Design of a Secure Distance-Bounding Channel for RFID

G.P. Hancke

ISG Smart Card Centre
Royal Holloway, University of London, Egham, TW200EX
ghancke@ieee.org

Abstract

Distance bounding is often proposed as a countermeasure to relay attacks and distance fraud in RFID proximity identification systems. Although several distance-bounding protocols have been proposed the security of these proposals are dependent on the underlying communication channel. Conventional communication channels have been shown to be inappropriate for implementing distance bounding, as these channels introduce latency that can be exploited to obscure attempted attacks. Distance-bounding channels for RFID tokens have been proposed but have failed to address distance fraud or have not been practically implemented in an RFID environment. This paper describes a near-field, bit-exchange channel design that minimises latency and allows for more secure distance-bounding measurements, while still allowing for a resource-constrained prover. Results from a proof-of-concept implementation is also presented, which illustrates that a channel that is resistant to both relay attacks and distance fraud is feasible in current RFID systems.

Keywords: RFID, contactless smart card, distance bounding

1. Introduction

RFID technology is a prevalent method for implementing proximity-based services in systems that need to link a person or object to a specific location or operational context. These systems operate on the assumption that the token is in close proximity to the reader because of the physical limitations of the communication channel. Proximity, and the associated trust, is especially important in secure RFID systems implementing applications such as payment, identification and access control. For example, upon scanning a access card a door is unlocked or when presenting a contactless credit card the payment is authorised and goods handed to the customer present. However, using only the physical characteristic of the communication channel is not suitable for securely proving the proximity of a token. An attacker can use a proxy-token and proxy-reader to relay the communication between a legitimate reader and token over a greater distance than intended, or simply extend the range of his own device by modifying the communication channel parameters, e.g. amplify the response.

Distance-bounding protocols determine an upper bound for the physical distance between two communicating parties based on the round-trip-time of cryptographic challenge-response pairs. This distance can then be used as a cryptographic proof of proximity. Distance-bounding protocols are meant to detect any extra delay in the prover's expected response as an increase in the round-trip-time extends the distance bound. Distance-bounding protocols can therefore be an effective way to prevent relay attacks as the attacker introduces a delay, even if it is only the additional propagation time between the proxy devices. The attacker cannot decrease the round-trip-time

by preemptively transmitting his response as he is forced to wait for the challenge, and as a result the probability of a successful distance fraud is reduced. Time-of-flight distance-bounding protocols must be integrated into the physical layer of the communication channel to accurately determine the distance between the prover and verifier. This means that the security of the distance bound depends not only on the cryptographic protocol itself but also on the practical implementation and the physical attributes of the communication channel. The communication channel used for the exchange must, therefore, not introduce any timing tolerances that the attacker can exploit to circumvent the physical distance bound. Communication channels used in HF RFID systems utilise error-correction and packet delimiters that introduce latency, and the physical transceiver architectures used have also been shown to be vulnerable to late-commit and clocking attacks, which allows the attacker to hide the extra time needed to relay data [14, 5]. The conventional channels currently used in RFID systems are therefore seen to be unsuitable for implementing secure distance bounding.

If distance-bounding is to be implemented in RFID systems then the ability of the underlying channel to generate an accurate and secure time measurement must considered. Any latency that could be exploited by an attacker would need to be identified and the resultant effect on security taken into account. The ideal situation would be to implement new distance-bounding channels that minimise latency and provide strong security properties, while still allowing for a resource-constrained RFID token. The paper starts by providing background information as to distance-bounding protocols and the general design requirements of a channel. Next we discuss

existing proposals for distance-bounding communication channels within the RFID environment and then describe in detail how a channel design suitable for near-field RFID tokens could be implemented. Finally, we present some initial results from a proof-of-concept implementation.

2. Distance-bounding background

Distance-bounding protocols calculate an upper bound for the physical distance between two communicating parties, the prover and verifier, based on the Round-Trip-Time (RTT) of cryptographic challenge-response pairs. Distance-bounding was first proposed by Brands and Chaum [2] in 1994 and since then distance-bounding protocols have become popular for doing secure neighbour detection [25] and proving proximity in relative location systems, such as in RFID and contactless smart card applications [15]. There are three main types of attacks that are presented in literature with regards to distance-bounding protocols:

- **Relay Attack:** A fraudulent third party tries to convince the verifier that the prover is in close proximity. Both the verifier and the prover are honest and unaware of the attack. This was first described as 'mafia fraud' in [9] and is also known as a 'wormhole' attack in sensor-network literature [17].

- **Distance Fraud:** The prover is fraudulent and tries to convince the verifier that he is closer than is actually the case. Distance fraud was first discussed in [2], and is known as 'wormhole' attacks in the sensor network context [18].

- **Terrorist Attack:** The prover collaborates with an attacker, who wants to convince the verifier that the real prover is in close proximity. The prover's motivation could either be that he is fraudulent, or that he is honest but being coerced by the attacker. The "terrorist attack" was first described in [8].

In most cases, protocols aim to prevent distance fraud and relay attacks. The distance-bounding process detects the extra delay introduced by the attacker when relaying the data. Even if the attacker could achieve zero processing delay in relaying the data, the extra distance propagated would increase the time-of-flight between prover and verifier. Distance fraud is prevented if the distance-bounding protocol ensures that a prover must wait for a challenge to arrive before sending a response. In other words, the value of the response must depend on the challenge and the prover must not be able to pre-emptively send a response, which would decrease the round-trip time and influence the distance bound. In some cases proposals dealing with trusted devices, e.g. contactless smart cards, will assume that the user does not have access to the key material and that distance fraud is therefore unlikely, i.e. the trusted device acts as the prover and is by definition not fraudulent [23]. The user could attempt to use the trusted device as an oracle, although that would more closely resemble a relay attack. Preventing the "terrorist attack" altogether is difficult as it is generally an accepted condition of security that the participants do not reveal their secrets. Some protocols do, however, force the prover to reveal a valuable secret, e.g. long term key, if it reveals all possible responses to a third party, thereby discouraging the prover from collaborating with an attacker [3, 20, 32].

2.1. Importance of the exchange stage

Distance-bounding proposals can further be classified by how they implement different stages of the distance-bounding process. Most of these distance-bounding protocols consist of three basic stages:

- **Setup:** The verifier and prover prepare for the exchange stage.
- **Exchange:** Timed exchange of challenge and response data.
- **Verification:** The verifier ensures that the exchange step has been executed faithfully and can therefore use the RTT to calculate the distance.

The timed exchange stage is the most critical, and the challenge-response format must be designed in such a way that an accurate distance-bound can be calculated while also ensuring that the cryptographic proof of proximity remains secure. To prevent distance-fraud the response must depend on the challenge. However, the possible variations in the processing delay t_{d} when the prover calculates the response affects the round-trip time measurement, which in turn causes the distance bound to be unreliable. The processing delay should therefore be reduced to limit the inaccuracy in the time measurements. Some protocols suggest that the prover calculates its 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 R based on the challenge C received from the verifier during the exchange stage, so t_{d} is significantly reduced. These protocols generally propose that the exchanges are constructed from single bits and that the function

$C \rightarrow R$ is implemented using a simple XOR ($R = M \oplus C$) or table look-up operations($R = M^0$ if $C = 0$, $R = M^1$ if $C = 1$) to minimise variation in t_{d}.

In protocols using pre-commitment, e.g. [2, 3], the prover prepares possible responses during the setup stage. For example, the verifier generates a random challenge bit string, $C = (C_1, C_2, \ldots, C_l)$, while the prover generates a response string, $M = (M_1, M_2, \ldots, M_l)$. The prover commits to M, e.g. by transmitting a collision-resistant message authentication code $h(K, M)$. The verifier then sends one C_i after another, which the prover receives as C'_i. It then instantly replies with a bit $R_i = C'_i \oplus M_i$, which is calculated by XORing each received challenge bit with the corresponding bit of M. Finally the prover reveals M and authenticates C', i.e. prover sends $MAC(C')_K$ to verifier.

In protocols using pre-computation, e.g. [13, 23, 27], the prover and the verifier calculates 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 K in order to calculate two n-bit response strings M^0 and M^1. If the prover receives challenge C_i it will respond with from the bit string indicated by the value of the challenge, $R = M_i^{C_1}$. Since the verifier also knows M^0 and M^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, thereby decreasing the data that needs to be transmitted.

A third distance-bounding approach is *timed authentication protocols*, the simplest form of ToF-based distance-bounding, where the basic idea is to

execute a conventional challenge-response authentication protocol under a very tight time-out constraint, e.g. [28, 30, 34]. For example, a verifier V transmits a random n-bit nonce $N_V \in_{\mathrm{R}} \{0,1\}^n$ to the prover P, who replies with a message-authentication code $h(K, N_V)$, where h is a keyed pseudo-random function and K is a shared secret key. 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. This results in possible inaccuracies in the round-trip time measurement. For example, a smart card could take 100 ms to compute a public-key signature and a 1% (1 ms) processing time variation could cause a 333 km error in the distance estimate. Furthermore, the long exchange strings introduce latency that an attacker can exploit. In general, this approach is not seen as a secure distance-bounding method [5].

3. Distance-bounding channels

Time-of-flight distance-bounding protocols are dependent on time measurements made at the physical layer of the communication channel to accurately calculate the distance between the prover and verifier. This means that the security of the distance bound depends not only on the cryptographic protocol itself but also on the practical implementation and the physical attributes of the communication channel executing the exchange stage. Conventional communication channels are designed for reliable data transfer. As a result, these channels feature redundancy and timing tolerances to prevent bit errors. Such latency introduces uncertainty into the distance-bounding measurement and can be exploited by an attacker not adhering to the com-

munication channel 'rules' to gain a time advantage. The time gained by the attacker can be used to obscure the extra time he introduces during a relay attack, or if the response reaches the verifier sooner than expected the attacker will be bounded within a smaller distance, thereby committing a successful distance fraud. Clulow et al. [5] showed how an attacker can gain a timing advantage by exploiting the time allowed by the verifier for the transmission of redundant data, such as framing and error correction, at the packet level of the communication layer. Hancke et al. [14] also demonstrated how the attacker can achieve similar timing benefits at the physical level, i.e. by exploiting timing tolerance in the coding and modulation stages of RF transceivers typically used in RFID systems. Systems planning to use distance-bounding protocols must, therefore, implement special low-latency channels.

3.1. Channel Design Requirements

A distance-bounding channel provides the verifier with a timed measurement, from which to determine the distance bound. This measurement must be as accurate as possible and should not include any additional latency that can compromise the security of the protocol. To protect against possible attacks at the physical and packet layer the designer of a distance-bounding protocol should optimize the choice of communication medium and transmission format according to the following principles [5]:

- Use a communication medium with a propagation speed that approaches the physical limit for propagating information through a vacuum.
- Use a communication format in which the recipient can instantly react

on 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.

- Minimize the length of the symbol used to represent each single bit or, if working with a baseband signal, the verifier should sample as early as possible during the bit period and base his decoding decision on the value of this single sample. This minimises the time that the attacker can potentially gain.

An additional communication channel, which shortens the bit period and allows for single bit exchanges is therefore required to obtain a useful distance bound. A practical implementation, however, must also take into consideration the limitations of the hardware and the operating environment. Some key issues influencing the design are:

- **Hardware constraints:** We assume that the verifier, e.g. the RFID reader, has more resources compared to the prover, e.g. the RFID token. Complex operations, such as variable delay lines and the generation of high-frequency sampling and synchronization clocks, should therefore be implemented by the verifier. For a reliable distance bound the prover's hardware must have a predictable processing delay t_{d}. The prover's system clock is derived from the received HF carrier, which means that it is not trusted. As a result, the prover should implement asynchronous circuitry that functions independently of the system clock.

- **Synchronisation:** The prover and verifier require a synchronization signal to exchange pulses and provide a timing reference. For example, the channel proposal described in Section 4.3 suggests that the prover and verifier trigger their operations on a zero-crossing of the carrier transmitted by the verifier. Using the carrier for loose synchronization is a possibility, although it might not be feasible for a bit exchange to occur on every zero crossing, i.e. this would require a low-resource 13.56 MHz token to transmit a bit every 36 ns. The prover and the verifier could, however, use the carrier to generate a lower-frequency synchronization signal. Other possibilities are the transmission of a preceding timing pulse, followed by the data pulse, or switching the carrier on and off.

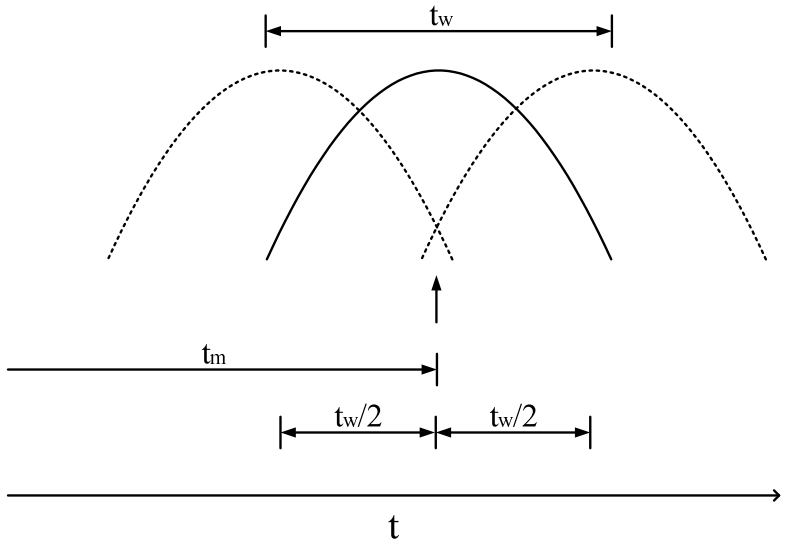


Figure 1: The effect of sample timing and pulse width on the distance bound.

- **Timing and distance resolution:** The timing accuracy influences

the resolution of the distance estimate. The simplest timing method is to start on the synchronization signal, wait for a fixed time t_{m}, and then sample once. If the correct response is sampled, the verifier can calculate the propagation time t_{p} from the round-trip time $t_{\mathrm{m}} = 2{\cdot}t_{\mathrm{p}}{+}t_{\mathrm{d}}$ and estimate the distance to the prover as $c \cdot (t_{\mathrm{m}} - t_{\mathrm{d}})/2$. It should be noted that this estimate only bounds the distance to the prover as the width of the response pulse t_{w} introduces some uncertainty. As shown in Figure 1, the actual distance to the prover can be anywhere between $c \cdot (t_{\mathrm{p}} - t_{\mathrm{w}}/4)$ and $c \cdot (t_{\mathrm{p}} + t_{\mathrm{w}}/4)$. This situation arises because the verifier does not know the exact time the response arrived, only that it sampled during the response. The verifier could have sampled right at the end, or the beginning, of the response, which means that t_{m} measured is $t_{\mathrm{w}}/2$ greater, or less, than the actual round-trip time. The resolution could be improved by sampling multiple times or decreasing t_{w}.

- **Bit representation:** The transmitted symbols that indicate the value of the challenge, or response bit, also affect the time measurement. In the ideal case, the symbols will be chosen in such a way that the verifier can distinguish between the different symbols, and also detect when no response is received. For example, an on-off keying scheme would make distance estimation more complicated since the verifier does not know whether it received a '0' or whether the response arrived before, or after, he sampled the channel. The two different symbols should also provide identical timing information. Symbols based on pulse-position schemes are therefore not suitable as these provide ambiguous timing, i.e. is

the pulse in the second time slot of the symbol, or is the pulse in the first time slot of a symbol transmitted later? Even if synchronization prevented this ambiguity, an attacker immediately knows that he needs to transmit a pulse in the second slot if the first is empty. If the attacker could relay this finding before the second slot started he would provide the correct response, within the required time, in approximately half of the exchanges.

- **Security of the distance bound:** We assume that a relay attacker will not cause any additional delay apart from the time it takes for the signal to propagate over the extra distance. This is the best theoretical attack scenario for the attacker as in practice additional delay could also be introduced by operations in the proxy token and reader. The attacker should not be able to decrease the processing time of the prover, which would enable the attacker to possibly obtain information about the response material from the legitimate prover before the verifier starts to transmit challenges. This means that the implementation should not depend on a clock derived from the data received or an external reader. As RFID tokens all derive their system clock from the reader, and currently contain no independent clock source, we suggest that the channel logic should be implemented using asynchronous logic that functions independently of the system clock. In theory, the prover's asynchronous logic and delay lines could be influenced, e.g. changing the temperature, although this would be difficult without having physical access to the prover's token, for example when covertly reading an RFID token in the victim's pocket. In the case of

distance fraud we assume that a dishonest prover can decrease t_d to zero, although the signal propagating over a longer distance will cause additional delay.

4. Proposed communication channels for distance-bounding

Published proposals for the implementation of RFID distance-bounding channels are currently confined to the HF RFID environment [13, 23, 27], although there are also some proposals for creating unforgeable RF channels that could be applied to prevent relay attacks, such as [7, 6].

4.1. Unforgeable channels

Several proposals attempt to construct unforgeable channels. If the verifier could reliably determine whether the source of the transmission is the legitimate prover. The basic principle in this case is that the proxy-prover will not be able to impersonate the real prover if he cannot exactly replicate the communication channel characteristics. For example, Alkassar, et al. [1] suggested that the use of channel-hopping radio could prevent relay attacks, as the attacker would find it difficult to track the prover's communication . This method, however does not provide for distance estimation apart from 'in communication range' and a dishonest prover is in position to commit distance fraud. The verifier can also try to uniquely identify the prover by using the physical characteristics of the channel. Rasmussen and Čapkun proposed that a verifier can construct a unique 'fingerprint' for each prover in a sensor network environment by using the attributes of the received RF signal [26] and this principle has been extended to RFID tokens by Danev *et al.* [6]. A proposal by DeJean and Kirovski [7] would allow a prover to identify itself by

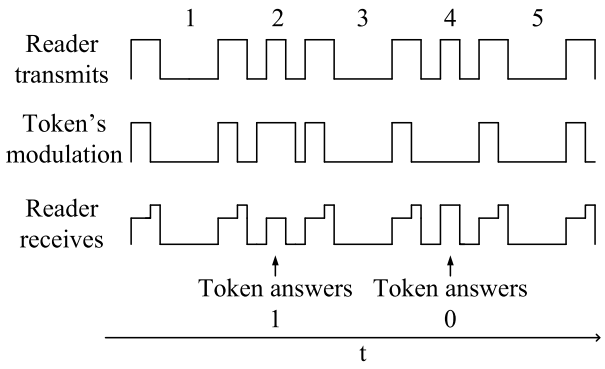
intentionally making its channel characteristics unique . This is achieved by placing a random constellation of conductive and/or dielectric objects within the token, which would alter the near-field response of a token when exposed to RF signals. Neither of these methods provide any accurate proximity information apart from 'in communication range', nor do they protect against a fraudulent prover. If the 'fingerprint' is strong, i.e. difficult to forge, this method could detect the use of a proxy device and therefore discourage either a relay or a terrorist fraud being executed.

Further proposals hide additional information within the transmitted data. In a scheme by Hu, et al. [17], geographical information, referred to as packet leashes, are added to transmitted data. This method, however, requires the verifier to know its location, which disqualifies it for two-party distance-bounding as it requires collaboration with additional parties. Kuhn [21] proposed that the prover transmits a hidden 'watermark' along with the data, which is subsequently revealed, so that the verifier can retroactively check whether the data it received was transmitted by the prover. These methods do discourage relay attacks, as the attacker would not be able to modify the source location information or recreate the watermark when relaying the data. Both these methods, however, are vulnerable if the prover acts dishonestly. The dishonest prover could reveal the watermark to an accomplice, i.e. commit terrorist fraud, or simply send fraudulent information to the verifier,i.e. commit distance fraud ,either by misrepresenting geographical information or by influencing the time-of-arrival GPS distance estimate by sending an incorrect transmission time.

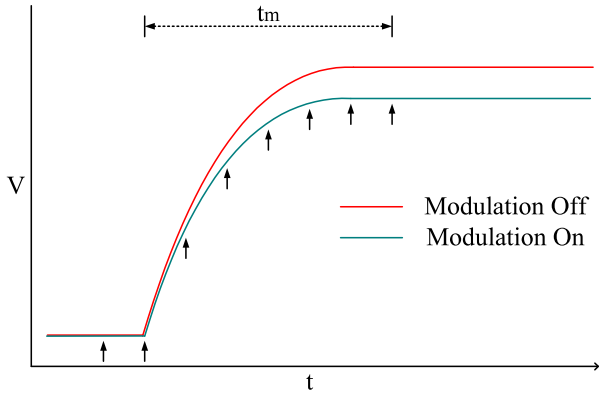
4.2. Carrier sampling

There are two proposals for a distance-bounding channel where the verifier directly samples the modulated carrier. This means that the verifier could determine the prover's response without performing traditional demodulation and decoding, thus reducing communication channel latency. Both proposals are tailored to the HF RFID environment and depend on the load modulation process, which allows the token to amplitude modulate the carrier transmitted by the reader.

In the proposal by Munilla, et al. [23], the reader transmits a periodic sequence of pulses that are 100% ASK modulated onto the carrier. The pulses act as synchronization bits with the periods in between, when the carrier is off, referred to as slots. In some slots, the reader will switch on the carrier for a short period of time to indicate that it wants a response. An example of how successive bits are exchanged is shown in Figure 2(a). The token knows when to expect these requests and preemptively switches its impedance to indicate the answer. When the reader then switches on the carrier, the envelope of the signal rises immediately to a level that indicates the token's answer state. To determine the tokens's response, the reader continuously samples the envelope of the carrier until it finishes rising and becomes stable, e.g. two successive samples are equal. Once the envelope reaches this steady state the verifier checks the amplitude level to see whether load modulation is on or off. The time it takes until the two levels can be reliably distinguished, and the difference between the envelope amplitude for the two states, depend on the distance between the token and the reader. The reader times from the point when it switches on the carrier, and the



(a) Example of a bit exchange sequence.

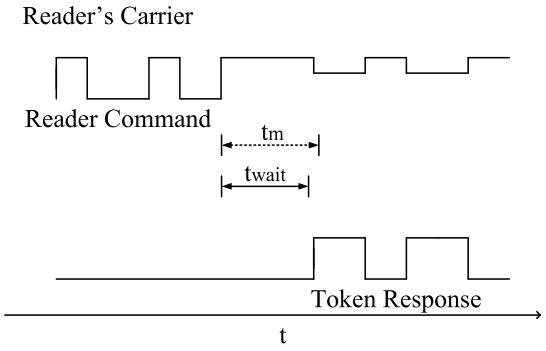


(b) Timing of a single bit exchange.

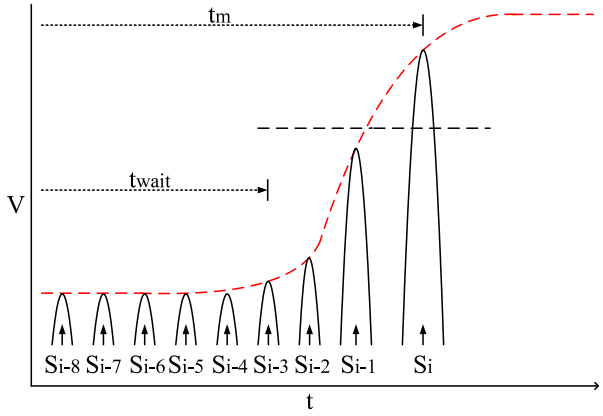
Figure 2: The void-challenge distance-bounding channel.

envelope starts rising, until the token's response is determined, as shown in Figure 2(b). The authors state that the timing resolution of the channel is less than 1 µs. Since the token knows when the reader will issue a challenge, and is in fact expected to respond preemptively, i.e. it knows the response before receiving the challenge, this implementation does not allow for the prevention of distance fraud. The token would also need to be protected against a proxy-reader transmitting a weak carrier, which appears to the

token to be 'off', to probe the state of the load early. Another practical requirement that must be taken into account it that the carrier, which is switched off regularly to indicate the slots, is also the source of power for the token so the duration of the slots must be chosen accordingly.



(a) Example of a challenge-response sequence.



(b) Timing the start of the token's response.

Figure 3: Accurately timing the token's response by early modulation detection.

The proposal by Reid, et al. assumes that the token will reply after a fixed time t_{wait} [27]. In practice the token waits for a pre-determined number of cycles of the 13.56 MHz carrier, which would synchronise its re-

sponse to an accuracy of 1/13.56 MHz = 75 ns. The reader times from the end of its command to the moment that the response is detected, with the distance-bounding time measurement then taken as $t_{\mathrm{m}} - t_{\mathrm{wait}}$. An example of a challenge-response sequence is shown in Figure 3(a). The time at which the response is received is measured using a special detector that tries to determine the exact moment that the amplitude of the carrier is first modulated. This involves sampling the peaks of the HF carrier and comparing the latest sample to a threshold calculated from the eight previous samples. The resolution of the system is once again dependent on the distance between the token and the reader, with the authors stating that a 300 ns resolution was obtained when the token and the reader were 4–5 cm apart.

This channel could be vulnerable to distance fraud if the prover does not wait t_{wait} and transmits its response pre-emptively. The authors also state their assumption that the token is protected against overclocking, where an attacker tries to speed up the prover's processing time by increasing the externally supplied clock, and that the RF carrier operates within the ± 7 kHz tolerance specified by the relevant standard. However, this does not seem to be a valid assumption for tokens currently available. It has been practically demonstrated that contactless tokens can be made to send the response 10-30 μsearlier if the attacker influences the system clock by transmitting a carrier of higher frequency [14].

4.3. Ultra-wideband pulses

Hancke and Kuhn proposed a crude ultra-wideband channel for near-field RFID systems [13] that aims to reduce latency that the attacker can exploit while also allowing for accurate distance measurement. Making the bit period

as short as possible would limit the time potentially gained by an attacker but this requirement might compromise the reliability of the channel. If this is the case the distance-bounding protocol using this channel would need to allow for bit errors during the timed exchange stage. The reader and the token use the 13.56 MHz carrier for loose synchronization and the response is sent immediately after receiving the challenge using an asynchronous circuit which limits the effect of overclocking attacks. A challenge and response bit exchange occurs on each rising edge, where the signal changes from a low to a high amplitude, of the carrier, as shown in Figure 4(a).

The reader starts timing on the zero-crossing of the carrier, waits for t_{t}, and then transmits the challenge bit C_i. The token also waits for the zero-crossing of the carrier before it starts the sampling process. The sampling time t_{s} is fixed and dependent on the token's hardware implementation. The reader tries to ensure that the token samples C'_i correctly by adjusting delay $t_{\mathrm{t}} \approx t_{\mathrm{s}}$, essentially aligning the challenge bit period with the time the token samples. By varying the delay t_{t} during the first few values of i until a delay has been found that results in the correct response bits, the reader can adjust itself automatically to any component tolerances and instabilities that may affect the exact sampling time in the token. After a brief processing delay t_{d} the prover transmits a response bit R_i. After time t_m the reader samples the channel to determine R'_i.

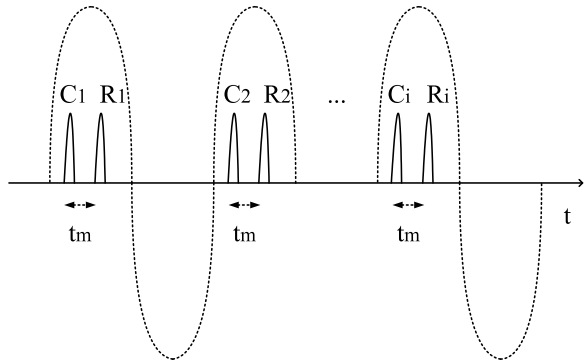
In a very simple implementation, the reader is equipped with two adjustable delay circuits. The first delay t_{t} is used to position the challenge bit such that the token has the best chance to sample the challenge bit correctly. The second delay element t_{m} is used to time the moment after the zero-

crossing when the reader has the best chance to sample the incoming R_i bit correctly. It is up to the reader to repeat the protocol and try different values for t_{t} and t_{m} until R'_i matches the expected result well. The total number of bits exchanged n should be chosen large enough such that enough bits remain to satisfy the security requirement of the challenge-response phase after the delay-element adjustment phase. In a more sophisticated implementation the verifier samples a response for multiple delays that are of interest, and then searches in the recorded results for the lowest value t_{m} with an acceptable response. In neither case are high clock-frequency circuits, or precise reference frequencies, needed in the token. The timing of a single bit exchange is shown in Figure 4(b).

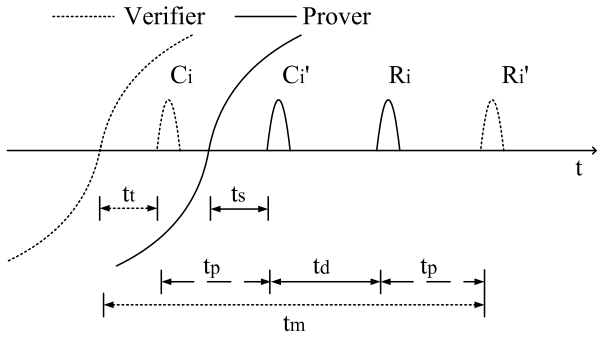
The propagation time t_p can then be calculated by the reader as follows: $t_{\mathrm{p}} = (t_{\mathrm{m}} - t_{\mathrm{t}} - t_{\mathrm{d}})/2$. As with all the other channels presented here, a prover could commit distance fraud if it managed to decrease the expected t_{d}. Minimizing t_{d} limits the amount of time the attacker could gain. Drimer and Murdoch recently used a similar technique to implement distance bounding to prevent relay attacks against contact smart cards [10].

5. Practical design for a near-field bit-exchange channel

The UWB channel approach described in Section 4.3 is the most suitable for implementing secure distance-bounding when taking into account the potential timing accuracy and the fact that it limits both relay attacks and distance fraud. Ultra-Wideband (UWB) communication has already been successfully implemented in active RFID systems for localisation [33], and also proposed for passive UHF RFID systems [35, 24]. UWB communication



(a) Example of a bit-exchange sequence.



(b) Timing of a single bit exchange.

Figure 4: Ultra-wideband pulse exchange using carrier synchronisation.

implements carrierless data transmission. This simplifies the transmitter and receiver architectures, which potentially results in small, inexpensive and low-power hardware. The allocated bandwidth allows for a large channel capacity, while the communication can be made resilient to noise and multi-path effects [12]. UWB channels can also be used for distance measurements with resolution of 30 cm or less [11]. Existing UWB channels are, however, primarily intended for data transfer, which means that they implement packet formatting to synchronise communication and increase reliability, which is

not ideal for secure distance bounding applications. Tippenhauer and Capkun describe a distance-bounding implementation using off-the-shelf UWB equipment but this approach is more suited to wireless sensor network applications [31].

The implementation of an elementary UWB distance-bounding channel for RFID tokens therefore remains a technical challenge. The initial channel proposal [13] did include some ideas on how to implement the channel, e.g. variable delay line to aid in synchronization and shift registers for storing R^0 and R^1, but practical issues such as generating synchronisation from the shared carrier, defining an accurate system timing model, building pulse transmitter/receiver architectures and implementing suitable response lookup and delay line circuits were not addressed. In this section, we describe a more detailed practical implementation for this channel, as shown in Figure 5. The symbol definitions and corresponding timing diagram are shown in Figures 6 and 7, while relevant variable definitions are shown in Table 1. The described channel implementation permits multiple exchanges of single challenge and response bits, and is tailored to protocols that prescribe this format of timed exchange, such as [13, 4, 3, 27, 32, 20, 19] and the single-bit exchange variant of [22].

The verifier starts by generating a suitable synchronization signal, e.g. an RF carrier. This signal is used to generate a clear timing reference, such as a rising edge, by both the verifier ($Sync\_V$), and the prover ($Sync\_P$). If the challenge bit C is '1' the transmitter will wait t_{t} after the timing reference and then instruct the transmitter to send a pulse. The time taken by the transmitter to generate and drive the pulse signal onto the antenna

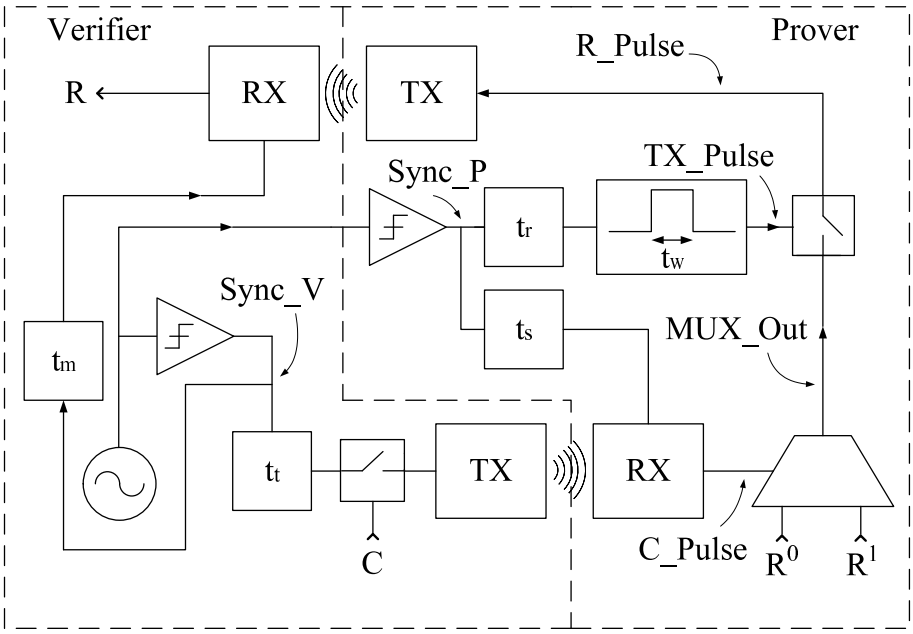


Figure 5: Overview of a wideband pulse distance-bounding system.

is t_{TX}. The transmitter's output is represented in the timing diagram by the signal V_{TX}. The prover waits for t_{s} after the timing reference signal and then samples its receiver's output P_{RX} to produce the signal *C_Pulse*. The prover allows for the time t_{RX} its receiver takes to detect and amplify the pulse signal. *C_Pulse* is then used to switch a 2-to-1 multiplexer to select one of R^0 or R^1 as a response. The output of a 2-1 multiplexer is one of two input signals determined by the state of a switching input, and can therefore be used to choose from the two possible response by using the value of the challenge as the switching input. The output of the multiplexer *MUX_Out* cannot be connected directly to the prover's transmitter. Due to the physical characteristics of a multiplexer, i.e. one path requires an additional NOT function, the time it takes to change the output when the input changes

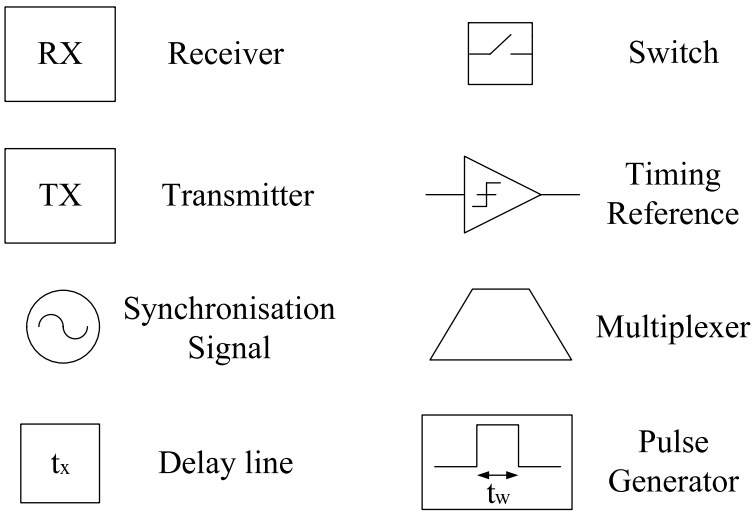


Figure 6: Symbol definitions.

from '1' to '0' might be different from when the input changes from '0' to '1'. This would make t_{d} dependent on C and complicate the round-trip timing. Connecting the multiplexer output directly to the transmitter also introduces a security vulnerability since it is possible for both R^0 and R^1 to appear on the multiplexer's output during each exchange cycle of the protocol. For example, if the previous *C_Pulse* was '1' then the multiplexer will output R^1 to start with, and then changes to R^0 if the current challenge is '0'. This might allow a proxy-prover to read out both response sequences, which would allow a proxy-prover to execute the protocol with 100% success. From a practical perspective the transmitter also requires a rising edge to generate a response pulse, which would not happen if two consecutive responses are '1'. These issues are solved by adding a switch to output the current response to the transmitter once the output of the multiplexer is stable. The prover waits for time t_{r} before generating a pulse *TX_Pulse* which switches the

multiplexer output through to the receiver. The 'off' state of the switch is '0' so a reponse of '1' will also generate the required rising edge to drive the transmitter. The falling edge, where the signal changes from a high to a low amplitude, of *TX_Pulse* can also be used to clock in the next values of R^0 and R^1. The width of the resultant *R_Pulse* can be adjusted by changing the width t_{w} of *TX_Pulse*. The response is transmitted back to the verifier, once again incurring some transmitting, receiving and propagation delay in the prover's transmitter and verifier's receiver, as indicated by signals P_{TX} and V_{RX} . Finally, the verifier samples its receiver's output t_{m} after the timing reference to determine response R.

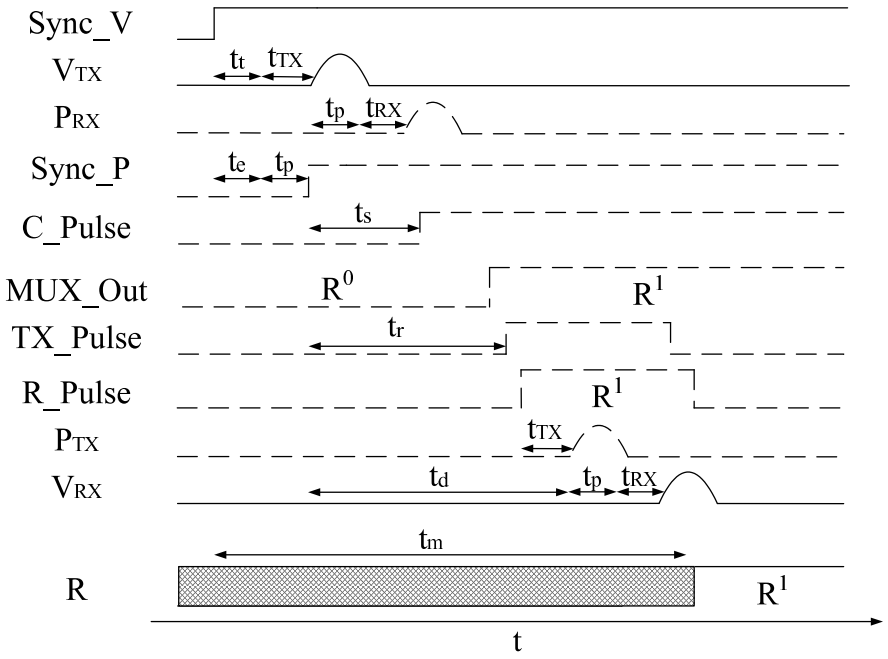


Figure 7: System timing diagram.

The delay lines in the prover, t_{r} and t_{s}, are fixed and it is assumed that these, along with t_{d}, are predictable and known by the verifier. The

t_{m}	Round Trip Time measurement taken by Verifier
Sync_V	Synchronisation signal generated by Verifier
Sync_P	Synchronisation signal received by Prover
t_{t}	Time delay after *Sync_V* that the Verifier transmits challenge C
t_{s}	Time delay after *Sync_P* that the Prover samples challenge C
t_{e}	Synchronisation time error between *Sync_V* and *Sync_P*
t_{p}	Propagation time between Verifier and Prover
t_{r}	The Prover's response delay, i.e the time between *Sync_P* and calculating the response
t_{d}	The Prover's processing time, i.e. the time between *Sync_P* and transmitting response R
t_{w}	The width of the challenge/response pulse
t_{TX}	Time delay in transmitter circuit of prover and verifier
t_{RX}	Time delay in receiver circuit of prover and verifier

Table 1: Variable definitions

delays in the verifier, t_{s} and t_{m}, are adjustable and should be configured during the early stages of the protocol. The expected round-trip time is $t_{\mathrm{m}} \approx 2 \cdot t_{\mathrm{p}} + t_{\mathrm{d}} + t_{\mathrm{e}} + t_{\mathrm{RX}}$, where t_{e} is the synchronization error between the time references in the prover and verifier. In near-field communication systems the expected propagation time is almost neglible, i.e. 10 cm is 300 ps, and the verifier should have an estimate for $\mathrm{t_d}$ and t_{RX}, so it can make a reasonable first approximation of t_{m}. At this stage the prover keeps responding with a '1', while the verifier increases t_{m} to the minimum value at which this response can be sampled reliably. Once the verifier completes his

adjustment it transmits a nonce to the prover, which indicates that the first calibration step is complete, and is also used to calculate R^1 and R^0. During the initial cryptographic exchanges the verifier will adjust t_t to ensure that the prover samples C reliably. It is assumed that the prover cannot provide the correct response if it does not receive the correct C. The verifier sets t_t to the maximum value at which the majority of the responses it receives are correct, which should results in $t_\mathrm{t} \approx t_\mathrm{e}$. t_s should have been set to $t_\mathrm{TX} + t_\mathrm{RX} + t_\mathrm{p}$, with $t_\mathrm{p} \to 0$, so if the prover cannot sample the challenge this is due to an error in the synchronization of the timing references. The uncertainty introduced by this synchronisation error can be accommodated in the distance bound as it is effectively measured by the value of t_t. Once t_m and t_t have been set, the verifier can evaluate the distance bound. Ideally, the verifier wants to calculate $d = c \cdot t_\mathrm{p}$ but he only has an approximate round-trip time measurement t_m, which includes various processing delays in addition to $2 \cdot t_\mathrm{p}$. It should also be kept in mind that the timing resolution of t_m is dependent on the pulse width and sampling method used, as described in the constraints on page 11. If the response is sampled once, as shown in Figure 1, the actual round-trip can be anywhere between $t_\mathrm{m} - t_\mathrm{w}/2$ and $t_\mathrm{m} + t_\mathrm{w}/2$. As the *upper* bound, i.e. the furthest distance the prover can be from the verifier, is required the verifier should allow for the latter time in his distance bound calculation. From Figure 7 and taking into account that $t_\mathrm{t} \approx t_\mathrm{e}$ the round-trip time can be defined as follows:

$$t_\mathrm{m} \approx 2 \cdot t_\mathrm{p} + t_\mathrm{d} + t_\mathrm{t} + t_\mathrm{RX} + t_\mathrm{w}/2 \tag{1}$$

When the distance bound is calculated two attacks should be considered: a relay attack by a third party attacker and distance fraud by a dishonest

prover. For the relay attack we assume that the third party attacker, who controls a proxy-prover and proxy-verifier, introduces no extra delay except the additional propagation time of the relayed communication. A third party attacker cannot decrease delay times, such as t_{r} and t_{s}, in the real prover as these are determined by asynchronous delay lines. There are arguments that delay lines can be sped up by cooling the token [10]. This requires that the prover's hardware is controlled by the attacker as it is not practically feasible to remotely cool the circuit, e.g. it is difficult to covertly spray liquid nitrogen on a victim's purse or pocket. As a result, the attacker cannot make the real prover answer earlier than expected and the verifier can store an accurate estimate of the processing time t_{d} of the real prover. If it is assumed that the attacker has the necessary control of the token to heavily influence its internal functioning then the security analysis of the system should rather use the distance-bound calculation taking into account the possibility of distance fraud. Similarly, we also assume that the attacker cannot manipulate the delay introduced by the receiver circuits in the prover and verifier. As a result the verifier also has an accurate estimate of the delay t_{RX} caused by his receiver circuitry. The verifier can then subtract the estimates of t_{d} and t_{RX} along with t_{t}, which was set earlier by the verifier, from t_{m} to get an accurate approximation of t_{p}. In the case of a relay attack the distance bound d can therefore be calculated as follows:

$$d = c \cdot (t_{\mathrm{m}} - t_{\mathrm{t}} - t_{\mathrm{d}} - t_{\mathrm{RX}})/2 \tag{2}$$

Substituting the approximation for t_{m} given in Equation 1 this simplifies to

$$d = c \cdot (t_{\mathrm{p}} + t_{\mathrm{w}}/4) \tag{3}$$

A dishonest prover can decrease his processing time t_{d} by implementing new receiver, transmitter and response look-up circuits that introduce less delay. Although it is probably not practically feasible, we consider the worst case scenario and assume that the dishonest prover could reduce his processing time t_{d} to zero, i.e. reply instantaneously without any delay. If $t_{\mathrm{d}} = 0$ is substituted in Equation 1 the round-trip time measurement taken by the verifier becomes

$$t_{\mathrm{m}} \approx 2 \cdot t_{\mathrm{p}} + t_{\mathrm{t}} + t_{\mathrm{RX}} + t_{\mathrm{w}}/2 \tag{4}$$

Substituting this modified estimate of t_{m} into Equation 2 results in

$$d = c \cdot ((t_{\mathrm{p}} + t_{\mathrm{w}}/4) - t_{\mathrm{d}}/2) \tag{5}$$

If the distance bound is calculated in this way, a fraudulent prover could therefore be up to $d = c \cdot t_{\mathrm{d}}/2$ further away and still appear within acceptable distance. For example, if the processing time is decreased by 12 ns then the dishonest prover could afford an additional 6 ns in the propagation time t_{p}, which means he can be approximately 2 m further from the verifier without his fraud being detected. A verifier allowing for distance fraud should therefore take into account that his expected value for t_{d} is not correct. As a result, the verifier does not use this estimate when calculating the distance bound. This approach results in a valid distance bound for both honest and dishonest provers, even though the actual distance to a honest prover is probably less since he does introduce processing delay t_{d}. It should, however, be remembered that the distance bound is not the actual distance to the prover. It is simply an upper bound on the distance between the verifier and the prover, i.e. the prover is within d, and if allowing for a dishonest

prover the worst case should be taken into account. If the verifier allows for possible distance fraud then the distance bound d should be calculated as

$$d = c \cdot (t_{\mathrm{m}} - t_{\mathrm{t}} - t_{\mathrm{RX}})/2 \tag{6}$$

Substituting the approximation for t_{m} given in Equation 1 this simplifies to:

$$d = c \cdot (t_{\mathrm{p}} + t_{\mathrm{d}}/2 + t_{\mathrm{w}}/4) \tag{7}$$

6. Proof-of-concept implementation

We performed some practical experiments focusing on three key functions: synchronisation, pulse transmission and asynchronous lookup. The electronic circuits described here are by no means the only solution, nor are they meant as reference designs for a commercial distance-bounding channel. Our goal was only to show that it is possible to implement these functions within a near-field environment using limited resources. The basic building blocks, e.g. delay lines, are based on reference designs, which we obtained from electronic circuit sources like [16].

6.1. Synchronisation

Earlier it was suggested that the carrier could be used for providing a time reference point. This was, however, not as straight forward as it appeared at first. The transmitted and the received versions of carrier are not exactly synchronised. This is to be expected since the prover token is in effect a complex rather then real load, which will cause a phase shift in the carrier on the token's side. In HF RFID systems the reader and token are inductively coupled, which means that the token is within the magnetic field (H-field)

of the reader. In other words, an alternating RF signal transmitted by the reader will induce a current to flow in the wound antenna coil of the token, which would result in a similar signal being generated on the antenna of the token. To optimise the transfer of power from the reader the token contains a resonant circuit constructed by connecting a capacitor (C) in parallel to the inductor (L) formed by the antenna coils. The resonant circuit amplifies the carrier signal if it corresponds to the circuit's resonant frequency, e.g. in HF systems the resonant frequency $f = (1/2\pi\sqrt{LC}) = 13.56$MHz. When dealing with alternating signals in the presence if of inductors and capacitors the behaviour of the signal is modeled using complex mathematics. In this case the effect of these components on the signal, i.e. the impedance, is represented by both an imaginary and real component, which essentially means that these components can influence both the amplitude of the signal and the phase of the signal. The phase of the signal is a representation of how the signal shifts in time in comparison to the original signal, i.e. how much it is delayed.

The phase shift between the reader's version of the carrier and the token's version depends on the capacitive and inductive components used in both the reader and token. Even if readers and tokens are similar in design there is a potential difference in component values. Readers usually contain an adjustable capacitor in the transmission circuit and there is a degree of variability in fixed-value components present in the token. To practically investigate this concept we build several resonant circuits attached to card-sized antenna coils. We then induced a carrier signal in these circuits using a selection of readers. In each case we compared the signal transmitted by the

reader to the signal available on the token's antenna. In general the error was minimal ($t_{\mathrm{e}} < 10$ ns), with some exceptions ($t_{\mathrm{e}} \approx 25$ ns). The synchronisation error should, however, compensated for in the distance bound calculation by the value of t_{t}.

If the unmodified carrier is used for synchronisation, the period of the HF carrier places a restriction on the maximum round trip time that can be measured. In a 13.56 MHz RFID system, t_{m} would need to be less than 73 ns since the prover should ideally be allowed to respond before being issued with another challenge. This also places restrictions on t_{d}, t_{RX} and t_{t}, which complicates the channel further. The prover and the verifier should therefore use a lower frequency synchronization signal derived from the carrier. This could be done by implementing a frequency divider, i.e. binary counter, in both the prover and verifier. At the start of the protocol the counters would need to be synchronised. This could possibly be done by resetting them if the carrier is switched off. Once the carrier is switched on again, both counters start on the first rising edge of the carrier. Their output will obviously also be influenced by any carrier synchronisation error.

Transmitting a pulse while the carrier is on also complicates the receiver architecture, as the receiver needs to distinguish a relatively weak pulse in the presence of a strong carrier. An alternative solution would be to switch the carrier off and use an envelope detector in both the prover and the verifier to perform synchronisation. When the verifier wishes to transmit a challenge it switches off the carrier and both parties generate a time reference when the falling edge of the envelope passes through a set threshold. The difference in the amplitudes of the transmitted and the received carrier can be

compensated for by making the threshold a percentage of the average carrier amplitude, rather than a fixed reference value. The synchronisation accuracy is effected by the component tolerances and phase of the carrier in the prover and verifier. Once again, these factors are compensated for in the distance bound calculation by the value of t_{t}.

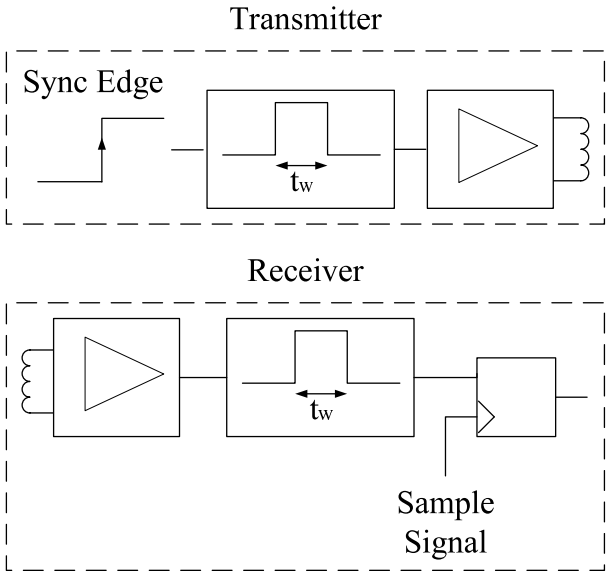


Figure 8: The transmitter and receiver architectures.

6.1.1. Pulse transmission

The simplest solution is to implement an on-off keyed pulse channel, i.e. pulse is '1', nothing is a '0'. The only drawback of this channel is that there will be no verifiable timing information when the response is '0', as nothing will be transmitted. This would make this channel unsuitable for doing distance estimation using a single exchange. However, taking into account that the channel is meant to be used in a distance-bounding protocol performing multiple exchanges, and that the verifier is only interested in the

answer at time t_{m}, this method is sufficient. The transmitter and receiver architectures are shown in Figure 8.

The transmitter generates a pulse of width t_{w} when a rising edge occurs on its input. The pulse generator can be constructed in a similar way as described earlier. The pulse is then transmitted using a buffer capable of sourcing enough current to drive the signal onto a small loop antenna. This buffer could be an amplifier, or simply multiple logic gates with the same output. The receiver also uses a small loop antenna, connected to an amplifier. This architecture works well when the pulse is not transmitted at the same time as the carrier. The first rising edge of the received signal is then used to generate a 'new' pulse of width t_{w}. This provides the sampler with a clean input pulse, irrespective of the quality of the received signal. The sampler is a D-type flip-flop (74HC74), clocked by an external sampling signal. We consider a design using a loop antenna as this technology is predominantly used in near-field tokens already and the process and cost of adding an additional loop should be minimal and based on existing designs.

6.2. Asynchronous response circuit

The prover needs to implement two delay lines, a multiplexer and a response switch. We assume that R^0 and R^1 are pre-loaded into two shift registers, and that they are both clocked onto the multiplexer inputs well before C is received. Alternatively, the next (R^0, R^1) pair can also be computed by iterating a pseudo random-bit generator well before C is received. A circuit that could be used to choose, and clock out the required response is shown in Figure 9. The entire circuit can be implemented using discrete logic and passive components.

The multiplexer is implemented using NOT (74HC04) and NAND (74HC00) logic gates, and outputs R^x where x is the value of the input. The input, *C_Pulse*, is the value sampled by the receiver after delay t_s. A delay line is implemented using NOT logic gates and an RC-network, which decrease the rise, or fall, time of the applied edge. Since the signal rises, or falls, more slowly it reaches the $L \leftrightarrow H$ threshold of the next gate later, which results in the output being a delayed version of the original signal edge. A similar delay line is implemented for t_r. The delayed edge then triggers a pulse generator, built using NOT gates and an RC-network. If a rising edge is applied to the RC-network the output immediately goes to a high level, but as the capacitor charges the level drops down again. If this charge-discharge cycle is applied to the input of a logic gate a pulse is generated at the output. In this case the positive pulse *TX_Pulse* is used to switch the output of the multiplexer to the transmitter. The switch is implemented using an AND (74HC08) logic gate.

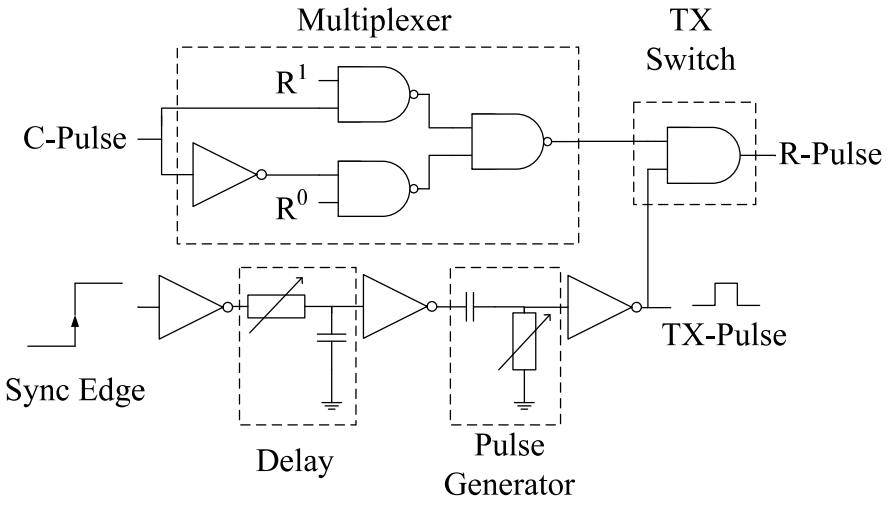


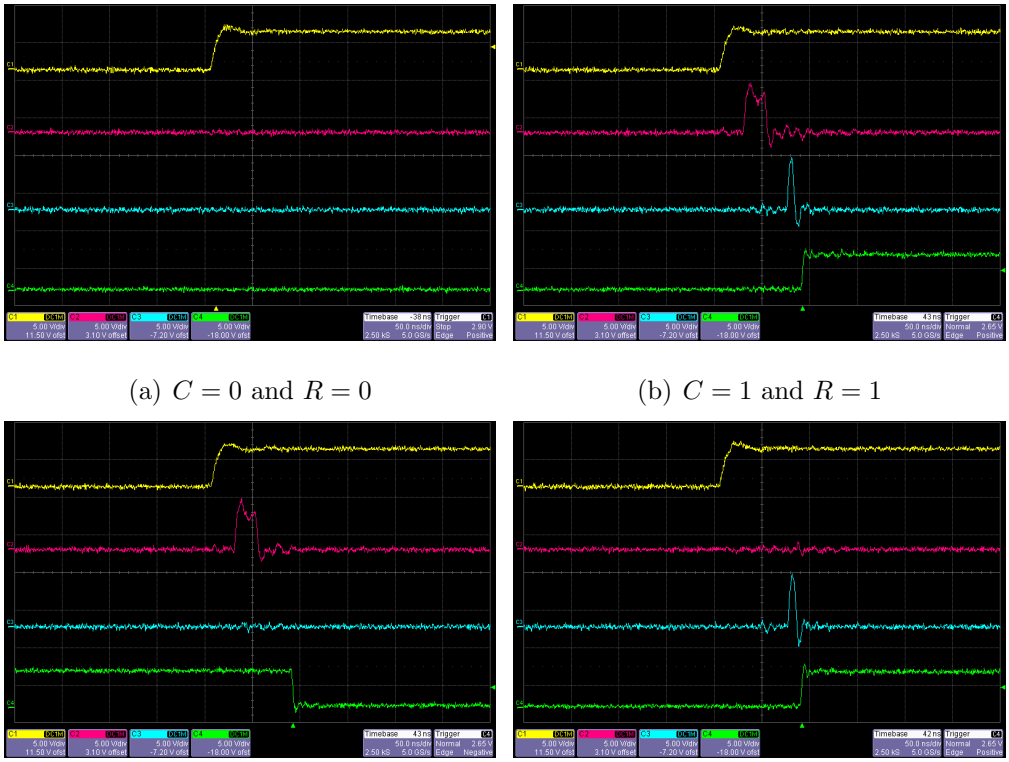
Figure 9: The prover's asynchronous circuit for choosing the response.

6.3. Experimental results

We implemented a simplified version of the proposed distance-bounding channel design. The experimental hardware implementation included:

- Two TX–RX links implementing a duplex channel between the prover and verifier.
- Delay lines t_{t}, t_{m}, t_{r} and t_{s}.
- An asynchronous lookup circuit.

C, R^0 and R^1 was implemented with user controlled switches and the delay lines t_{m} and t_{t} was adjusted manually. For the experiment a 100 kHz clock signal, connected to both the prover and verifier, was used for synchronisation. This is not realistic, but allowed us to simulate synchronization errors by delaying the signal to the prover. Some initial results are shown in Figure 10. The width of the transmitted pulses was approximately 10 ns, although in the figure the prover's receiver generates a wider *C_Pulse*. This was to show that a narrow pulse can be stretched to provide the prover with a greater time window to sample correctly. This would simplify the initial setup when adjusting t_{t} as it shortens the time taken before the prover samples at the right time and starts responding with the correct responses. The security of the distance bound is not affected since t_{t} will still be adjusted to the maximum value, i.e. where the prover samples just after the pulse edge. If t_{e} was zero the experimental system had a round trip time t_{m} of approximately 75 ns. t_{RX} of the verifier was approximately 14 ns.



(a) $C = 0$ and $R = 0$

(b) $C = 1$ and $R = 1$

(c) $C = 1$ and $R = 0$

(d) $C = 0$ and $R = 1$

Figure 10: Different bit exchanges on the experimental distance-bounding channel. The top trace shows the synchronising edge, the second trace shows the output of the prover's receiver, the third trace shows the output of the verifier's receiver and the bottom trace shows the response sampled by the verifier after t_{m}. The initial state, the left hand side, of the bottom trace shows the response sampled during the previous exchange.

Given that the propagation time in a near-field system is negligible, $t_{\mathrm{p}} \approx 300$ ps while $t_{\mathrm{m}} \approx 75$ ns, t_{p} effectively tends toward zero. As a result Equation 3 on page 29 and Equation 7 on 31 can be modified to $c \cdot (t_{\mathrm{w}}/4)$ and $c \cdot (t_{\mathrm{d}}/2 + t_{\mathrm{w}}/4)$ respectively. If the measured round-trip time t_{m} in this channel implementation is 75 ns the upper bound on the distance is calculated as follows:

- An honest prover is within $d = c \cdot (2.5 \text{ ns}) \approx 1$ m of the verifier, even if a third party attacker executed a relay attack.
- If the prover is not trusted and in a position to execute distance fraud, the verifier concludes that the prover is within $d = c \cdot (60 \text{ ns}/2 + 2.5) \approx 11$ m.

The accuracy and integrity of a distance estimate based on a single challenge-response exchange is crucial to a distance-bounding protocol. However, the number of challenge-response exchanges that can be performed is also relevant to the security of the protocol. The probability that an attacker would be able to successfully execute a distance fraud or a relay attack is dependent on the number of challenge-response exchanges made. For example, the success probability of an attacker committing a distance fraud or a relay attack against the protocol proposed by Brands and Chaum [2] is $(1/2)^n$, where n is the number of challenge-response bits exchanged. These probabilities vary from protocol to protocol, e.g. the relay attack probability for the protocol proposed by Hancke and Kuhn is only $(3/4)^n$ [13]. The exact reasons for some protocol having different attack resistance are outside the scope of this paper, but it is clear that some protocols would therefore require

more exchanges to take place to achieve an equivalent level of security. In HF RFID systems, the protocols might be required to complete within a limited transaction time due to operational constraints. One example is a transport system where an e-ticket needs to be efficiently validated as to maximise the throughput of travelers during busy periods. The practical transaction limit for such a transaction has been stated to be 350 ms [29]. In our proof-of-concept implementation we did not try to optimise the repetition rate of the challenge-response exchanges. However, the channel did reliably exchange a challenge-response pair every 10 µs, as the practical exchange repetition rate was essentially the same as the frequency of the experimental synchronization signal (100 kHz). This means that a distance-bounding protocol with $n = 128$ would complete the exchange stage in little more than 1 ms (1.28 ms). A distance-bounding protocol is only one of several operations that would need to be executed during a transaction, but the time needed by the channel to exchange a relatively large amount of challenge-response bits is not significant. For example, the time taken to exchange $n = 128$ would comprise less than 0.5% of the total time allowed for the transport transaction mentioned above.

7. Conclusion

Vulnerabilities in the underlying communication channel undermine the security of distance-bounding protocols. Conventional communication channels are therefore unsuitable for implementing distance-bounding protocols. Special consideration must be given to the communication channel used for distance bounding, and the designer must include any potential vulnerabili-

ties into the final distance bound estimate. Ideally, distance bounding should be implemented using a specially designed channel. Current relay-resistant channel designs are based on promising ideas but are not yet a complete solution. For example, unforgeable channels provide no distance information, and might be vulnerable to distance fraud, carrier sampling techniques presented for the RFID environment fail to protect against distance fraud and UWB channels suitable for distance-bounding have not been been practically implemented in the RFID environment.

A channel exchanging single short pulses would offer the best distance-bounding characteristics but would require specially designed hardware. We describe how a bit-exchange channel could be implemented in the near-field environment using an improvised wideband-pulse channel. The design takes into account the identified system considerations and it is shown how key functions of this design could be achieved using simple hardware. Some practical results are also presented from an experimental bit-exchange channel based on this design. The described channel implementation permits multiple exchanges of single challenge and response bits, and could therefore be used to implement a number of distance-bounding protocols in literature, such as [13, 4, 3, 27, 32, 20, 19] and the single-bit exchange variant of [22]. The experimental hardware bounds $d \approx 1$ m for an honest prover and $d \approx 11$ m for a fraudulent prover. This compares favourably to existing proposals, which do not consider distance fraud and bounds 1 µs $\overset{\Delta}{\approx}$ 300 m and 300 ns $\overset{\Delta}{\approx}$ 100 m, while also requiring modifications to the reader and token. We hope that this work will encourage further research on implementing communication channels for distance-bounding protocols and the

development of a prototype contactless token that implements these channels to execute distance-bounding protocols. The antenna design, bit-error rate, exchange repetition rate and the operating range for active and passive tokens should also be investigated further.

[1] A. Alkassar, C. Stuble and A. Sadeghi. *Secure Object Identification: or Solving the Chess Grandmaster Problem.* Proceedings of New Security Paradigms Workshop, pp 77–85, 2003.

[2] S. Brands and D. Chaum. *Distance Bounding Protocols.* Advances in Cryptology, EUROCYPT '93, Springer-Verlag LNCS 765, pp 344–359, May 1993.

[3] L. Bussard and W. Bagga. *Distance-Bounding Proof of Knowledge to Avoid Real-Time Attacks.* Proceedings of IFIP International Information Security Conference, pp 223–238, June 2005.

[4] S. Čapkun, L. Buttyán and J. Hubaux. *SECTOR: secure tracking of node encounter in multi-hop wireless networks*, Proceedings of ACM Workshop on Security in Ad Hoc and Sensor Networks (SASN), ACM Press, 2003.

[5] J. Clulow, G.P. Hancke, M.G. Kuhn, T. Moore. *So Near and Yet So Far: Distance-Bounding Attacks in Wireless Networks.* European Workshop on Security and Privacy in Ad-Hoc and Sensor Networks (ESAS), Springer-Verlag LNCS 4357, pp 83–97, September 2006.

[6] B. Danev, T.S. Heydt-Benjamin and Srdjan Capkun. *Physical-layer*

Identification of RFID Devices, Proceedings of USENIX Security Symposium, 2009.

[7] G. DeJean and D. Kirovski. *RF-DNA: Radio-Frequency Certificates of Authenticity*, 9th International Workshop Cryptographic Hardware and Embedded Systems (CHES 2007), Springer-Verlag LNCS 4727, pp 346–363, September 2007.

[8] Y. Desmedt. *Major security problems with the 'unforgeable' (Feige)-Fiat-Shamir proofs of identity and how to overcome them.* Proceedings of SecuriCom 88, pp 15-17, 1988.

[9] Y. Desmedt, C. Goutier and S. Bengio. *Special Uses and Abuses of the Fiat-Shamir Passport Protocol.* Advances in Cryptology (CRYPTO), Springer-Verlag LNCS 293, pp 21, 1987.

[10] S. Drimer and S.J. Murdoch. *Keep Your Enemies Close: Distance Bounding Against Smartcard Relay Attacks.* Proceeding USENIX Security Symposium, pp 87–102, August 2007.

[11] R.J. Fontana, E. Richley and J. Barney. *Commercialization of an ultra wideband precision asset location system.* Proceedings of IEEE Conference on Ultra Wideband Systems and Technologies, pp 369–373, November 2003 2004.

[12] M. Ghavami, L.B. Michael and R. Kohno. *Ultra wideband signals and systems in communication engineering.* Wiley, 2004.

[13] G.P Hancke and M.G. Kuhn. *An RFID distance bounding protocol.* Proceedings of IEEE/CreateNet SecureComm, pp 67–73, September 2005.

[14] G.P Hancke and M.G. Kuhn. *Attacks on Time-of-Flight Distance Bounding Channels.* Proceedings of the First ACM Conference on Wireless Network Security (WISEC'08), pp194–202, March 2008.

[15] G.P. Hancke, K. Mayes, and K. Markantonakis. *Confidence in Smart Token Proximity: Relay Attacks Revisited.* Elsevier Computers& Security, June 2009.

[16] P. Horowitz and W. Hill. *The Art of Electronics*, 2nd Edition, Cambridge University Press, 1989.

[17] Y.C. Hu, A. Perrig and D.B. Johnson. *Packet leashes: A defense against wormhole attacks in wireless networks.* Proceedings of INFOCOM, pp 1976–1986, April 2003.

[18] Y.C. Hu, A. Perrig and D.B. Johnson. *Wormhole attacks in wireless networks.* IEEE Journal on Selected Areas in Communications (JSAC), pp 370–380, 2006.

[19] C.H. Kim and G. Avoine, *RFID distance bounding protocol with mixed challenges to prevent relay attacks.* Proceedings of Conference on Cryptology and Network Security, LNCS 5888, pp 119-133, 2009.

[20] C.H. Kim, G. Avoine, F. Koeune, F.X. Standaert and O. Pereira. *The Swiss-Knife RFID Distance Bounding Protocol*, Proceedings of Information Security and Cryptology, pp 98-115, December 2008.

[21] M.G. Kuhn. *An asymmetric security mechanism for navigation signals.* 6th Information Hiding Workshop, Springer-Verlag LNCS 3200, pp 239–252, May 2004.

[22] C. Meadows, P. Syverson and L. Chang. *Towards More Efficient Distance Bounding Protocols for Use in Sensor Networks*. Proceedings of Securecomm and Workshops, pp 1–5, August 2006.

[23] J Munilla and A. Peinado. *Distance bounding protocols for RFID enhanced by using void challenges and analysis in noisy channels*. Wireless Communications and Mobile Computing, Vol. 8, pp 1227–1232, 2008.

[24] Y. Niu, M.B. Nejad, H. Tenhunen and L.R. Zheng. *Design of a Digital Baseband Processor for UWB Transceiver on RFID Tag*. Proceedings of Advanced Information Networking and Applications Workshop, pp 358–361, May 2007.

[25] P. Papadimitratos, M. Poturalski, P. Schaller, P. Lafourcade, D. Basin, S. Čapkun and J-P Hubaux. *Secure Neighborhood Discovery: A Fundamental Element for Mobile Ad Hoc Networking*. IEEE Communications Magazine, February 2008.

[26] K.B. Rasmussen and S. Čapkun. *Implications of Radio Fingerprinting on the Security of Sensor Networks*. Proceedings of IEEE SecureComm, 2007.

[27] J. Reid, J.M.G Nieto, T. Tang and B. Senadji. *Detecting Relay Attacks with Timing-Based Protocols*. Proceeding 2nd ACM Symposium on Information, Computer and Communications Security, pp 204–213, March 2007.

[28] N. Sastry, U. Shankar and D. Wagner. *Secure verification of location*

claims. Proceedings of ACM Workshop on Wireless Security, pp 1–10, 2003.

[29] C. Sheehan. *Why legacy payment schemes are inadequate for mass transit*. Contactless News, March 2010. Cited 28 March 2010. `http://www.contactlessnews.com/2010/03/23/why-legacy-payment-schemes-are-inadequate-for-mass-transit`

[30] C. Tang and D. O. Wu. *Distance-Bounding Based Defense Against Relay Attacks in Wireless Networks*. IEEE Transactions on Wireless Communications, Vol. 6, No. 11, pp 4071–4078, November 2007.

[31] N.O. Tippenhauer and S. Capkun. *ID-based Secure Distance Bounding and Localization*. Proceedings of European Symposium on Research in Computer Security (ESORICS), 2009.

[32] Y.J. Tu, S. Piramuthu. *RFID distance bounding protocols*. Proceedings of First International EURASIP Workshop on RFID Technology, September 2007.

[33] Ubisense. White papers and datasheets, 2003–2006. `http://www.ubisense.net`

[34] B. Walters and E. Felten. *Proving the location of tamper resistant devices*. February 2003. `ftp://ftp.cs.princeton.edu/techreports/2003/667.pdf`

[35] P. Yu, P. Schaumont and D. Ha. *Securing RFID with Ultra-Wideband Modulation*. Proceedings of Workshop on RFID Security (RFIDSec),pp 27–39, July 2006.