Chapter 5

SECURE LOCALISATION SYSTEMS

Most of the proposed localisation algorithms studied the problem of location discovery in a non-adversarial environment. Although these types of algorithm are vulnerable to several types of security attack, less work has been done to implement secure localisation algorithms that are able to work in a hostile environment. The very nature of WSNs enables the attackers to abuse the localisation systems. Malicious nodes could claim fake locations that are not where they are physically located. An attacker could compromise, or masquerade as, a beacon node and send incorrect location information. Localisation in a hostile environment is a critical problem in WSNs because compromising the localisation system could disturb and subvert the entire functioning of the WSN.

Localisation systems consist of three major components: distance estimation, position computation and localisation algorithm. These components are strongly connected. Compromising one of them could affect the entire localisation system. However, securing the first two components greatly enhances the security of the third component and the entire localisation system.

Several techniques can be used to implement secure distance-estimation components, such as directional antennas, RF fingerprinting, centralised systems and distance bounding. A distance-bounding approach has several characteristics that make it very suitable to be used in WSNs. Distance bounding can function within an ad-hoc, mobile environment with any number of nodes and different types of network topology. Distance bounding does not consume valuable resources and it can be executed in minimal time.

Distance bounding involves only two nodes, to estimate the distance between them, which makes it a good choice for the ALWadHA algorithm. ALWadHA, as has been shown, uses a

subset of a few references compared with other secure localisation algorithms. Therefore, ALWadHA can use the distance-bounding method to estimate the distance to the references within the selected subset without involving other references. This can also reduce the overhead that could be caused by estimating the distance to all available references. Using a distance-bounding approach will assist ALWadHA to accomplish several design objectives, such as self-organising properties, simplicity, energy efficiency, localisation and security.

Distance bounding can be used to implement secure distance-estimation components. However, this is not enough to secure the entire localisation system, and more techniques should be used to secure the other components. Therefore, ALWadHA relies not only on using a secure distance-bounding protocol, but also uses a robust position-computation component that tolerates the existence of malicious nodes.

5.1 THE SECURITY OF LOCALISATION SYSTEMS

Localisation systems play a key role in WSNs, because in addition to locating events they could be used as the base for routing, density control, tracking and other services and protocols. This role makes the localisation system a target of attackers. An attacker that compromises the localisation system could disturb the entire functioning of WSN and lead to incorrect plans and decisions. Therefore, as stated above, WSNs require a secure localisation algorithm, which not only solves the problem of localisation, but should also be resistant to the presence of fraudulent, malicious and/or compromised nodes.

5.1.1 Attacks on localisation systems

As mentioned in Section 2.2.1, localisation systems consist of three main components: distance/angle estimation, position estimation and localisation algorithm. These components are strongly dependent on one another. Malfunctioning in any of these components could affect the entire localisation system, and lead to incorrect estimation. Because of the strong relationship between them, any of these components can be targeted by an attack on a localisation system, making these systems very fragile and hard to secure [18].

5.1.1.1 Attacks on distance/angle estimation

Different approaches can be used for distance/angle estimation, such as directional antennas, RF fingerprinting (communication neighbour authentication), connectivity (in range) and distance bounding. Practically, these approaches use several techniques, including RSS, ToA, TDoA, AoA, RTT and hop-count based ones.

RSS is not a secure method, since attackers can easily alter transmission power by either amplifying or attenuating a signal. Changing the transmission power makes the neighbour nodes think the compromised node is nearer or farther away than it really is. In the ToA and TDoA techniques, an attacker can delay the transmission time to pretend that it is further away. On the other hand, there are no measures in place to prevent the compromised node from lying about the sending time and appearing closer to the neighbour nodes as a result. Secure AoA technique requires that the nodes must have synchronised clocks to ensure that a transmission was made to multiple references at the same time. However, time synchronisation to the accuracy required for distance estimation is a challenge in wireless networks. In the hop-count-based technique, the node first estimates the average distance to neighbouring nodes, then it uses this average distance to estimate the distance for other nodes by multiplying this average distance by the number of hops to these nodes. A compromised node can deliberately change the computed hop count to mislead other nodes about its real distance.

In a compromised environment, both RSS and ToA techniques can be targeted by changing the physical medium, for example by introducing noise, obstacles or smoke. Also, the AoA technique can be compromised by deploying magnets in the sensor field [18].

The next two components of localisation systems rely on accurate distance estimates, so if the system is to be secured, the designer must consider the possible vulnerabilities of these techniques that could be exploited by an attacker. Sections 5.2 to 5.6 investigate the security issues of these different approaches and techniques and show that distance-bounding approaches that are based on RTT and follow the four principles proposed by Clulow *et al.* [89], could achieve secure distance estimation between two nodes.

5.1.1.2 Attack on position computation

In addition to the distance estimations of at least three references, the node also needs to know the location of these references in order to estimate its position. An attack on the distance estimation can indirectly affect the position computation. However, some attacks can affect this component directly by advertising incorrect information about the compromised node's location. As illustrated in Figure 5.1, an attacker may provide incorrect location information by either pretending to be an honest reference node (Figure 5.1.a) or compromising beacon nodes (Figure 5.1.b). In both types of attack, unknown nodes which received the incorrect location information will determine their locations incorrectly.

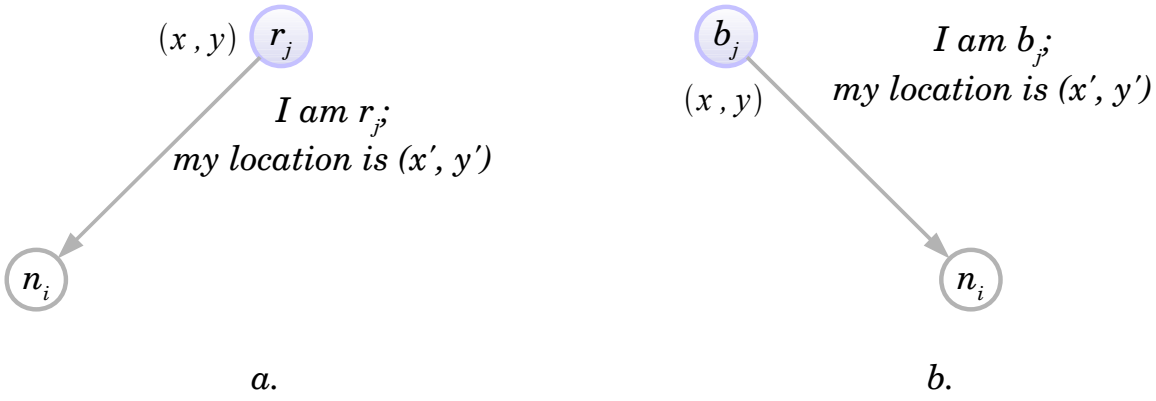

Figure 5.1. Attacks against position computation. a. Dishonest reference node, b. Compromised beacon node

The second type of attack is more difficult to achieve than the first one, since an attacker would have to be able to compromise a beacon node first to perform the second attack, while in the first type an attacker can directly advertise the incorrect location information, pretending to be an honest reference node. Several localisation algorithms, for example Nearest algorithm, do not distinguish between the reference and the beacon nodes. Therefore, both of these attacks are equivalent for them, and so an attacker can simply perform the first attack. On the other hand, other localisation algorithms distinguish between these two types of node. For example, the ALWadHA algorithm assigns the highest probability of accuracy to beacon nodes to enable the unknown nodes to select them first. The NDBL algorithm uses a different weight for beacon nodes. In addition to

the other benefits of this distinction, such as achieving better accuracy, an attacker has to perform the second type of attack in order to increase the erroneousness of position computation.

5.1.2 Secure localisation algorithms

Secure localisation algorithms use several techniques to deal with the security problems of location discovery in WSNs. The main techniques used are cryptography, detecting and blocking compromised nodes, making statistical decisions or filtering the positions used in computations [18].

5.1.2.1 Cryptography

Cryptography can be used to achieve several security requirements of localisation systems, including firstly *authentication*, where nodes accept information only from authorised references, secondly *message integrity*, thirdly the *availability of the required information* to estimate a node's position, fourthly *non-repudiation*, which can be used to make sure that the references are unable to deny the location information sent by them, and fifthly, *confidentiality*, to prevent malicious nodes from collecting nodes' location information and from claiming a different legitimate location in the network.

Cryptography does not solve all the security problems of a localisation system. An attacker may disclose the key of the compromised node and forge a “location response” packet. An attacker could also replay “location response” packets intercepted in a different location. Moreover, cryptography could require extensive resources that may not be available in the resource-constrained WSNs. Therefore, additional security techniques should be used to protect localisation systems. Several localisation algorithms use non-cryptographic security techniques and rely on cryptography as a second line of defence, such as the algorithms proposed in [56, 90-94].

5.1.2.2 Detection of malicious nodes

The second technique for securing localisation systems is to detect and block malicious nodes. In this technique the node tries to detect malicious nodes and then does not use their

location information to compute its position. Several methods are used to detect malicious nodes. For example, Liu *et al.* [56] compare the measured distance (using RSS, ToA, AoA, etc.) and the distance estimated by using the advertised location information. A great difference between these two distances indicates that the location information could be sent by malicious nodes. Another method is to report the location information to a central authority, which manipulates this information and filters out malicious nodes accordingly [56]. A distributed method, proposed by Srinivasan *et al.* [95, 96], enables beacons to monitor their neighbour nodes and exchange information to allow other nodes to choose trustworthy beacons based on a quorum voting approach.

Zhong *et al.* [97] propose two localisation algorithms and prove that when the number of malicious beacons is less than a specific threshold ((*n* - 3) / 2), where *n* is the total number of nodes, the localisation error is bounded, assuming that the measurement error is ideally small. In [98] Zeng *et al.* introduce a powerful attack, called "Pollution attack", that can succeed and seriously distort the location estimation even when the number of malicious beacons is less than the lower bound and the measurement error is practically small.

5.1.2.3 Robust position computation

Implementing a robust position computation component is another technique that can be used to deal with malicious nodes. This technique enables the nodes to estimate their position with acceptable accuracy even in the presence of malicious nodes. This technique is based on using statistical and outlier filtering methods to deal with malicious nodes. Li *et al.* [73] propose the use of LMS as an improvement over the LS method for achieving robustness to attacks. LMS tolerates up to 50% outliers and still provides correct estimation. However, the original LS method outperforms LMS in the absence of attacks.

The authors of [41] investigate two types of attack-resistant location estimation techniques to tolerate the malicious attacks against range-based location discovery in WSNs. In the first technique the unknown nodes defeat malicious attacks by checking the consistency of references and then removing the inconsistent malicious references. This technique starts by using the entire set of references and then it gradually removes the most suspicious references till it gains a certain level of consistency, which depends on the measurement error of an estimated location. The authors develop an incremental MMSE approach to

reduce the computation cost, but it increases the size of the required memory. The second technique is called voting-based location estimation, which quantises the deployment field into a grid of cells, and then the unknown node determines how likely it is in each cell based on each reference. After the unknown node has processed all references, it chooses the cell(s) with the highest vote and uses its (their) geometric centroid as the estimated location of the sensor node. However, specifying the voting by each reference at each cell of the grid requires a high computation cost.

Misra *et al.* [99] propose a convex optimisation-based localisation scheme that can accurately localise a node, as long as the number of neighbouring malicious beacons is lower than a critical threshold value. This scheme computes the geometric centre of the intersection of circles corresponding to location references. Since it uses a distance-bounding protocol, it assumes that the attackers can only enlarge the distance measurements. This scheme is also able to identify a significant number of malicious nodes.

5.1.2.4 Location verification

In fact, this technique can be used only to verify the location estimation results from the overall localisation system, thus it can enhance the security level of the three localisation system components. Du *et al.* [100] propose several ways to verify the estimated location. Their techniques are independent of the localisation system and they can be performed after estimating the location. The authors use deployment knowledge, with a group-based deployment model, to compare the actual observation of the deployment knowledge and the observation obtained from the estimated location. If the two observations deviate substantially from each other, the node knows that the estimated position is inconsistent with the real location and it can be rejected.

In [101] Hwang *et al.* propose a secure localisation algorithm that allows all nodes to play the role of verifier to detect the existence of malicious nodes. Each node generates a local map, checks whether its measured ranges are consistent with its ranges in the map and then finds the largest consistent subset, which could contain all the honest nodes.

Wei *et al.* [102] propose two location verification algorithms; greedy filtering by matrix

(GFM) and trustability indicator (TI). In GFM, the verification centre identifies inconsistent locations using several matrices based on neighbourhood observations and sensors' location estimations. In TI, the verification centre calculates indicators for the sensors. The sensor's location will be accepted if its final indicator is greater than a threshold.

In [103] Ekici *et al.* propose a Probabilistic Location Verification algorithm to verify the nodes' location with a small number of trusted verifiers. The verifiers use the number of hops and Euclidean distance to other nodes to determine the plausibility of the claimed location and then create an arbitrary number of trust levels in the location claimed. Finally, a central node collects these two values (i.e. the plausibility and the trust levels) from all the verifiers to decide whether it will accept or reject the claimed location.

5.2 SECURE DISTANCE ESTIMATION

Distance estimation is the first component of localisation systems, and is responsible for determining the physical distance between nodes. In hostile environments this distance estimation process must be executed using secure protocols to maintain the integrity of services that rely on this ‘next-hop’ information. There are a number of possible approaches to implementing secure distance estimation: directional antennas, RF fingerprinting, centralised systems, location-based and distance-bounding approaches [104].

Directional antennas triangulate position using AoA, which requires static references with antenna arrays providing fixed reference directions, e.g. [44]. To be secure, these nodes must also have synchronised clocks to ensure that a transmission was made to multiple references at the same time, i.e. from the same location. But time synchronisation to the accuracy required for distance estimation is a challenge in wireless networks. To determine the distance to another node would also require that node to be covered by at least two references. This limits the topology and the connection structures to a network ‘cloud’, where all nodes are covered by multiple references. For example, AoA does not provide secure distance estimation in point-to-point connections, where only two nodes are communicating with each other.

Centralised approaches work on the assumption that there are many nodes that can collaborate and aggregate data to a central system controller, which can check for suspicious node activity (for example, if two nodes receive the same transmission and it is impossible for that transmitting node to be within the communication range of both, e.g. [46, 73]). The effectiveness of these approaches relies on the network consisting of a large number of nodes capable of providing simple ancillary measurements. However, this approach requires that all data should be transmitted to a centralised authority that would be able to search connectivity graphs for wormholes or orchestrate network-wide node localisation. Such a high volume of communications might not be practical for a resource-constrained WSN.

RF fingerprinting characterises the physical communication channel aspects of a node to generate a unique identifier that is difficult to forge, e.g. [45]. This is a promising approach to identifying nodes uniquely. However, each device needs to be characterised, and if nodes are mobile and subject to varying multi-path and path-loss effects, the fingerprint might need to be revised often. This reduces the practicality of using fingerprinting in WSNs if the topology is bound to change quickly and if ad-hoc connections are often made to new nodes entering the network's area of coverage.

Location-based methods require the physical location of each node to be known at node level; for example, each node is equipped with a GPS unit, or at system level using a localisation scheme. Examples of such systems are described in [18]. Determining the location at node level requires additional network infrastructure and resources, especially indoors, where GPS is not as effective, and a system-wide localisation scheme has the same disadvantages as previously discussed for centralised approaches. Localisation schemes still rely on accurate neighbour detection, and several secure localisation proposals use distance-bounding protocols for the underlying distance estimation between nodes [29].

Distance bounding is a secure neighbour detection method that determines an upper bound for the physical distance between two communicating parties based on the RTT of cryptographic challenge-response pairs. Brands and Chaum proposed the first true distance-bounding protocol [105], and several new protocols have been proposed since. Most distance-bounding protocols have been proposed for providing security services in

radio frequency identification (RFID), e.g. [106, 107], and WSN environments, e.g. [47, 108]. In the RFID environment distance bounding can be used to prove the proximity of a RFID token cryptographically to a reader, while in WSNs its ability to verify the physical proximity of "neighbour nodes" makes it a key building block in secure routing and localisation methods.

5.2.1 Distance bounding as a possible solution for ALWadHA

Components of a sensor network should ideally be designed to be as autonomous as possible, which means that a secure distance-estimation method that can be performed locally by any node, with minimal collaboration with other network entities, is preferable. Distance bounding requires a node that can make round-trip time measurements, that is, it requires a suitable RF channel and an accurate, albeit unsynchronised, clock, but can function within an ad-hoc, mobile environment with any number of nodes and network topology. Moreover, distance-bounding protocols can be executed in minimal time, while not consuming valuable resources that could be allocated to the main network services.

These characteristics make distance bounding a good approach for secure distance estimation. Using a distance-bounding method will assist ALWadHA to accomplish several design objectives such as self-organising properties, simplicity, energy efficiency, localised algorithm and security. Unlike other localisation algorithms, ALWadHA does not select a subset of references based on the estimated distance to the available references; rather it initially selects this subset of references based on their probability of accuracy. Each reference node is able to determine its probability of accuracy without relying on collaborating information from a centralised controller or other neighbours. After selecting a subset of references, the node estimates the distance only to the selected subset of references. Therefore, the use of a distance-bounding approach by ALWadHA will not add an overhead that could be caused by estimating the distance to all available references. Estimating the distance between two nodes using distance bounding involves only these two nodes. Therefore, ALWadHA can use a distance-bounding method to estimate the distance to the references within the selected subset without involving other references.

5.3 DISTANCE-BOUNDING PROTOCOLS

Distance bounding involves only two parties: a *prover* and a *verifier*, and provides the verifier with cryptographic proof of the maximum physical distance to the prover. The verifier relies exclusively on information gained from executing the protocol with the prover. As the verifier requires a reliable and secure estimate of the distance to the prover, distance-bounding protocols should be integrated into the underlying communication channel. The security of the protocol therefore depends not only on the cryptographic mechanisms, but also on how the physical attributes of the communication channel are used to measure proximity.

Time-of-flight (ToF) and RSS are often used for distance estimation between two nodes. RSS is, however, not a secure method since attackers can easily alter RSS, by either amplifying or attenuating a signal. ToF measures elapsed time for a message exchange and then estimates the distance based on the communication medium's propagation speed. Both RF and ultrasound channels (US) have been used in ToF systems. The propagation speed of sound is much slower than that of radio waves. As a result, an attacker can intercept the US communication and forward it over a faster communication medium to an accomplice closer to the verifier, or prover, thus reducing the time measurement and decreasing the distance estimate. RF channels resist this simple relay attack and are often proposed as the channel of choice for implementing distance-bounding systems. WSNs are potentially suited to implement RF ToF distance bounding. High bandwidth communication channels proposed for use in WSNs, such as ultra-wideband (UWB), are also suitable for fine resolution time-of-flight distance estimation.

There are two well-known examples of how ToF is generally used to determine distance: ToA and RTT. In ToA systems the estimate is made purely on data transmitted in one direction, from the prover to the verifier.

The distance between two devices can be calculated as

$$d = c.t_p \tag{5.1}$$

where c is the propagation speed and t_p is the one-way propagation time between the transmitter and the receiver. A distance-bounding protocol using this approach could

possibly be implemented as described below if both the verifier and the prover share a common, high-precision time base, such as secure GPS receivers. The verifier sends out a challenge α at time t_0, which the prover receives at time $t_0 + t_p$. The prover then replies with a message-authenticated data packet $(t_0 + t_p, \alpha)$, where α is the challenge he received. The verifier checks the message-authentication code of this packet with the shared key and also whether $t_p \leqslant d/c$, where d is the upper bound for the distance and c is the speed of light. An attacker relaying the challenge would introduce extra propagation delay, which would be reflected in t_p, and as a result the verifier would conclude that the prover is outside the allowable distance bound. There are, however, some problems with this implementation. An accurate time base shared between devices is difficult to achieve, so this method is not practically feasible, especially in wireless network nodes. This protocol also assumes that the prover is a trusted entity, thus making the prover responsible for taking the time measurement. There are no measures in place to prevent the prover from lying about $t_0 + t_p$ and appearing closer to the verifier as a result. From a security perspective this is not acceptable, as a distance-bounding protocol should provide the verifier with reliable proof that the prover is actually within a certain distance, even in cases where the prover is not trusted.

The problem of maintaining a shared time base in both the verifier and the prover could be addressed by making distance measurements based on RTT. In RTT systems, only the verifier is required to maintain an accurate time base and the prover is also no longer responsible for providing the security-critical time measurements. The distance between the prover and verifier can be calculated as

$$d = c \,.\, (t_m - t_d)/2 \tag{5.2}$$

$$t_m = 2 \,.\, t_p + t_d \tag{5.3}$$

where c is the propagation speed, t_p is the one-way propagation time, t_m is the measured total round-trip time and t_d is the prover's processing delay between receiving a challenge and sending a response.

A distance-bounding protocol using RTT could possibly be implemented as follows: The verifier again sends a challenge α at time t_0, which the prover receives at time $t_0 + t_p$. Once the challenge has been received the prover sends a response β in the opposite

direction at time $t_1 = t_0 + t_p + t_d$, which the verifier receives at time $t_2 = t_0 + 2 . t_p + t_d$. The verifier determines $t_m = t_2 - t_0$, checks that the response is correct and finally confirms that $t_p \leqslant d/c$, where d is once again the upper bound for the distance. Although this protocol is an improvement on the first example, extra care must be taken to specify the nature of the response. If the prover simply echoed the challenge received α' it would provide limited security, as any attacker who controlled a proxy token close to the verifier would be able to do this and achieve an acceptable measurement $t_p \leqslant d/c$. Ideally, the response should depend on the challenge, i.e. $\beta = f(\alpha)$, as this is also an effective measure against fraudulent provers. If this was not the case, and β was merely a random nonce agreed on beforehand between the prover and verifier, the prover could send β before receiving α and effectively decrease the round-trip time. Specifying that β is a function of α forces the prover to wait until he receives the challenge and prevents him from pre-emptively transmitting his response.

5.3.1 Distance-bounding attacks

Distance-bounding protocols should be resistant to attacks from a fraudulent prover and a malicious third party. There are three types of attack that can be prevented by distance-bounding protocols, namely distance fraud, mafia fraud and terrorist fraud.

5.3.1.1 Distance fraud

A fraudulent prover wants to convince the verifier that its physical distance is closer to the verifier than is actually the case, as shown in Figure 5.2.a. Distance-bounding protocols prevent distance fraud if the prover is not in a position to transmit its answer pre-emptively. In other words, the response is made dependent on the challenge. A possible scenario of distance fraud in WSNs is if a node that is outside the allowable network coverage area tries to initiate a connection by pretending to be within the allowable area.

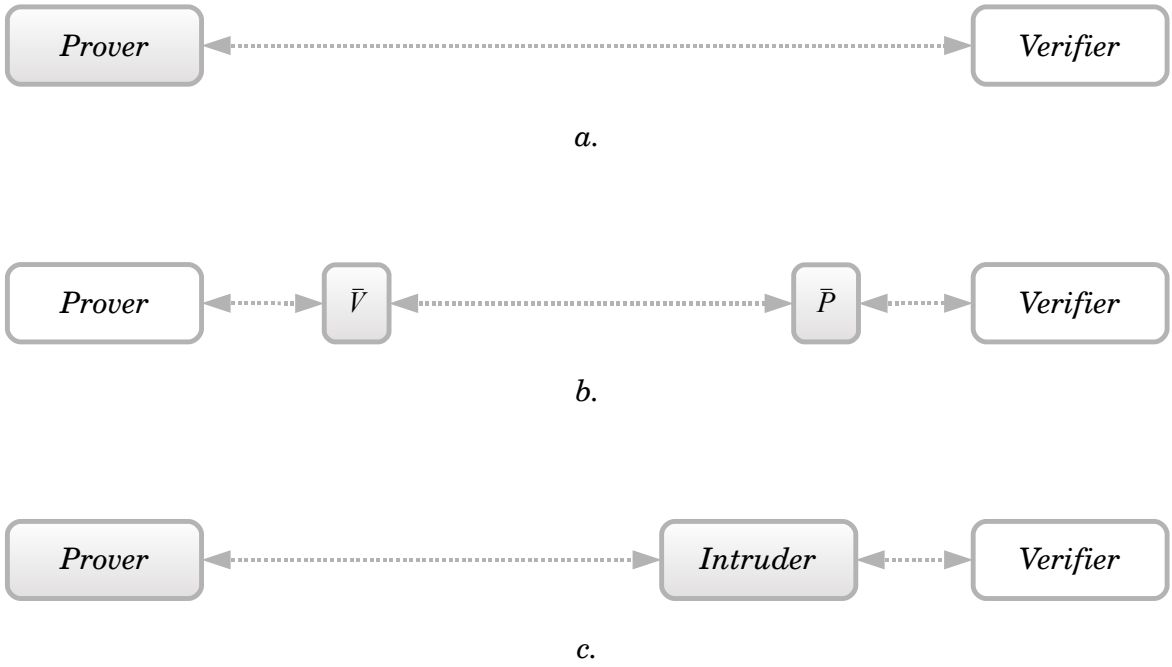

Figure 5.2. Main attacks that distance-bounding protocols aim to prevent. White boxes are the honest parties, while the grey ones are the dishonest parties. a. Distance fraud, b. Mafia fraud, c. Terrorist fraud

5.3.1.2 Mafia fraud

In this fraud, the prover (P) and the verifier (V) are honest, while a third party attacker attempts to carry out the attack. As depicted in Figure 5.2.b, the attacker consists of a dishonest prover and dishonest verifier $\{\bar{P}, \bar{V}\}$ interacting with the honest verifier and prover respectively. This fraud enables the attacker to convince V that P is in close proximity, and does not require the attacker to circumvent the application level security. The attacker does not need to know any key material; in fact he does not even need to know the content of the communication he relays. Mafia fraud was first described by Desmedt [109] but similar attacks in WSN and RFID environments have also been described as wormhole or relay attacks respectively. Mafia fraud is detected by distance-bounding protocols, as the attacker introduces additional delay into the RTT measurement, for example, even if theoretically the attacker's hardware introduces no delay, the time taken for the relayed data to propagate the extra distance will add to the RTT. The most likely scenario of a mafia-type fraud in WSNs is a wormhole attack where two nodes forward data and routing packets between remote areas of the network. This artificially

short routing path changes the true topology of the network, which potentially places the attacker in a position to carry out further attacks on the network and the data transmitted over this path.

5.3.1.3 Terrorist fraud

A third-party attacker and a dishonest prover collaborate to perform terrorist fraud, as shown in Figure 5.2.c. The fraudulent prover, who is far from the honest verifier, assists an attacker to convince the verifier that the prover is in close proximity. Terrorist fraud was first described in [109]. The prover ideally does not reveal his secret information to his accomplice, who will be able to impersonate the prover if he obtains this secret information. In order to prevent terrorist fraud, some distance-bounding protocols therefore ensure that if the response is revealed in advance, the accomplice will learn the secret key of the prover.

5.3.2 Types of distance-bounding protocol

Since Brands and Chaum first described distance-bounding protocols in 1993 [105] several protocols have been published that allow a verifier to determine an upper bound on the physical distance to a specific prover. Most distance-bounding protocols have been proposed for WSN and RFID environments. These proposals can be classified by how they implement different stages of the distance-bounding process. Most of these distance-bounding protocols consist of three basic stages: the *setup* stage, where the verifier and prover prepare for the *exchange* stage, the timed exchange of challenge and response data, and the *verification* stage that ensures that the exchange step has been executed faithfully. In the literature there are three different types of distance-bounding proposals [89].

Timed authentication protocols are the simplest form of ToF-based distance bounding, with the verifier timing normal, authenticated data exchanges. The basic idea is to execute a challenge-response authentication protocol under a very tight time-out constraint, which was a concept first proposed by Desmedt [109]. For example, a verifier V transmits a random n-bit nonce $N_V \in_R \{0, 1\}^n$ to the prover P, who replies with a message-authentication code $h(S, N_V)$, where h is a keyed pseudo-random function and S is a

shared secret key. Numerous protocols have been proposed using different constructions for pseudo-random functions keyed with shared secrets, public-key mechanisms or trusted third parties. Examples of such protocols are the secure neighbour detection protocol proposed by Hu *et al.* [110], and the protocol proposed by Tang *et al.* [111]. This set of protocols generally time an exchange without considering variations in the processing time of the response or the format of the challenge and response. The possible variations in the processing delay t_d affect the time measurement, which in turn causes the distance bound to be unreliable. For example, a node that usually takes 100 ms to compute a public-key signature and has a 1% (1 ms) processing time variation could cause a 333 km error in the distance estimate. Furthermore, a fraudulent prover has scope to speed up the processing time and transmit his response earlier than expected.

To reduce, or accurately predict the processing delay t_d, some protocols suggest that the prover determines the possible responses before the exchange stage. This is usually accomplished by using *pre-commitment* or *pre-computation*. Now the prover only has to choose a response β based on the challenge α received from the verifier during the exchange stage, so t_d is significantly reduced. These protocols generally propose that the function $\alpha \rightarrow \beta$ be implemented using a simple XOR or table look-up operations to minimise variation in t_d. In protocols using pre-commitment, the prover prepares possible responses during the setup stage. For example, the verifier generates a random challenge bit string, $\alpha = (\alpha_1, \alpha_2, \cdots, \alpha_l)$, while the prover generates a response string, $R = (R_1, R_2, \cdots, R_l)$. The prover commits to R, for example by transmitting a collision-resistant message authentication code $h(S, R)$. The verifier then sends one α_i after another, which the prover receives as α_i'. It then instantly replies with a bit $\beta_i = \alpha_i' \oplus R_i$, which is calculated by XORing each received challenge bit with the corresponding bit of R. Finally the prover reveals R and authenticates α', i.e. the prover sends $MAC(\alpha')_S$ to the verifier.

In protocols using pre-computation, the prover and the verifier calculate the possible response strings before the exchange stage starts. For example, the verifier and the prover first exchange nonces N_V and N_P. Both the prover and the verifier then use a pseudo-random function F and a shared key S in order to calculate two n-bit response strings R^0 and R^1:

$$(R_1^0, R_2^0, R_3^0, \cdots, R_l^0, R_1^1, R_2^1, R_3^1, \cdots, , R_l^1) := F_S(N_V, N_P) \quad . \tag{5.4}$$

Since the verifier also knows R^0 and R^1 at this stage, the prover is effectively committed to two response strings, without explicitly making a commitment during setup. As a result the prover does not have to open his commitment during the verification stage, thus decreasing the data that need to be transmitted.

Although most protocols using pre-commitment or pre-computation propose the exchange of single-bit challenges and responses, there are some protocols which use a single multi-bit exchange instead, for example Third Party Location Proofs by Waters and Felten [112], the multi-bit exchange variant of Brands and Chaum proposed in [29] and the protocol by Nikov and Vauclair [113]. Communication channels usually transmit multi-bit data packets rather than single bits, so exchanging a multi-bit bit stream is easier to implement with off-the-shelf components. The problem is that these channels usually add some extra formatting information, such as trailers, headers and error-correction measures that introduce latency. This could be as simple as adding parity, start and stop bits to the data sent. An attacker may not be restricted by regular implementations and could reduce latency introduced by the communication layers to commit distance fraud or hide mafia fraud [89]. For example, an attacker could ignore packet framing and start to calculate his response before receiving the packet trailer information, and as a result be in a position to respond earlier than expected. A verifier expecting the prover to adhere to the communication channel 'rules' would therefore measure a reduced round-trip time. Even if a nonce was exchanged without any formatting, or error correction, an attacker could still gain a timing advantage. As shown in [114], an attacker can gain a timing advantage from tolerances introduced in the modulation and encoding stages, while also having the option of decreasing the decoding process of a legitimate device if the data recovery clock is derived from the transmitted data. The exchange of multi-bit data packets over normal communication layers therefore does not achieve the secure timing measurement required. In contrast, a multiple single-bit exchange provides the highest time resolution, as it depends only on the propagation time, pulse width and processing delay of a single bit. Multiple-timed bit exchanges may appear inefficient, but multiple measurements increase accuracy and confidence. An added performance benefit of transmitting single bits is that it reduces the communication overhead. This decreases the protocol execution time,

especially in resource-constrained environments, such as RFID, where the capacity of the communication channel is limited.

5.3.3 Principles of secure distance bounding

To protect against attacks at the physical and packet layer of the communication channel, Clulow *et al.* [89] propose that the designer of a distance-bounding protocol should optimise the choice of communication medium and transmission format according to the following principles:

- **Principle 1:** Use a communication medium with a propagation speed which approaches the physical limit for propagating information through space-time.
- **Principle 2:** Use a communication format in which the recipient can instantly react to the reception of each individual bit. This excludes most traditional byte or block-based communication formats, and in particular any form of redundancy such as error-correction and packet delimiters such as headers and trailers.
- **Principle 3:** Minimise the length of the symbol used to represent each single bit.
- **Principle 4:** The distance-bounding protocol should be designed to cope with bit errors during the rapid single-bit exchange, because the previous principle may limit the reliability of the communication channel.

Principle 4 is of particular importance in WSNs, as it ensures that the system is more resilient as a whole and that ad-hoc connections can quickly and reliably be formed even if some communication errors occur. A rapid bit exchange channel, with minimal latency needed for accurate distance bounding, has no error correction and might be implemented on resource-constrained devices that contain simple radio receivers more susceptible to noise, so that bit errors could occur. It is therefore an advantage if a distance-bounding protocol is resistant to bit errors during the exchange stage, rather than incurring the time cost of rerunning the entire protocol. Hancke and Kuhn (HK) [106] propose that protocols handle errors by defining a bit-error threshold, and that the protocol therefore successfully completes even if some responses are incorrect. Although this method is easier to apply to

pre-computation protocols with no verification stage, these authors also point out that further protocols can tolerate bit errors during the exchange stage using a threshold method, as long as the challenge bits, α_i', received by the prover and the response bits, β_i, sent by the prover are transmitted over an error-corrected channel during the verification stage.

An alternative to pre-commitment protocols is to use error-correcting codes (ECC), as proposed by Singelée and Preneel (SP) [115]. $ECC(l + k, l)$ is applied to the generated strings response string R and this results in a new string. The protocol now continues exactly as before except that there are now $l + k$ timed bit exchanges instead of l. After the bit exchanges, the verifier reconstructs R from the challenges it sent and the responses it received. Finally, the verifier uses the ECC to correct any bit errors in the reconstructed R. The verification stage then proceeds as normal. The disadvantage of using ECC for error resistance is that it increases the protocol execution time. This is a result of the node performing additional ECC calculations and also transmitting additional redundant bits, which are used to detect and correct bit errors in the challenge and response strings. Recently, Munilla and Peinado [116] proposed two effective attacks against the SP protocol. These two attacks increase the adversary's success probability. Moreover, they showed that when the number of allowed failures increases, the false acceptance probability of the SP protocol can be higher than that of the HK protocol.

5.4 COMPARISON FRAMEWORK

The implementation of distance-bounding protocols can differ in a number of ways. For example, not all protocols are designed to be resistant to distance, mafia and terrorist fraud, while some protocols might be optimised to reduce communication or intended for use in systems with resource-constrained nodes. As a result, the security, resource requirements and execution time vary for each protocol. Distance-bounding protocols should not adversely affect the quality of service and should execute in minimal time without consuming resources that could be allocated to main network applications. This section defines the metrics of a framework which will be used to compare selected protocols. The framework consists of the following metrics:

- **Attacks prevented:** Not all protocols that estimate the distance between nodes are necessarily designed to be secure, e.g. [117]. This investigation considers only distance-bounding protocols that prevent one or more of the attacks described in Section 5.3.1, and it will show which attack(s) they prevent.

- **Attack success probability:** Security can be quantified by the probability that an attacker may succeed in executing a chosen attack, which is obviously considered to be a critical parameter for choosing a distance-bounding protocol. For each protocol, the probable success of the attacker in committing the attack(s) that these protocols are meant to prevent will be shown.

- **Cryptographic primitives required:** In order to run a distance-bounding protocol, a selection of cryptographic primitives are needed, such as random number generators, hash/pseudo-random functions, symmetric cryptography and/or asymmetric cryptography. Cryptographic primitives affect not only security but also the resources required to implement and execute the protocol in practice. Therefore, it is important to consider the types of primitives needed to implement a protocol and also how many times during each protocol run this primitive is used in a computation.

- **Memory:** This represents the amount of memory space required to store the functional variables required by each distance-bounding protocol during execution. For example, the protocol needs to store the challenges, responses, nonces, keys, and the like. In order to compare the memory overhead required by each protocol, the following length variables are assigned: K_{SYM} for the symmetric cryptography keys, K_{PUB} for asymmetric cryptography keys, F for a hash or pseudo-random function, MAC for MAC authentication (this indicates the cost of having an additional MAC key, security best practice, or storing the intermediate result from the symmetric encryption function) and N_{RAND} for the random number. For instance, if a protocol uses symmetric cryptography, for which it needs to store a shared symmetric key, and also requires one hash function and two random numbers, the memory needed is: $m = K_{SYM} + F + 2 * N_{RAND}$.

- **Error resilience:** Resilience against channel errors is important for distance-bounding protocols' robustness, especially for those using the rapid bit exchange

phase, since they are typically sensitive to channel errors. Some of the distance-bounding protocols that will be discussed in this chapter are designed to be tolerant to faults occurring during transmission, and so they can be used in noisy environments. Others are able to handle channel errors if the protocol is modified.

- **Required computation:** Efficient computation should be taken into consideration in order to implement a distance-bounding protocol that executes within a specified time, especially if the protocol will be required to run using limited resources. The computational efficiency of each distance-bounding protocol is compared, based on the number of variables or values that need to be computed during each protocol run. The values considered are hash or pseudo-random function results, a symmetric encryption operation, MAC computation and random number generation. This is a slightly crude metric, as the exact times of executing these functions may vary, especially considering the different algorithm options for each computation. However, in the case of distance-bounding protocols where the number of exchanges is not considerably larger than the input block sizes of the primitives, it is feasible that execution times could be comparable, e.g. a symmetric encryption, a MAC or a hash of a short string could be equivalent.

- **Data transmission:** The execution time of the protocols is compared with regard to the amount of data that needs to be transmitted. The speed of the communication link could influence the overall execution time of the protocol, and for nodes that are required to facilitate quick route discovery and connection setup, the communication volume and protocol execution time need to be minimised. It is assumed that each protocol uses the same communication channel, with the total communication time T needed to execute the protocol being equal to the total number of exchanged bits multiplied by the transmission time period (P_B) of each bit, or line coding symbol representing a bit.

5.5 SELECTED DISTANCE-BOUNDING PROTOCOLS

It is not feasible to discuss and compare all existing distance-bounding and secure-ranging proposals in the literature. Therefore, the comparison is limited to protocols adhering to the

'principles of secure distance bounding' as defined in Section 5.3.3. Please note that Principle 2 implies that all these methods would require a special bit-exchange channel for maximum security. However, all these protocols can be converted to a single multi-bit exchange if a special channel is not available. In such a case the effects of the physical communication layer attacks demonstrated in [89, 114] should be taken into account. Examples of such a security analysis for a UWB and IEEE 802.15.4 radio channel can be found in [118, 119]. Please note that the investigated distance-bounding protocols were originally developed for different application scenarios. For example, protocols designed for RFID might place more emphasis on being suitable for devices with resource constraints, while protocols designed for nodes with more resources, which might be found in some WSNs, might rather concentrate on adding functionality or increased resistance to attacks. All protocols discussed here can be applied to WSNs for secure distance estimation.

5.5.1 Brands and Chaum's distance-bounding protocol

Brands and Chaum (BC) [105] presented the first distance-bounding protocol based on measuring the RTT for a single-bit challenge-response exchange. As shown in Figure 5.3, the prover commits to a new random *n*-bit string *R* (e.g. transmits hash value $h(S, R)$). The verifier generates a random *n*-bit string α. In the challenge-response exchange phase the verifier sends one challenge bit α_i to the prover, who replies with the bit $\beta_i = \alpha_i \oplus R_i$. The verifier measures the RTT between sending the bit α_i and receiving the bit β_i. In the end the prover sends *R* in addition to the signed bit strings to the verifier, who checks the correctness of $\beta_i = \alpha_i \oplus R_i$.

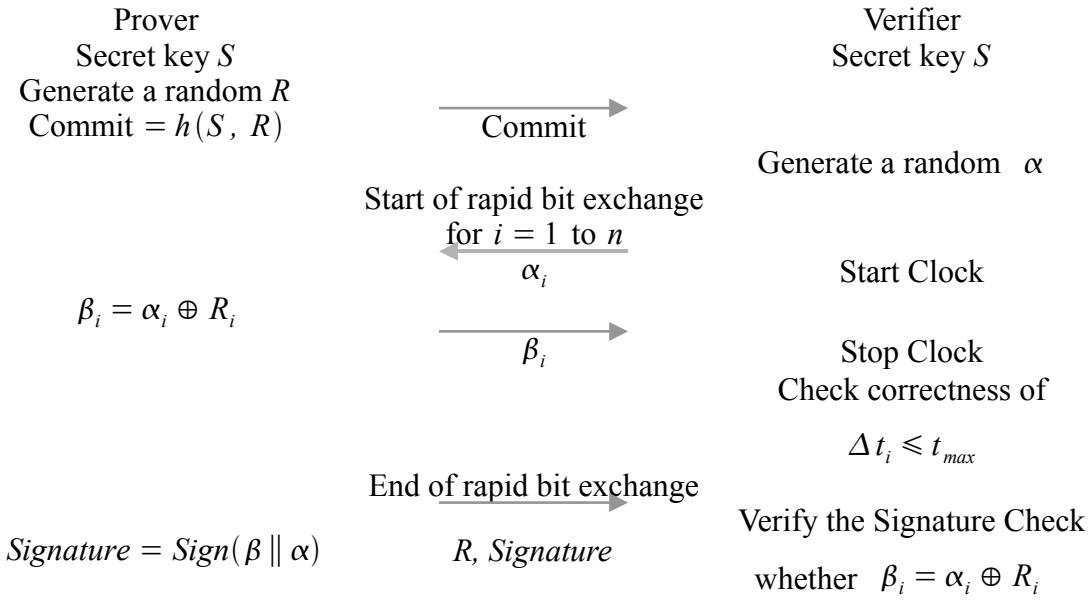

Figure 5.3. Brands and Chaum's Protocol

5.5.2 Bussard and Bagga's distance-bounding protocol

Bussard and Bagga (BB) [120] proposed a protocol called Distance-Bounding Proof of Knowledge (DBPoK) protocol, which combined the original distance-bounding proposal [105] and zero-knowledge proof of knowledge protocols [121]. The DBPoK protocol relies on public key cryptography with a certificate generated by a trusted authority.

The first stage of the DBPoK protocol is called the bit commitment stage. The prover generates a random key K to encrypt its private key S, which results in $e = \varepsilon_K(S)$, and then the prover commits to both K and e. During the bit-exchange stage, the prover responds with either K_i or e_i, depending on the challenge bit received from the verifier. The prover opens the commitments, which are checked by the verifier, on the released bits of K and e in the commitment opening stage. During the proof of knowledge stage, the prover convinces the verifier in a zero-knowledge interaction that he performed the first three stages.

5.5.3 Čapkun et al.'s distance-bounding protocol

Čapkun *et al.* (CBH) [108] proposed a protocol for mutual authentication with distance bounding (MAD). The protocol basically modified the BC protocol to allow for mutual

distance bounding. The verifier and the prover generate two random numbers each (r, r' and s, s' respectively) and exchange the commitment of these numbers. In the fast challenge-response phase the verifier starts by sending $\alpha_1 = r_1$, to which the prover replies with $\beta_1 = s_1 \oplus \alpha_1$. The exchange continues with the verifier sending $\alpha_i = r_i \oplus \beta_{i-1}$ and the prover replying with $\beta_i = s_i \oplus \alpha_i$. During the exchange stage both parties measure the time taken for the response to arrive after sending a challenge. During the authentication phase, the two parties both open commitments (r' and s') and transmit an MAC message containing the response-challenge strings, which both can then verify.

5.5.4 Hancke and Kuhn's distance-bounding protocol

Hancke and Kuhn (HK) [106] proposed a pre-computation protocol that did not require additional data to be transmitted during the verification stage. The protocol, as shown in Figure 5.4, requires the verifier V and prover P to share a common secret key S. During the setup stage V and P exchange random nonces N_V and N_P, and then they compute two n-bit sequences, R^0 and R^1, using a pseudo-random function h of the key and the concatenation of the nonces N_V and N_P. During the single-bit challenge-response exchange V generates and sends an unpredictable random challenge bit α_i, to which P responds instantly with a bit from either R^0 or R^1 based on the value of α_i. During the verification stage the verifier checks that the responses received match the possible responses it calculated during the setup stage. HK protocol is designed to tolerate transmission errors during the exchange stage and a verifier will accept a prover if only k out of n bits are correct.

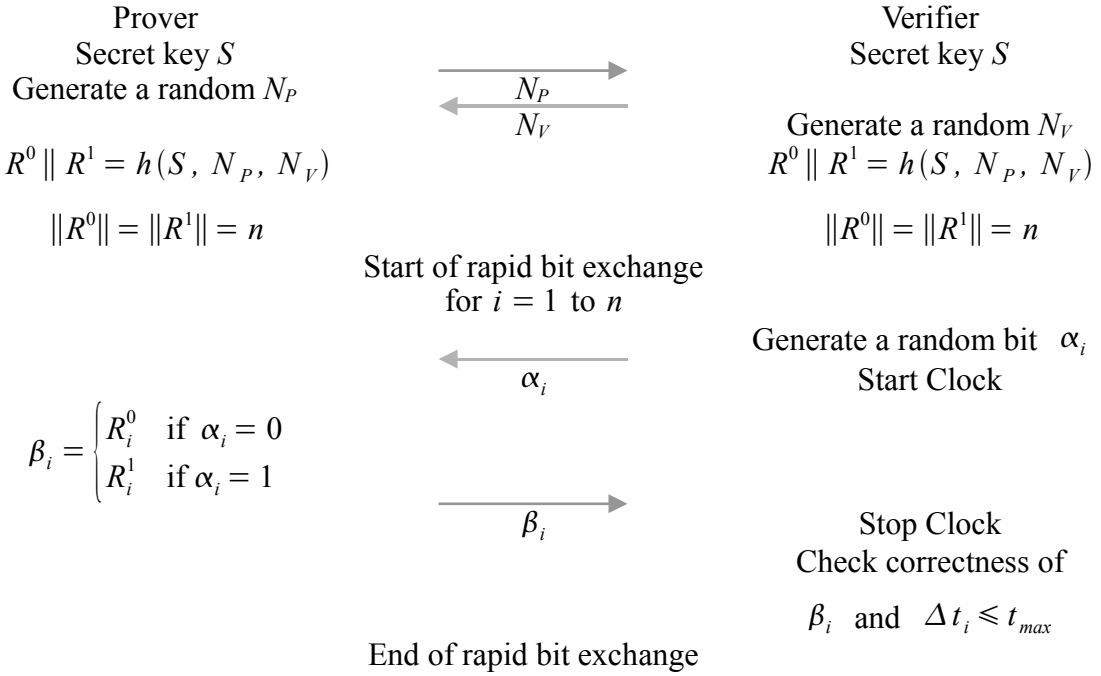

Figure 5.4. Hancke and Kuhn's protocol

5.5.5 Reid et al.'s distance-bounding protocol

Reid *et al.* (RNTS) [122] modified HK protocol [106] to be resistant to terrorist fraud by applying the terrorist fraud deterrent method proposed by BB protocol [120]. A verifier and a prover exchange their identities and nonces and then they use a pseudo-random function h to derive a session key k, which is XORed to a shared key S to get the ciphertext c. In the fast challenge-response phase the prover responds to the verifier with either c_i or k_i, depending on the value of α_i . The prover cannot reveal c_i and k_i to an adversary, as the adversary can then compute S and then impersonate the prover in more than a single run of the protocol.

5.5.6 Tu and Piramuthu's distance-bounding protocol

Tu and Piramuthu's (TP) proposed protocol [123] uses the same principles as were used in [105, 106, 122]. Their main contribution is dividing the fast challenge-response phase into four loop iterations using different hash outputs; they reduce the success probability of an adversary to $(9/16)^n$. However, Kim *et al.* [124] showed that this protocol is not secure against an active adversary; a mafia fraud attacker which could change the challenge bits

and relay multiple protocol runs would recover the secret shared key.

5.5.7 Munilla and Peinado's distance-bounding protocol

Munilla and Peinado (MP) [107] modified HK protocol in order to reduce the adversary's mafia fraud success probability. They used an additional void challenge in addition to the two existing bit values that can be transmitted. A void challenge is basically a period in which the verifier does not send any challenge bit. If a mafia fraud attacker tries to query the prover pre-emptively to read out possible response bits, he might send a challenge in a time slot in which the prover knows it should not expect a challenge, and therefore the attack will be detected. In order to use the void challenge, the protocol requires that the verifier and the prover have crude synchronisation, although it has been shown that the protocol can be modified to eliminate this requirement.

5.5.8 Kim and Avoine's distance-bounding protocol

Kim and Avoine (KA) [125] modified the previous protocol [107] by using mixed challenges. The challenges are divided into two categories: random challenges and predefined challenges. The first method uses random bits generated by a verifier and the second uses predefined bits known to the verifier and the prover in advance. After exchanging nonces, the verifier and the prover compute a $4n$-bit sequence $T \| D \| R^0 \| R^1$. The string T indicates which challenges a verifier sends; string D is the predefined challenges and strings R^0 and R^1 are the responses used by a prover. The random challenges, which are unknown to the prover, ensure that the protocol is still secure against distance fraud, as the prover would not be able to respond pre-emptively to these challenges. The predefined challenges, of which the prover knows the value beforehand, allows the prover to detect whether an attacker is transmitting the query, i.e. if the challenge received is different from what is expected, it did not originate from the verifier.

5.5.9 Kim et al.'s distance-bounding protocol

Kim *et al.* (KAKSP) [124] proposed a distance-bounding protocol based on the authentication protocols MAP1 [126] and MAP1.1 [127]. In order to achieve distance

bounding, they added a timed challenge-response phase to the MAP1.1 protocol and then they modified it to ensure the privacy of the prover. The first two phases of this protocol are, however, very similar to the first two phases of the RNTS protocol [122]. In the verification stage the prover and the verifier authenticate each other by exchanging messages consisting of a keyed pseudo-random function of the exchanged bits. The prover also sends the challenges it received and responses sent in plaintext. This allows the verifier to incorporate error resistance into the protocol run using the threshold method.

The verifier must perform an exhaustive search over its database that grows linearly with the number of nodes deployed in order to find the prover's identity and key, which should be used to verify the prover's authentication message. Using this protocol in a system with many nodes may therefore not be efficient. Peris-Lopez *et al.* [128] have shown that an attacker who observes multiple protocol runs can recover the prover's secret key.

5.5.10 Meadows et al.'s distance-bounding protocol

The protocol proposed by Meadows *et al.* (MSC) [47] can be seen as a hybrid of a pre-computation and pre-commitment protocol. During the setup phase the verifier requests that the protocol must commence and sends a nonce to the prover. The prover calculates a response string R by hashing its ID and a random nonce N^P it generates. The verifier sends a random challenge, which the prover XORs to R and then transmits back. Finally, the prover sends an authentication message to the verifier, including the random nonce used to calculate R. In this protocol variation only the prover calculates the response string and there is no explicit commitment before the exchange stage. The prover is prevented from pre-emptively responding with $R' \oplus \alpha$, as it will be impossible to find a value for N^P so that $h(N^P, P) = R' \oplus \alpha$ before the verification stage. The advantage of this protocol is that it involves minimal data transmission and does not require the nodes to share a key before the verification stages, which means the verifier can make initial estimates of the distances to other nodes and determine its neighbours before keys are exchanged.

5.5.11 Avoine *et al.*'s technique

Avoine *et al.* [129] presented a generic technique called MULtiState Enhancement

(MUSE) that can be used by the existing distance-bounding protocols. MUSE extended the void challenges concept, which was introduced in MP protocol [107], to *p*-symbols, where $p \geqslant 2$. In other words, instead of using binary messages {0, 1}, MUSE uses *p*-state messages {0, 1, 2, …, *p*-1}. The use of *p*-state messages decreases the adversary's success probability without changing the number of exchange messages. Therefore, MUSE could improve the performance of distance-bounding protocols that use this technique. However, using the MUSE technique requires more memory, exchanges a higher number of bits, and in practice it could be difficult to implement the multistate messages required by this technique. In the comparison, the MUSE-4 HK will be considered, which is an improvement over the four-state message approach of HP protocol.

5.5.12 Avoine and Tchamkerten's distance-bounding protocol

Protocols belonging to the HK family do not distinguish authentication from proximity check. Avoine and Tchamkerten (AT) introduced a tree-based RFID distance-bounding protocol [130], which distinguishes authentication from proximity check. The tree-based RFID protocol performs the authentication before the fast phase, using the first *m* bits of the keyed-hash value. The prover and the verifier use the remaining 2^{n+1} - 2 bits of the keyed-hash value to label a full binary tree of depth *n*. The left and the right edges are labelled 0 and 1, respectively. Each node in the tree (except the root) is associated with a value of a particular bit from the 2^{n+1} - 2 bits. The edge and the node values represent the verifier's challenges and the prover's replies, respectively.

Compared with HK protocol, this protocol reduces the probability of a successful attack to $(1/2)^n(n/2+1)$, but it requires more memory space. However, this protocol can use multiple trees to balance the false-acceptance rate (FAR) and memory requirement. Using multiple trees requires storing $\alpha(2^{k+1}-2)$ bits for the fast phase where α is the number of the trees, *k* is the depth of the trees and $n = \alpha k$. This memory is much less than the one required for using a single tree $2^{n+1} - 2$. With a multiple tree FAR is equal to $(2^{-k}(k/2+1))^{\alpha}$. The required memory shown in Table 5.1 and Figure 5.8 considers using eight trees with depth 8.

5.5.13 Trujillo-Rasua et al.'s distance-bounding protocol

The authors of [131] introduced a graph-based distance-bounding protocol (TMA). The graph consists of $2n$ nodes $\{q_0, q_1, \cdots, q_{2n-1}\}$ and $4n$ edges. Both the verifier and the prover start at node q_0. The next node will be specified based on the challenge bit and the corresponding edge. The prover and the verifier either move to the next node (q_{i+1}) or to the node after two steps (q_{i+2}). However, TMA protocol can be simply represented without involving any graph, as shown in Figure 5.5; also l can be eliminated, which is the complementary of s.

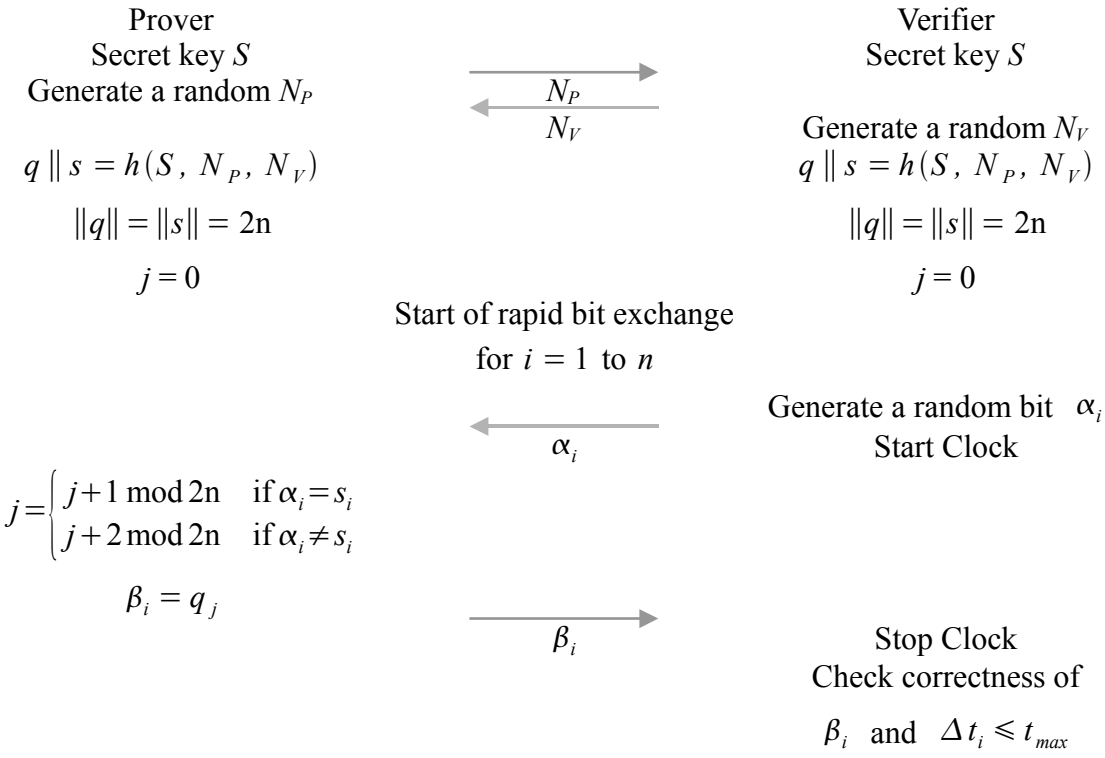

Figure 5.5. Trujillo-Rasua *et al.*'s protocol

5.5.14 Peris-Lopez et al.'s distance-bounding protocol

Peris-Lopez *et al.* [128] presented a passive attack against KAKSP protocol, which is also exploitable against RNTS and TP protocols. This attack allows an adversary to discover the long-term secret key of the prover owing to the weak protection mechanism adopted against terrorist fraud attacks. Therefore, this attack wrecks all the security properties claimed by these protocols. Peris-Lopez *et al.* introduced design guidelines to propose a

secure and efficient distance-bounding protocol. These guidelines were used to design a new protocol called Hitomi. Hitomi protocol was inspired by KAKSP protocol.

In KAKSP protocol, knowing Z^0 and Z^1 makes it easy to calculate the value of secret key x, since $x = Z^0 \oplus Z^1$. In contrast, in Hitomi protocol there is a non-linear relation between Z^0 and Z^1; moreover, no information on the secret key x is revealed through the responses bits. However, Hitomi protocol requires more memory space and bits transmission than does KAKSP protocol.

5.6 COMPARISON OF DISTANCE-BOUNDING PROTOCOLS

This section presents a comparison of the distance-bounding protocols discussed in Section 5.5. The comparison is based on the proposed framework defined in Section 5.4, with the following metrics: attacks prevented, success probability of an adversary to perform distance fraud, mafia fraud or terrorist fraud, the protocol error resilience, requirement of special channel, protocol efficiency based on the primitives used by the prover, the amount of memory space required by the prover, the computation required by the prover and the amount of data exchanged between the two parties. For the sake of comparing the different protocols, the resource variables were assigned the following values: K_{SYM} = 128 bits, K_{PUB} = 3072 bits, F = 256 bits, MAC = 128 and N_{RAND}= 64 bits. These values were chosen based on the cryptographic functions that are most likely to be used, taking into account the minimal recommendations for choosing secure algorithms up to the year 2030 by the National Institute of Standards and Technology [132]. For K_{SYM} and MAC AES128 is used, and to achieve equivalent 128-bit security, F is generated using SHA-256 and the public key K_{PUB} is an RSA key of length 3072. If K_{PUB} was implemented using elliptic curve cryptography the key length would be 256 - 383 bits. To allow for equal comparison of all protocols, n needs to be smaller than $F/3 \approx 85$ (Munilla's protocol needs to split F into three equal length registers). The number of iterations in the exchange phase is therefore chosen to be n = 64 (simply the closest number smaller than 85 that is a power of 2).

Kara *et al.* [133] show that the success probability of an adversary to perform distance fraud against HK protocol is $(3/4)^n$. Therefore, in Table 5.1, this value will be considered for all protocols that have a lookup from registers, such as HK, RNTS and MP protocols. A

summary of the comparison is shown in Table 5.1. If required, the reader can generate a similar table tailored to his system design by assigning his system's values and substituting these values into the equations given in Table 5.2.

Table 5.1. Comparison of selected distance-bounding protocols

Protocol	Attacks			SPA			ER	Mem bits	Computation			TD
	DF	MF	TF	DF	MF	TF			RN	PRF	Asm	bits
BC	√	√		$(1/2)^n$	$(1/2)^n$		½	704	1	2		576
BB	√	√	√	$(1/2)^n$	$(1/2)^n$	$(1/2)^n$	½	6912	3	4	1	3392
CBH	√	√		$(1/2)^n$	$(1/2)^n$		½	1152	2	4		1024
HK	√	√		$(3/4)^n$	$(3/4)^n$		√	258	1	1		256
RNTS	√	√	√	$(3/4)^n$	$(3/4)^n$	$(3/4)^n$	√	384	1	1		256
TP	√	√	√	$(1/2)^n$	$(9/16)^n$	$(9/16)^n$	½	640	1	5		1280
MP	√	√		$(3/4)^n$	$(5/8)^n$		√	641	1	2		512
KA	√	√		$(7/8)^n$	$\sim(1/2)^n$		√	260	1	1		256
KAKSP	√	√	√	$(1/2)^n$	$(1/2)^n$	$(3/4)^n$	√	960	1	3		832
MSC	√	√		$(1/2)^n$	$(1/2)^n$		½	320	1	2		384
MUSE	√	√		$(1/4)^n$	$(7/16)^n$		√	768	1	1		384
AT	√	√		$(1/2)^n$	$(1/2)^n * (n/2+1)$		√	4336	1	1		512
TMA	√	√					√	512	1	1		256
Hitomi	√	√	√	$(1/2)^n$	$(1/2)^n$	$(3/4)^n$	√	1152	3	4		1024

Legend	
DF	Distance fraud
MF	Mafia fraud
TF	Terrorist fraud
SPA	Success probability of an adversary
n	Number of iteration
ER	Error resilience
Mem	Memory
RN	Random number
PRF	Pseudo-random function
Asm	Asymmetric cryptography
TD	Transmitted data
½	Possible with modifications, added overhead

Table 5.2 summarises the amount of memory required and the transmitted data exchanged between the prover and verifier for each protocol based on the cryptographic primitives defined in the framework.

Table 5.2. Memory and transmitted data based on the defined primitives

Protocol	Memory	Transmitted data
BC	$K_{SYM} + F + 3n + MAC$	$F + 3n + MAC$
BB	$2\,K_{PUB} + K_{SYM} + 6n + F$	$5n + K_{PUB}$
CBH	$K_{SYM} + 2F + 4n + 2MAC$	$2F + 4n + 2MAC$
HK	$K_{SYM} + 2N_{RAND} + 2$	$2N_{RAND} + 2n$
RNTS	$K_{SYM} + 2N_{RAND} + 2n$	$2N_{RAND} + 2n$
TP	$K_{SYM} + 2N_{RAND} + 2n + F$	$2N_{RAND} + 2n + 4F$
MP	$K_{SYM} + 2N_{RAND} + 2n + 1 + F$	$2N_{RAND} + 2n + F$
KA	$K_{SYM} + 2N_{RAND} + 4$	$2N_{RAND} + 2n$
KAKSP	$K_{SYM} + 2N_{RAND} + 3n + 2F$	$2N_{RAND} + 3n + 2F$
MSC	$K_{SYM} + 2N_{RAND} + n$	$2n + MAC + 2N_{RAND}$
MUSE	$K_{SYM} + 2N_{RAND} + nplog_2(p)$	$2N_{RAND} + 2nlog_2(p)$
AT	$K_{SYM} + 2N_{RAND} + 2^{n+1} - 2$	$2N_{RAND} + F + 2n$
TMA	$K_{SYM} + 2N_{RAND} + 4n$	$2N_{RAND} + 2n$
Hitomi	$K_{SYM} + 4N_{RAND} + 4n + 2F$	$4N_{RAND} + 4n + 2F$

Legend	
K_{SYM}	Symmetric key
K_{PUB}	Asymmetric key
N_{RAND}	Random number
MAC	Message Authentication Code
F	Pseudo-random function
n	Number of iteration
p	Number of state

5.6.1 Security

In order to compare the security of these distance-bounding protocols, given a fixed number of challenge-response exchanges, the comparative success probability of an adversary to perform distance, mafia and terrorist fraud with number of iterations $n = 64$ is computed. The results are shown in Table 5.3. The probability of an attack succeeding decreases as the number of iterations increases. Increasing the number of iterations does, however, involve a trade-off with both memory used and the amount of data transmitted increasing with the number of iterations, as shown in Figure 5.6 and 5.7.

Table 5.3. The success probability of an adversary (SPA)

Protocol	SPA		
	DF	MF	TF
BC	5.42×10^{-20}	5.42×10^{-20}	
BB	5.42×10^{-20}	5.42×10^{-20}	5.42×10^{-20}
CBH	5.42×10^{-20}	5.42×10^{-20}	
HK	1.01×10^{-08}	1.01×10^{-08}	
RNTS	1.01×10^{-08}	1.01×10^{-08}	1.01×10^{-08}
TP	5.42×10^{-20}	1.02×10^{-16}	1.02×10^{-16}
MP	1.01×10^{-08}	8.64×10^{-14}	
KA	1.94×10^{-04}	5.42×10^{-20}	
KAKSP	5.42×10^{-20}	5.42×10^{-20}	1.01×10^{-08}
MSC	5.42×10^{-20}	5.42×10^{-20}	
MUSE	2.94×10^{-39}	1.05×10^{-23}	
AT	5.42×10^{-20}	1.79×10^{-18}	
Hitomi	5.42×10^{-20}	5.42×10^{-20}	

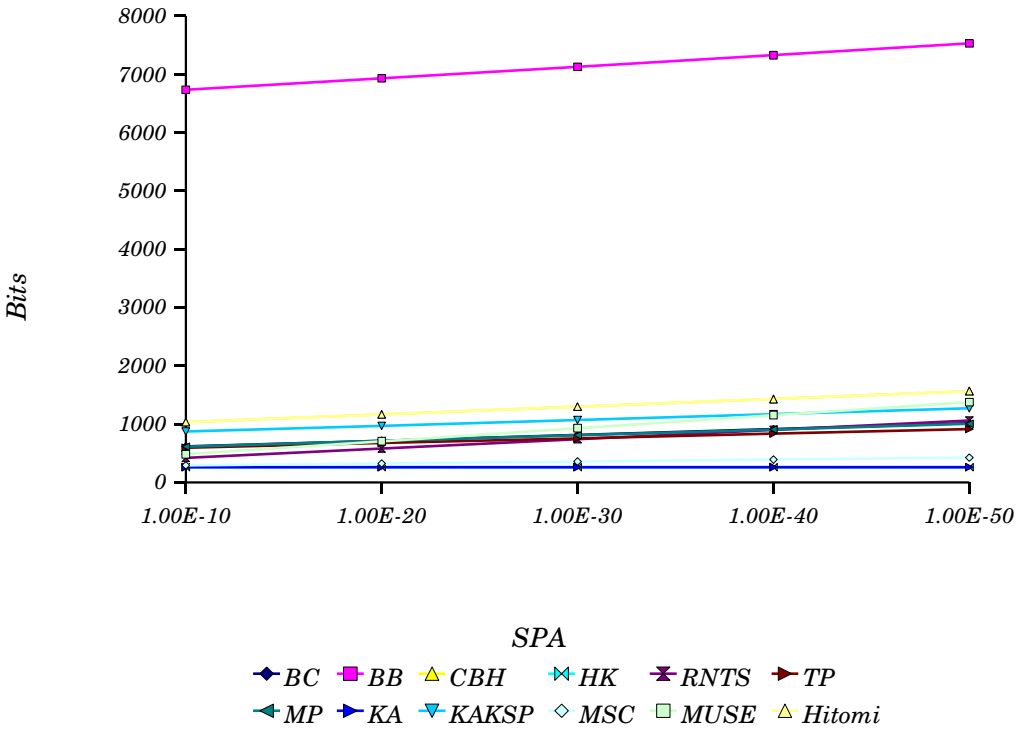

Figure 5.6. Memory vs probability of attack for mafia fraud (SPA)

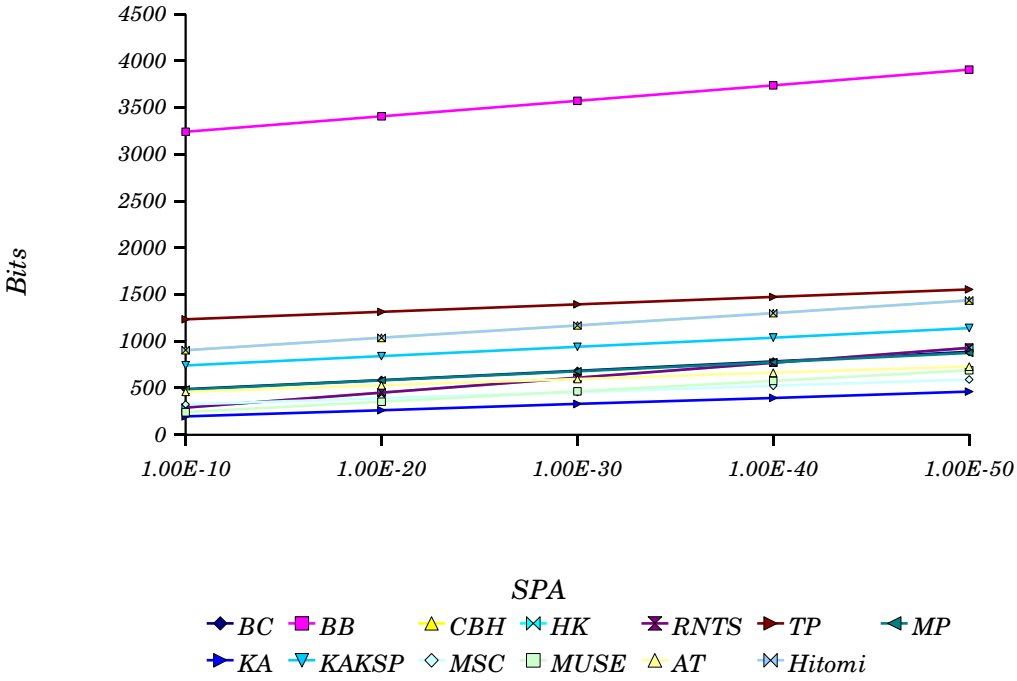

Figure 5.7. Transmitted data vs success probability of attack for mafia fraud (SPA)

5.6.1.1 Current vs k-previous challenge dependent

Recently, Kara *et al.* [133] classified distance-bounding protocols having bitwise fast phases and no final signature into two types: Firstly, *current challenge dependent* (CCD), where each response bit depends only on the current challenge, and secondly, *k-previous challenge dependent* (*k*-PCD), where each response bit depends on the current and *k*-previous challenges. Kara *et al.* showed the theoretical security bounds for these two types. For CCD protocols, they showed that there is a trade-off between mafia fraud and distance fraud, namely $P_{maf} + P_{dis} \geqslant 3/2$, where P_{maf} and P_{dis} are the success probability for mafia fraud and distance fraud respectively. Also, they proved that there is a security limit concerning mafia fraud such that $P_{maf} \geqslant 3/4$. Therefore, if the security level for distance fraud is ideal (i.e. $P_{dis} = 1/2$), then the protocol is completely vulnerable to mafia fraud (i.e. $P_{maf} = 1$).

To improve the security level of these protocols, Kara *et al.* suggested extending these protocols to become *k*-PCD protocols. In these protocols, $P_{maf} + P_{dis} \geqslant 5/4$, while $P_{maf} \geqslant 5/8$. As a case study, they illustrated two natural extensions on HK protocol among 1-PCD protocols. From the first extension, they achieved $P_{dis} \geqslant 1/2$ and

$P_{maf} \geqslant 3/4$. For the second one, they achieved $P_{dis} \geqslant 5/8$ and $P_{maf} \geqslant 5/8$.

The authors conjectured that the attacks used in the analysis are the best generic attacks mounted on CCD and *k*-PCD protocols. However, this could be not the case, and so finding other ways to implement these attacks to produce different trade-off curves can be regarded as an open problem.

5.6.2 Memory and transmitted data

In order to compare the memory requirements and data transmission of the selected distance-bounding protocols, the comparative amount of memory required by the prover to run each protocol, and the number of bits exchanged between the prover and the verifier are computed. For the purpose of comparison the following values are used: K_{SYM} = 128 bits, K_{PUB} = 3072 bits, F = 256 bits, MAC = 128 and N_{RAND} = 64 bits and the number of iterations is n = 64. The results are shown in Figure 5.8. The total transmission time can be calculated by multiplying the number of transmitted bits with the channel bit period P_B. Figure 5.9 shows the effect of increasing the number of iterations on the data transmitted between the prover and the verifier. Figure 5.10 shows the effect of increasing the number of iterations n on the memory required by each protocol.

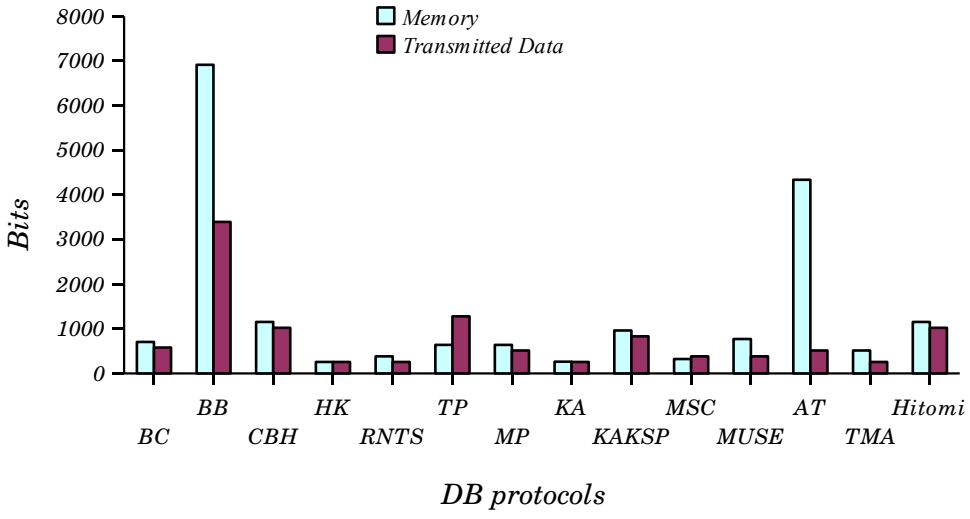

Figure 5.8. The required memory and transmitted data by the selected DB protocols (n = 64)

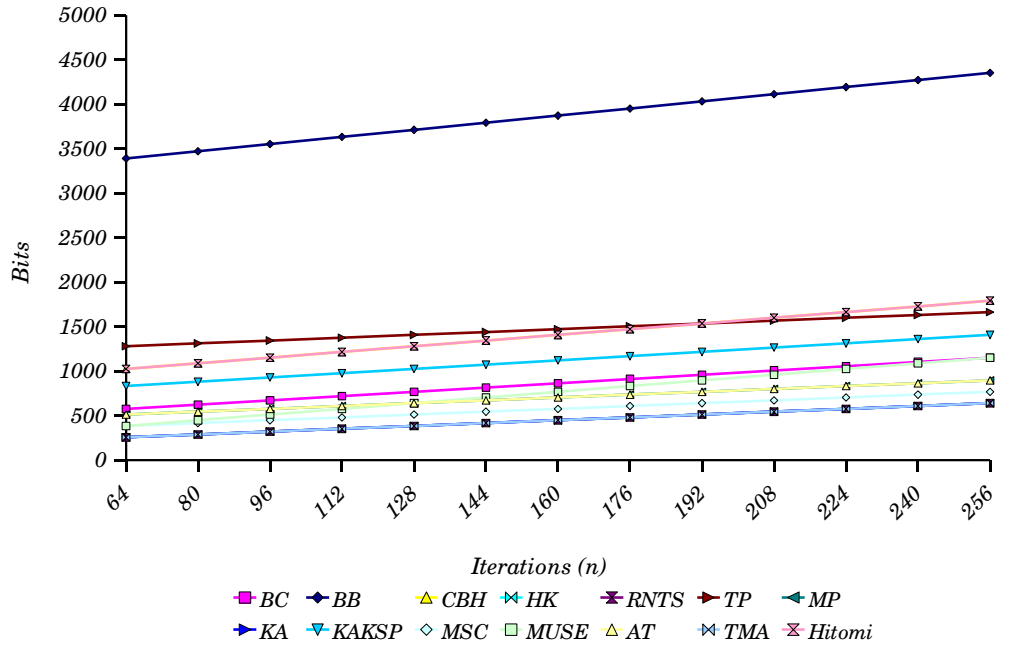

Figure 5.9. Transmitted data vs number of iterations (n)

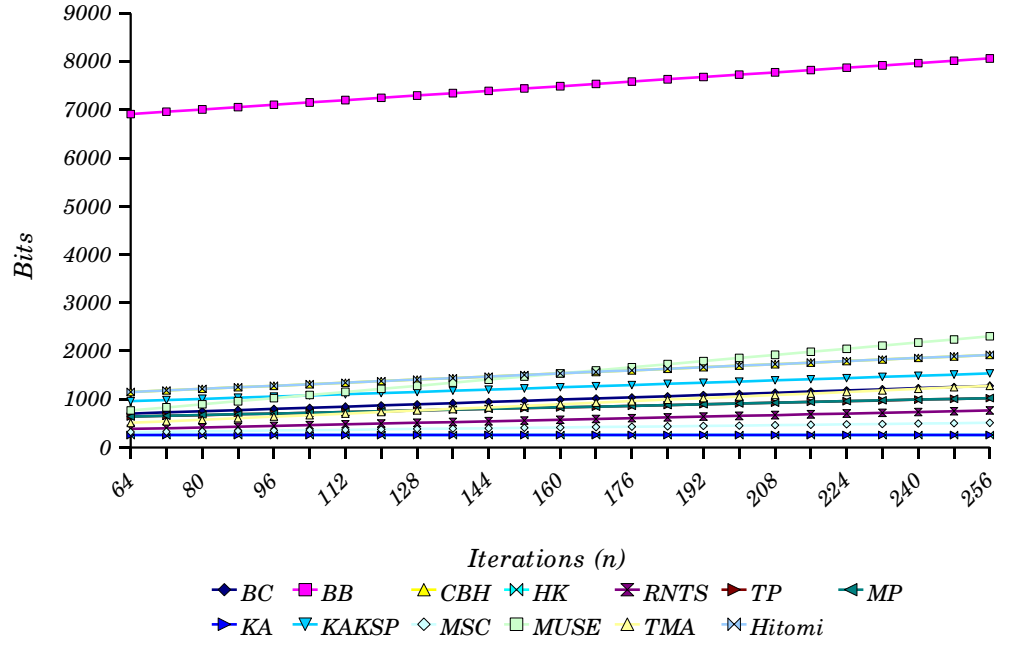

Figure 5.10. Memory vs number of iterations (n)

5.6.3 Computation

To compare the computational efficiency of each distance-bounding protocol, the number of variables or values that need to be computed during each protocol run are considered. The values considered are hash or pseudo-random function results, a symmetric encryption operation, MAC computation and random number generation. Figure 5.11 shows the number of computations needed by the prover for each protocol run. This is a slightly crude metric, as the exact times of executing these functions may vary in practice, especially considering the different algorithm options for each computation. However, in the case of the example given, the number of exchanges is not considerably larger than the input block sizes of the primitives, and it is therefore feasible that execution times could be comparable, e.g. a keyed pseudo-random function and a MAC should require a comparable time to compute. The protocol by BB does not appear in this figure, since it requires asymmetric cryptography, which requires more resources and cannot be compared like-with-like with the other processes.

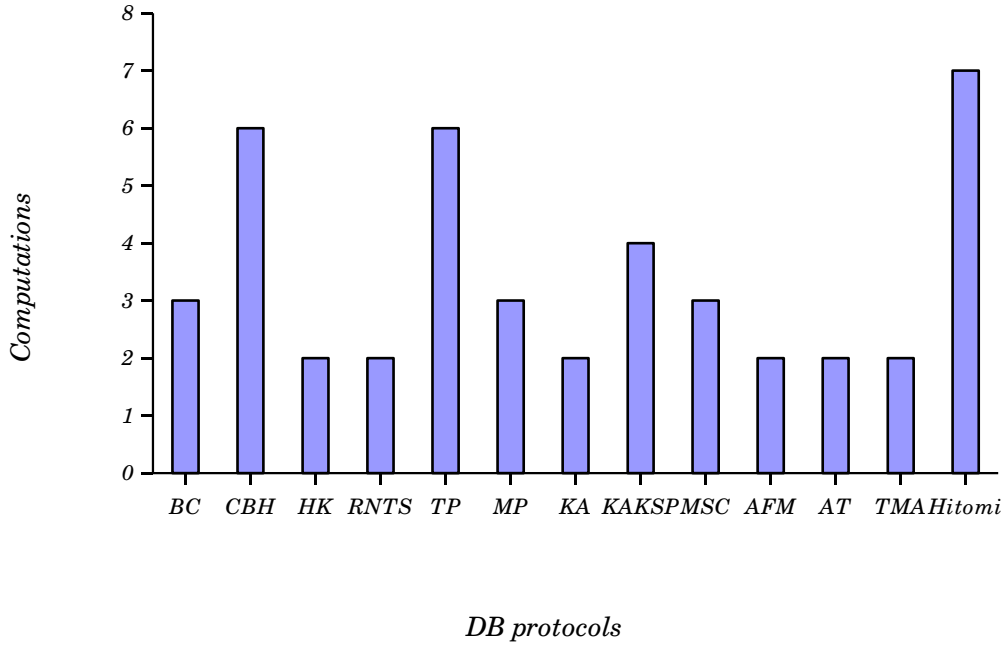

Figure 5.11. The number of computations required by the prover

5.6.4 Choosing a suitable protocol

There is no single protocol that is ideal under all design conditions. Each of the selected protocols discussed in Section 5.5 has advantages and disadvantages, and given the right design conditions any of these protocols can be argued to be 'most suitable'. Given the basic trade-offs and results presented in this section, a system engineer should be able to get a good indication of which protocol could be used in his system. For example, if transaction time and resource constraints were not an issue, the protocol by BB provides the best level of security for all types of attack (as shown in Table 5.2). If the designer wanted to use minimal resources and wanted the protocol to complete in the shortest time possible, taking into account error correction, the HK, RNTS and KA proposals are possible choices. These three protocols require the least computation (Figure 5.11), transmit the lowest number of bits (Table 5.1, Figure 5.8) and require the least memory of protocols that do not have to be modified to accommodate bit errors (Table 5.1, Figure 5.8). In this case, the most suitable protocol choice could depend on the attacks that are to be mitigated(Table 5.1) and attack success (Table 5.1). RNTS protocol is the only choice for protection against terrorist fraud and offers equal security to HK protocol in mafia and distance fraud, at a cost of needing almost 50% additional memory. KA offers the strongest security against mafia fraud but is comparatively weak against distance fraud. If the attack probability needs to be lowered, Figure 5.6 and 5.7 would give an indication of the additional resources and transaction time required. There are also design factors beyond security, memory and transaction time, which were highlighted in Section 5.4. For example, if mutual distance bounding is required, CBH's MAD protocol would be the only option.

In general, the amount of resources required and the time needed for execution are reasonable for a WSN environment. If implemented using typical cryptographic algorithms in use today, 10 out of the 14 protocols require less than a kilobit of data to be stored and transmitted, which appears feasible for all but the most resource-constrained platforms. At the same time, the security level obtained using these algorithm choices is relatively strong, with the probability of an attack succeeding exceeding 10^{-8} in all but one case.

5.7 ATTACK RESISTANCE OF ALWADHA ALGORITHM

The previous two sections provided a comparative study of selected distance-bounding protocols that achieve the four principles proposed by the authors of [89]. This comparison could act as a guide for choosing a suitable distance-bounding protocol for implementing a secure distance estimation component in WSNs. However, implementing a secure localisation system should not rely only on using a secure distance-bounding protocol, for the following reasons:

- It could be difficult to achieve the four principles proposed by Clulow *et al.* [89]. For example, using a rapid bit exchange phase, in which the recipient can instantly react to the reception of each individual bit, requires a special type of channel that may not be available because of a certain limitation in the sensor nodes used.

- Distance-bounding protocols can only prevent malicious nodes from pretending that they are closer. However, an attacker can still pretend that it is more distant from the node that sent the "location request" packet than it really is.

- Using the RTT technique requires interaction between the two parties involved in the localisation system (the unknown and the other references), i.e. the unknown sends a "location request" packet, the neighbouring references send the "location response" packets and then the unknown estimates the round-trip time of the sending and receiving packets. However, some scenarios require using a passive localisation system, in which the references only cooperate with one another to estimate the unknown location. For example, references could locate the position of an attacker which is trying to interfere with the network. This type of attacker would not respond to the request sent by these references, and so they could simply use the RSS of the signals sent by this attacker to estimate the distance to it and then estimate its position.

- A correct distance estimation does not mean that the location information received is correct. As illustrated in Figure 5.12, a compromised beacon node (b_j) sends incorrect location information to an unknown pretending that it is in the location (x', y'). The distance between the beacon node and unknown node is d, the distance between the incorrect location (x', y') and the unknown is also d. Distance-bounding

protocol will indicate that the estimated distance is correct. However, the unknown node could determine its location incorrectly.

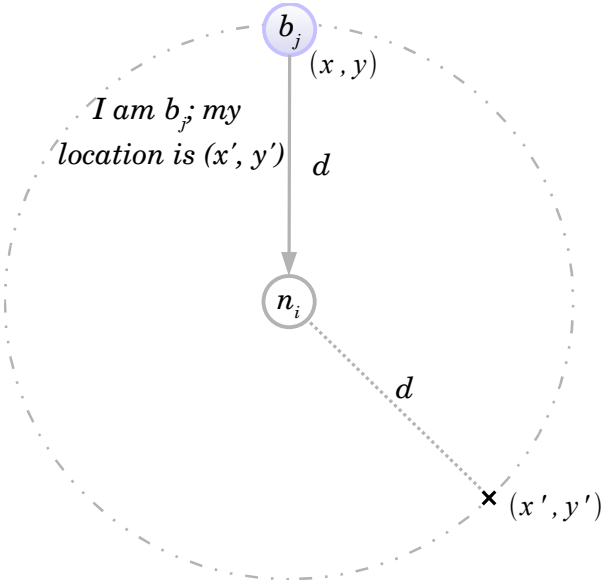

Figure 5.12. Compromised beacon node sends incorrect location information that is the same distance (d) to the unknown node

Therefore, the ALWadHA algorithm does not rely only on using a secure distance-bounding protocol but it also uses a robust position computation component that tolerates the existence of malicious nodes. The ALWadHA algorithm uses three types of filter to select the proper subset of references, as shown in Figure 4.3. Filter three is used to eliminate those references with a high distance error. On the other hand, this filter could be used to eliminate malicious nodes from the selected subset of references. The compromised nodes provide incorrect location information to mislead other nodes; however; pretending to be in an incorrect location increases the difference between the measured and estimated distance, which makes it easy to be detected by filter three, and as a result these malicious nodes will not participate in the localisation process. In fact, the goal of the ALWadHA algorithm is not to detect these malicious nodes; rather it is to enable nodes to live with them without disturbing the location estimation.

5.8 SIMULATION RESULTS

This section evaluates the resistance of the localisation algorithms against the two types of attack shown in Figure 5.1. In both types the attacker provides an incorrect location reference to mislead other nodes, which could determine their location incorrectly. The performance of the localisation algorithms is evaluated based on two metrics, firstly location error created by malicious nodes, and secondly the number of malicious nodes in the network. The random deployment characteristics described in Section 3.3.3 will be used.

5.8.1 Dishonest reference nodes

Four malicious nodes were distributed randomly in the network, with each sub-area having one malicious node (0.25 malicious node per r_{tx}^2). These malicious nodes pretended to be honest references and sent incorrect location references. The error of their location was generated randomly, using a normal random variant with a mean of 0.1% of the real location and a standard deviation changed from 1% to 50% of the real location. The mean error of location estimation is recorded at the end of the run time.

Figure 5.13 shows that increasing the erroneousness of the malicious nodes' location dramatically increases the mean error of estimated location in all algorithms except ALWadHA. The maximum mean error of the ALWadHA algorithm is equal to 4.76% of the transmission range, which occurs when the standard deviation is equal to 20% of the malicious nodes' location ($L_{malicious}$). Increasing the standard deviation reduces the mean error gradually till it reaches 3.51% at the standard deviation of 50% of $L_{malicious}$. The reason for this improvement is that increasing the erroneousness of the malicious nodes' advertised location also increases the difference between the measured and the estimated distance, and so filter three detects these nodes and eliminates them from the selected subset.

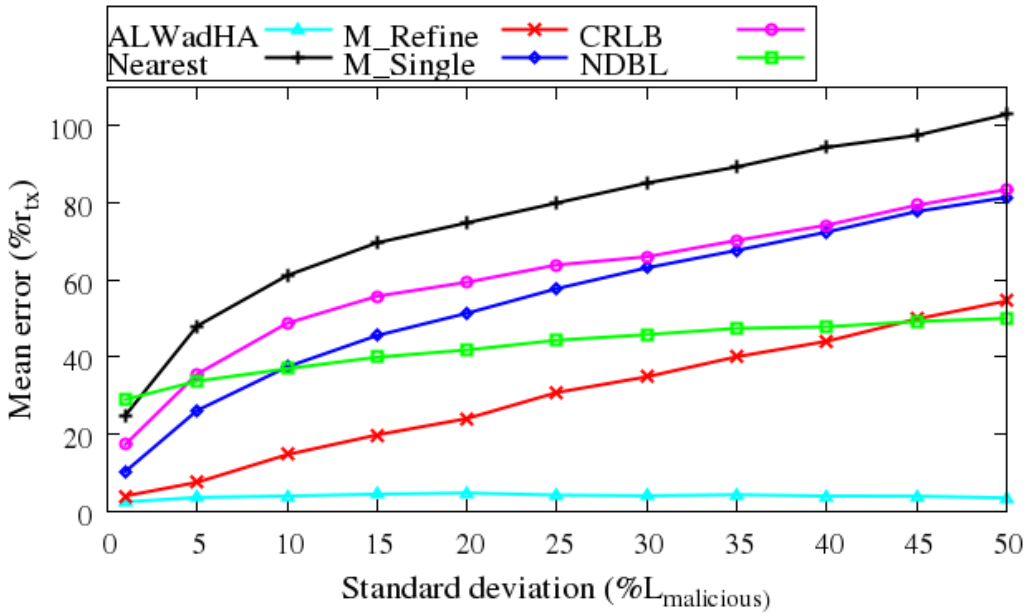

Figure 5.13. Location estimation error, four dishonest references

In the second experiment, the standard deviation was fixed to 50% of $L_{malicious}$ and the number of malicious references was changed from 4 to 16. Note that the number of beacon nodes used in the network is only 12. Figure 5.14 shows that ALWadHA outperforms other localisation algorithms and still achieves good accuracy of location estimation in spite of the increase in the number of malicious references. The mean error of location estimation of the ALWadHA algorithm in the presence of 16 malicious references is equal to 7.72% of the transmission range, which is much lower than that of the other localisation algorithms.

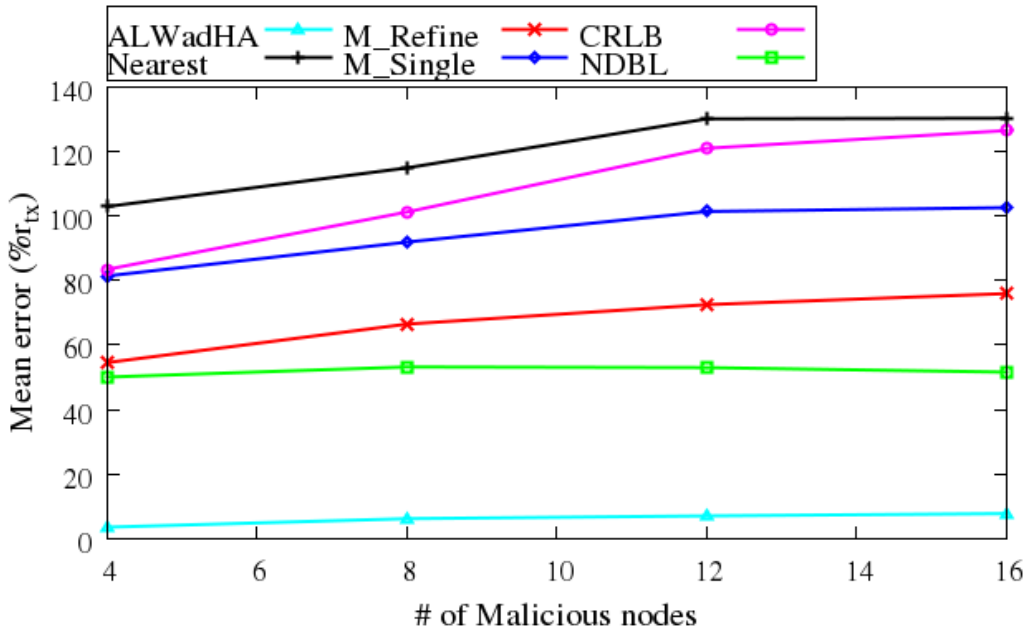

Figure 5.14. Location estimation error, standard deviation is equal to 50% of $L_{malicious}$

5.8.2 Compromised beacon nodes

The previous two experiments were repeated, using compromised beacon nodes instead of dishonest references. The results of these two experiments are shown in Figure 5.15 and 5.16. Compared with the previous two experiments, Figure 5.13 and 5.14, the Nearest, CRLB, M_Single and M_Refine algorithms yielded the same results, because these localisation algorithms do not distinguish between the reference and beacon nodes. For example, the Nearest algorithm selects the closest three nodes regardless of the type of these nodes, whether they are references or beacons. Therefore, an attacker has no need to perform the second type of attack and simply performs the first one. ALWadHA and NDBL algorithms distinguish between the reference and beacon nodes. Therefore, the mean error of location estimation is higher in the second type of attack. Figure 5.15 shows again that the maximum mean error is also at the standard deviation of 20% with a value of 6.43% of transmission range, while the mean error at a standard deviation of 50% is only 4.45% of the transmission range. Figure 5.16 shows that, in spite of there being 16 compromised beacons, which outnumber the existing benign beacons, the mean error of location estimation using ALWadHA algorithm is only 15.39% of the transmission range. This error is much less than that achieved by the other localisation algorithms.

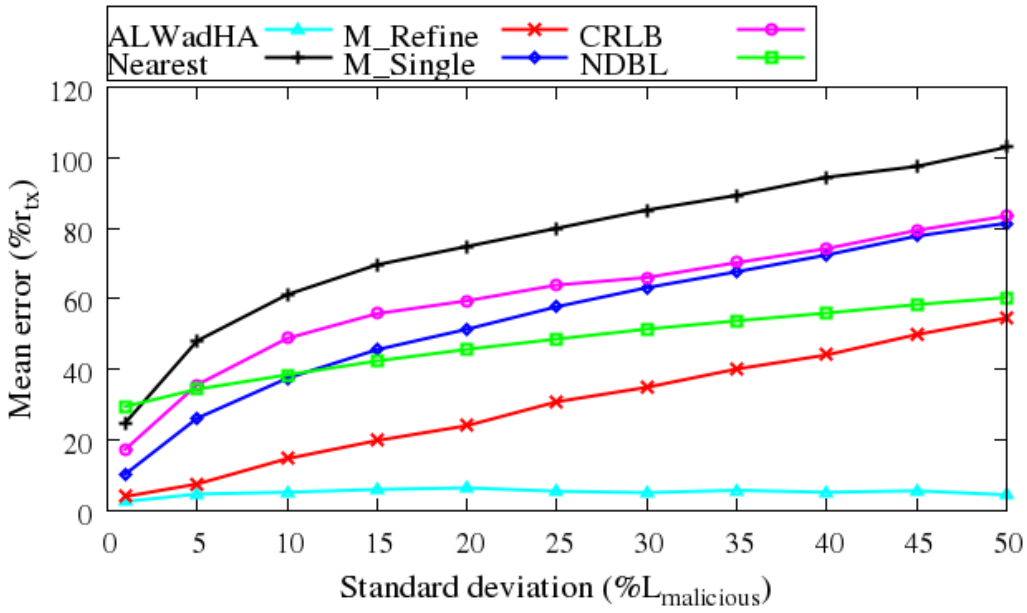

Figure 5.15. Location estimation error, four compromised beacon nodes

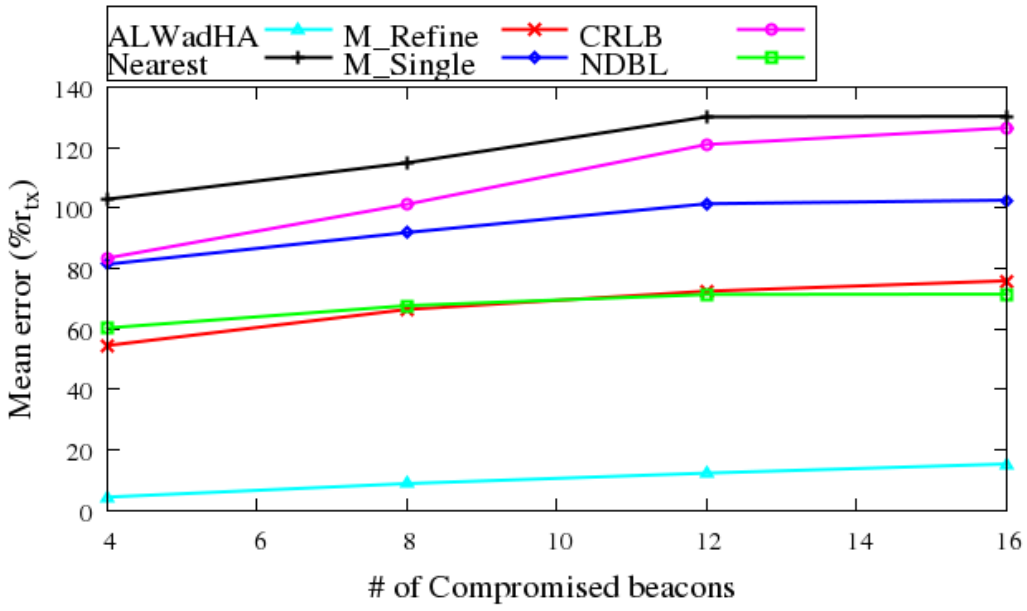

Figure 5.16. Location estimation error, standard deviation is equal to 50% of $L_{malicious}$

5.9 CHAPTER CONCLUSIONS

The security of a localisation system is a critical issue, because compromising the

localisation system could disturb the entire functioning of WSNs. Attacks against localisation systems can be classified mainly into two categories: attacks on distance-estimation components and attacks on position-computation components. Compromising one of them could affect the entire localisation system. Therefore, a secure localisation system needs to take into account the security of these two components.

Several techniques can be used to implement secure distance-estimation components. However, this investigation suggested the use of a distance-bounding approach as a promising solution for ALWadHA. The chapter discussed the basics and the characteristics of a distance-bounding approach. It evaluated and compared several secure distance-bounding protocols, using different metrics. This discussion showed that distance-bounding protocols can be chosen that will perform well in a WSN environment and add minimal overheads to ALWadHA. Moreover, using a distance-bounding method will assist ALWadHA to accomplish several design objectives. Distance bounding can be done within an ad-hoc, mobile environment with any number of nodes and network topology, which does not introduce any conflict with the self-organising design objective targeted by ALWadHA. Distance bounding is a simple protocol that does not need synchronised clocks. Distance bounding involves only two nodes to estimate the distance between them, therefore it assists ALWadHA in achieving the simplicity, energy-efficiency and localised-design objectives. Lastly, using a secure distance-bounding protocol that adheres to the "principles of secure distance bounding" as defined by Clulow *et al.* [89], will assist ALWadHA to implement a secure localisation system.

ALWadHA uses a robust position computation technique that tolerates the existence of malicious nodes. The attack resistance of the ALWadHA algorithm was investigated simulating two types of attack. In the first type an attacker pretended to be an honest reference node, while in the second type an attacker compromised a beacon node. In both types of attack, the attacker sent incorrect location information to neighbouring nodes to mislead them in their location estimations. Simulation results showed that ALWadHA is able to determine the location of nodes where malicious nodes exist without undermining accuracy. Moreover, increasing the advertised location erroneousness of these malicious nodes makes it easier to detect these nodes and prevent them from taking part in the position computation.