Distance bounding

Why distance bounding?

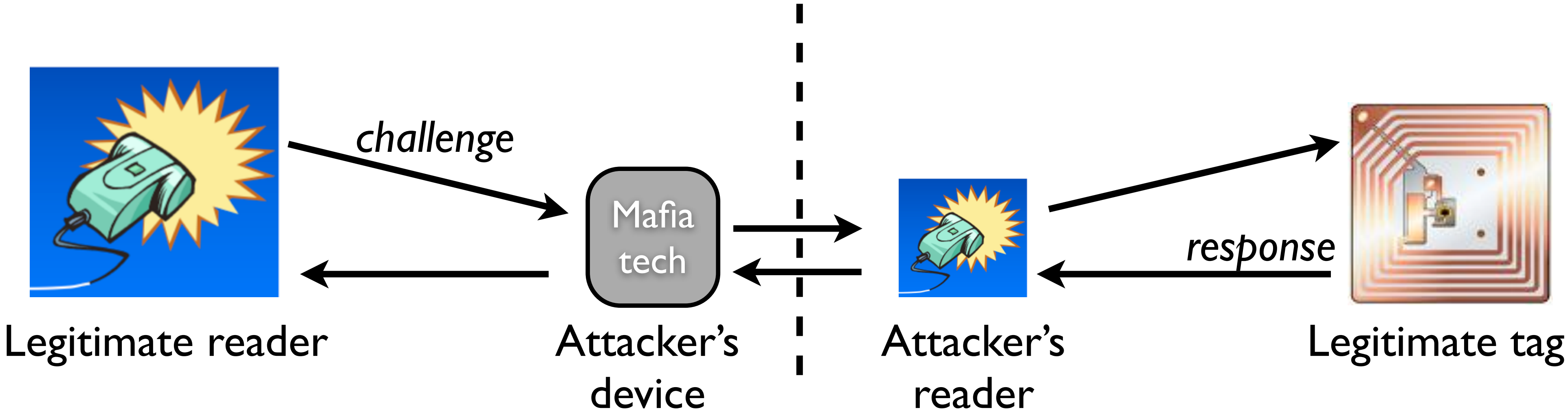
Authentication alone may not be sufficient

- physical access to buildings etc.
  - watch out for relay attack

Two main types of attack

- "Mafia Fraud"
- "Terrorist Fraud"

Relay attacks: Mafia Fraud

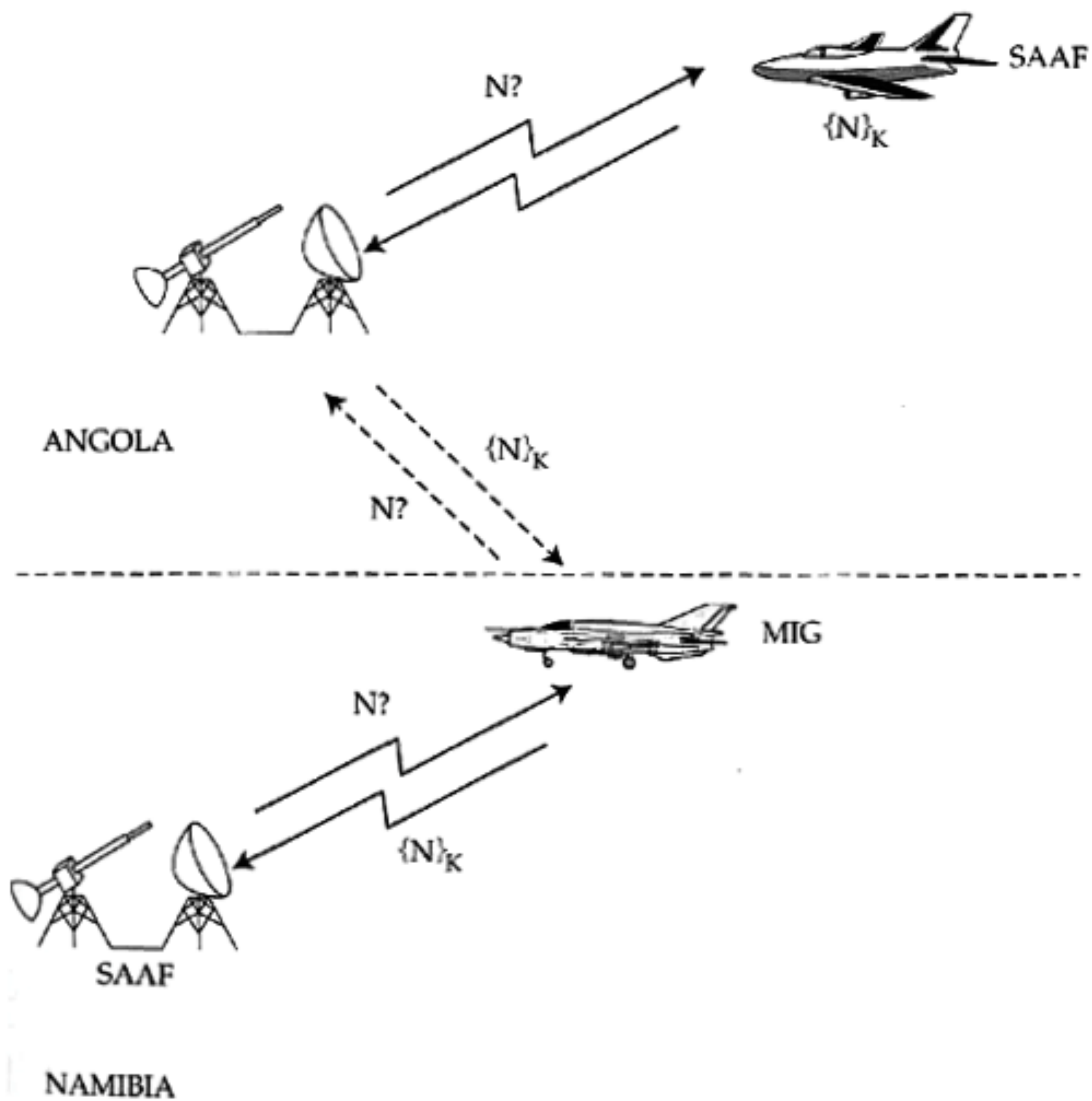


<u>Unaware of attack</u>

Authentication without distance checking

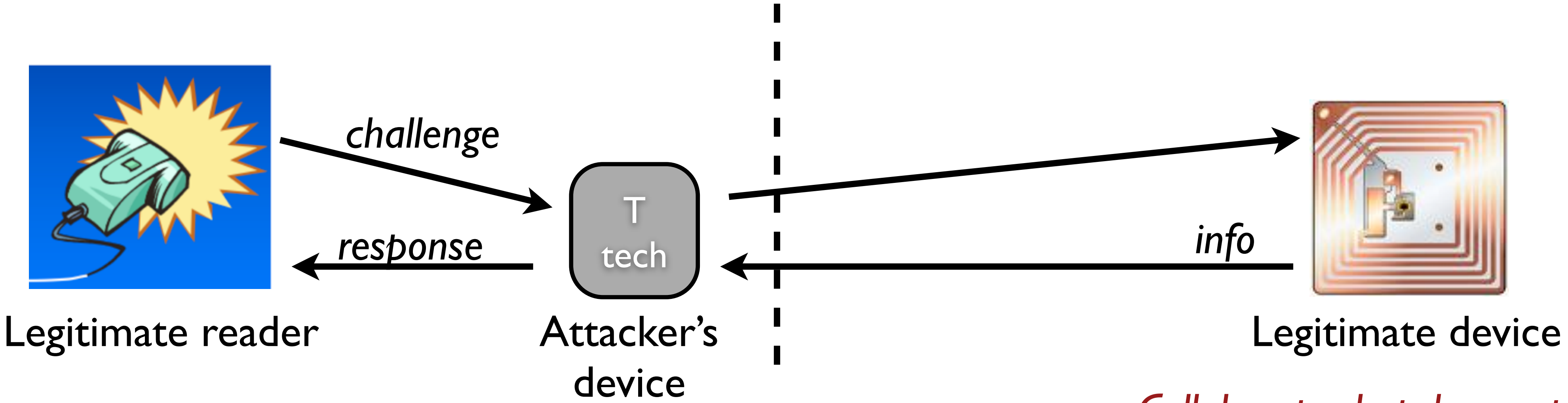
- *Correct response*
- *from legitimate tag*
- *... but attacker gets access!*

Famous urban myth: Mig-in-the-middle attack



N?
SAAF
{N}K
ANGOLA
{N}K
N?
MIG
N?
{N}K
SAAF
NAMIBIA

Relay attacks: Terrorist Fraud



Collaborates but does not reveal long-term secrets

More powerful than Mafia fraud:

- *legit device does not have to be tricked,*
- *... does not have to follow the protocol*
- *... can provide more info than just response*

Countermeasures

What to do against relay attacks?

- Ask the prover where he is
  - but he could be lying
- Signal strength
  - can be spoofed
- Measure the distance to the prover
  - "distance bounding"
  - nothing travels faster than light $c = 2.99792458 \cdot 10^8$ m/s
  - infer distance from traveling time of signal

300 meters per microsecond

Distance bounding

Demand response within time t_{max}

- travel time to distance x_{max} and back
- allow some "slack" time for computations
- dist. measurement & proof of knowledge at the same time

$$t_{\max} = 2\frac{x_{\max}}{c} + t_{\text{slack}}$$

$$x_{\text{spoofable}} = \tfrac{1}{2}c\,t_{\max} = x_{\max} + \tfrac{1}{2}c\,t_{\text{slack}}$$

has to be very small

Distance bounding: practical problems

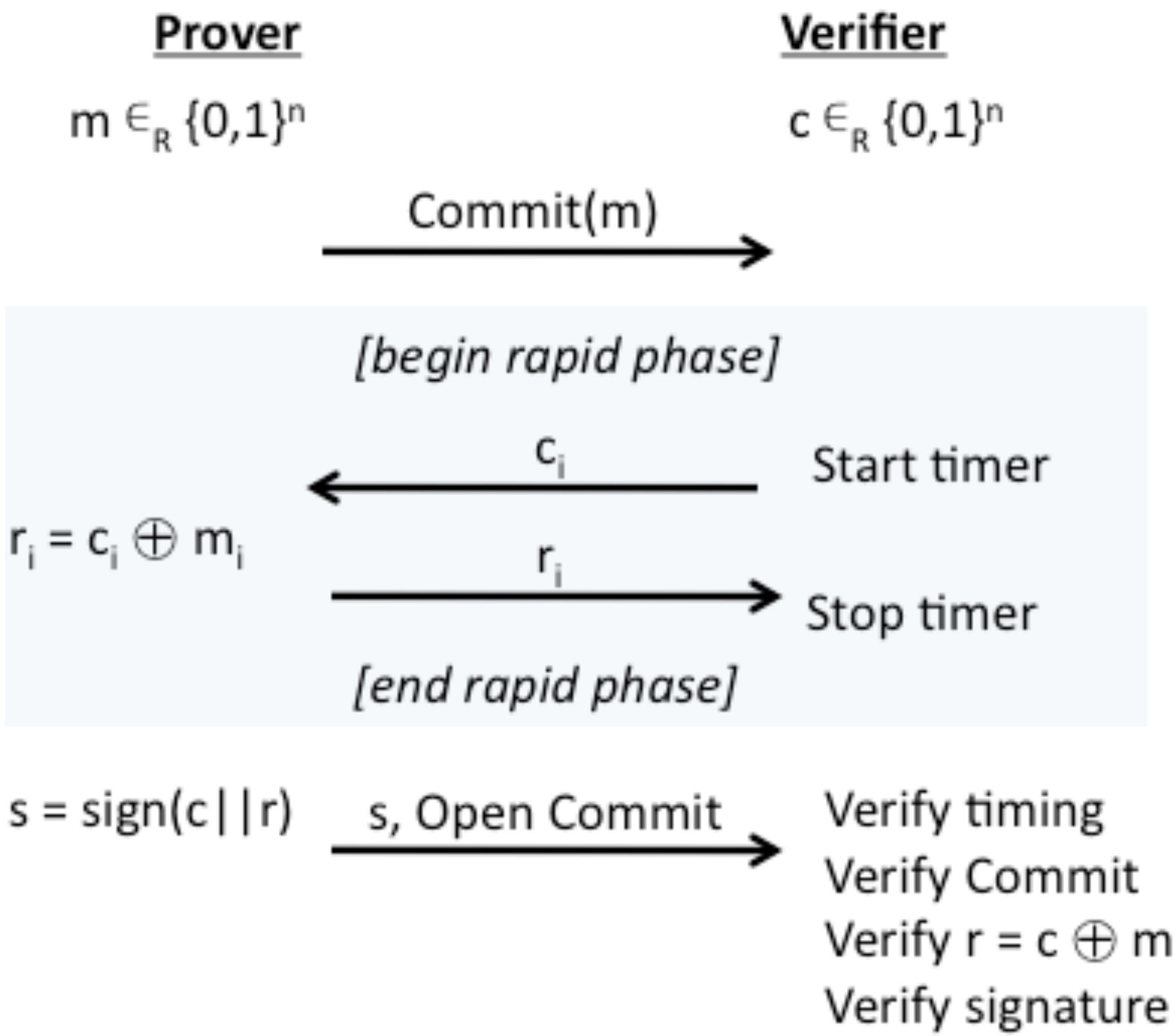
t_{slack} must be very small

- no (heavy) computations
  - addition lasts too long
  - *but still cryptographic challenge-response protocol !*
- delays inside prover device become problematic
  - missed cycles, bus speed, etc
- no error correction
  - we have to live with transmission errors

Solving the practical problems

- no (heavy) computations
  - split protocol into slow and quick phase
  - prover creates LUT in slow crypto phase
  - verifier: unpredictable selection from LUT in quick phase
- delays inside prover
  - LUT sitting right "next to" emitter
- no error correction
  - decide afterwards if there were transmission errors

The Brands-Chaum protocol



Question time

Do you see a problem with the Brands-Chaum protocol ?

Swiss Knife protocol (2008)

<u>Reader</u> has DB $\{ID, x\}$		<u>Tag</u> (ID, x)
Random N_A; random d (Hamm.weight m)	$\xrightarrow{N_A,\, d}$	Random N_B
	$\xleftarrow{N_B}$	$Z^0 = f_x(C_B, N_B)$; $Z^1 = Z^0 \oplus x$; For $i = 1$ to m { j = index of next 1 in d; $R^0{}_i = Z^0{}_j$; $R^1{}_i = Z^1{}_j$ }

Rapid bit exchange

For $i = 1$ to m

Random bit c_i ; start clock	$\xrightarrow{c_i'}$	$r_i = \begin{cases} R^0{}_i & \text{if } c_i' = 0 \\ R^1{}_i & \text{if } c_i' = 1 \end{cases}$
Stop clock; store Δt_i	$\xleftarrow{r_i}$	

Find matching (ID, x) in DB; compute R^0, R^1; $err_c = \#\{i: c_i' \neq c_i\}$; $err_r = \#\{i: c_i' = c_i \wedge r_i \neq R^{c_i}{}_i\}$; $err_t = \#\{i: c_i' = c_i \wedge \Delta t_i > \Delta t_{max}\}$; if $err_c + err_r + err_t \geq T$ reject;	$\xleftarrow{t_B,\, c_1', \dots, c_m'}$	$t_B = f_x(c_1', \dots, c_m', ID, N_A, N_B)$
$t_A = f_x(N_B)$	$\xrightarrow{t_A}$	Check t_A

Question time

Why is the Swiss Knife protocol
secure against the Mafia Fraud ?

Is it secure against the Terrorist Fraud?

Still too slow!

State of the art hardware:

- analog → digital conversion: 50 ns
- all conversion steps together: 170 ns
  (26 meters)

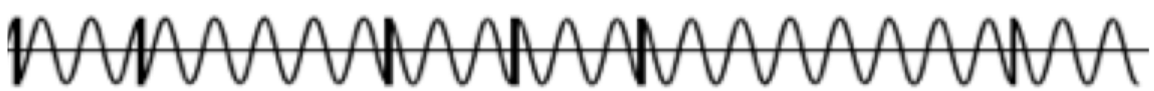
Only analog processing is fast enough!

Analog challenge-response

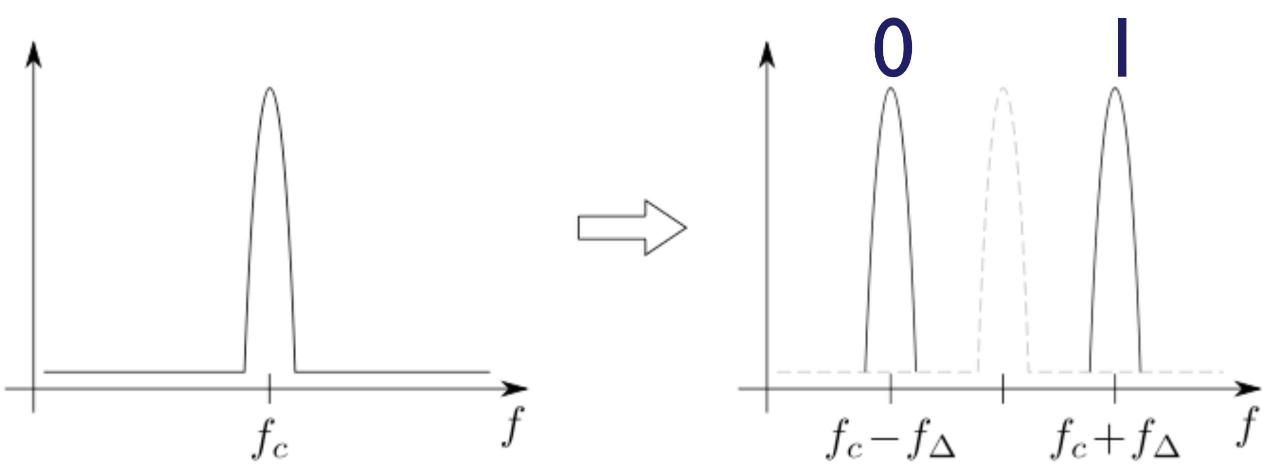
Rasmussen & Čapkun 2010

- Adaptation of Brands-Chaum
- Single register R.
- **CRCS**: *Challenge Reflection with Channel Selection.*

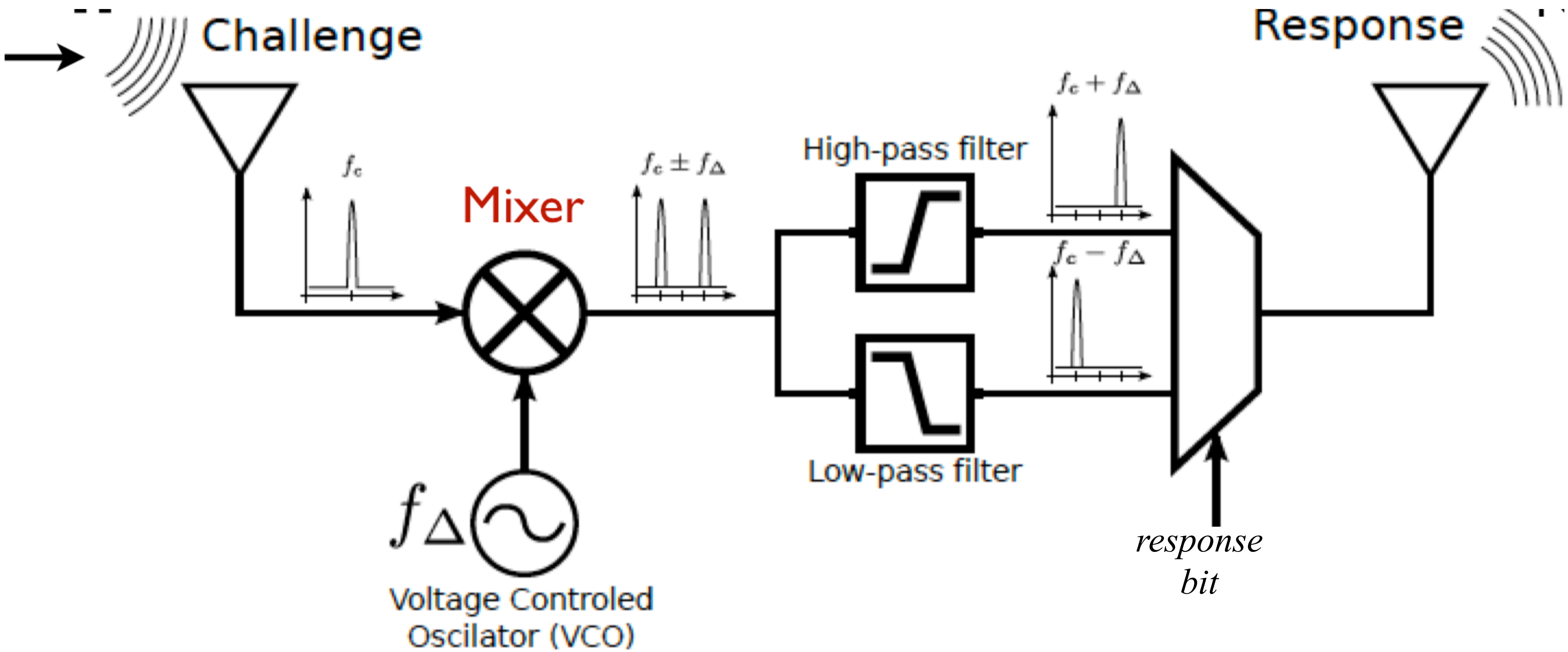
Challenge: unpredictable signal c(t) at frequency f_c



Response: reflection of c(t) at shifted frequency



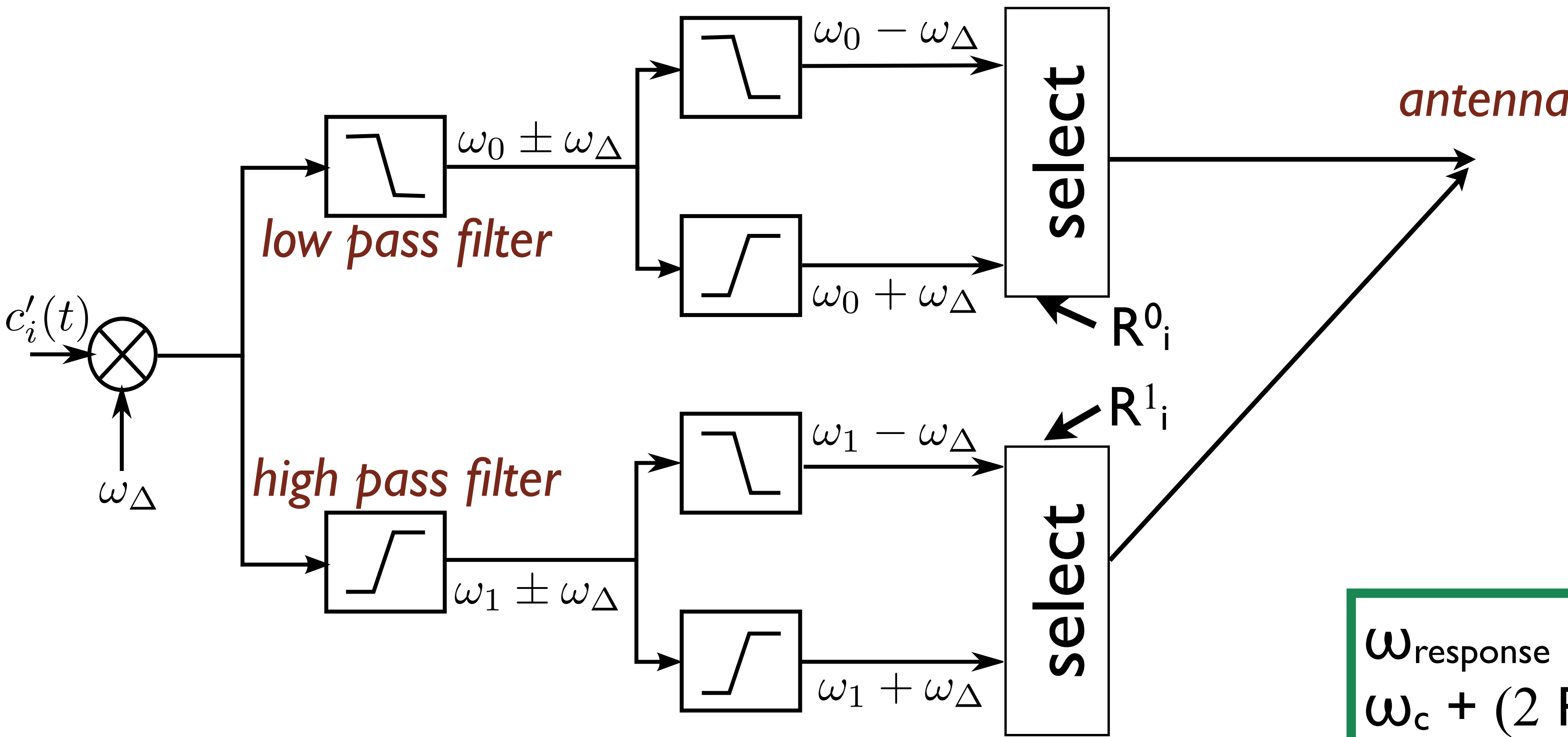
Challenge Reflection with Channel Selection



< 1 nanosecond !

Analog version of Swiss Knife rapid phase

- Challenge: $c_i(t)$ at frequency ω_c (ω_0 or ω_1).
- Response: reflection of $c_i(t)$ at shifted frequency; shift depends on R^c_i.



$\omega_{response}$ =
$\omega_c + (2\, R^c_i - 1)\, \omega_\Delta$