the processing delay can be relaxed when using broadcast mode! This is very important from a practical point of view! To make the scheme more robust, one could use more than 3 verifiers, although this is not entirely necessary from security point of view.

VI. CONCLUSION

In this paper, we gave an overview of the different techniques that can be used to design a distance bounding protocol. The most promising solution is measuring the time of flight of electromagnetic signals during a challenge-response protocol. There are however some practical problems which limit the use of such protocols. There are three major attack scenarios: distance fraud attacks, mafia fraud attacks and terrorist fraud attacks. Some distance bounding protocols are resistant to these attacks. However, they need trusted hardware or expensive modular exponentiations to accomplish this. It is easy to extend such a secure distance bounding protocol with 1 verifier to a protocol with 3 synchronized verifiers. These “extended protocols” can be used as a building block for location verification schemes. For security and efficiency reasons, it is recommended to place some restrictions on the location of the prover and to use broadcast mode. The exact location of the prover can be found, even if signals are delayed by an attacker. Using broadcast mode also relaxes the restrictions on the processing delay.

ACKNOWLEDGMENT

Dave Singelee is funded by a research grant of the Institute for the Promotion of Innovation by Science and Technology in Flanders (IWT). This work was also supported by the Concerted Research Action (GOA) Ambiorics 2005/11 of the Flemish Government.

REFERENCES

- [1] L. Bussard, “Trust Establishment Protocols for Communicating Devices,” Ph.D. dissertation, ENST Paris, October 2004, 233 pages.
- [2] J. Hightower and G. Borriello, “Location systems for ubiquitous computing,” *IEEE Computer*, vol. 34, no. 8, pp. 57–66, August 2001.
- [3] D. Denning and P. MacDoran, “Location-Based Authentication: Grounding Cyberspace for better Security,” in *Computer Fraud and Security*. Elsevier, 1996.
- [4] B. Waters and E. Felten, “Proving the Location of Tamper-Resistant Devices,” <http://www.cs.princeton.edu/bwaters/research/location-proving.ps>, 2003.
- [5] Bluetooth Specification, <https://www.bluetooth.org/spec/>.
- [6] DEF CON, “Computer Underground Hackers Convention,” <http://www.defcon.org>.
- [7] P. Bahl and V. Padmanabhan, “RADAR: An In-Building RF-based User Location and Tracking System,” in *Proceedings of IEEE INFOCOM '00*, vol. 2, 2000, pp. 775–784.

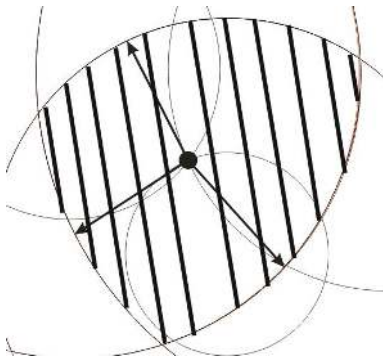


Fig. 9. Geometrical properties of broadcast mode

- [8] S. Brands and D. Chaum, "Distance-Bounding Protocols," in *Advances in Cryptology - EUROCRYPT '93*, ser. Lecture Notes in Computer Science, LNCS 765. Springer-Verlag, 1994, pp. 344–359.
- [9] N. Sastry, U. Shankar, and D. Wagner, "Secure Verification of Location Claims," www.cs.berkeley.edu/~nks/locprove/csd-03-1245.pdf, 2003.
- [10] T. Kindberg and K. Zhang, "Validating and Securing Spontaneous Associations between Wireless Devices," in *Proceedings of the 6th Information Security Conference (ISC03)*, ser. Lecture Notes in Computer Science, LNCS 2851. Springer-Verlag, 2003, pp. 44–53.
- [11] Y. Desmedt, "Major Security Problems with the "Unforgeable" (Feige)–Fiat–Shamir Proofs of Identity and how to overcome them," in *Proceedings of SecuriCom '88*, 1988, pp. 15–17.
- [12] S. Čapkun, L. Buttyán, and J. Hubaux, "SECTOR: Secure Tracking of Node Encounters in Multi-hop Wireless Networks," in *Proceedings of the 1st ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN '03)*, 2003, pp. 21–32.
- [13] B. Waters and E. Felten, "Secure, private proofs of location," Princeton Technical Report TR-667-03, 2003.
- [14] M. Wiener, "Performance Comparison of Public-Key Cryptosystems," in *RSA Laboratories Cryptobytes*, vol. 4(1), 1998, pp. 1–5.
- [15] A. Hodjat and I. Verbauwhede, "The Energy Cost of Secrets in Ad-Hoc Networks," in *Proceedings of the IEEE CAS Workshop on Wireless Communications and Networking*, 2002.