Using Proximity as an Authentication Mechanism For Protocol-Parties

Nico Petzendorfer
School of Computation, Information and Technology
Technische Universität München
Garching, Germany
nico.petzendorfer@tum.de

***Abstract*—This document presents possible protocols for using distance as a factor for authentication, where other approaches like confirming passwords out-of-band are inconvenient. It shows how time-of-flight based distance-bounding can be used to obtain a physically secure upper bound on distance. Furthermore, it demonstrates how this distance-bound can be used for both authenticating known devices and pairing unknown devices. Additionally, an existing protocol using ultrasound is analyzed. It becomes clear that devices need to either have fast processors, special hardware for acceleration, or the protocol needs to be weakened, which introduces possible security issues.**

***Index Terms*—distance-bounding, time-of-flight, pairing, authentication**

I. Introduction

There exist many methods of authentication, which can be more or less useful depending on the use-case. One simple method is using some form of hardware-token: A small device that holds a shared key and automatically communicates with other devices to authenticate and trigger certain actions (e.g., opening a door).

However, there exist attacks on this form of authentication, like mafia fraud, terrorist fraud or distance fraud. To remedy these attack possibilities, such devices could additionally use proximity as a second factor of authentication: The authenticating hardware token needs to be nearby, so that it can be assumed the person holding the token really does want to execute the authenticated action or can at least recognize if the authentication is being used by an attacker.

There are multiple ways to check if the devices are in proximity: The most simple possibility is to just rely on the used communication-channel being location limited. For example with radio communication the signal power decreases with increasing distance from the sender until it eventually is so low that it cannot be discerned from noise. In this case, the devices are unable to communicate with one another and therefore unable to authenticate. Still, this approach can easily be attacked by simply relaying the signals.

Another possibility is for the devices to rely on local signal variations. Nearby devices receive similar changes in signals while farther away multiple effects influence the changes in signal. Because these effects are rather unpredictable, it is unlikely an attacker can convince the devices to authenticate. While this method can determine proximity (to some accuracy), it is not flexible, as the maximum allowed distance where devices are considered close can not be chosen.

This paper suggests using time-of-flight based distance-bounding. Distance-bounding is the process of determining an upper bound on the distance between two devices. This can be done by measuring the time a message takes to travel (fly) between the two devices. The resulting value is not just a binary decision of nearby or not, but a maximum distance the devices are apart. We show that this approach to determine proximity is physically immune to the specified attacks, though it does require fast calculations. To remedy this, we point out what parts of the protocol can be relaxed, as well as what risks of new attacks these relaxations bring. promising!

Furthermore, this idea of using distance as authentication can be extended to pairing devices which do not have any shared secrets: Current implementations for secure pairing of devices require the users to confirm the pairing out-of-band, e.g., by entering a password on one device, which is shown on the other device to be paired with. While this can be used to securely pair devices, it is not convenient to enter a password when pairing or might not be feasible because of limited I/O-capabilities. When the devices simply check their distance and only pair if they are close to each other, the user does not need to do any additional action besides initiating the pairing. For an attacker to pair, the attacker would need to be inside the allowed range and could therefore be easily spotted by the user. However, the same computational requirements in regard to calculation speed are also needed for this use-case. We show once again how these requirements could be relaxed and what security issues are introduced. For this, we analyze security issues in the key establishment protocol using time-of-flight based distance-bounding in [1].

II. Related Work

A multitude of protocols to authenticate devices based on distance exist. Here is an overview of some of them.

other overviews ?

A. Amigo and Poster

Both Amigo [2] and Poster [3] measure a shared radio environment to check for proximity. Because of reflections and

multipath effects,[1] only close devices are able to read the same signatures from the shared radio environment.

With Amigo, successful pairing was achieved with devices $5\,\mathrm{cm}$ apart and attackers with a range or over $3\,\mathrm{m}$ were successfully blocked. However, attackers or other nodes at $1\,\mathrm{m}$ were able to authenticate most of the time,[2] though generating local entropy[3] (e.g., by waving hands around the pairing devices) would prevent attackers at $1\,\mathrm{m}$ from successfully authenticating.

Poster achieved successful pairing with devices $1\,\mathrm{m}$ apart while attackers over $3\,\mathrm{m}$ away were blocked.

B. *Ensemble*

Ensemble [4] is similar, but instead of only the two pairing devices measuring signal strength of transmission, it relies on additional readings from trusted devices the user carries (like smartwatches, smartphones, or tablets). Each of these trusted devices measure the signal strength variations and cast a vote depending on the similarity in signal strength of the two pairing devices.

Successful pairing was achieved with devices $10\,\mathrm{cm}$ apart most of the time, while attackers starting from $1\,\mathrm{m}$ were mostly blocked with false acceptances lowering with increasing distance. Ensemble already had acceptable results with only one additional device, though the acceptance rate of close devices and rejection rate of far devices increased with a growing number of devices. The surrounding environment also had some influence on the results, as reflections and noise level from other devices varied.

C. *Good neighbor and Wanda*

Good neighbor [5] also measures signal strength, but with two antennas on one device to compare signal falloff over a known distance. Because the signal falloff is not linear, a transmitter close to one of the antennas will have a large signal strength difference between the two antennas, while far away transmitters will have similar signal strength.

The presented prototype was always able to reject attackers more than $20\,\mathrm{cm}$ away while accepting closer devices in over 90%.

Wanda [6] is an extension to the Good neighbor protocol, that additionally allows sending information in-band.

D. *Move2Auth and Shake well before use*

Move2Auth [7] was designed for securely authenticating a smart-device to a WiFi network, where it is not possible to enter a SSID and password directly. Where current devices would temporarily create a WiFi network the user can then log in to and enter the SSID and password, Move2Auth suggests pairing a smartphone to the smart-device based on distance and then transmitting the required data over the created secure channel. This prevents possible impersonation attacks, where an attacker would create a separate WiFi network with the name of the smart-device to have the user connect to and enter the network credentials in a fake form, therefore transmitting them to the attacker and granting them full access to the home network.

Move2Auth achieves this secure pairing by having the user either move his smartphone towards and away from the smart-device or rotate it and measuring the resulting signal strength variations.

Close devices would always authenticate correctly while distant devices would always fail to authenticate. An attacker manipulating his transmission strength almost never managed to authenticate himself and even knowing the randomized gestures would only increase the successful percentage to about $8\,\%$.

Shake Well Before Use [8] is similar, but requires both devices to move together synchronously and then using the devices' accelerometer data. They present two different protocols:

- ShaVe (*shaking for verification*) performs an initial key exchange, then compares the sensor data to check if the devices are close to each other (i.e., shaken in the same way), and accepts the key only if this is the case.
- ShaCK (*shaking to construct a key*) matches features extracted from the sensor data to generate a common key.

Both protocols managed to have no false positives while having about ten percent false negatives. Due to the way these protocols work, an attackers distance to the pairing devices does not affect his chances of a successful attack (as long as they have line of sight to try and match the movements).

E. *ProxiMate*

ProxiMate [9] uses a similar approach to ShaCK by generating keys from local information. Instead of using common movement, it generates the keys based on the shared radio environment using either amplitude or phase. For a used transmission wavelength[4] of λ, the devices were able to successfully reject attackers more than 0.4λ away, while legitimate devices should be no more than 0.1λ away when using amplitude and 0.05λ when using phase. With amplitude-based key extraction, an attacker could increase the chances of getting the same key by controlling the radio source.[5]

F. *Key Establishment Using Secure Distance-Bounding Protocols*

The proposed key establishment protocol in [1] uses time-of-flight based distance-bounding to securely pair nearby devices. The pairing devices communicate over ultrasound. This protocol will be analyzed further in section IV-C.

[1] Due to reflections, a signal can take multiple paths (e.g., the direct path and/or an indirect path where the signal bounces off a surface) from a sender to a receiver. This results in a fading pattern where the signal amplitude and phase vary depending on location.

[2] Over 60%, depending on the parameters used even up to 100%.

[3] Signal variations that mostly only influence the signals near the pairing devices and can therefore not easily be measured by an attacker.

[4] When using WiFi or Bluetooth at $2.4\,\mathrm{GHz}$, the wavelength λ would be about $12\,\mathrm{cm}$, though other frequencies can be used as well.

[5] As ProxiMate relies on a public radio source (e.g., TV or the activity causes by other devices' WiFi communication), the derived keys also depend on these signals. If an attacker can influence these signals, they can also influence the derived key. Additionally, the attacker could simply transmit on the used frequency with a high power, therefore drowning out the original radio source and having the devices solely depend on his transmission.

III. Background

In the following, we will introduce some basic concepts mentioned in this paper. Additional resources for further reading will be referenced as well.

A. Location limited channels

A location limited channel is a communication channel, where messages interchanged over that channel can only be received within a certain radius around a transmitter.

One reason could be, that the communication would require line-of-sight, meaning that objects like walls can block the signal transmission. An example for this includes trying to communicate with a flashlight.

Another reason could be signal falloff following the inverse square law: As the signal spreads and takes up more space the further away it gets, the energy of the signal is distributed over the larger space, therefore the local energy is lower. At some distance, the signal is then too weak to be received or accurately depicted from the noise floor.[6]

Location limited channels are generally vulnerable against multiple attacks, like an attacker relaying the signal, transmitting with high power, or having sensitive and/or directional receivers.

Examples include radio frequency, which follows the inverse square law, and can be blocked by objects depending on its frequency.

B. Distance-Bounding

Distance-bounding finds an upper bound for the distance between two devices. The device that wants to prove it is in a certain radius around the other is called the prover, while the other device that wants to verify the claim is called the verifier. There exist different possibilities of gaining such an upper bound. The communication parties can use a location limited channel, measure a shared environment like radio waves or acceleration, or measure time-of-flight of their messages (how long it takes for their messages to travel between them) [11].

C. Attacks

Depending on the method used for distance-bounding, various attacks may be possible with different goals. The most important ones are listed below:

1) Impersonation Fraud: Impersonation Fraud [12] is a class of attacks where an attacker can reliably pretend to be a valid prover whenever they want to. The verifier then does not recognize the attacker as such, and therefore believes they are communicating with a valid prover.

[6]This follows from the Shannon–Hartley theorem $C = W \log_2 \frac{P+N}{N}$ in [10]: The noise level N can be thought of as roughly constant (or at least not as varying with distance). The bandwidth W also remains constant throughout a transmission, as this is set in the physical specification of the protocol. When the signal power P then goes towards zero, the channel capacity C goes towards zero as well, meaning no information can be transmitted anymore.

2) Mafia Fraud: Mafia Fraud was initially described in [13]. It is a relay attack, wherein two attackers create a tunnel that relays messages between a prover and a verifier where the two authenticating parties would be out of range when using the location limited channel alone. One attacker will then act as a verifier to the real prover and the other attacker will then act as a prover to the real verifier. The two authenticating parties do not know this relaying is happening and think they are close to each other, as they can receive each other's signals.

Take for example your car key supporting keyless entry: Normally your car unlocks when your key is inside a specified radius around the car, determined by the location-limited nature of radio transmission. Now say you are multiple kilometers away (so definetly outside the range of the location limited channel) from your car and have your car key with you. A pair of attackers, one near you and the other near your car, could then simply forward all messages between the car and the key, therefore virtually bringing them close together and unlocking your car.

3) Terrorist Fraud: Terrorist Fraud [14] is also a relay attack, except the prover actively works together with an attacker: The prover is outside the range of the verifier, but wants to appear like they are near. Therefore, they proxy their communication through an attacker that is near the verifier. Since no shared secrets are leaked to the attacker, the attacker will not be able to impersonate the prover in the future. Still, the evil prover successfully helped the attacker in passing the authentication checks.

4) Distance Fraud: Distance fraud [11] is when a malicious prover manages to convince the verifier a proof with a lower distance than in reality is correct without the help of any other parties. This could stem from a predictable challenge, where the prover sends out his response before receiving all (or any) parts of the challenge from the verifier. Another reason for distance fraud when using location limited channels could be a sensitive and high power transceiver.

5) Distance Hijacking: Distance Hijacking [15] is when a dishonest prover is outside the allowed range of a verifier, but hijacks the communication between the verifier and an honest prover in range of the verifier. The dishonest prover lets the honest prover perform a distance-bounding routine with the verifier, but then acts as if the communication actually happened between the dishonest prover and the verifier by sending their identity. If the protocol is not built to detect such attacks, the verifier then has a valid distance-bound and wrongly attributes this to the dishonest prover, therefore authenticating this out of range prover.

IV. Authentication using distance

Devices can use multiple possibilities to authenticate themselves. An additional factor to prevent relay attacks can be distance: The devices check whether they are in proximity and only authenticate if they are.

In the following, we will first look into distance-bounding by using time-of-flight. Then we will see how this can be applied

for authenticating devices with shared secrets and later how this can be used for pairing previously unknown devices.

A. Physically secure distance-bounding using time-of-flight

There are many possibilities of gaining a distance bound, as can be seen in section II. However, some of these methods do not only depend on the strength of the chosen keys and random numbers, but instead rely on external factors such as radio noise, signal strength or signal reflections, that could possibly be controlled, monitored or simulated by an attacker.

A way to gain a distance bound, where it is physically impossible to report a smaller distance,[7,8] is to measure the time it takes for a signal to get to the other device and back. This is called time-of-flight measuring.[9]

In the following, we will look how these protocols can work (in practice), by examining how a device $\mathcal{A}$ can get an upper bound on its distance to a device $\mathcal{B}$.

A straightforward example for such a protocol would be a simple ping, as can be seen in Figure 2a: $\mathcal{A}$ sends an echo request to $\mathcal{B}$, who immediately answers with an echo reply.[10] Then, $\mathcal{A}$ can check the ping-time (the time between sending the echo request and receiving the echo response) and calculate an upper bound on distance using the following formula:

$$d \leq d_{max} = \frac{1}{2} * c * t\,.$$

Here, t is the measured ping-time and c is the speed of light in vacuum. The factor of $\frac{1}{2}$ is due to the fact that the signal traveled the distance twice (once from $\mathcal{A}$ to $\mathcal{B}$, then back from $\mathcal{B}$ to $\mathcal{A}$). It is easy to see how this upper bound is correct: If the prover would hold back the message or take time processing the message, the measured time

$$t' = t_{travel} + t_{process} \geq t_{travel} = t$$

would be longer than the time-of-flight. Therefore, the real distance can only be lower than the calculated distance-bound, as

$$d'_{max} = \frac{1}{2} * c * t' \geq \frac{1}{2} * c * t = d\,.$$

Additionally, due to the speed of light in a vacuum being always constant, no matter the relative movement of the two devices,[11] and the impossibility for information to travel faster than the speed of light,[12] the speed v of the signal is always at most c. The measured time would then be

$$t' = \frac{d}{v} \geq \frac{d}{c} = t$$

[7]According to the current knowledge in physics.

[8]Except if random numbers or generated keys are too weak.

[9]This also has applications in various other fields for measuring distance.

[10]This method also allows establishing distance-bounds of devices on the internet using the ICMP/ICMPv6 protocol, although these messages will not be routed directly, are not traveling through vacuum and are not prioritized, which is why they tend to be too large.

[11]Resulting from the special theory of relativity in [16].

[12]At least according to our current state of physics.

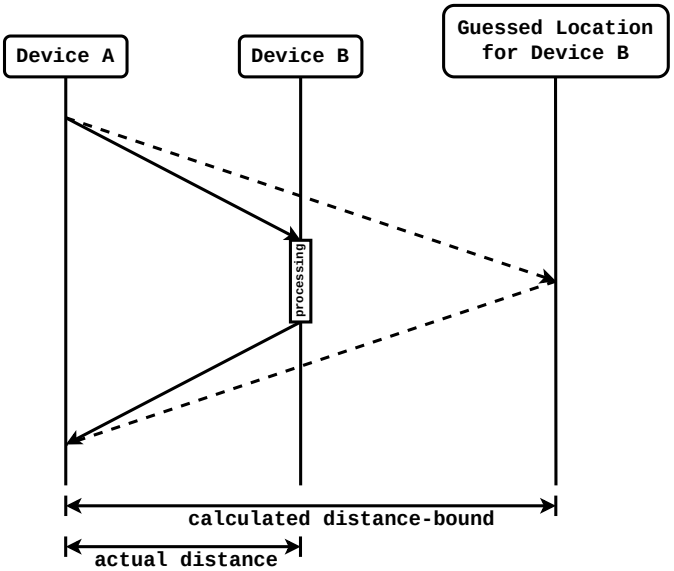


Figure 1: The calculated distance bound for a device if the signal travels slower than the speed of light and the device takes some time processing. The guessed signal path is represented by the dashed line. Time is modeled vertically and distance is modeled horizontally. Therefore, if an arrow is slanted steeper, the signal travel speed is slower, as more time is needed for the same distance.

which again leads to the real distance only being lower than the calculated distance-bound, as

$$d'_{max} = \frac{1}{2} * c * t' \geq \frac{1}{2} * c * t = d\,.$$

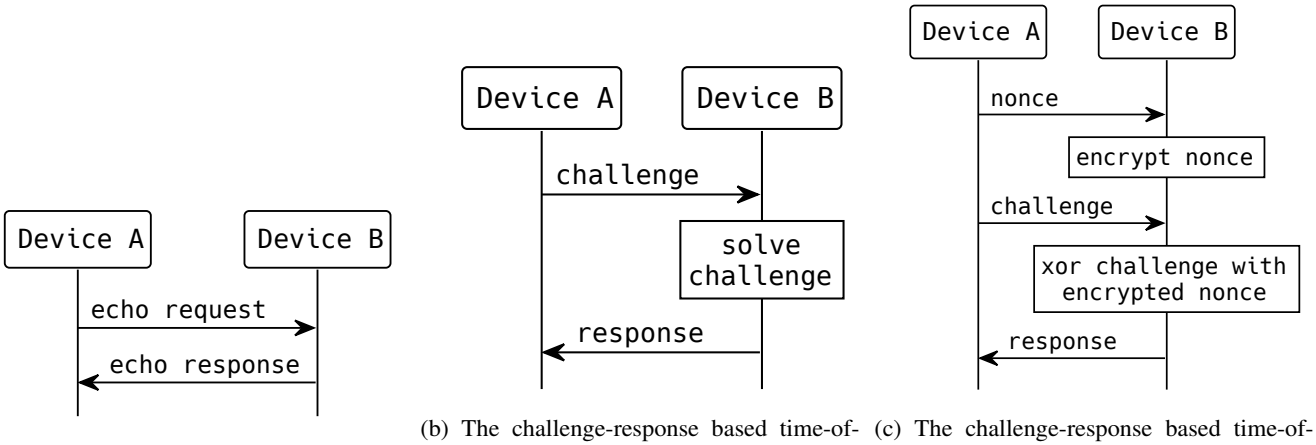
An illustration of this can be seen in Figure 1.

However, it mostly is not sufficient to send unauthenticated messages, as their authenticity cannot be known in a shared/public medium, therefore essentially enabling distance hijacking. Additionally, $\mathcal{B}$ could continually send out echo replies without any request, which would lead to $\mathcal{A}$ getting a wrong measured time of $t = 0\,\mathrm{s}$ and therefore lead to distance fraud.

An easy addition to remedy this issue is to use a challenge-response procedure: The two devices have shared keys.[13] Instead of just sending a plain echo request, $\mathcal{A}$ sends a challenge to $\mathcal{B}$, such as a nonce to encrypt. $\mathcal{B}$ solves the challenge and sends the response back to $\mathcal{A}$, who has measured the time of this exchange and now checks for a correctly solved challenge (such as a correctly encrypted nonce) to ensure authenticity of $\mathcal{B}$. The protocol is then immune against distance hijacking, as a dishonest prover can not claim the response as his, because it depends on the secret key of the honest prover which differs from the key of the dishonest prover. This protocol can be seen in Figure 2b.

As we have shown, this additional processing time for solving the challenge only increases the distance-bound, but does not

[13]These can be pre-shared keys or keys obtained from a key-exchange.



(a) The simple time-of-flight protocol.

(b) The challenge-response based time-of-flight protocol.

(c) The challenge-response based time-of-flight protocol using xor.

Figure 2: The presented time-of-flight protocols.

invalidate it, except when the challenge is predictable. Even with a truly random (and therefore unpredictable) number, if the first response-bits can already be calculated from only parts of the challenge, $\mathcal{B}$ could start sending the response before having received the full challenge, therefore giving a too small distance-bound. This means, the first response-bit needs to depend on the last challenge bit, which is given with most cryptographic hashing- and encryption-functions due to their diffusion-property described in [17].

However, this approach can be problematic, as the distance-bound can become unusable. The time taken for sending the messages can be compensated by $\mathcal{A}$ simply measuring only the time taken between the last bit of the challenge and the first bit of the response. Still, there is an additional distance error d_{proc} due to the time t_{proc} taken for processing, which can be calculated as follows:[14]

$$d_{proc} = \frac{1}{2} * c * t_{proc}\,.$$

Additionally, as $f = \frac{1}{t}$, we get the distance error per clock cycle d_{clk} with frequency f by substituting time in the previous formula for frequency from

$$d_{clk} = \frac{1}{2} * \frac{c}{f}\,.$$

As a rough estimate, we can use a CPU with a clock speed of 3 GHz and get an additional distance of

$$d_{clk,3\,\mathrm{GHz}} \approx \frac{3 \times 10^8\,\mathrm{m\,s^{-1}}}{2 * 3 \times 10^9\,\mathrm{Hz}} = 0.05\,\mathrm{m} = 5\,\mathrm{cm}$$

per clock cycle. As encryption usually takes multiple clock cycles, this can lead to unusably large distance-bounds.

This means, the challenge-solving needs to be able to be calculated quickly, while still being secure. An inspiration can be taken form the CTR mode of block-ciphers: $\mathcal{A}$ sends a nonce to $\mathcal{B}$, which $\mathcal{B}$ encrypts but does not send to $\mathcal{A}$, as shown in Figure 2c. This time is not taken into account in the time-measurement. Now, $\mathcal{A}$ sends a new nonce to $\mathcal{B}$, which $\mathcal{B}$ xors with the previous encryption-result and sends back to $\mathcal{A}$. This time is measured, and $\mathcal{A}$ once again checks if the challenge was solved correctly.

Since we did not use a cryptographic function for mapping from challenge to response, but instead a plain xor, the first response-bit is now only dependent on the first challenge-bit. To remedy this, the challenge can simply be sent in reverse, therefore having the first response-bit (only) depend on the last challenge-bit. Since xor is a single operation instead of a full encryption-algorithm, this additional error on the distance-bound is considerably lower than the one in the previous protocol. If the precision is still too bad, the challenge-solving could be accelerated by a special piece of hardware which takes over the communication channel and performs the xor and reversing.

These protocols only allow $\mathcal{A}$ to get the distance-bound, which can, however, easily be resolved by performing a second pass with switched roles. A similar protocol based on commitment-schemes, which lets both parties get a distance-bound in one pass, can be seen with the MAD protocol from [18]. This protocol additionally increases the security by measuring the time multiple times, however it is not resistant against distance hijacking attacks.

B. Authentication of known devices

This physically impossible-to-cheat upper-bound on distance can now be used for known devices to check if they are close to each other. In such a scenario, a device $\mathcal{P}$ wants to prove its proximity to a device $\mathcal{V}$, which wants to verify the proximity-claim. $\mathcal{P}$ is therefore called *prover* and $\mathcal{V}$ is called *verifier*. The devices are considered known devices, because they have some pre-shared secret, like a common key.

Where a simple implementation relying solely on the location-limited nature of a communication channel is sus-

[14] Following from $\frac{1}{2} * c * t + \frac{1}{2} * c * t_{proc} = \frac{1}{2} * c * (t + t_{proc}) = \frac{1}{2} * c * t' = d'_{max} = d_{max} + d_{proc}$

ceptible to relay attacks as described in section III-C,[15] these are physically impossible with time-of-flight based distance-bounding: Even if attackers would relay the signal between prover and verifier, the signal travel time can only increase as was shown in section IV-A. Therefore, if the verifier accepts the distance-bound of such a relayed proximity-proof as close enough, the distance-bound of a direct connection would also have been accepted as close enough, as it would have been smaller.

An example for a use case of such a system can be found with unlocking a car: The car should only be unlocked, if the key is close enough, so that the owner sees the unlocking of his car.[16] The car might deem a radius of up to $30\,\mathrm{m}$ acceptable for unlocking, therefore will unlock if it receives a valid response, which took at most

$$t_{max} = \frac{2d_{max}}{c} \approx \frac{2 * 30\,\mathrm{m}}{3 \times 10^8\,\mathrm{m\,s^{-1}}} = 200\,\mathrm{ns}\,.$$

This shows a problem, that was already described in section IV-A: The challenge-solving needs to be fast, as the range the key gets accepted in shrinks the longer the key takes to generate an answer. For car keys, this is an even bigger problem, as in order to save cost and have the battery last longer, they usually do not include high clock-speed processors.[17]

When the devices are not only known, but also trusted, one could theoretically subtract either a constant time (or distance) as a margin for processing or a time reported by the prover: This would not reduce the resistance against mafia fraud, as this relaying could not reduce the processing time. An external attacker could only unlock the car from farther away, if they have the key used in solving the challenges and can use a faster processor for doing so.[18] However, the attacker does not gain anything from this, as they have now essentially stolen the car key and would get close to the car for entering anyway. In other scenarios, this can cause a problem though: The prover could copy his own key to a faster machine or report a longer processing time and use that for solving the challenges, therefore successfully committing distance fraud.[19]

C. Pairing previously unknown devices

Distance-bounding can also be used to pair two nearby devices (imagine connecting to a public printer). These do not share any secrets a priori, but want to create a secure session that prevents attackers from eavesdropping on their connection. These devices can be considered as not knowing each other.

Current possibilities for achieving a secure connection include confirming the connection out-of-band by either showing a code on both devices and having the user confirm they match or showing a code on one device and requiring the user to enter it on the other. This is one of the possible authentication mechanisms for Bluetooth, though this is not enforced by the protocol and unauthenticated pairings are allowed (which could theoretically be denied by one of the pairing devices) [21]. While this can be a secure way to ensure only the wanted devices are paired (even though there exist attacks on the Bluetooth implementation [22], [23]), it is not convenient to confirm or enter the code and may not be viable, as the device may not have a sufficient display or keypad to show or enter such codes.

A more seamless and easier approach for the user, that also would not require any additional hardware on the devices,[20] would be successful pairing of the devices using proximity: A user would initiate pairing on their device. After that, the devices exchange keys to create a secure channel, which could be realized using the Diffie-Hellman key exchange protocol, as was originally described in [24]. However, the devices do not know whether they have exchanged keys and hence created a private channel with each other or with an attacker. Therefore, the devices perform a time-of-flight based distance-bounding routine using the exchanged keys, as described in section IV-A. Using this gained distance-bound, the devices can now decide if they are in proximity: If the upper bound is lower than some set distance d_{pair} (e.g., $20\,\mathrm{cm}$), they will accept the connection and continue communication over the established channel. If, however, the distance was higher than the allowed maximum, the devices will drop the connection by simply discarding the exchanged keys. A high-level overview for this algorithm can be seen in Figure 3.

A similar protocol using ultrasound instead of radio waves was proposed in [1] and will be analyzed further later in this section. It is, however, not immediately obvious if such a pairing-routine does indeed disallow any external attackers from pairing with the devices or eavesdropping on their communication.

Assuming that the used key-exchange itself is secure, and any attacker is computationally bounded (i.e., cannot brute-force the key),[21] the created channel is indeed secure, but the devices have no way of confirming the identity of their communication-partners. This confirmation is then gained by performing the distance-bounding routine.

We have already shown, that the time-of-flight based distance-bounding using the speed of light, as described in section IV-A, is secure and physically impossible to cheat (except brute-forcing). Consequently, the devices can fully trust their distance-bounds for their communication-partners. Since the devices will reject connections with partners they cannot prove being in proximity,[22] an attacker would also need to be inside the radius of d_{pair}. This is the reason for the choice of a low

[15] Examples include the attacks in [19].

[16] Except when the key is stolen, which these protocols can – of course – not protect against.

[17] Examples for chips used in the automotive field include the SPC58 2B Line by STMicroelectronics [20], which has a clock speed of $80\,\mathrm{MHz}$.

[18] Note that the distance error from processing scales linearly with the inverse of the used frequency. For example, the step from $300\,\mathrm{MHz}$ to $3\,\mathrm{GHz}$ reduces the distance error caused by processing time to a tenth.

[19] This is not a problem for the car-key scenario, as successfully unlocking your car from a greater distance when having access to the car key does not benefit you.

[20] Apart from the hardware the devices use to communicate and possibly some way to initiate pairing.

[21] Otherwise, another key-exchange protocol should be used.

[22] As distance-bounding only creates an upper-bound, but no lower-bound, it is only possible to prove proximity, but not remoteness.

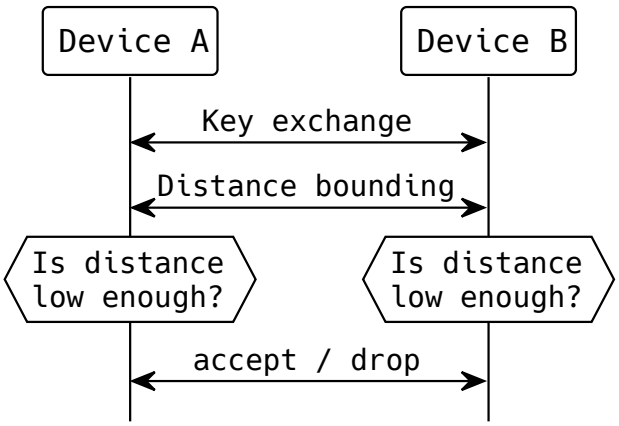


Figure 3: An overview for a protocol for pairing two devices without any previous knowledge.

d_{pair}: The users should be able to easily spot any attackers in proximity. While a radius of about $20\,\mathrm{cm}$ around ones device is easy to check for attackers, a larger radius would make it difficult, as radio-enabled devices became smaller,[23] so they could easily be hidden. Additionally, a larger radius might reach areas behind walls or on other floors, where it is impractical (if not impossible) to search for attackers. This is a reason why the protocols in section II are oftentimes not secure, as some of these protocols allow pairing with devices meters apart. Moreover, while the methods relying on variations in radio environment or movement are hard to attack, as the variations and local signals are hard to predict, their security is not physically impossible to cheat in contrast to time-of-flight based distance-bounding.

This makes the proposed protocol secure against the attacks mentioned in section III-C: It is secure against mafia attacks, and in extension all relay attacks, as we have shown that the relaying devices and the device the signal is relayed to would also need to be in the radius of d_{pair}. Not only would the relaying devices then be easily spotted, but the relaying would also be useless, as the recipient would need to be brought into the range of d_{pair} and could therefore directly pair with the other device. Terrorist fraud is impossible for the same reasons. Furthermore, distance fraud is also impossible, as already shown in section IV-A, because the reported upper bound on distance can never be lower than the actual distance. If an attacker was inside the radius of d_{pair}, they could successfully impersonate the pairing devices. However, since we specifically chose d_{pair} so low, that any attackers would easily be spotted, impersonation fraud is not possible as long as the user can be sure they want to pair with the device they see.

Although the distance needs to be this low, this also creates challenges for the hardware running these protocols: As we discussed in section IV-A, the distance-bound grows by $d_{ckl,3\,\mathrm{GHz}} \approx 5\,\mathrm{cm}$. Even if the pairing devices were immediately next to each other, the devices would only be allowed to take around 4 clock cycles maximum for responding to the challenge.[24] This essentially means, that the allowed range d_{pair} would need to be enlarged, weakening the security of this protocol due to the problems mentioned above, or the devices need additional specialized hardware, which can solve these challenges faster than a processor.

One can not simply take the same approach as in section IV-B, where the devices would simply subtract a fixed known processing time, as the pairing devices might have different computational power. Additionally, it would not be sensible to trust any processing times reported by the other device, as this device is untrusted in this stage of the protocol. An attacker could then report processing times higher than the real times and successfully make the device they want to pair with think they are nearby, therefore successfully committing distance fraud. Most important though, it can not be assumed that an attacker would also use such computationally weak devices and not some hardware accelerated equipment, which would lead to an attacker being able to cheat on the distance-bounds.

As was already mentioned, a different approach was taken in [1]: Instead of communicating over radio waves and using the speed of light for calculating distance-bounds, they communicated over ultrasound and used the speed of sound for distance-bounds. As the speed of sound c' is considerably lower than the speed of light, the computing requirements of the devices can be considerably lower, because the error per clock cycle when using ultrasound is only

$$d'_{clk,3\,\mathrm{GHz}} = \frac{c'}{2f} \approx \frac{340\,\mathrm{m\,s^{-1}}}{2 * 3\,\mathrm{GHz}} \approx 57\,\mathrm{nm}$$

which is significantly smaller than the allowed $20\,\mathrm{cm}$. Even though the used distance-bounds are not physically uncheatable anymore, this does not create any problems on first sight: As the devices only send and receive ultrasound waves, an attacker would need to convert the ultrasound waves to a faster channel inside the radius of d_{pair} and would therefore be easily visible.

However, any leaks on or possibilities of information injection over faster side-channels can be used to break the security of the distance-bounding routine. A faster side-channel is a channel, which is not the channel the devices use to communicate and where information can be transmitted faster. This is where the relative recently discovered light commands in [27] come into play: Here, lasers were used to enter commands into voice assistants. This conversion of light to sound is possible because of the way MEMS microphones work, though tests on condenser microphones allowed injecting sound using lasers as well. This can not only be applied to transmitting voice commands, but any signal, such as the communication in the distance-bounding protocols. Therefore, we have found a way to inject information over a faster side-channel. Still, the attacker can only be about $2 * d_{pair}$ (i.e., $40\,\mathrm{cm}$) away, as

[23] Examples for such devices include microcontrollers like Espressifs ESP family of chips [25] or even more featured devices like Flipper Devices' Flipper Zero [26].

[24] Ignoring any additional time it would take for the processor to receive/send the data over the radio.

Excellent!

they still need to receive the signal over the slow ultrasound channel and hence could be spotted rather easy.

However, there exist some possibilities to leak the transmission over a faster side-channel as well: As sound waves are just vibrations of air, they also vibrate any objects. While these vibrations are extremely fast ($> 20\,\mathrm{kHz}$ for ultrasound), a fast sampling of these vibrations (for example by using a camera with a sufficiently high framerate) could allow extracting the sound. These vibrations are usually relatively small, but there exist laser-microphones [28], which allow recording sound using these tiny vibrations by shining a laser onto a reflective surface and measuring the power variations in the reflected beam. This now allows an attacker to be up to $881\,040 * d_{pair}$ (i.e., about $176\,\mathrm{km}$) away[25] (depending on the distance of the vibrating object used to receive the devices transmission and requiring line-of-sight) and still successfully pair with the devices.

Granted, this attack is rather theoretical, as any movement of the two devices would require the attacker to reorient his transmitting laser-beam onto the devices microphone and getting a good signal from the laser microphone requires a good shiny surface in range that vibrates with the ultrasound communication. It does, however, show, that while weakening the factors of the time-of-flight based distance-bounding protocol from section IV-A can be desireable to decrease the required processing speed, it can also lead to unintended side-channel attacks, as it is not physically secure anymore.

V. Evaluation

The proposed protocols have not been tested for this paper. Future work is needed in this regard: For one, the feasability of the secure distance-bounding protocol using the speed of light needs to be checked. At the start, this may be implemented using normal processors for a proof of concept. Later iterations may start with custom FPGAs to then go towards custom chip designs.

Additionally, the laser-based attacks on the distance-bounding protocol using sound from [1] are currently only theoretical. The protocol using sound needs to be implemented and the possibility of the mentioned attacks must be checked.

We have, however, shown, that implementing the physically secure protocol using the speed of light is challenging and weakening the protocol to reduce the required computational power can lead to unforeseen security issues.

VI. Discussion

These protocols can be used to secure various authentications which are susceptible to relay attacks. This can reach from the aforementioned keyless car entry and pairing of devices to securing entry using RFID cards.

[25]Following from the relation of the speed of light c to the speed of sound c' because

$$d_{pair} = \frac{1}{2} * c' * \frac{2 * d_{attack}}{c} = \frac{c'}{c} * d_{attack} \implies d_{attack} = \frac{c}{c'} * d_{pair}$$

While it has been shown, that processors need to be relatively fast for these protocols to be secure, some of the relaxations discussed might make sense depending on the use case, especially for the authentication of known devices. And if a high level of security needs to be achieved, the additional cost for the required hardware will not be the limiting factor. Furthermore, if specialized chip-designs become available and are mass-produced or integrated into microcontrollers, the additional cost for these protocols will be negligible while at the same time increasing security. These hardware-accelerations would additionally allow slow processors to also perform these distance-bounding routines.

VII. Conclusion

We have shown how using proximity as an authentication factor can be more convenient than confirming the pairing out-of-band using human interaction. We introduced multiple existing protocols that can be used for this purpose. Furthermore, we mentioned why protocols depending solely on the location limited nature of the used communication channel are generally vulnerable to relay attacks like mafia fraud and terrorist fraud, as well as distance fraud. We then introduced time-of-flight based distance-bounding using the speed of light. Not only did we show how this is physically secure, but also how a protocol can be built around this theoretical principle.

However, we have seen that this method does have computational requirements which may not be feasible to implement in small devices. We have shown that depending on the use case, small relaxations may be possible. Additionally, we analyzed a bigger relaxation inspecting an existing protocol, which used the speed of sound. This leads to far lower requirements on processing power. We have however seen how such a relaxation makes the protocol lose its physical uncheatability. We provided a theoretical proof of concept using recent discoveries in related fields.

Future work developing hardware that can provide the fast computation of responses needed for this method of distance-bounding is required. Furthermore, these devices' reliability and accuracy will need to be tested then. Additionally, the ultrasound based protocol will need to be implemented and the practicality of the proposed attacks will need to be checked.

References

[1] D. Singelée and B. Preneel, "Key establishment using secure distance bounding protocols," in *2007 Fourth Annual International Conference on Mobile and Ubiquitous Systems: Networking & Services (MobiQuitous)*. IEEE, 2007, pp. 1–6.

[2] A. Varshavsky, A. Scannell, A. LaMarca, and E. De Lara, "Amigo: Proximity-based authentication of mobile devices," in *UbiComp 2007: Ubiquitous Computing: 9th International Conference, UbiComp 2007, Innsbruck, Austria, September 16-19, 2007. Proceedings 9*. Springer, 2007, pp. 253–270.

[3] H. Shafagh and A. Hithnawi, "Poster: come closer: proximity-based authentication for the internet of things," in *Proceedings of the 20th annual international conference on Mobile computing and networking*, 2014, pp. 421–424.

[4] A. Kalamandeen, A. Scannell, E. de Lara, A. Sheth, and A. LaMarca, "Ensemble: cooperative proximity-based authentication," in *Proceedings of the 8th international conference on Mobile systems, applications, and services*, 2010, pp. 331–344.

[5] L. Cai, K. Zeng, H. Chen, and P. Mohapatra, "Good neighbor: Ad hoc pairing of nearby wireless devices by multiple antennas." in *NDSS*, 2011.

[6] T. J. Pierson, X. Liang, R. Peterson, and D. Kotz, "Wanda: securely introducing mobile devices," in *IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications*. IEEE, 2016, pp. 1–9.

[7] J. Zhang, Z. Wang, Z. Yang, and Q. Zhang, "Proximity based iot device authentication," in *IEEE INFOCOM 2017-IEEE conference on computer communications*. IEEE, 2017, pp. 1–9.

[8] R. Mayrhofer and H. Gellersen, "Shake well before use: Intuitive and secure pairing of mobile devices," *IEEE Transactions on Mobile Computing*, vol. 8, no. 6, pp. 792–806, 2009.

[9] S. Mathur, R. Miller, A. Varshavsky, W. Trappe, and N. Mandayam, "Proximate: proximity-based secure pairing using ambient wireless signals," in *Proceedings of the 9th international conference on Mobile systems, applications, and services*, 2011, pp. 211–224.

[10] C. E. Shannon, "Communication in the presence of noise," *Proceedings of the IRE*, vol. 37, no. 1, pp. 10–21, 1949.

[11] S. Brands and D. Chaum, "Distance-bounding protocols," in *Workshop on the Theory and Application of of Cryptographic Techniques*. Springer, 1993, pp. 344–359.

[12] G. Avoine and A. Tchamkerten, "An efficient distance bounding rfid authentication protocol: balancing false-acceptance rate and memory requirement," in *Information Security: 12th International Conference, ISC 2009, Pisa, Italy, September 7-9, 2009. Proceedings 12*. Springer, 2009, pp. 250–261.

[13] Y. Desmedt, C. Goutier, and S. Bengio, "Special uses and abuses of the fiat-shamir passport protocol," in *Advances in Cryptology—CRYPTO'87: Proceedings 7*. Springer, 1988, pp. 21–39.

[14] S. Bengio, G. Brassard, Y. G. Desmedt, C. Goutier, and J.-J. Quisquater, "Secure implementation of identification systems," *Journal of Cryptology*, vol. 4, no. 3, pp. 175–183, 1991.

[15] C. Cremers, K. B. Rasmussen, B. Schmidt, and S. Capkun, "Distance hijacking attacks on distance bounding protocols," in *2012 IEEE symposium on security and privacy*. IEEE, 2012, pp. 113–127.

[16] A. Einstein, "Zur elektrodynamik bewegter körper," *Annalen der physik*, vol. 4, 1905.

[17] C. E. Shannon, "A mathematical theory of cryptography," *Mathematical Theory of Cryptography*, 1945.

[18] S. Čapkun, L. Buttyán, and J.-P. Hubaux, "Sector: secure tracking of node encounters in multi-hop wireless networks," in *Proceedings of the 1st ACM workshop on Security of ad hoc and sensor networks*, 2003, pp. 21–32.

[19] A. Francillon, B. Danev, and S. Capkun, "Relay attacks on passive keyless entry and start systems in modern cars," in *Proceedings of the Network and Distributed System Security Symposium (NDSS)*. Eidgenössische Technische Hochschule Zürich, Department of Computer Science, 2011.

[20] *SPC58 2B Line - 32 bit Power Architecture automotive MCUSingle core 80Mhz, 1MByte Flash, ASIL-B*, STMicroelectronics, rev 4.

[21] K. Ren. Bluetooth pairing part 1 – pairing feature exchange. [Online]. Available: https://www.bluetooth.com/blog/bluetooth-pairing-part-1-pairing-feature-exchange/

[22] Y. Shaked and A. Wool, "Cracking the bluetooth pin," in *Proceedings of the 3rd international conference on Mobile systems, applications, and services*, 2005, pp. 39–50.

[23] M. von Tschirschnitz, L. Peuckert, F. Franzen, and J. Grossklags, "Method confusion attack on bluetooth pairing," in *2021 IEEE symposium on security and privacy (SP)*. IEEE, 2021, pp. 1332–1347.

[24] W. DIFFIE and M. E. HELLMAN, "New directions in cryptography," *IEEE TRANSACTIONS ON INFORMATION THEORY*, vol. 22, no. 6, 1976.

[25] Espressif systems chipsets. [Online]. Available: https://www.espressif.com/en/products/socs

[26] Flipper zero. [Online]. Available: https://flipperzero.one/

[27] T. Sugawara, B. Cyr, S. Rampazzi, D. Genkin, and K. Fu, "Light commands: Laser-based audio injection attacks on voice-controllable systems," in *29th USENIX Security Symposium (USENIX Security 20)*, 2020.

[28] S. D. Koloydenko and K. V. Tcyguleva, "Laser microphone surveillance," in *2021 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (ElConRus)*. IEEE, 2021, pp. 1991–1995.