Distance bounding protocols
If you think cryptography is the answer to your problem, then you don't know what your problem is.

The "mafia" fraud

- The top-secret area contains big military secrets (crashed UFOs, mind-control technologies, etc.)
- The "men in black" employees access the top-secret area with a contactless smart card.
The “mafia” fraud

• Suppose that:
  • The smart cards cannot be stolen (man in black are very professional)
  • The smart cards cannot be cloned (asymmetric cryptography with tamper-proofness)
  • The authentication protocol between verifier and prover is correct (BAN logic proof)
  • The employed crypto primitives are unforgeable (the cryptoanalyzer are good in maths)
• There is still a way to completely break the system
The “mafia” fraud

Prover

secret key: $k$

Verifier

nounce: $a$

prover's key: $k^1$

beacon

hello, $P$

$a$

$\text{sign}_k(a)$
The “mafia” fraud

• Build a relay link (possibly an Internet link) which makes a legitimate verifier (V) communicate with a far away legitimate prover (P)

Welcome back, P!
Come in!

Thank you!
(Har har!)

 relay link

Legitimate prover  Proxy verifier  Proxy prover  Legitimate verifier
The “mafia” fraud

- Legitimate prover
- Proxy verifier
- Relay link
- Proxy prover
- Legitimate verifier
The “mafia” fraud

Prover ➔ relay link ➔ Verifier

secret key: $k$

nounce: $a$

prover's key: $k^1$

$\text{hello, } P$

$a$

$\text{sign}_k(a)$
The “mafia” fraud

- *False assumption:* "If two devices can hear each other, then they are close to each other"
- Sometimes called "relay attack", "wormhole attack"
- Other examples: credit card payments, car stealing, wireless routing
Mafia fraud against chip&pin payments
Mafia fraud against chip&pin payments

Evil shop (restaurant)

- PIN 123456
- 1 lunch: 20€
- modified POS
- modified card
- P'

Good shop (jewelry)

- PIN 123456
- 1 diamond: 20,000€
- regular POS
- modified card
- P'

Bank
Mafia fraud against chip&pin payments
Mafia fraud against PKES

- Passive Keyless Entry and Start
Mafia fraud against PKES
Mafia fraud against PKES

<table>
<thead>
<tr>
<th>Car model</th>
<th>Relay cable</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>7 m</td>
<td>30 m</td>
<td>60 m</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>open</td>
<td>go</td>
<td>open</td>
<td>go</td>
<td>open</td>
</tr>
<tr>
<td>Model 1</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Model 2</td>
<td>✓</td>
<td>✓</td>
<td>A</td>
<td>A</td>
<td>A</td>
</tr>
<tr>
<td>Model 3</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Model 4</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Model 5</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Model 6</td>
<td>✓</td>
<td>✓</td>
<td>A</td>
<td>A</td>
<td>A</td>
</tr>
<tr>
<td>Model 7</td>
<td>✓</td>
<td>✓</td>
<td>A</td>
<td>A</td>
<td>-</td>
</tr>
<tr>
<td>Model 8</td>
<td>✓</td>
<td>A</td>
<td>✓</td>
<td>A</td>
<td>-</td>
</tr>
<tr>
<td>Model 9</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Model 10</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
</tbody>
</table>

- Without amplification
- With amplification
- Not tested

2010
Wormhole attack

- False assumption: "if A hears an (authenticated) beacon message from B, then B and A are in the proximity"
- The adversary establishes a (wireless) link between two far away nodes (the wormhole)
Wormhole attack

- A and B become *de facto* neighbours
- The wormhole is controlled by the adversary
- The adversary can suppress the traffic partially or totally
Wormhole attack

- From the routing point of view, the wormhole link is very convenient.
- The adversary can split a wireless network in (roughly) two parts.
Distance bounding protocol

- **Countermeasure**: precisely measure the round trip-time between a challenge and a response messages
- If the round-trip time is too large, reject the authentication (a mafia fraud could be present!)
- This is not enough!
- The adversary could build a relay link and actively anticipate the challenge and response messages
- The challenges and the responses must be **externally unpredictable**
Brands-Chaum protocol (type I)

Prover

Verifier

secret key: $k$

public key: $k^1$

$N$ random bits: $b_i$

$N$ random bits: $a_i$

Response time: $Tr$

Round-trip times: $RTT_i$

$m = a_1|b_1|...|a_N|b_N$

$x \times N$ times:

$sign_k(m)$
Brands-Chaum protocol (type I)

• Two general phases
  • **Rapid bit exchange** (*real-time*): challenge and response bits are exchanged, the round-trip time is precisely measured
    - challenge and response bits are *externally unpredictable*
    - channel speed is *impassable* (typically radio or electrical, avoid sound!)
  • **Signature**: the prover signs the challenge and response bits with a secret
    - the device which sent the responses proves to be the prover
Brands-Chaum protocol (type I)

Prover \[\smiley\] \quad \leftrightarrow \quad \text{Verifier} \[\smiley\]

Round-trip times: \( RTT_i \)

Challenge and response bits are externally unpredictable

\( m = a_1 | b_1 | ... | a_N | b_N \)

\( \text{sign}_k(m) \)
Brands-Chaum protocol (type I)

$m = a_1b_1|a_2b_2|...|a_nb_n$

$sign_k(m)$

Round-trip times: $RTT_i$

Channel speed is impassable
Distance bounding protocol

The verifier:

1. Executes the protocol
2. Verifies the validity of the signature
3. Computes the *measured distance* $D$ as:
   
   $$D = \max(\text{RTT}_i) \times c/2$$

   $c = \text{speed of light}$

4. Verifies that the measured distance is within a *proximity distance* $D_{\text{max}}$

   $$D \leq D_{\text{max}}$$
Distance bounding protocol

- The real distance $d$ is given by:
  \[ d = \text{mean}(RTT_i - Tr) \times \frac{v}{2} - d_{\text{NLOS}} \]
  
  $Tr$: response time
  
  $v < c$: real signal speed
  
  $d_{\text{NLOS}}$: component due to the non-line-of-sight path of the signal
Distance bounding protocol

• The measured distance is always longer than the real one:
  \[ D \geq d \]
  \[ (\max(RTT_i) \times c/2) \geq (\text{mean}(RTT_i - Tr) \times v/2 - d_{\text{NLOS}}) \]

• The term with the biggest impact is Tr

• If we design the prover to respond in \( Tr \geq Tr_{\text{min}} \) time, we can measure a more accurate distance (accuracy improvement):
  \[ D = \max(RTT_i - Tr_{\text{min}}) \times c/2 \]
Distance bounding protocol

• Honest case:

- Proximity distance $D_{\text{max}}$
- Measured distance $D$
- Distance bounding
- Real distance $d$

• Adversarial case:

- Proximity distance $D_{\text{max}}$
- Measured distance $D$
- Distance bounding
- Real distance $d$
Distance bounding protocol

• To mount a Mafia fraud, the adversary should build a *time-gaining relay*

• A time-gaining relay is a link that delivers *in advance* the challenge and/or the response bits

• However:
  • She cannot guess them in advance (unpredictability)
  • She cannot make them travel quicker than light (unpassability)
Distance bounding protocol

- What is the probability of successfully performing a time-gaining mafia fraud?
- The adversary has to anticipate $N$ bit exchanges
- For each bit exchange, she has to guess and anticipate the response (or the challenge)
  $$P_{1\text{-round}} = 1/2$$
- Overall adversarial success probability:
  $$P_{\text{adv}} = (1/2)^N \quad (\text{negligible with } N)$$
  with $N=128$: $P_{\text{adv}} = 3 \times 10^{-39}$
Distance bounding protocol

The prover is surely in this circle

\[ D \]

\[ D_{\text{max}} \]
Other types of frauds

- **Mafia fraud**: an external adversary builds a relay link between P and V, and makes the distance appear shorter (*time-gaining relay*)
- **Distance fraud**: P itself is malicious, and makes its distance from V appear shorter (*time-gaining response*)
- **Terrorist fraud**: a malicious P colludes with an external adversary to make its distance from V appear shorter
- **Distance hijacking**: a malicious P leverages on another (honest) P to make its distance from V appear shorter
Other types of frauds

- Mafia fraud
- Distance fraud
- Distance hijacking
- Terrorist fraud

- Is the prover honest? 
  - yes
  - no
- Is only the prover involved in the attack? 
  - yes
  - no
- Is one of the other involved parties honest? 
  - yes 
  - no
Distance fraud

Prover

Verifier

$N$ random bits: $b_i$

secret key: $k$

$m = a_1|b_1|...|a_N|b_N$

$x N$ times:

$a_i$

$b_i$

$\text{sign}_k(m)$

$N$ random bits: $a_i$

public key: $k^1$
Distance fraud

Prover

Verifier

$N$ random bits: $b_i$

secret key: $k$

$m = a_1|b_1|...|a_N|b_N$

$xN$ times:

$sign_k(m)$

Round-trip times: $RTT_i$

$N$ random bits: $a_i$

public key: $k^1$
Distance fraud

- **Countermeasure**: the response bits must depend on the challenge bits
- **Challenge-response function**: $b_i = f_{cr}(a_i)$
- In this way, the response bits becomes *externally predictable* (mafia fraud vulnerability!)
Distance fraud

Prover

Verifier

secret key: $k$

public key: $k^{-1}$

$N$ random bits: $a_i$

$m = a_1|b_1|...|a_N|b_N$

$\text{sign}_k(m)$

$x \times N$ times:

$\begin{align*}
  b_i &= f_{cr}(a_i) \\
  a_i
\end{align*}$
Distance fraud

Prover

Verifier

secret key: \( k \)

public key: \( k^1 \)

\( N \) random bits: \( a_i \)

\( b_i = f_{cr}(a_i) \)

\( m = a_1|b_1|...|a_N|b_N \)

\( \text{sign}_k(m) \)
Commitment scheme

• Bob and Alice want to play *rock-paper-scissors* by email

• Suppose there is not a trusted third entity

• If Alice first sends to Bob her choice (for example “*paper*”), Bob can cheat by changing his choice on-the-fly (for example “*scissors*”)

• Who plays for second *always wins*
Commitment scheme

- A commitment scheme is a cryptographic protocol which allows a party to commit to a value without revealing it.

- “To commit to a value” = to be forced to use a particular value afterward.
Commitment scheme

Alice

random bit-string: salt

commit phase

hash("paper", salt)

open phase

"rock"

salt
Distance fraud

- Challenge-response function: \( b_i = f_{cr}(a_i) \)
- Challenge-response function: \( b_i = f_{cr}(a_i, c_i) \)
- The prover commits the bits \( c_i \)
Distance fraud

- The challenge-response function cannot be too complex, because the prover must respond timely

\[ b_i = f_{cr}(a_i, c_i) = a_i \oplus c_i \]
Brands-Chaum protocol (type II)

Commit phase:

- $N$ random bits: $c_i$
- Random bit-string: $\textit{salt}$

Open phase:

- $m = a_1|b_1|...|a_N|b_N$
- $b_i = a_i \oplus c_i$
- $\text{hash}(c_1|...|c_N|\textit{salt})$ x $N$ times:

- $m = a_1|b_1|...|a_N|b_N$
- $\textit{salt}$, $\text{sign}_k(m)$

It resists against mafia fraud and distance fraud.
Brands-Chaum protocol (type II)

- Four general phases
  - **Commit**: the prover "promises" to use a particular sequence of bits $c_i$, without revealing it
  - **Rapid bit exchange (**real-time**)**: challenge and response bits are exchanged, the round-trip time is precisely measured
    - response bits must depend on the challenge bits and the committed bits
  - **Commit open**: the prover reveals the committed bits
  - **Signature**: the prover signs the challenge and response bits with a secret
Brands-Chaum protocol (type II)

- The verifier:
  1. Executes the protocol
  2. Verifies the validity of the commitment
  3. Verifies the validity of the signature
  4. Computes the *measured distance* $D$ as:
     \[ D = \max(RTT_i) \cdot c/2 \]
     
     \[ c = \text{speed of light} \]
  5. Verifies that the measured distance is within a *proximity distance* $D_{\text{max}}$
     \[ D \leq D_{\text{max}} \]
Brands-Chaum protocol (type II)

- Can we make the accuracy improvement?
  \[ D = \max(RTT_i - Tr_{\text{min}}) \times c/2 \]

- No, because we cannot trust the prover to respect a minimal response time \( Tr_{\text{min}} \)

- A dishonest prover could have a more powerful hardware
  - compute quicker the challenge-response function
  - respond quicker
Brands-Chaum protocol (type II)

• What is the probability of successfully performing a time-gaining mafia fraud?

\[ P_{adv} = (1/2)^N \]

• What is the probability of successfully performing a distance fraud?

• For each bit exchange, the dishonest prover has to guess the response:

\[ P_{adv} = (1/2)^N \]
Overview

- A *distance bounding protocol* is a protocol that permits us to establish a *secure upper bound* \((D)\) to the distance between a “prover” and a “verifier”

\[ d_{V-P} \leq D \]

- The basic idea is to precisely measure the *round-trip time* between two unpredictable messages (a challenge and a response)
Overview

- **Brands-Chaum protocol (type I)** is a distance bounding protocol capable of resisting to *mafia fraud* (external adversary with relay link)
  - \( P_{\text{adv}} = (1/2)^N \)

- **Brands-Chaum protocol (type II)** is a distance bounding protocol capable of resisting to mafia fraud and *distance fraud* (dishonest prover that responds in advance)
  - Mafia fraud: \( P_{\text{adv}} = (1/2)^N \)
  - Distance fraud: \( P_{\text{adv}} = (1/2)^N \)
Distance bounding on RFID tags

Verifier (reader)

Prover (tag)
Distance bounding on RFID tags

- RFIDs are resource-constrained
  - It is expensive to equip them with unpredictable random number generators
- Wireless channels are noisy
  - The signature fails if one of the challenge-response bits gets corrupted
- RFIDs have an external (and untrusted) clock source
  - Overclock attacks are possible
Channel noise

Prover

Verifier

hash(c_1|...|c_N|salt)

x N times:

a_i

b_i = a_i \oplus c_i

m = a_1|b_1|...|a_N|b_N

salt, sign_k(m)

Reliable communication
- Forward Error Correction (FEC)
- Ack/retransmit
- etc...

Unreliable communication
Channel noise

Prover

Verifier

\[ m = a_1 | b_1 | ... | a_N | b_N \]

\[ b_i = a_i \oplus c_i \]

\[ \text{hash}(c_1 | ... | c_N | \text{salt}) \]

\[ x \text{ N times:} \]

\[ \text{salt, sign}_k(m) \]

corrupted challenge bit
the signature fails
corrupted response bit
the signature fails
Channel noise

- Probability of protocol failure:

\[ P_{\text{fail}} = 1 - (1 - BER_{V-P})^N(1 - BER_{P-V})^N \]

\( BER_{V-P}, BER_{P-V} \): bit error rates of P-V and V-P channels

- With \( BER_{V-P} = BER_{P-V} = 10^{-3} \) and \( N=128 \):

\[ P_{\text{fail}} = 23\% ! \]
Channel noise

- Performing a *reliable* rapid bit exchange (for example with CRC) is burdensome:
  - More than one bit for every challenge and for every response
- ... and insecure:
  - The dishonest prover could ignore the challenge's CRC and anticipate the response
Channel noise

Prover

Verifier

\[ b_i = a_i \oplus c_i \]

\[ m = a_1 | b_1 | ... | a_N | b_N \]

Send again \( m \) with a reliable channel

The verifier checks how many corrupted bits
Hancke-Kuhn protocol

Prover

Verifier

secret key: $k$

nonce: $N_v$

$x \times N$ times:

$b_i = \begin{cases} 
  m_i & \text{if } a_i = 0 \\
  n_i & \text{if } a_i = 1 
\end{cases}$

$<m, n> = \text{MAC}_k(N_v)$

$N_{\text{correct}} = \text{number of correct responses}$

It resists against mafia fraud and distance fraud
Hancke-Kuhn protocol

• The verifier counts the number of correct responses $N_{\text{correct}}$

• If the correct responses are $\geq$ a threshold $N_{\text{accept}}$, the authentication is accepted
Noise tolerance

• Probability of protocol failure:

\[ P_{\text{fail}} = \sum_{i=0}^{N_{\text{accept}}-1} \binom{N}{i} (1-\epsilon)^i \epsilon^{n-i} \]

where “\( \epsilon \)” is the probability of receiving a corrupted response

\[ \epsilon = \frac{BER_{p-v} + (1-(1-BER_{v-p})(1-BER_{p-v}))}{2} \]

• With \( BER_{v-p} = BER_{p-v} = 10^{-3} \), \( N=128 \), and \( N_{\text{accept}}=124 \):

\[ P_{\text{fail}} = 2 \times 10^{-6} \] (Brands-Chaum was \( P_{\text{fail}} = 23\% \))
Hancke-Kuhn protocol

- Challenge-response function implemented with *shift registers*

$$b_i = f_{cr}(a_i, m, n)$$

externally unpredictable
Hancke-Kuhn protocol

- Two general phases
  - **Secret initialization**: prover and verifier agree to an *externally unpredictable secret* \((m, n)\)
  - **Rapid bit exchange + signature (*real-time*):** challenge and response bits are exchanged
    - The signature is contextual with the rapid bit exchange
Hancke-Kuhn protocol

- The prover is not required to produce (and commit to) an unpredictable quantity
- No final signature
  - In practice, the *response bits* are the signature
- The overall quantity of messages is decreased (time efficiency)
- It is possible to tolerate a certain number of wrong response bits, due to channel noise
Double-chance guessing attack

- Hancke-Kuhn distance bounding is vulnerable to *double-chance guessing*!
  - the adversary tries to guess the challenge bit
  - if she fails, she has *another chance* by trying to guess the response bit
Double-chance guessing attack

\[ \langle m, n \rangle = \text{MAC}_k(N_v) \]

\[ b_i = \begin{cases} 
  m_i & \text{if } a_i = 0 \\
  n_i & \text{if } a_i = 1 
\end{cases} \]

\[ x \times N \text{ times:} \]

- case of \( a'_i = a_i \)
Double-chance guessing attack

\[ \langle m, n \rangle = \text{MAC}_k(N_v) \]

\[ b_i = \begin{cases} 
  m_i & \text{if } a_i = 0 \\
  n_i & \text{if } a_i = 1 
\end{cases} \]

Case of \( a'_i \neq a_i \)
Double-chance guessing attack

- What is the probability of successfully performing the double-chance guessing attack?
- For each bit exchange, she has to perform double-chance guessing
  \[ P_{1\text{-round}} = \frac{1}{2} + \frac{1}{2} \times (\frac{1}{2}) = \frac{3}{4} \]
- Overall adversarial success probability:
  \[
  P_{adv} = \sum_{i=N_{accept}}^{N} \binom{N}{i} \left(\frac{3}{4}\right)^i \left(\frac{1}{4}\right)^{N-i}
  \]
  with \(N=128\) and \(N_{accept}=124\): \(P_{adv} = 10^{-12}\)
  (Brands-Chaum was \(P_{adv} = 3 \times 10^{-39}\))
Overclocking attack

- RFIDs do not have an internal clock source
- Their clock source is untrusted
- An external adversary could overclock a legitimate prover to get the responses in advance
- Countermeasure: shift registers challenge-response function
Overclocking attack

XOR-based version of Hancke-Kuhn protocol is vulnerable to overclocking attack
Overclocking attack

\[ c = \text{MAC}_k(N_v) \]

processing time (speed-up)

\[ a_i' \oplus b_i' = c_i \]

\[ b_i = a_i \oplus c_i \]

\[ N_v \times N \text{ times:} \]

overclock!
Overclocking attack

\[ <m,n> = \text{MAC}_k(N_v) \]

processing time (speed-up)

\[
\begin{align*}
    b_i &= \begin{cases} 
        m_i & \text{if } a_i=0 \\
        n_i & \text{if } a_i=1
    \end{cases} \\
    \text{she discovers } m_i \text{ or } n_i \text{ (not both)}
\end{align*}
\]

\[ x N \text{ times:} \]

\[
\begin{align*}
    m_i & \text{ or } n_i \text{ (not both)} \\
    \text{if } a_i=a'_i \text{ she can produce the correct } b_i, \\
    \text{otherwise she has to } \text{guess it}
\end{align*}
\]
Overclocking attack

- For each bit exchange:
  - the adversary discovers a register bit, and hopes that it is the “useful” one \((P=1/2)\)
  - if she fails, she tries to guess the response bit \((P=1/2)\)
  
  \[P_{1\text{-round}} = \frac{3}{4}\]

- Overall adversarial success probability:

  \[P_{adv} = \sum_{i=N_{\text{accept}}}^{N} \binom{N}{i} \left(\frac{3}{4}\right)^i \left(\frac{1}{4}\right)^{N-i}\]

  (same as double-chance guessing attack)
Overclocking attack

- To be sure of being successful, the adversary should perform twice all the $N$ bit exchanges with the legitimate prover, and discover both $m$ and $n$ registers.
- She should perform a huge overclock, which easily avoidable by means of a low-pass filter on the clock source.
Frame-based distance bounding

20 meters
Frame-based distance bounding

- Prover and verifier are far away (up to 20-30m)
- Every message comes with a *PHY header* (for time synchronization and demodulation infos)
- The PHY headers are *very long* (longer than payloads): 1024 symbols or more
- Sending short headers would require to transmit them with very high transmission power
- This is not permitted by the telecommunications regulator agencies (for example FCC in the USA)
Frame-based distance bounding

- It is burdensome to send *single bits*
  - A very long PHY header for each bit!
- … and insecure:
  - A dishonest prover could leverage on the latency times to anticipate the transmission of the response bit
Frame-based distance bounding

- The rapid bit exchange phase is replaced by a frame exchange phase.
- Instead of performing $N$ challenge-response rounds, we perform a single round with two $N$-bit frames.
Frame exchange phase
(honest prover)

Prover

Verifier

preamble

payload

trailer

$a_1, a_2, \ldots, a_N$

$b_1, b_2, \ldots, b_N$

response time $Tr$

round-trip time $RTT$
Frame exchange phase (honest prover)

- **Timeline representation:**

  ![Timeline representation diagram]

- Can we use the same timing if we want to defend against *distance fraud* too?
Frame exchange phase (honest prover)

- A frame-based challenge-response function is too complex to be computed on-the-fly

- A (classic) bit-based challenge-response function is used
Distance fraud

Prover

Rx

Tx

preamble

pld

trl

\( a_1, a_2, \ldots, a_N \)

\( b_1, b_2, \ldots, b_N \)

response

time gain

time

\( f_{cr}(\cdot) \)

response time \( Tr \)
Distance fraud

- *Countermeasure*: the challenge-response phase is performed in a *full-duplex* way
- Each response bit is sent *just after* having received the correspondent challenge bit
Frame exchange phase (dishonest prover)
Frame exchange phase (dishonest prover)

Prover

Verifier

$\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_N$

preamble

payload

trailer

$a_1, a_2, \ldots, a_N$

round-trip time $RTT$
References

  - (Only sections 1 and 2)

Contact me for questions:

pericle.perazzo@iet.unipi.it