Security of Wireless Networks
Lecture 2

Srdjan Čapkun
(delivered by Aanjhan Ranganathan)
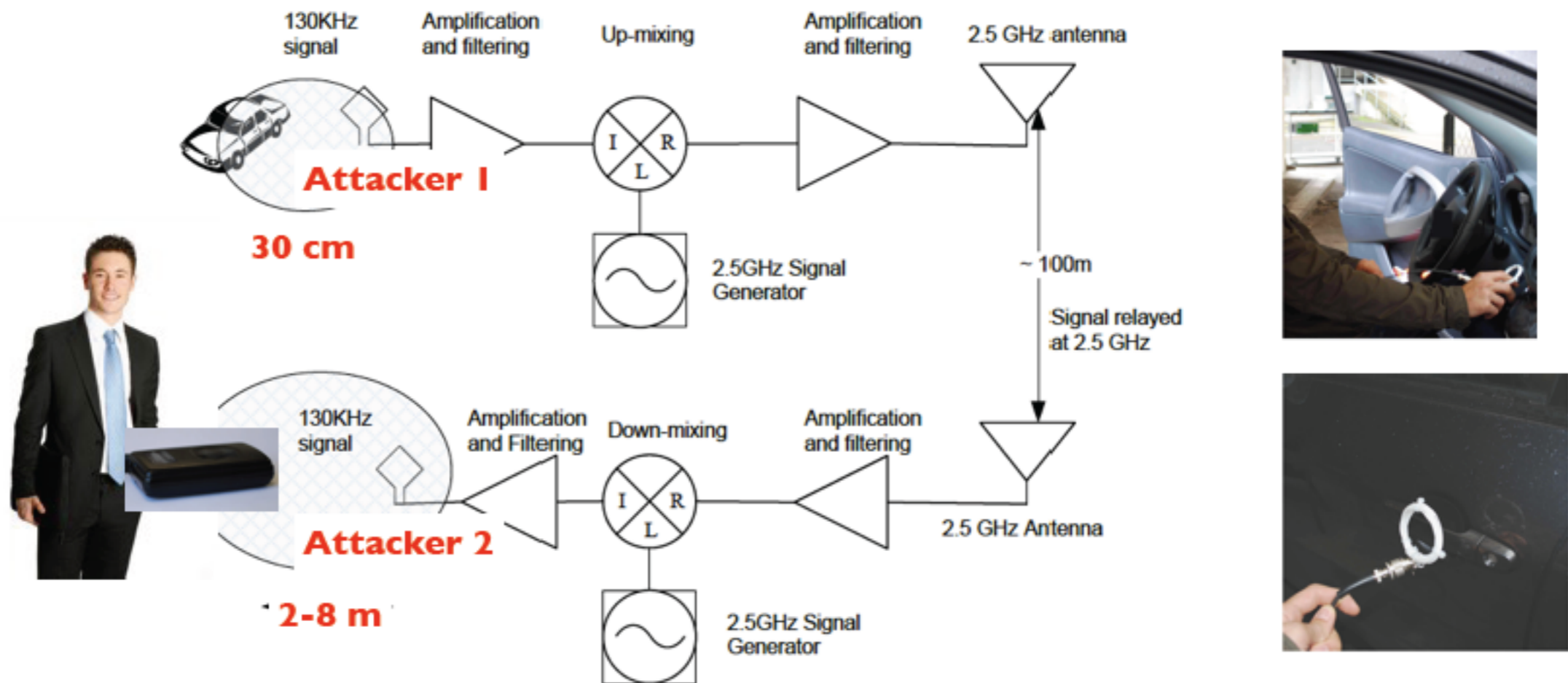*Department of Computer Science*
ETH Zurich

Secure Proximity Verification

Example: PKES
(deployed by all major car manufacturers)

PKES: Key is "in pocket" - car opens when the key is *close to the car*
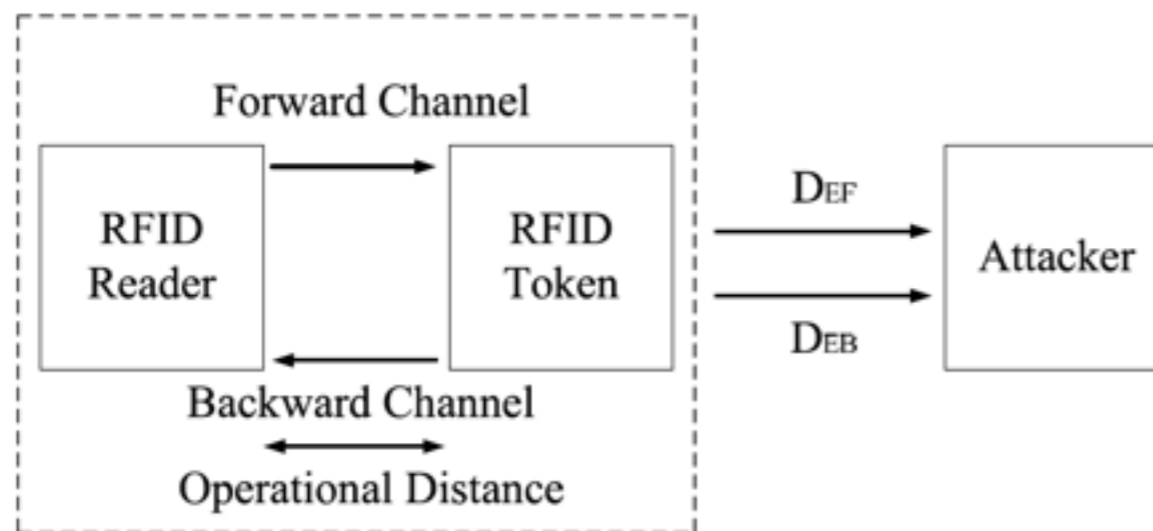
- *Relay attack* *[FrancillonNDSS11]*

- Tested on 10 car models from 8 manufacturers
- Manufacturers are now redesigning Entry and Start Systems

Example: RFID / NFC communication

Do LF/HF RFID/NFC systems provide guarantees on the communication range?

- HF RFID, ISO 14443 and ISO 15693 [Hancke10]

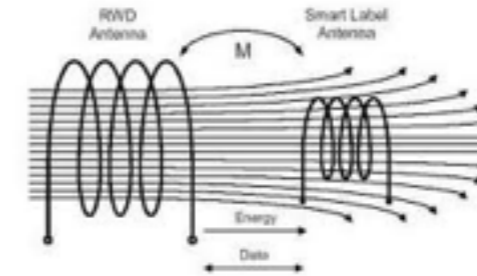|	ISO 14443A	ISO 14443B	ISO 15693
Entrance hall			
1 m	FB	FB	FB
2 m	FB	FB	FB
3 m	Fx	xB	Fx
4 m	Fx	xx	Fx
5 m	Fx	xx	Fx
Lab corridor			
1 m	FB	FB	FB
2 m	FB	FB	FB
3 m	FB	FB	Fx
4 m	Fx	xB	Fx
5 m	Fx	xx	Fx

Table 1: Eavesdropping results: F – Forward channel recovered, B – Backward channel
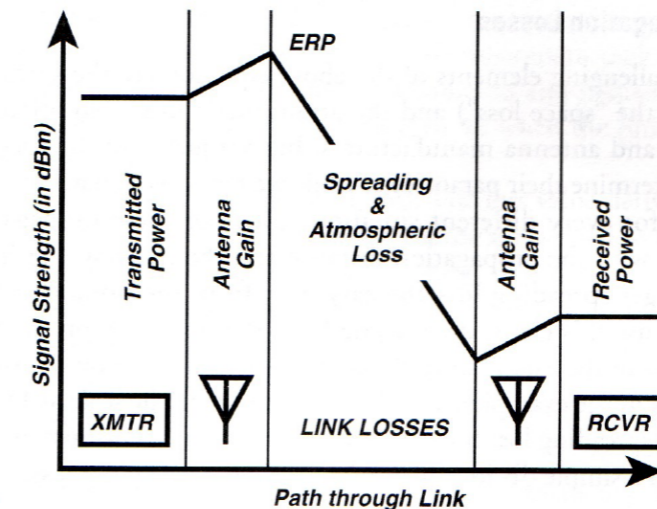
Secure Proximity Verification?

Secure Proximity Verification

- Inductive Coupling

- Radio Communication

Communication DOES NOT imply physical proximity.
(in adversarial environments)

To calculate the received signal level (in dBm), add the transmitting antenna gain (in dB), subtract the link losses (in dB), and add the receiving antenna gain (in dB) to the transmitter power (in dBm).

©D. Adamy, A First Course on Electronic Warfare

As shown in PKES systems, relying on the reduced communication range is either not convenient or not secure.
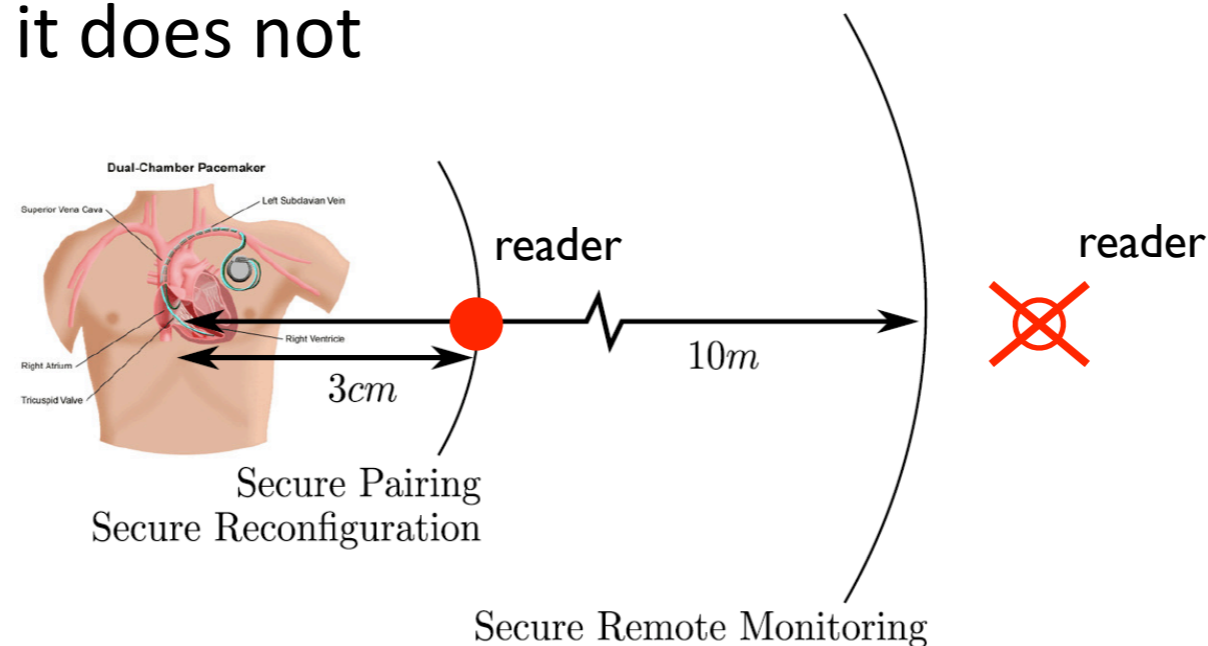
- *We need a difficult problem to hold on to.*

Solution: Secure Proximity Verification **using secure ranging**.

Secure Proximity Verification

One (untrusted) device wants to *prove to be close* to another device.

- e.g., if a reader is close to the pacemaker, it gets access, otherwise it does not

Dual-Chamber Pacemaker

3cm — Secure Pairing / Secure Reconfiguration
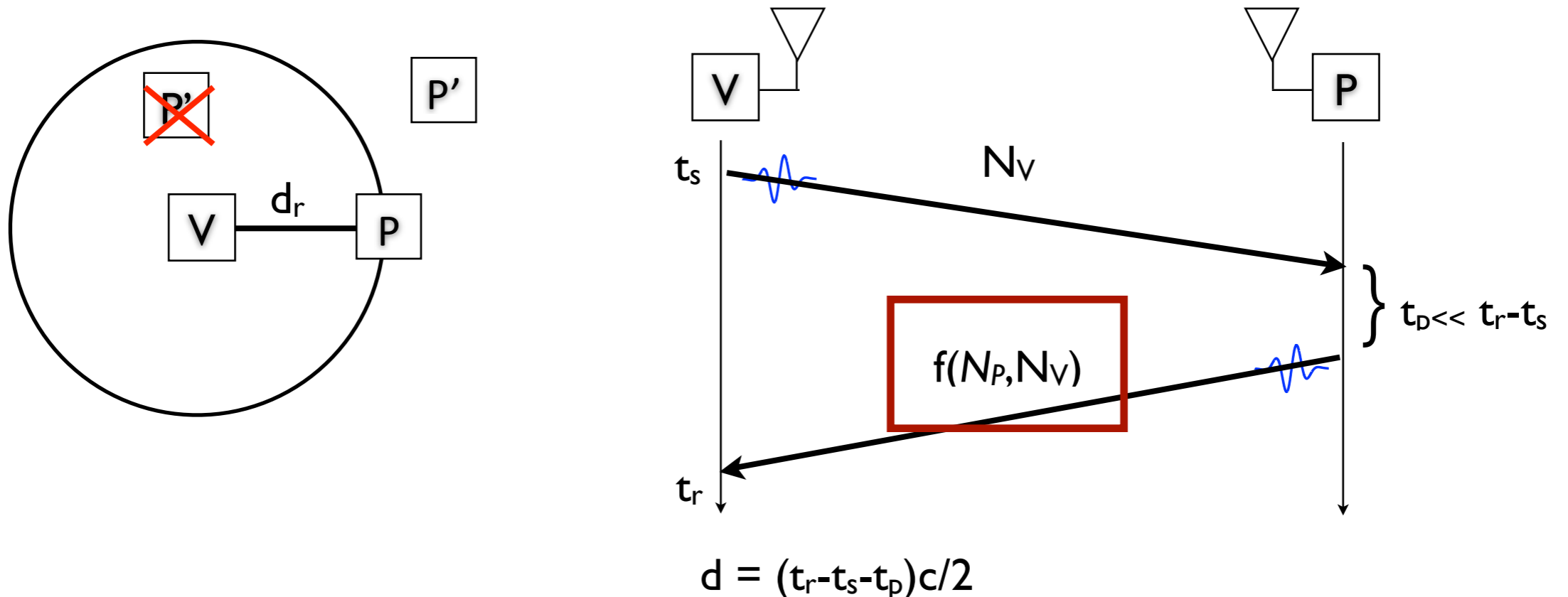
10m

reader

reader

Secure Remote Monitoring

Two devices want to *verify if they are indeed close.*

- e.g., a car and a key want to verify if they are physically close

1m
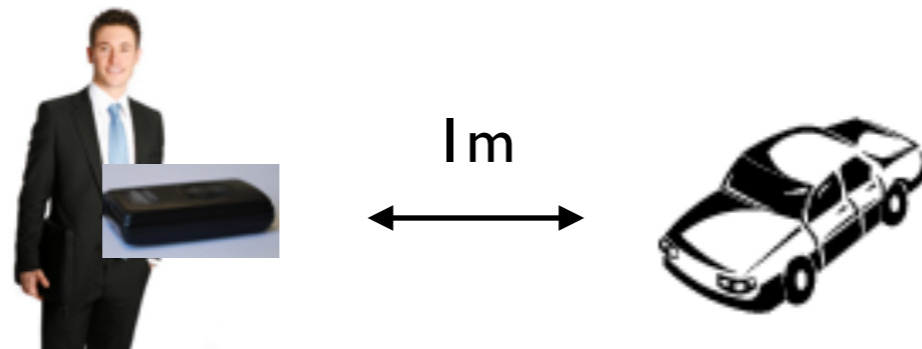
Distance Bounding [BrandsChaum93]

Basic Idea

$$d = (t_r - t_s - t_D)c/2$$

Property:

Measured distance *d* should be an *upper bound* on the true distance d_r between V and P.

Distance Bounding [BrandsChaum93]

msc Signature-based Brands and Chaum protocol

Prover	Verifier
P	V

$m_i \in_{\mathcal{R}} \{0,1\}$

$commit(m_1,\ldots,m_k)$

$\alpha_i \in_{\mathcal{R}} \{0,1\}$

Rapid bit exchange

α_i

$\beta_i \leftarrow \alpha_i \oplus m_i$

β_i

$c \leftarrow \alpha_1|\beta_1|\cdots|\alpha_k|\beta_k$

(open commit), $sign(c)$

Verify commit
$c \leftarrow \alpha_1|\beta_1|\cdots|\alpha_k|\beta_k$
verify $sign(c)$
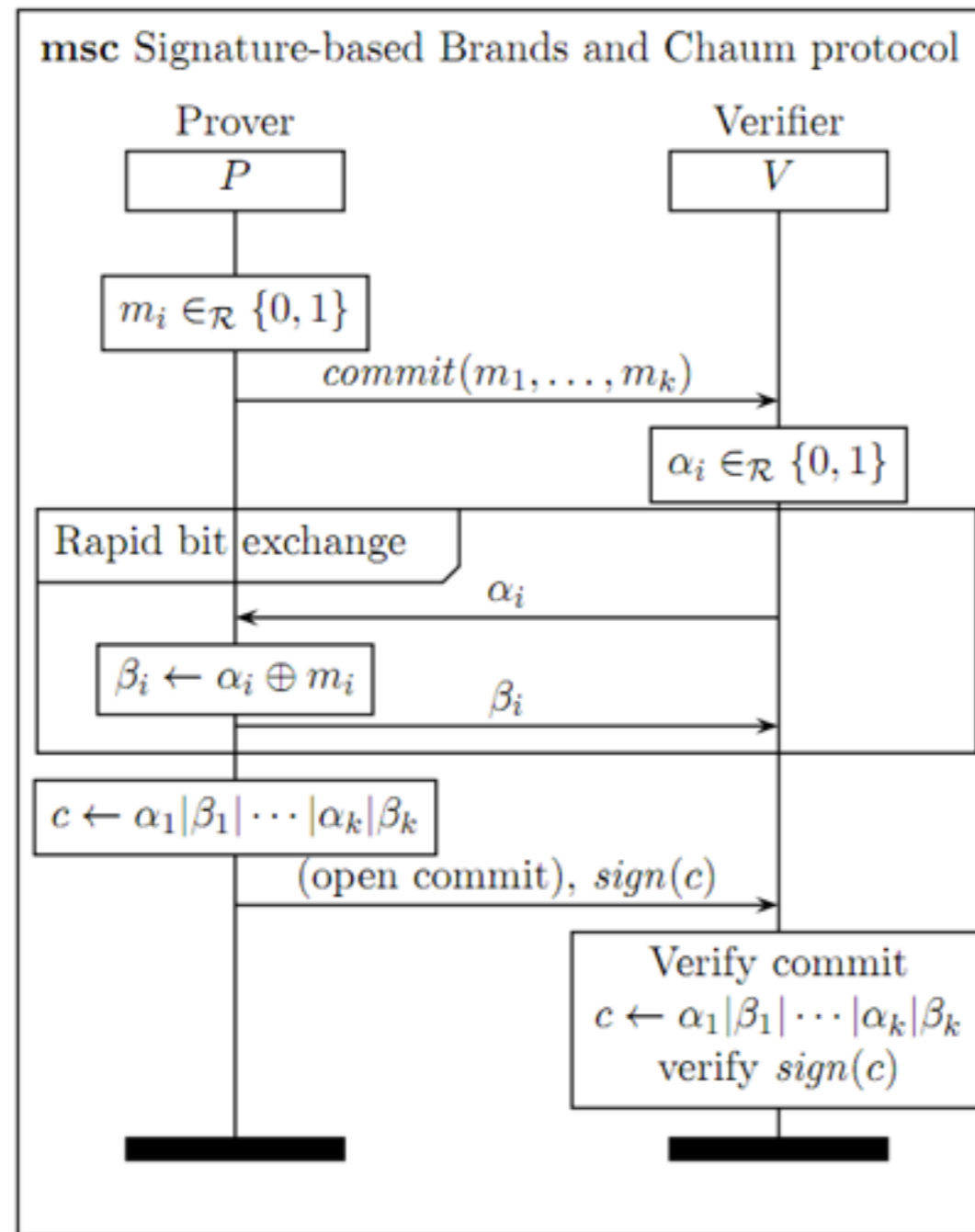
Distance Bounding: *f() and t_p*

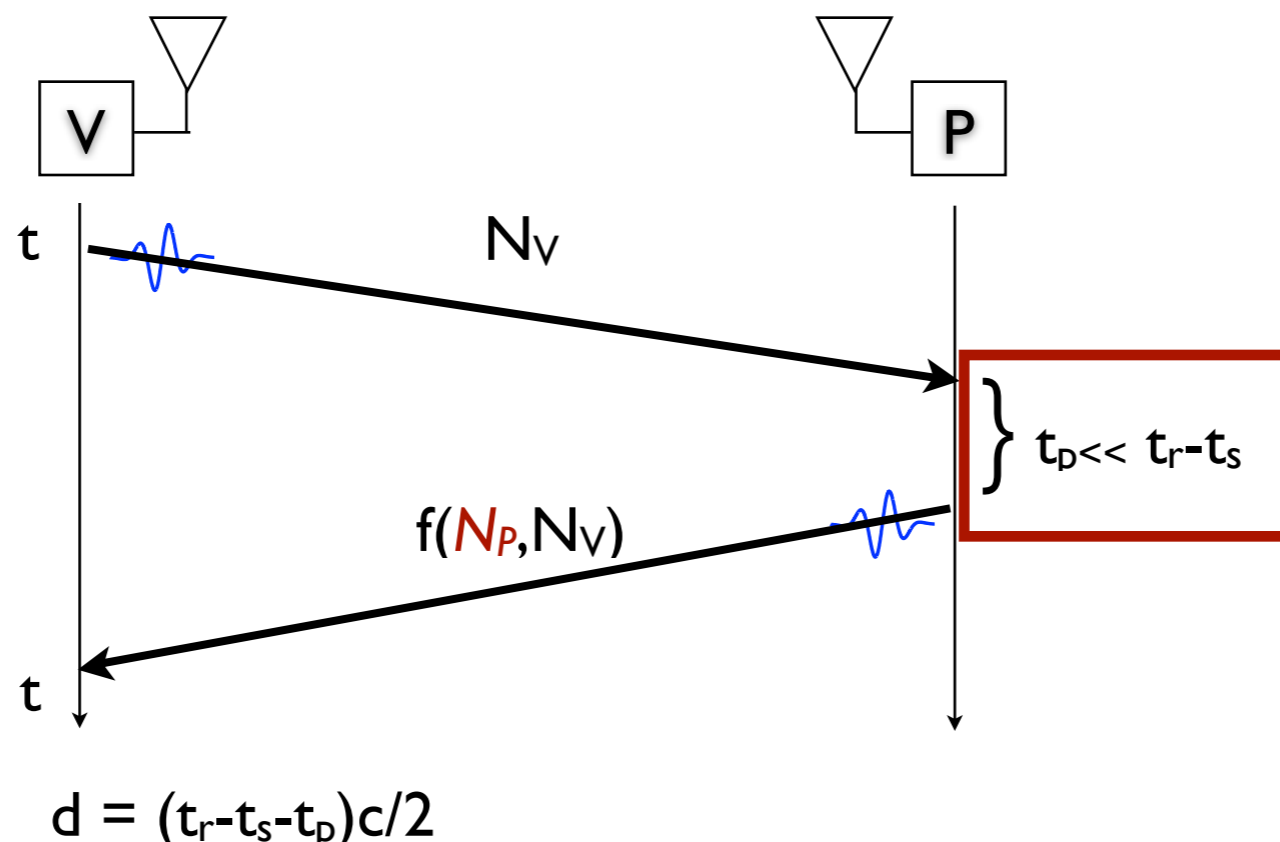Provers should ***quickly*** *receive N_V, compute $f(N_V,N_P)$ and send $f(N_V,N_P)$*

- The verifier estimates prover's processing = t_p
- If attacker's processing = 0 then he *can cheat by $t_p/2$*
- Thus ideally $t_p=0s$, in most applications $t_p=1\text{-}2ns$ (15-30cm)
- t_p needs to be *stable and* ***short***

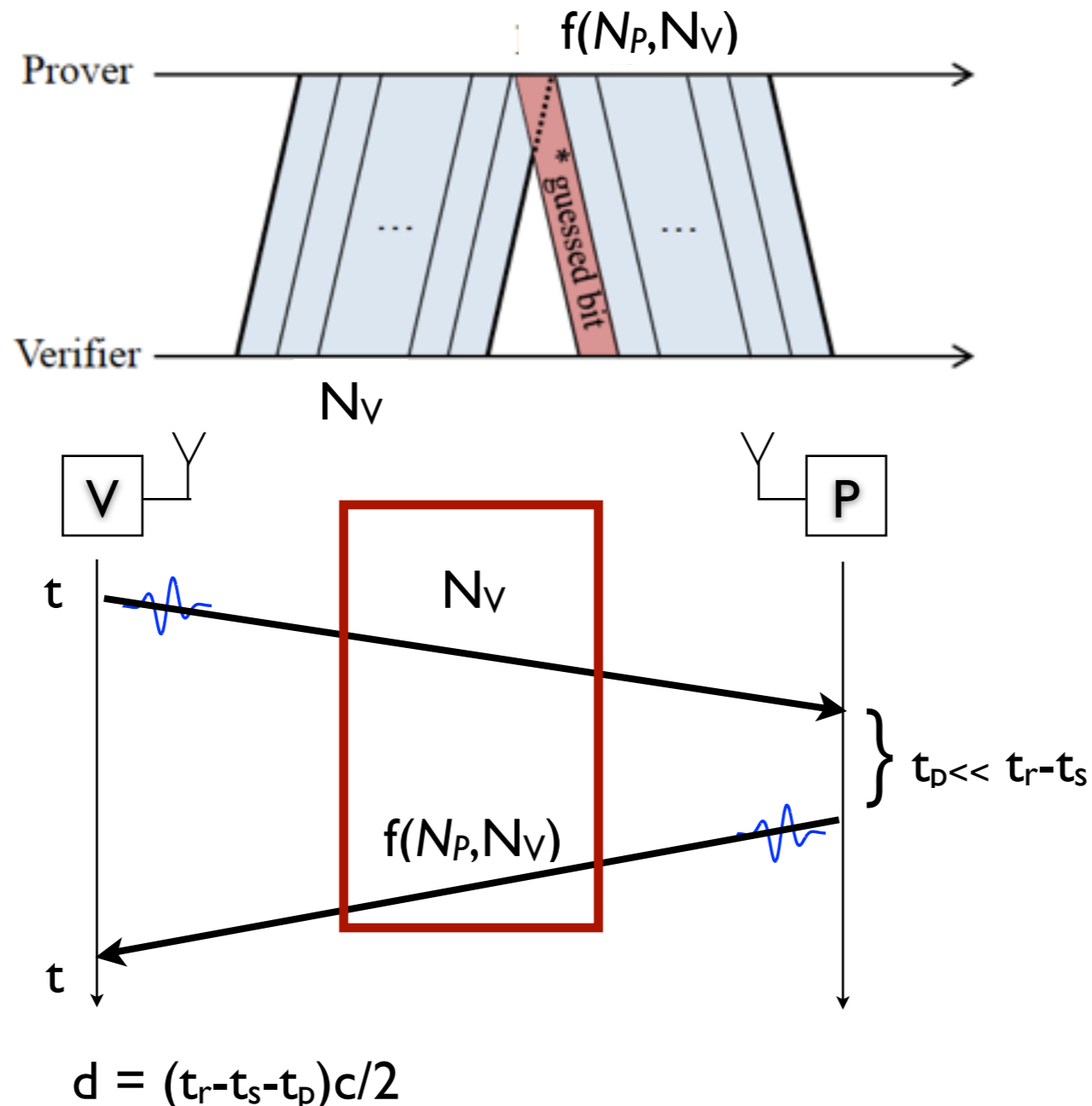*Main assumption: we do not control the prover*

$$d = (t_r-t_s-t_p)c/2$$

Distance Bounding: N_V

N_V and f(NV,NP) should be "short" in the # of bits [HankeKuhn]

- short compared to the required accuracy / security

$$d = (t_r - t_s - t_p)c/2$$

Distance Bounding: *symbols*

Assuming $|N_V|=1bit$, the symbols should be short as well

- short compared to the required accuracy / security
- Early Detection
- Late Commit
- Note: *channel spread does not help*

symb

t_{symb}

t_{detect}

ETH Zürich
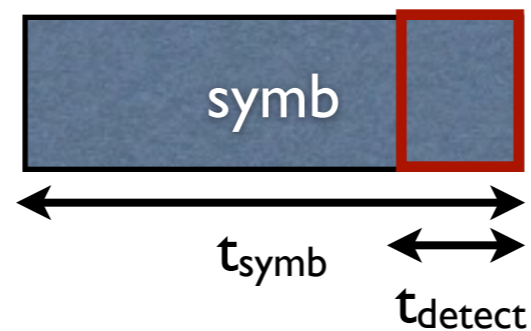
Distance Bounding: *symbols*

Assuming $|N_V|$=1bit, the symbols should be short as well

- short compared to the required accuracy / security
- Early Detection
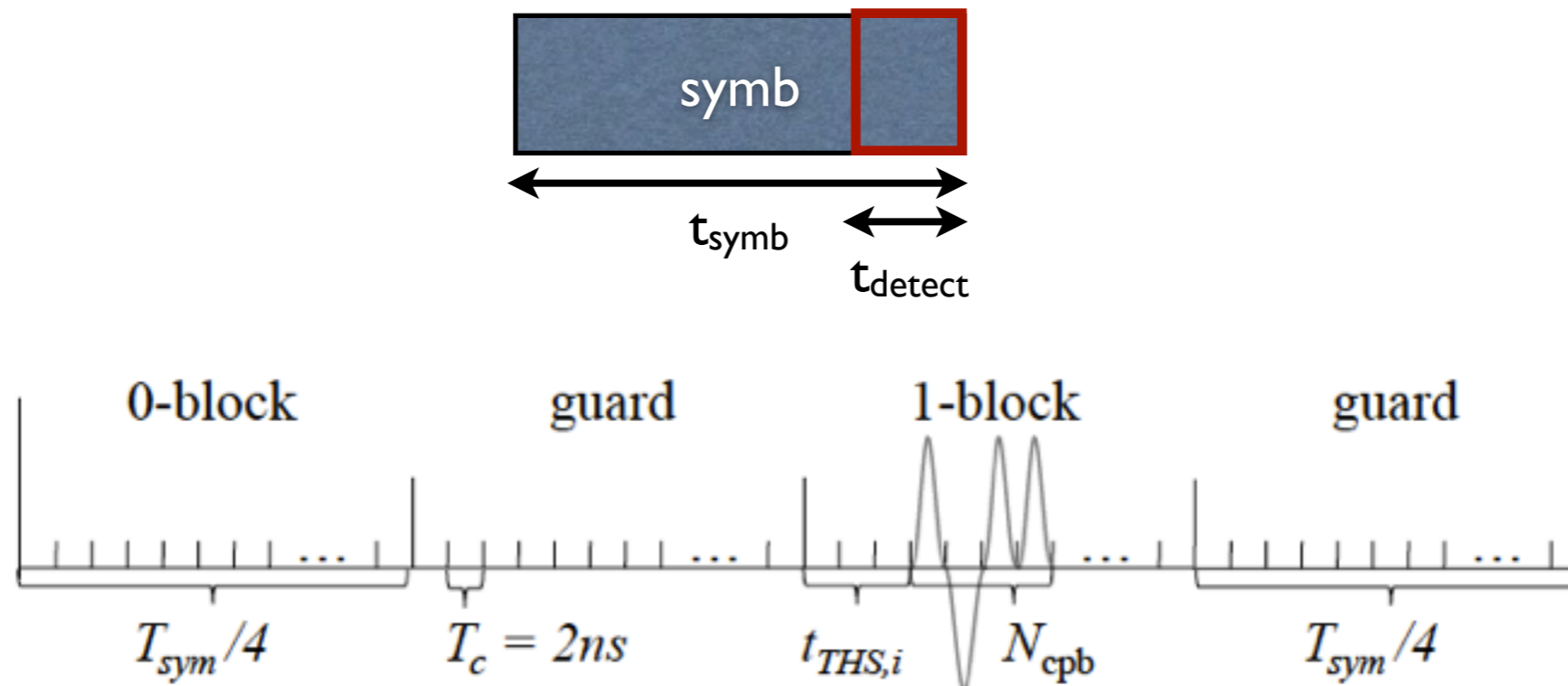- Late Commit
- Note: *channel spread does not help*

Figure 4.2: IEEE 802.15.4a data symbol structure [Poturalski2011]

Distance Bounding
## *experiments on 802.15.4a (IR UWB)*

[Poturalski2011]

		No guessing		Max. guessing gain	
		(relay) time-gain	distance-decrease	(relay) time-gain	distance-decrease
En.D. against En.D.					
Malicious Prover	ED-only	$T_{\text{sym}}/4 + (t_{\text{det}} - t_{\text{det}}^{\text{A}})/2$	86m	$+ t_{\text{THS}}^{\text{max}}$	$+74$m
	LC-only	$T_{\text{sym}}/4 + t_{\text{PLC}}/2$	86m	$+ t_{\text{THS}}^{\text{max}}$	$+74$m
	ED+LC	$T_{\text{sym}}/2 + (t_{\text{PLC}} + t_{\text{det}} - t_{\text{det}}^{\text{A}})/2$	171m	$+ t_{\text{THS}}^{\text{max}}$	$+74$m
Relay Attack	ED+LC	$T_{\text{sym}}/2 + t_{\text{PLC}} - t_{\text{det}}^{\text{A}}$	171m	$+0$	$+0$m
Rake against En.D.					
Malicious Prover	ED-only	$T_{\text{sym}}/2 + (t_{\text{det}} - t_{\text{det}}^{\text{A}})/2$	162m	$+ T_{\text{sym}}/4 + t_{\text{THS}}^{\text{max}}$	$+151$m
	ED+LC	$3/4 \cdot T_{\text{sym}} + (t_{\text{PLC}} + t_{\text{det}} - t_{\text{det}}^{\text{A}})/2$	248m	$+ T_{\text{sym}}/4 + t_{\text{THS}}^{\text{max}}$	$+151$m
Relay Attack	ED+LC	$T_{\text{sym}} - t_{\text{THS}}^{\text{max}} + t_{\text{PLC}} - t_{\text{det}}^{\text{A}}$	251m	$+ T_{\text{sym}}/2 + 2 \cdot t_{\text{THS}}^{\text{max}}$	$+302$m
	ED-only	$T_{\text{sym}}/2 - t_{\text{THS}}^{\text{max}} - t_{\text{det}}^{\text{A}}$	79m	$+ T_{\text{sym}}/2 + 2 \cdot t_{\text{THS}}^{\text{max}}$	$+302$m
Rake against Rake					
Malicious Prover	ED-only	$(t_{\text{det}} - t_{\text{det}}^{\text{A}})/2$	5m	$+ T_{\text{sym}}/4 + t_{\text{THS}}^{\text{max}}$	$+151$m
	LC-only	$t_{\text{PLC}}/2$	5m	$+ T_{\text{sym}}/4 + t_{\text{THS}}^{\text{max}}$	$+151$m
	ED+LC	$(t_{\text{det}} + t_{\text{PLC}} - t_{\text{det}}^{\text{A}})/2$	10m	$+ T_{\text{sym}}/2 + t_{\text{THS}}^{\text{max}}$	$+228$m
Relay Attack	ED+LC	$t_{\text{PLC}} - t_{\text{det}}^{\text{A}}$	10m	$+0$	$+0$m

Table 4.2: Upper-bound on (relay) time-gain and (relay) distance-decrease of various PHY attacks in various "adversarial receiver against honest receiver" configurations. The left column presents conservative attacks, that work with 100% success probability. The right column presents the maximal additional time-gain/distance-decrease that can be achieved by combining PHY attacks and guessing attacks (when time guessing probability approaches the guessing probability of pure guessing attacks). Time-gain is expressed in terms of T_{sym} – data symbol duration, $t_{\text{det}} = 48\text{-}60ns$ – detection time of honest receivers without ED-countermeasure, $t_{\text{det}}^{\text{A}}$ – detection time of the adversary, $t_{\text{PLC}} < t_{\text{det}}$ – pulse LC delay, $t_{\text{THS}}^{\text{max}}$ – maximum time-hopping offset. The distance-decrease is shown for the IEEE 802.15.4a mandatory modes and delay values that maximize the distance-decrease.

Distance Bounding: *symbols*

Chirp SS ranging (802.15.4) systems strongly affected*

- long symbol lengths allow for simple ED and LC attacks
- Early Detection
- Late Commit

t_{ED}

Commit at least $T_{chirp} - t_{LC}$
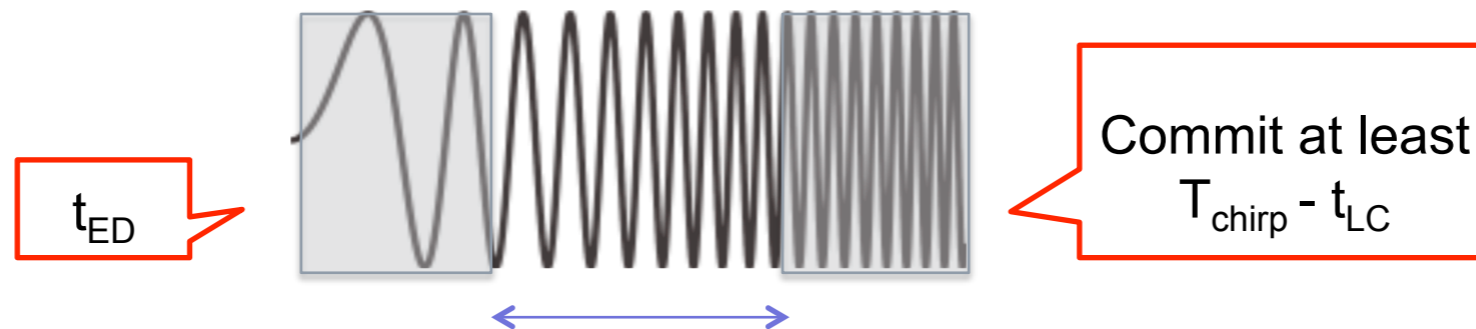
$$t_{GAIN} = t_{LC} - t_{ED} - t_{HW}$$

$$D = c \times t_{GAIN}$$

*[RanganathanWisec12]

ETH Zürich

Realization of RF Distance Bounding:
Processing Function $f(N_v, N_p)$

f(Nv,Np) is computed by the prover:

- takes as input Nv (received from the verifier)
- takes as input Np (locally generated by the prover)
- Should allow that the prover: *receives Nv, computes and outputs f(Nv,Np)* **in a short time (few ns)**

DB protocols in the literature:

[BethDesmedt] sign(N_V); h(N_V); mac(N_V); E(N_V); ... => t_p >> ns

[BrandsChaum, *CapkunInfocom05*, ...] *XOR* => t_p = ?

[HanckeKuhn, *TippenhauerESORICS09*, ...] *bit comparison* => t_p = ?

> 20 proposed protocols, not one was *fully* implemented

Can the proposed DB protocols be realized?

ETH Zürich

Realization of RF Distance Bounding:
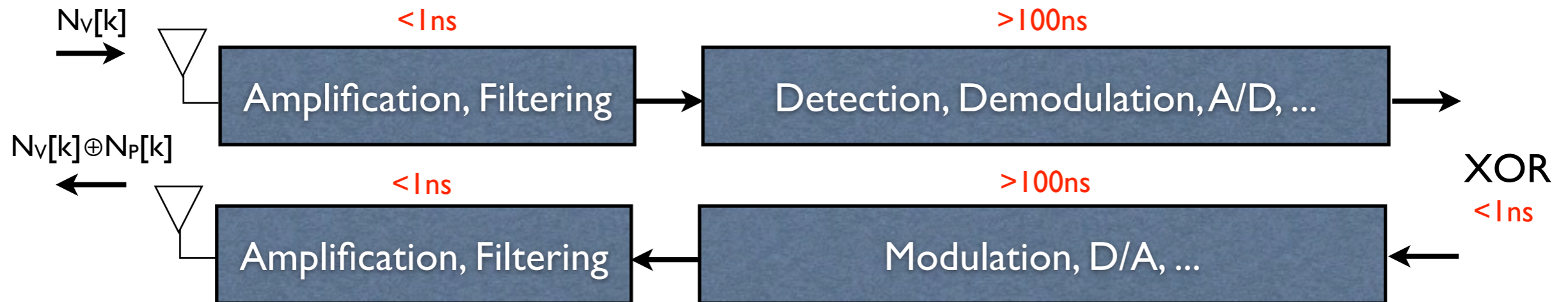## *Processing Function $f(N_v, N_p)$*

[BethDesmedt] sign(); h(); mac(); E(); ... => t_p >> ns

[**BrandsChaum**, ...] *XOR* => t_p = ? (nx100ns ?)

[**HanckeKuhn**, ...] *bit comparison* => t_p = ? (nx100ns ?)

[RasmussenSec09, ...] *CRCS (analog modulation)* => t_p < 1ns

... > 20 proposed protocols

Can we use functions that require interpretation (demodulation) Nv ?

Realization of RF Distance Bounding:
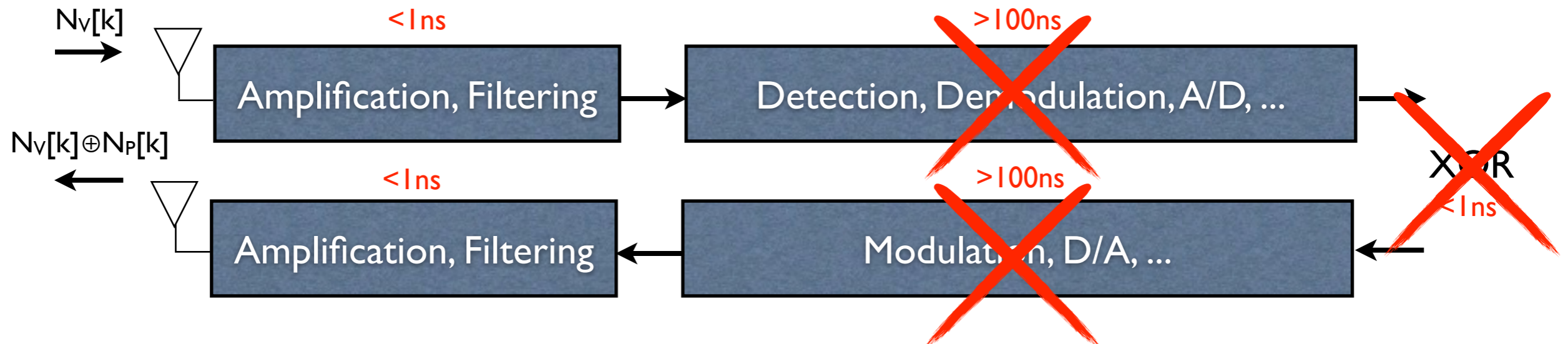## *Processing Function $f(N_v, N_p)$*

[BethDesmedt] sign(); h(); mac(); E(); ... => t_p >> ns

[**BrandsChaum**, ...] *XOR* => t_p = ? (nx100ns ?)

[**HanckeKuhn**, ...] *bit comparison* => t_p = ? (nx100ns ?)

[RasmussenSec09, ...] *CRCS (analog modulation)* => t_p < 1ns

... > 20 proposed protocols

$N_V[k]$ <1ns   >100ns

| Amplification, Filtering | Detection, Demodulation, A/D, ... |

$N_V[k] \oplus N_P[k]$ <1ns   >100ns   XOR   <1ns

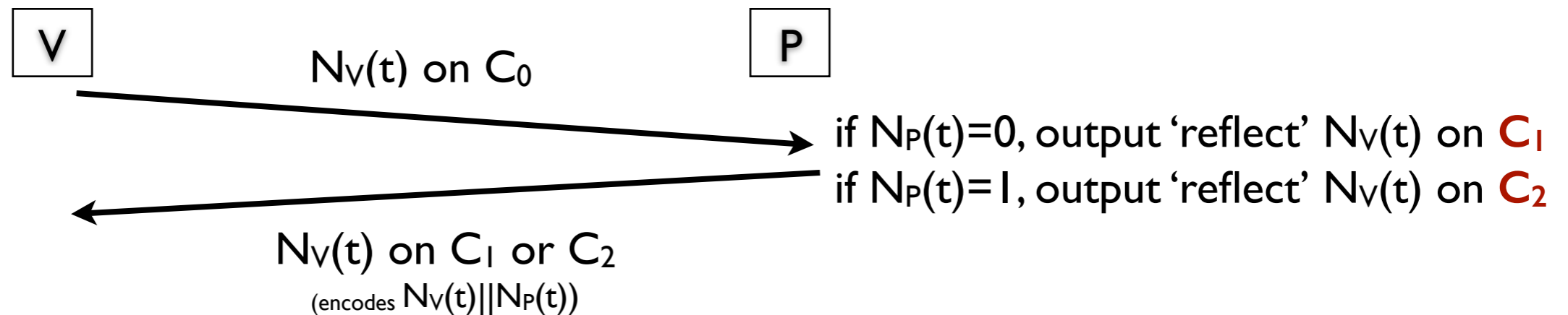| Amplification, Filtering | Modulation, D/A, ... |

Can we use functions that require interpretation (demodulation) Nv ?

ETH Zürich

A new Function: CRCS

Our approach: *Challenge Reflection with Channel Selection*

- Prover does not interpret Nv
- All *time-critical* processing is done in *analog*
- Verifier does "all the work"

Main idea (C_0, C_1, C_2 are channels)

V		P

$N_V(t)$ on C_0

if $N_P(t)=0$, output 'reflect' $N_V(t)$ on C_1

if $N_P(t)=1$, output 'reflect' $N_V(t)$ on C_2

$N_V(t)$ on C_1 or C_2

(encodes $N_V(t)\|N_P(t)$)

A new Function: CRCS

Our approach: *Challenge Reflection with Channel Selection*

- Prover does not interpret Nv
- All *time-critical* processing is done in *analog*
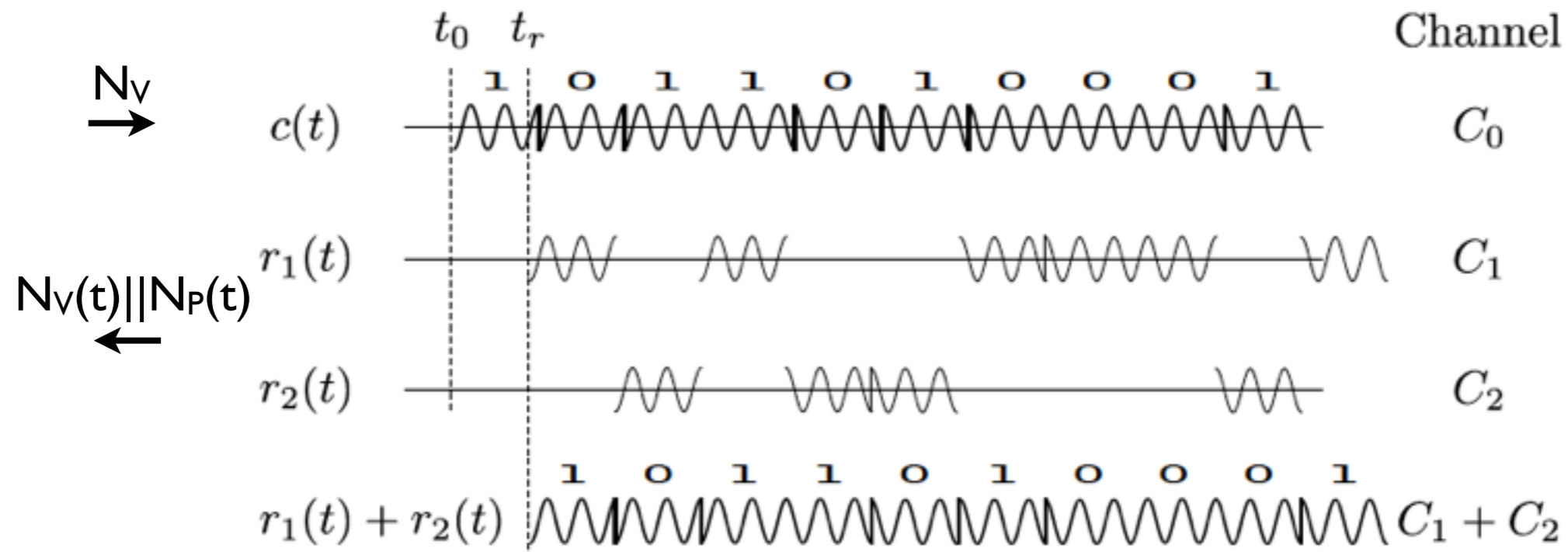- Verifier does "all the work"

Main idea (C_0, C_1, C_2 are channels)

A new Function: CRCS

Our approach: *Challenge Reflection with Channel Selection*

- Prover does not interpret N_V
- All *time-critical* processing is done in *analog*
- Verifier does "all the work"

Main idea (C_0, C_1, C_2 are channels)

A new Function: CRCS

Implementation of CRCS

Mixer up+down converts the input signal

A new Function: CRCS

Implementation of CRCS

$t_p < 1ns$, st. dev. 61ps, full duplex
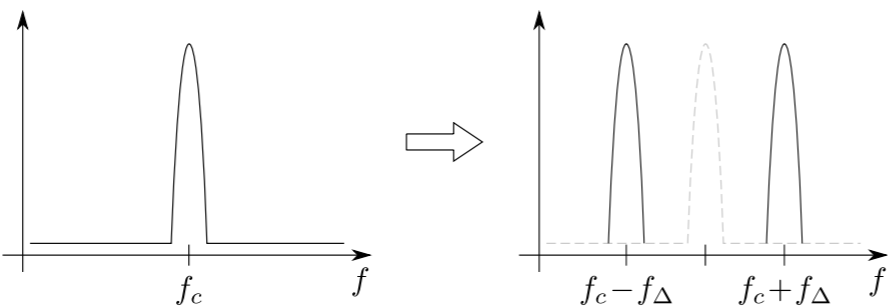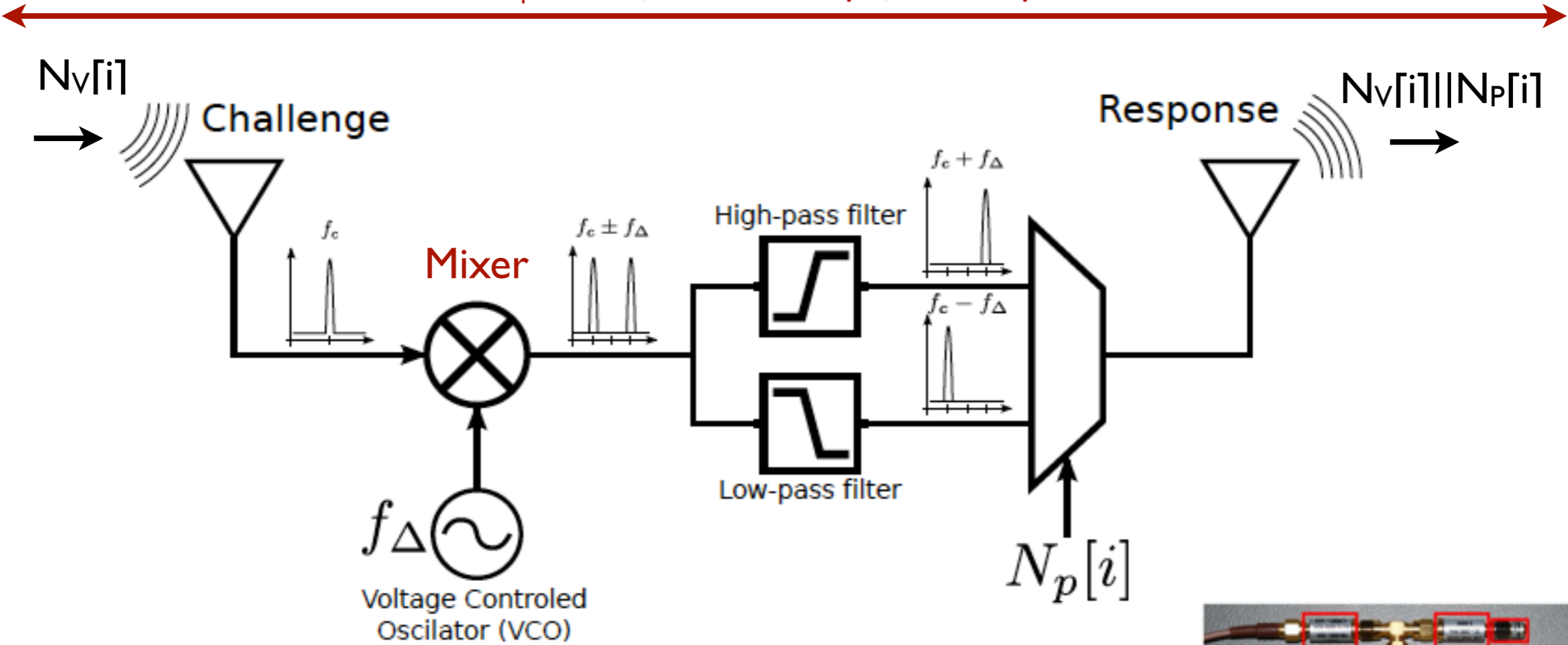
Mixer up+down converts the input signal
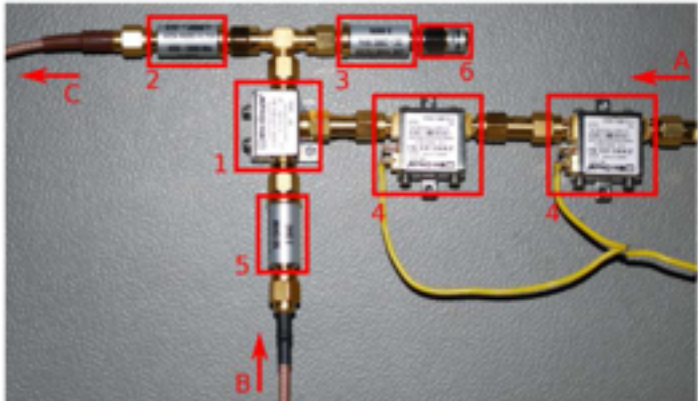
A new Function: CRCS

CRCS++ (measured at the input/output of the prover)
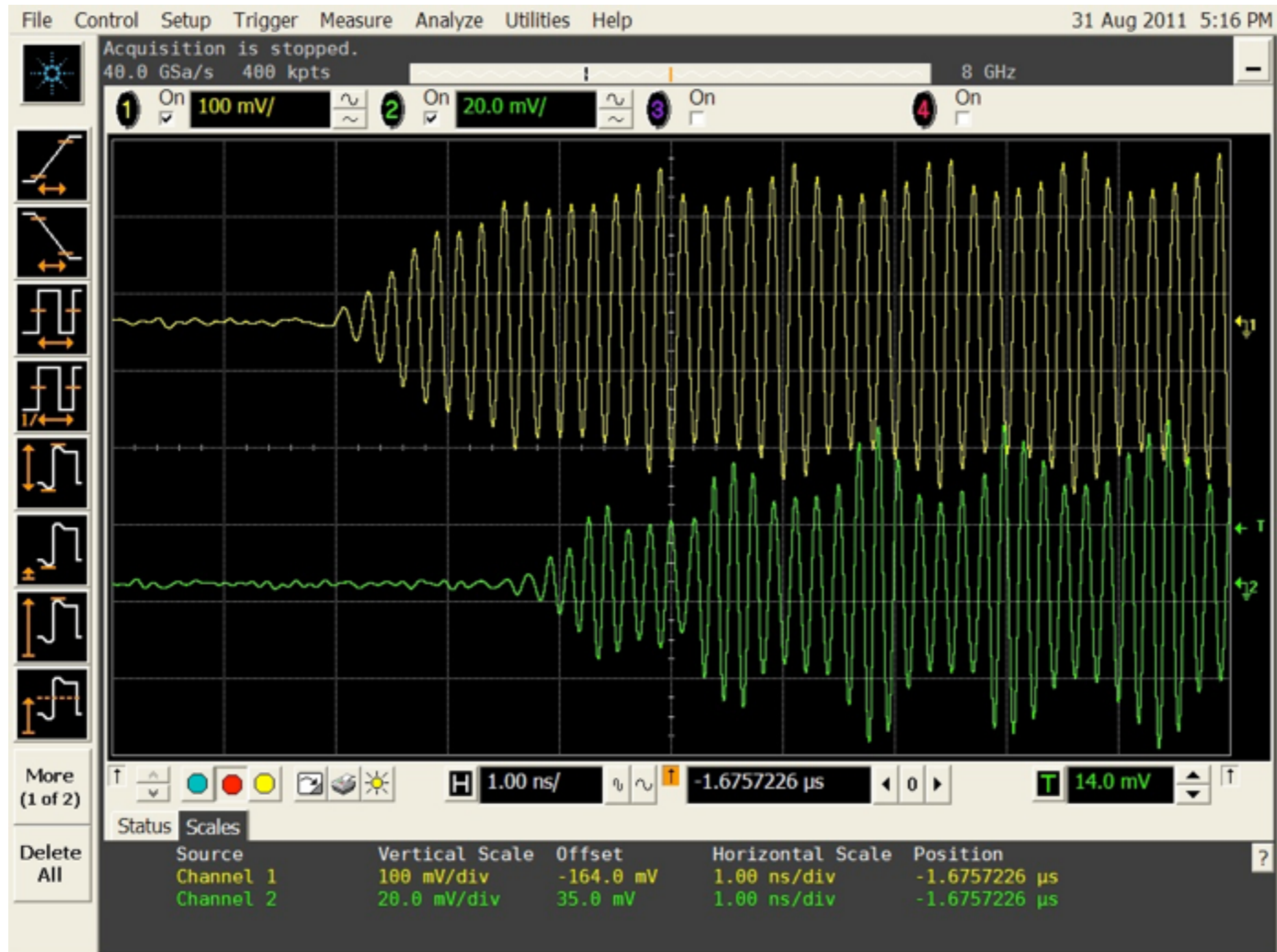
Two basic Attacks on DB protocols

Distance Fraud

- dishonest prover pretends to be closer to the verifier
- "pacemaker scenario"
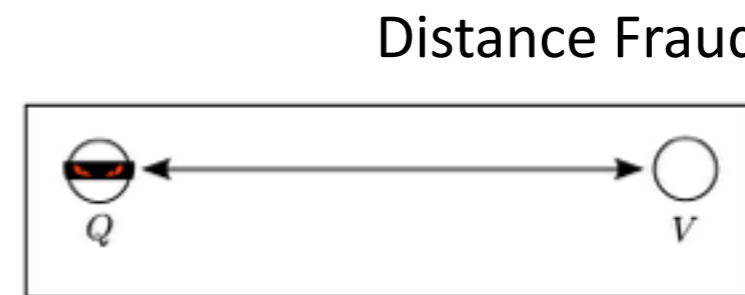
Distance Fraud

Mafia Fraud

- honest prover
- attacker convinces verifier and prover that they are closer
- relay attack ("car and key scenario)
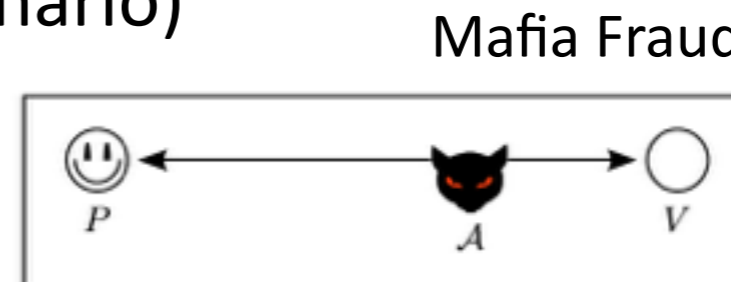
Mafia Fraud

A new Function: CRCS

CRCS-based DB protocol *(vs Distance and Mafia Fraud)*

P (Prover) V (Verifier)

Pick N_p

$c_p \leftarrow commit(N_p, P)$

$$\xrightarrow{\quad c_p \quad}$$

Pick N_v

- -

fast phase $r \leftarrow CRCS(N_v, N_p)$ $\xrightleftharpoons[\quad r \quad]{\quad N_v \quad}$ Record Δt

- -

$n \leftarrow delay()$ $N_p' \leftarrow channel(r)$

$N_v' \leftarrow signal(r)$

$n' \leftarrow delay(r)$

$$\xrightarrow{\quad sign(c_p, n, V, N_p, N_v) \quad}$$

Verify $\{\Delta t,\ n = n',$

$N_v' = N_v,\ N_p' = N_p,$

$sign(c_p, n, V, N_p, N_v)\}$

A new Function: CRCS

CRCS-based DB protocol *(vs Distance and Mafia Fraud)*

P (Prover) $\qquad\qquad\qquad\qquad\qquad\qquad$ V (Verifier)

Pick N_p

$c_p \leftarrow commit(N_p, P)$

$$\xrightarrow{\quad c_p \quad}$$

Pick N_v

- -

fast phase $\quad r \leftarrow CRCS(N_v, N_p)$ $\qquad \overset{N_v}{\underset{r}{\rightleftarrows}}$ $\qquad\qquad$ Record Δt → *distance*

- -

$n \leftarrow delay()$ $\qquad\qquad\qquad\qquad\qquad\qquad\qquad$ $N_p' \leftarrow channel(r)$

$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad$ $N_v' \leftarrow signal(r)$

$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad$ $n' \leftarrow delay(r)$

$$\xrightarrow{\quad sign(c_p, n, V, N_p, N_v) \quad}$$

Verify $\{\Delta t,\ n = n',$

$N_v' = N_v,\ N_p' = N_p,$

$sign(c_p, n, V, N_p, N_v)\}$

ETH Zürich

A new Function: CRCS

CRCS-based DB protocol *(vs Distance and Mafia Fraud)*

P (Prover) $\qquad\qquad\qquad\qquad$ V (Verifier)

Pick N_p

$c_p \leftarrow commit(N_p, P)$

$$\xrightarrow{\quad c_p \quad}$$

Pick N_v

- -

fast phase $\quad r \leftarrow CRCS(N_v, N_p)$ $\qquad\xrightarrow[\;r\;]{N_v}\qquad$ Record Δt → *distance*

- -

$n \leftarrow delay()$ $\qquad\qquad\qquad\qquad\qquad\qquad N'_p \leftarrow channel(r)$

$$N'_v \leftarrow signal(r)$$

$$n' \leftarrow delay(r)$$

slow phase
interpretation
of N$_v$

$$\xrightarrow{\; sign(c_p, n, V, N_p, N_v) \;}$$

Verify $\{\Delta t,\; n = n',$

$$N'_v = N_v,\; N'_p = N_p,$$

$$sign(c_p, n, V, N_p, N_v)\}$$

A new Function: CRCS

CRCS-based DB protocol *(vs Distance and Mafia Fraud)*

P (Prover) $\qquad\qquad\qquad\qquad\qquad\qquad$ V (Verifier)

Pick N_p

$\boxed{c_p \leftarrow commit(N_p, P)}$

DH and MF protection $\qquad\qquad\qquad \xrightarrow{\quad c_p \quad}$

$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad$ Pick N_v

- -

fast phase $\quad r \leftarrow CRCS(N_v, N_p)$ $\qquad \begin{array}{c} \xleftarrow{\ N_v\ } \\ \xrightarrow{\ r\ } \end{array}$ $\qquad$ Record Δt → *distance*

- -

$n \leftarrow delay()$ $\qquad\qquad\qquad\qquad\qquad\qquad$ $N_p' \leftarrow channel(r)$

$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad$ $N_v' \leftarrow signal(r)$

slow phase $\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad$ $n' \leftarrow delay(r)$

interpretation

of N_v $\qquad\qquad \xrightarrow{\ sign(c_p,n,V,N_p,N_v)\ }$

$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad$ Verify $\{\Delta t,\ n = n',$

$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad$ $N_v' = N_v,\ N_p' = N_p,$

$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad$ $sign(c_p, n, V, N_p, N_v)\}$

A new Function: CRCS

CRCS-based DB protocol *(vs Distance and Mafia Fraud)*

P (Prover) $\qquad\qquad\qquad\qquad$ V (Verifier)

Pick N_p

$c_p \leftarrow commit(N_p, P)$

DH and MF protection

$$\xrightarrow{\quad c_p \quad}$$

DF protection $\qquad\qquad\qquad\qquad\qquad$ Pick N_v

fast phase $\quad r \leftarrow CRCS(N_v, N_p)$ $\qquad \xrightarrow[\quad r \quad]{\quad N_v \quad}$ $\qquad$ Record $\Delta t \rightarrow$ *distance*

$n \leftarrow delay()$ $\qquad\qquad\qquad\qquad\qquad\qquad$ $N_p' \leftarrow channel(r)$

$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad$ $N_v' \leftarrow signal(r)$

slow phase $\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad$ $n' \leftarrow delay(r)$

interpretation

of N_v

$$\xrightarrow{\quad sign(c_p,n,V,N_p,N_v) \quad}$$

$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad$ Verify $\{\Delta t,\ n = n',$

$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad$ $N_v' = N_v,\ N_p' = N_p,$

$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad$ $sign(c_p, n, V, N_p, N_v)\}$

A new Function: CRCS

CRCS-based DB protocol *(vs Distance and Mafia Fraud)*

P (Prover) $\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad$ V (Verifier)

Pick N_p

$$c_p \leftarrow commit(N_p, P)$$

DH and MF protection

$$\xrightarrow{\quad c_p \quad}$$

DF protection $\qquad\qquad\qquad\qquad\qquad$ Pick N_v

fast phase $\quad r \leftarrow CRCS(N_v, N_p)$ $\qquad$ $\overset{N_v}{\underset{r}{\rightleftarrows}}$ $\qquad$ Record Δt → *distance*

$n \leftarrow delay()$ $\qquad\qquad\qquad\qquad\qquad\qquad$ $N_p' \leftarrow channel(r)$

$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad$ $N_v' \leftarrow signal(r)$

slow phase $\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad$ $n' \leftarrow delay(r)$

interpretation

of N_v

$$\xrightarrow{\quad sign(c_p, n, V, N_p, N_v) \quad}$$

IF protection

$\qquad\qquad\qquad\qquad\qquad\qquad\qquad$ Verify $\{\Delta t,\ n = n',$

$\qquad\qquad\qquad\qquad\qquad\qquad\qquad$ $N_v' = N_v,\ N_p' = N_p,$

$\qquad\qquad\qquad\qquad\qquad\qquad\qquad$ $sign(c_p, n, V, N_p, N_v)\}$

A new Function: CRCS

CRCS-based DB protocol *(vs Distance and Mafia Fraud)*

P (Prover) V (Verifier)

Pick N_p

$$c_p \leftarrow commit(N_p, P)$$

DH and MF protection

$$\xrightarrow{\quad c_p \quad}$$

DF protection   Pick N_v

fast phase $r \leftarrow CRCS(N_v, N_p)$

$$\xrightarrow{\quad N_v \quad}$$
$$\xrightarrow{\quad r \quad}$$

Record $\Delta t \rightarrow$ *distance*

MF protection $n \leftarrow delay()$

$$N'_p \leftarrow channel(r)$$
$$N'_v \leftarrow signal(r)$$
$$n' \leftarrow delay(r)$$

slow phase
interpretation
of N_v

$$\xrightarrow{\quad sign(c_p, n, V, N_p, N_v) \quad}$$

IF protection

Verify $\{\Delta t,\ n = n',$
$N'_v = N_v,\ N'_p = N_p,$
$sign(c_p, n, V, N_p, N_v)\}$

A new Function: CRCS

Mafia Fraud Detection *(physical layer)*

reported to V for
MF protection

fixed bit width
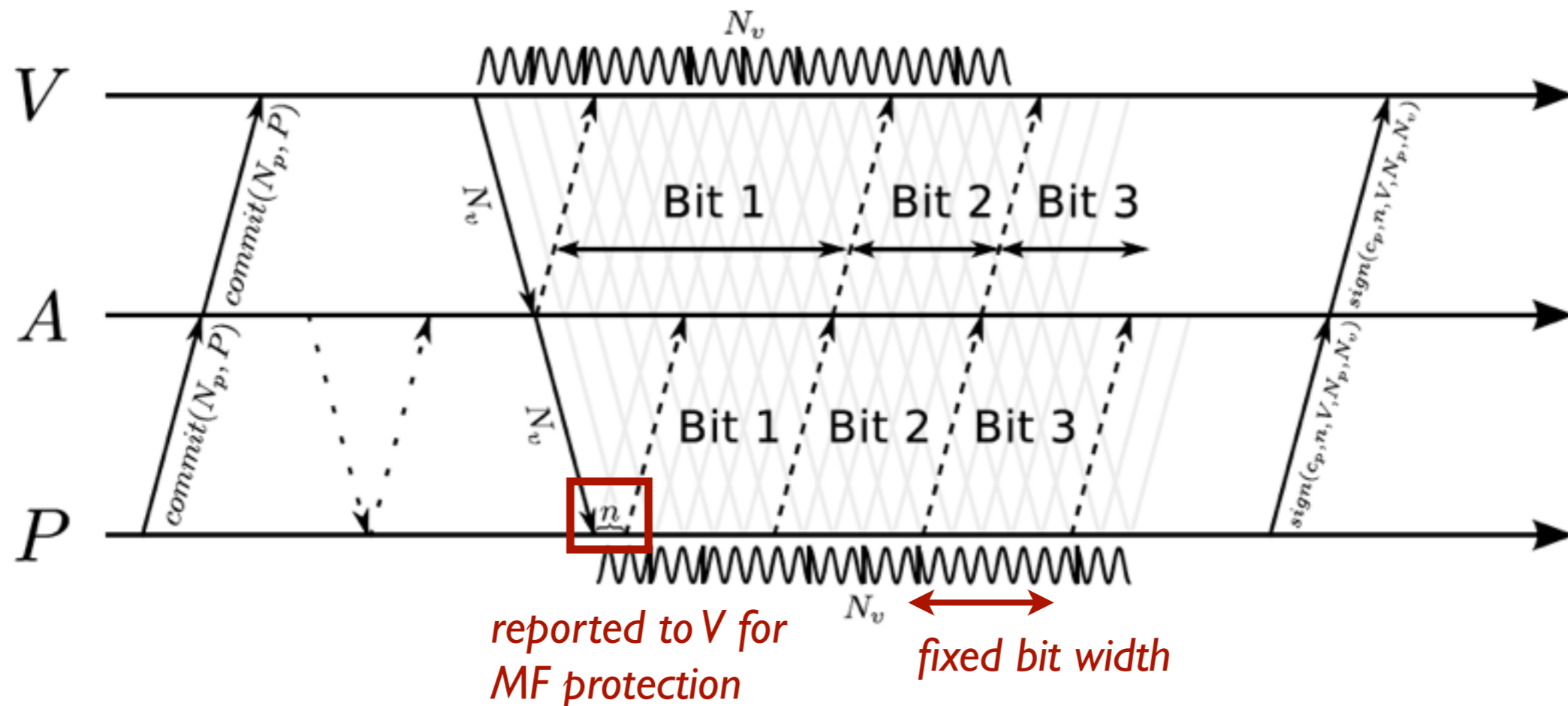
MF attack: $\frac{1}{2^{|N_p|}}$; DF attack: $\frac{1}{2^{|N_v|}}$

CRCS eliminates early detection, late commit attacks

Ongoing work on CRCS

Using CRCS the prover also reflects noise

=> CRCS increases complexity of the Verifier

In essence, CRCS trades
- robustness for increased security
- reduces complexity of the prover but increases the complexity of the verifier
- range might be affected by the use of CRCS (?)

What I didn't talk about (synchronization, preambles, ...).

Ongoing implementations ...

...

Other Implementation Efforts

Going back to XOR.

- What is the "fastest" implementation that we can make with f(Nv,Np) = Nv⊕Np?

- What kind of a receiver are we considering?
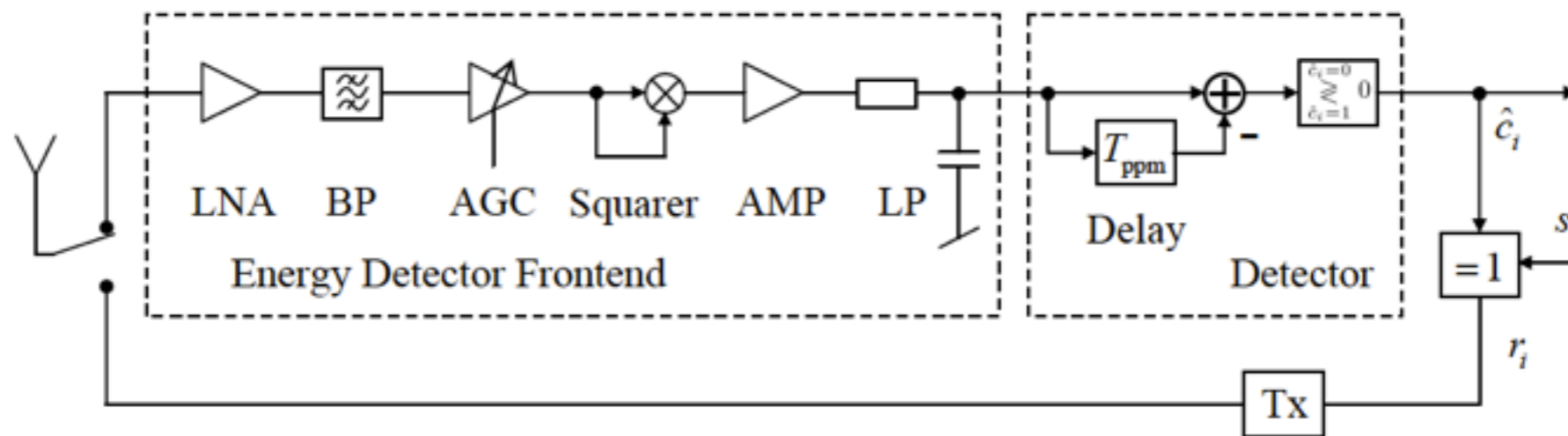
- A different modulation (SEM vs BPPM)
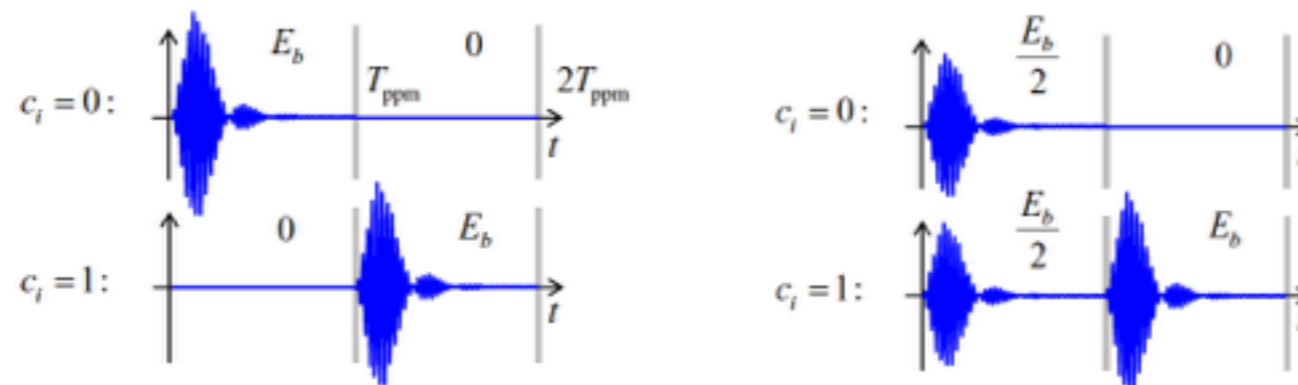
Protocol Analysis

Two main protocol constructs:

- Hancke-Kuhn
- Brands-Chaum

Three main attacks considered:

Distance Fraud

Terrorist Fraud
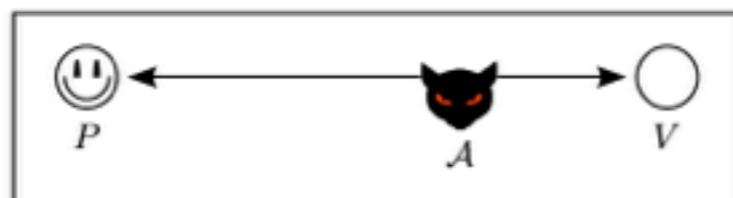
Mafia Fraud

Protocol Analysis

Two main protocol constructs:

- Hancke-Kuhn
- Brands-Chaum

msc Signature-based Brands and Chaum protocol

Prover: P

Verifier: V

$m_i \in_\mathcal{R} \{0,1\}$

$commit(m_1, \ldots, m_k)$

$\alpha_i \in_\mathcal{R} \{0,1\}$

Rapid bit exchange

α_i

$\beta_i \leftarrow \alpha_i \oplus m_i$

β_i

$c \leftarrow \alpha_1|\beta_1|\cdots|\alpha_k|\beta_k$

$(open\ commit),\ sign(c)$

Verify commit
$c \leftarrow \alpha_1|\beta_1|\cdots|\alpha_k|\beta_k$
verify $sign(c)$
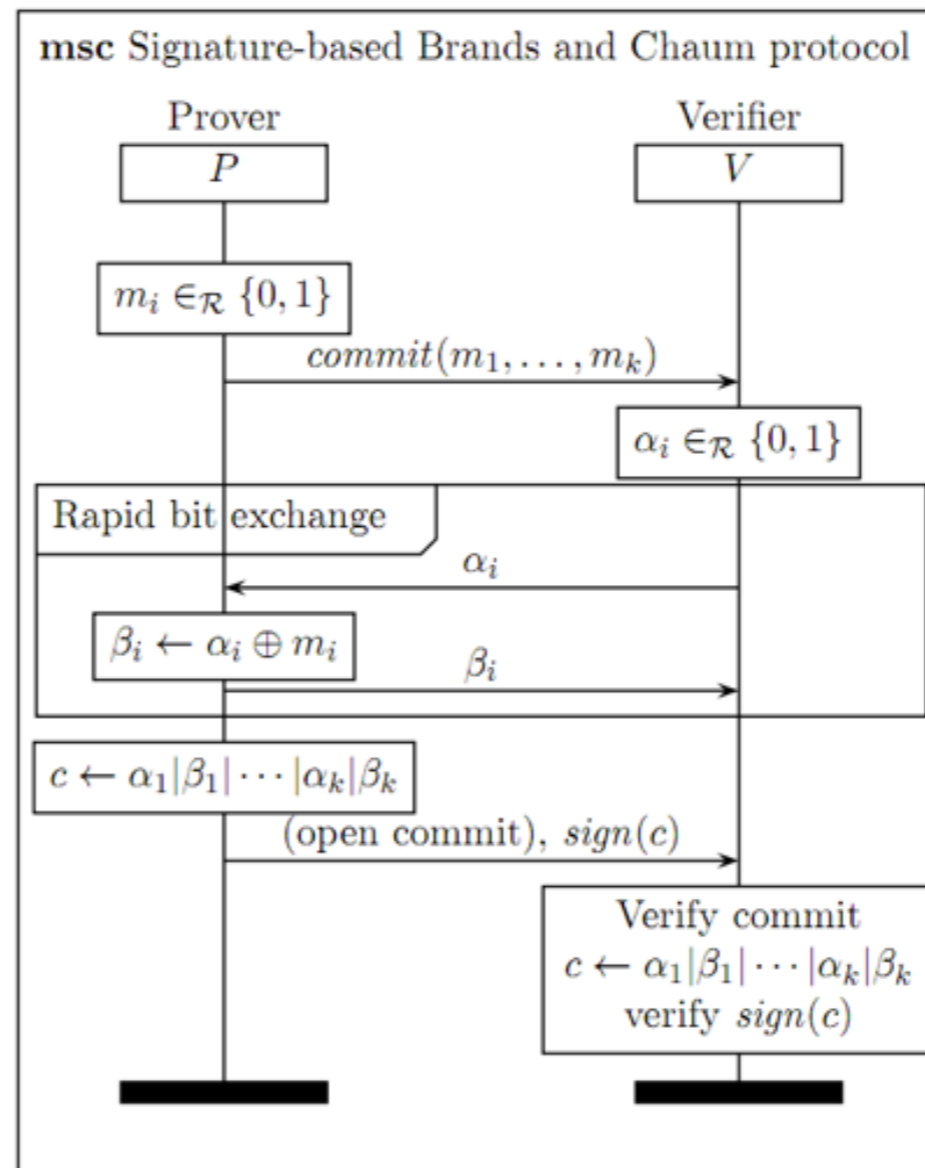
Protocol Analysis

Novel attack: Distance Hijacking
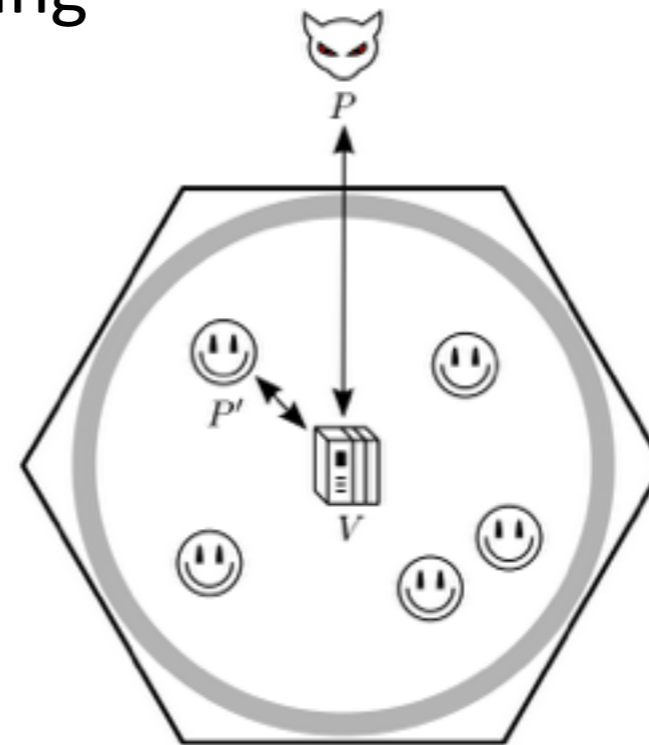
Distance Fraud
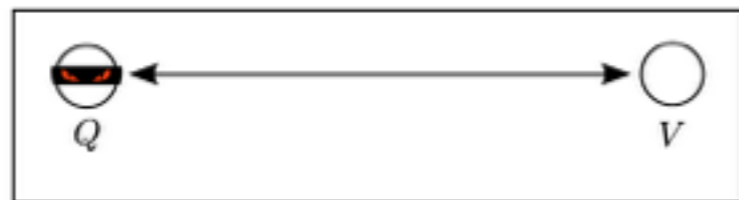
Terrorist Fraud

Mafia Fraud

Distance Hijacking

ETH Zürich
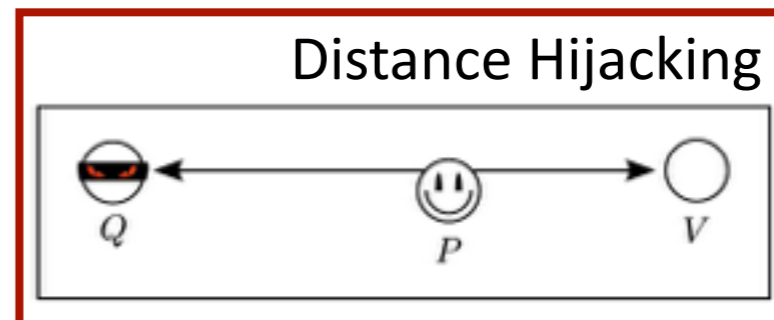
Protocol Analysis

Distance Hijacking on Brands and Chaum
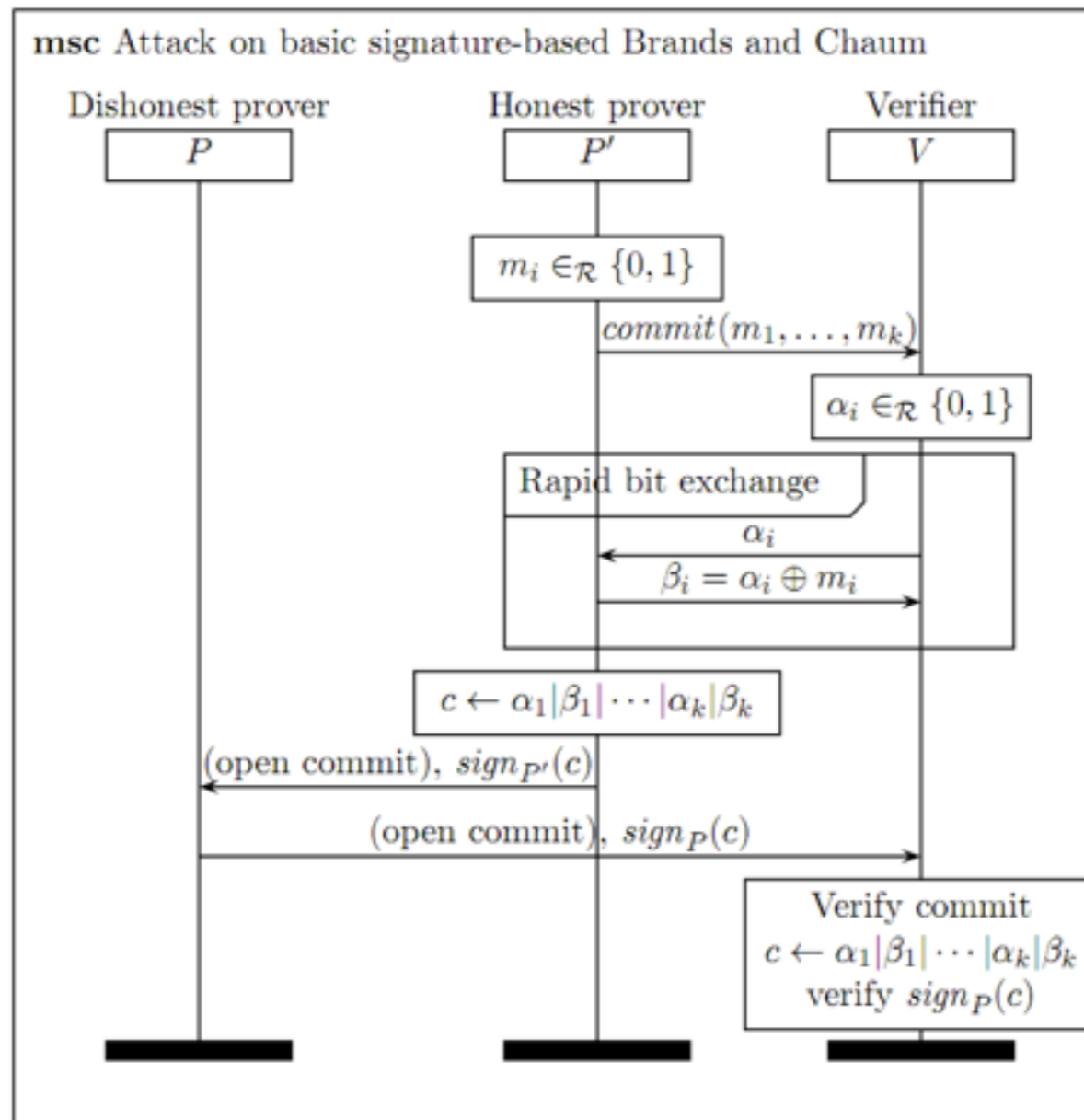
Protocol Analysis

More Distance Hijacking
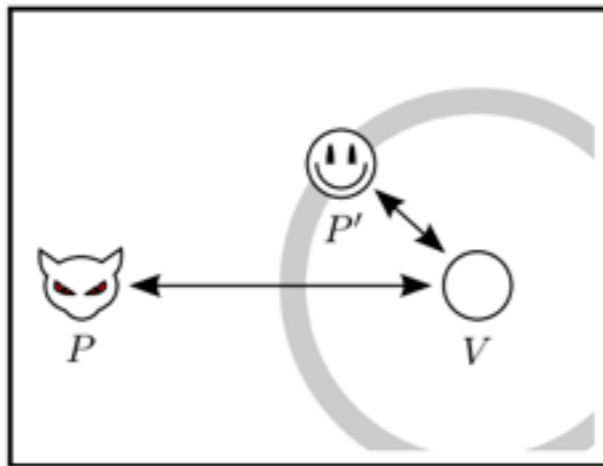
Figure 7: Scenario in which V accepts protocol sessions from multiple provers, here P and P', where Distance Hijacking may be a threat.
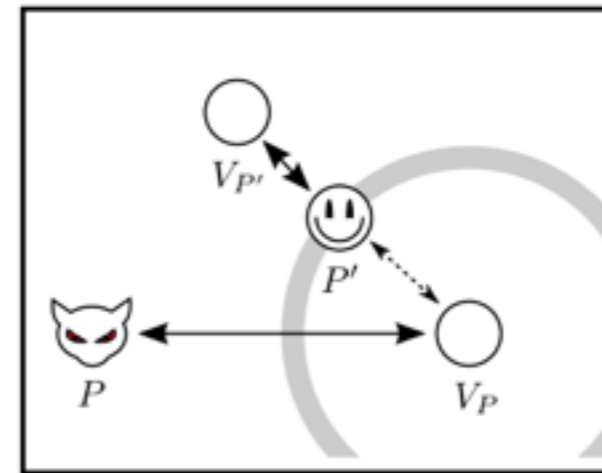
Figure 8: Scenario with multiple prover/verifier pairs, where V_x only accepts sessions from x. Even in this case, Distance Hijacking may be possible.

Protocol Analysis

Attack on Hancke-Kuhn (indirect)
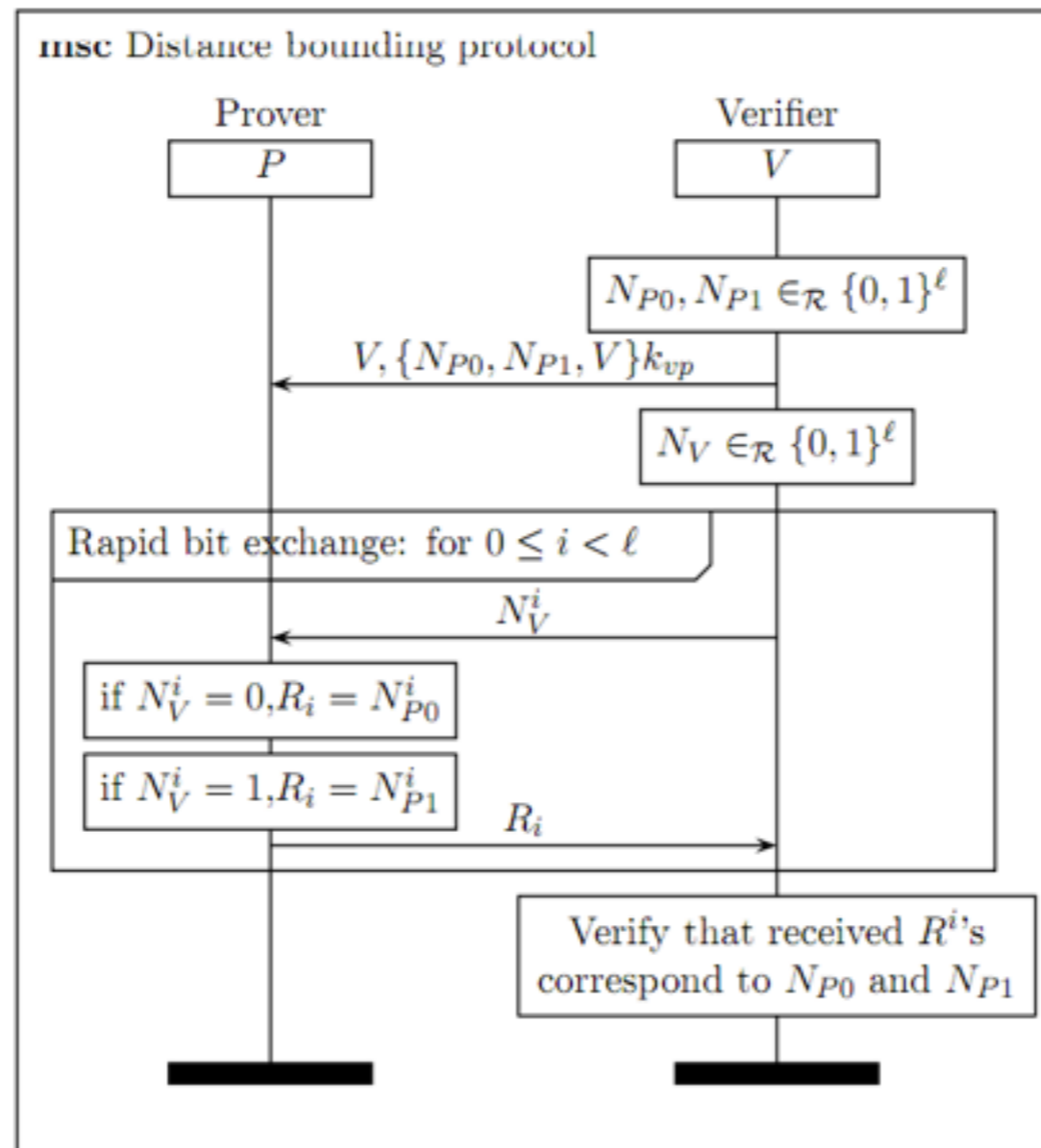
DB Protocol Analysis (Formal)

Authentication and Key Establishment protocols

- analyzed in the Dolev-Yao model
- no notions of location, channel characteristics, (or time)
- the same frameworks cannot analyze DB protocol

Some new framework can capture physical properties *(time, location, physical layer) e.g.,* [Basin10]

- Model based on experiments with real systems
- Enables formal analysis of DB protocols
- Captured new attacks on DB that we missed in the informal analysis

Other frameworks: Avoine, Meadows,

Game is not over … (ref. Distance Hijacking attacks)

One Use of DB -> Authentication Based on *Absence* Awareness

How would Proximity-Based Access Control be implemented?
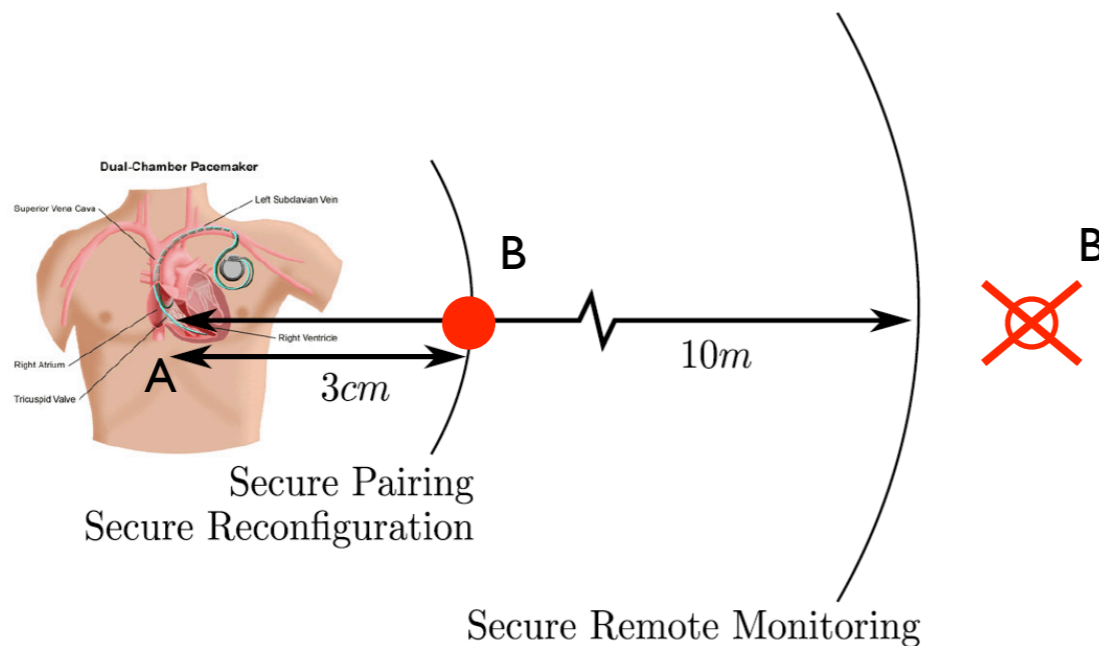
1. A verifies proximity of B
2. A establishes a *shared secret key* with B (e.g., pairing using DH)
3. The key is used to enforce access control

One Use of DB -> Authentication Based on *Absence* Awareness

How would Proximity-Based Access Control be implemented?
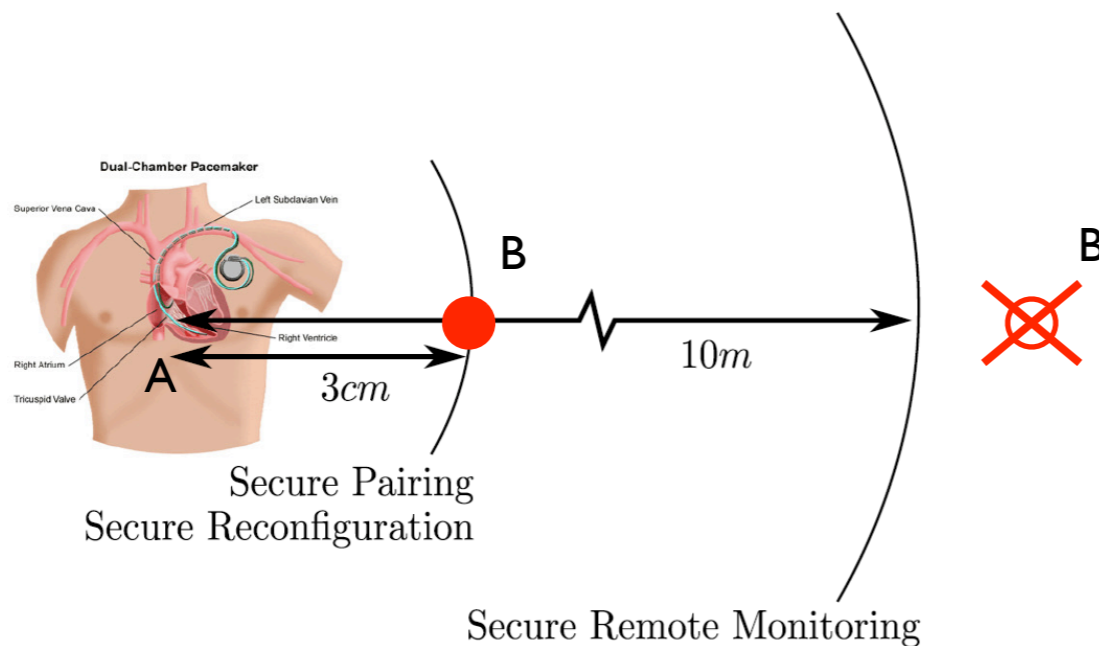
1. A verifies proximity of B
2. A establishes a *shared secret key* with B (e.g., pairing using DH)
3. The key is used to enforce access control

The protocol needs to ensure that the *key is bound to the distance*.

One Use of DB -> Authentication Based on *Absence* Awareness

How would Proximity-Based Access Control be implemented?

1. A verifies proximity of B
2. A establishes a *shared secret key* with B (e.g., pairing using DH)
3. The key is used to enforce access control

The protocol needs to ensure that the *key is bound to the distance*.

ETH Zürich
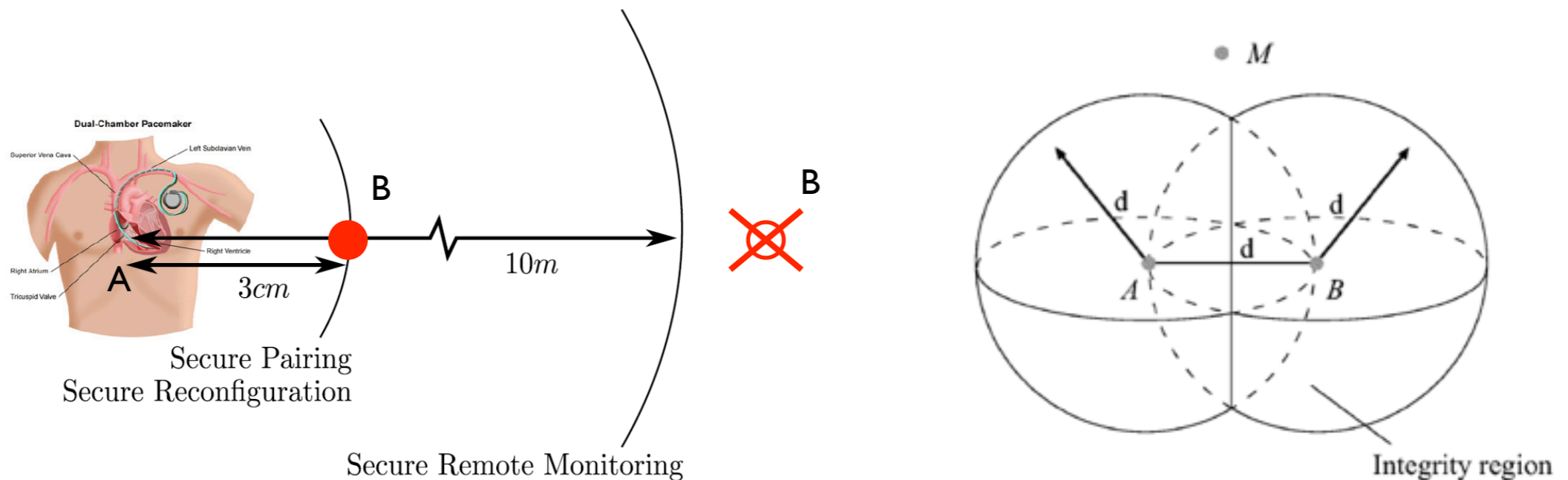
Secure Localization

*From Proximity Verification
to Location Verification and Secure Localization*

Secure Localization

User's perspective:

to obtain a correct information about its own location

Infrastructure perspective:

to obtain a correct information about the location of a device

Secure localization goals

- Compute a 'correct' location of a (trusted) device in the presence of an attacker. *(Secure Localization)*
- Verify the correctness of a location of an untrusted device. (that e.g., claims a certain location) *(Location Verification)*

ETH Zürich

Secure Localization Schemes

- Verifiable Multilateration
- Location Verification with Hidden and Mobile Stations
- *Secure Broadcast Localization and Time Synchronization (will be covered later in the lectures)*

ETH Zürich

Distance Bounding

- P can always pretend to be further from V
- M can always convince P and V that they are further away

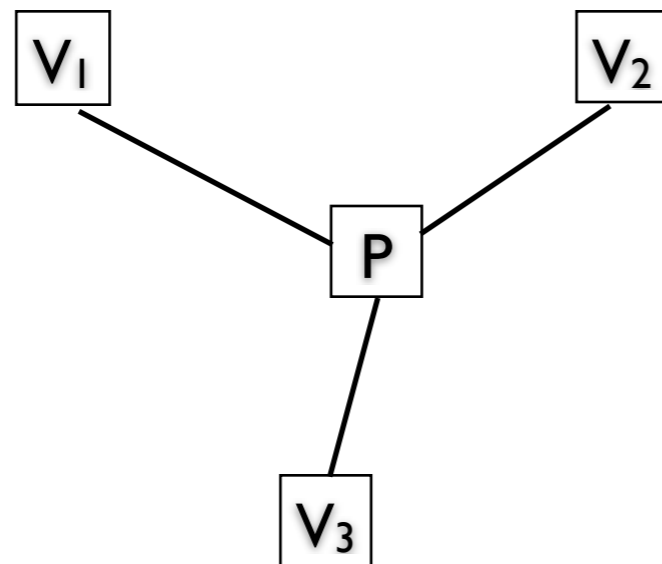=> *Distance **enlargement is easy, distance reduction is prevented** using distance bounding protocols*

Ranging

$t_p << t_r - t_s$

$d = (t_r - t_s - t_p)c/2$

ETH Zürich

Distance Bounding

- P can always pretend to be further from V
- M can always convince P and V that they are further away

=> *Distance **enlargement is easy, distance reduction is prevented** using distance bounding protocols*

Ranging

$$d = (t_r - t_s - t_p)c/2$$

$$t_p << t_r - t_s$$

ETH *Zürich*

Distance Bounding

- P can always pretend to be further from V
- M can always convince P and V that they are further away

=> *Distance **enlargement is easy, distance reduction is prevented** using distance bounding protocols*

Ranging

$$d = (t_r - t_s - t_p)c/2$$

$$t_p << t_r - t_s$$

ETH Zürich

Distance Bounding

- P can always pretend to be further from V
- M can always convince P and V that they are further away

*=> Distance **enlargement is easy, distance reduction is prevented** using distance bounding protocols*

Ranging

$$d = (t_r - t_s - t_p)c/2$$

$$t_p << t_r - t_s$$

ETH *Zürich*

Verifiable Multilateration

Distance enlargement is easy, distance reduction is prevented using distance bounding protocols

- *So can we realize Location Verification or Secure Localization using Distance Bounding protocols?*

ETH Zürich

Verifiable Multilateration

Distance enlargement is easy, distance reduction is prevented using distance bounding protocols

- *So can we realize Location Verification or Secure Localization using Distance Bounding protocols?*

ETH Zürich

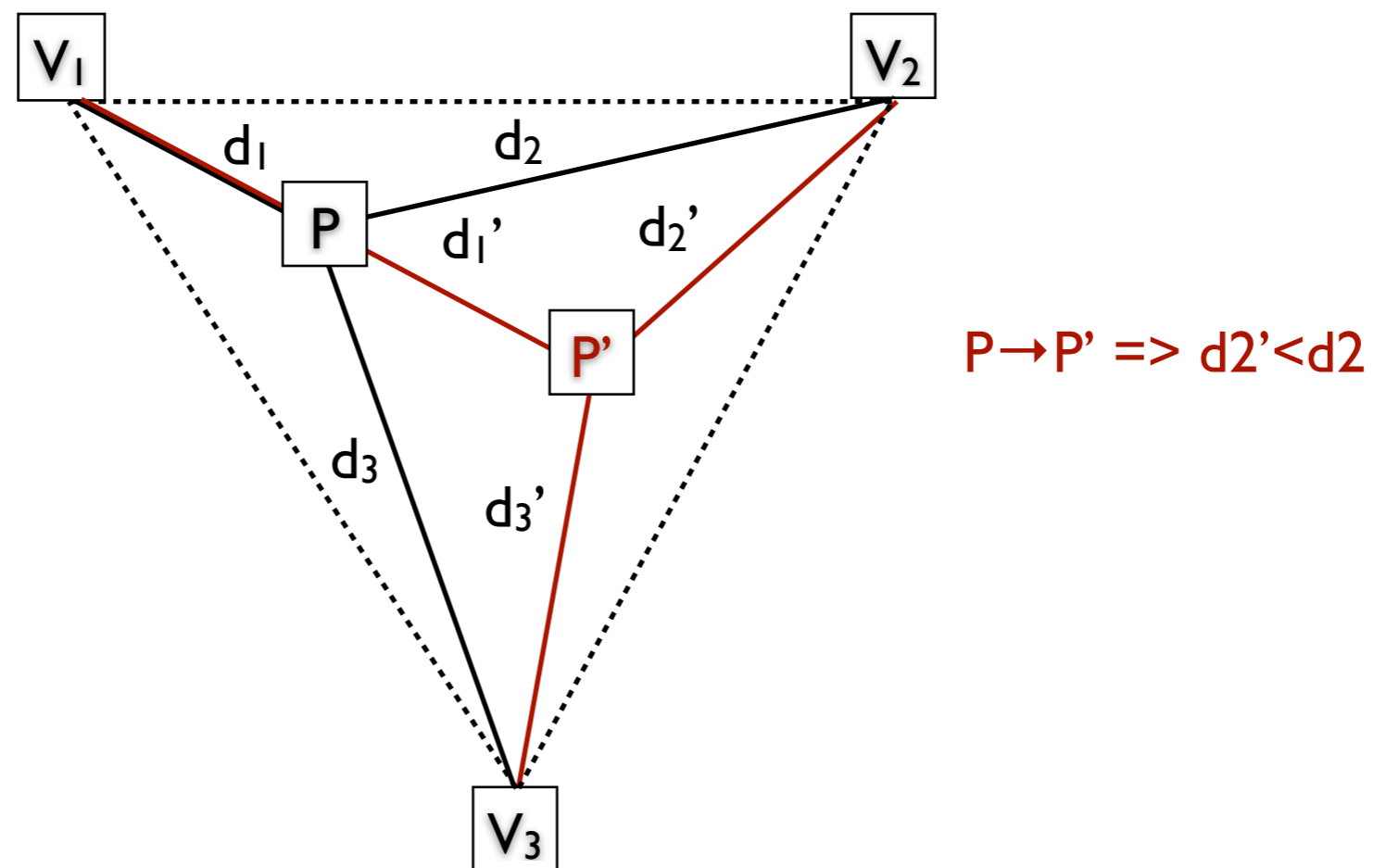# Verifiable Multilateration

Verifiable Multilateration in 3 steps:

1. Verifiers (known locations) form a *verification triangle*.
2. Based on the measured distance bounds, compute the location of the Prover.
3. *If the computed location is in the verification triangle, the verifiers conclude that this is a correct location.*

ETH Zürich

Verifiable Multilateration

Verifiable Multilateration in 3 steps:

1. Verifiers (known locations) form a *verification triangle*.
2. Based on the measured distance bounds, compute the location of the Prover.
3. *If the computed location is in the verification triangle, the verifiers conclude that this is a correct location.*

ETH Zürich

Verifiable Multilateration

Verifiable Multilateration in 3 steps:

1. Verifiers (known locations) form a *verification triangle*.
2. Based on the measured distance bounds, compute the location of the Prover.
3. *If the computed location is in the verification triangle, the verifiers conclude that this is a correct location.*

P→P' => d2'<d2

ETH Zürich

Verifiable Multilateration

Properties:

1. *P cannot successfully claim to be at P'≠P, where **P' is within the triangle***

2. *M cannot convince Vs and P that P is at P'≠P where **P' is within the triangle***

3. *P or M can spoof a location from P to P' where P' is **outside the triangle***

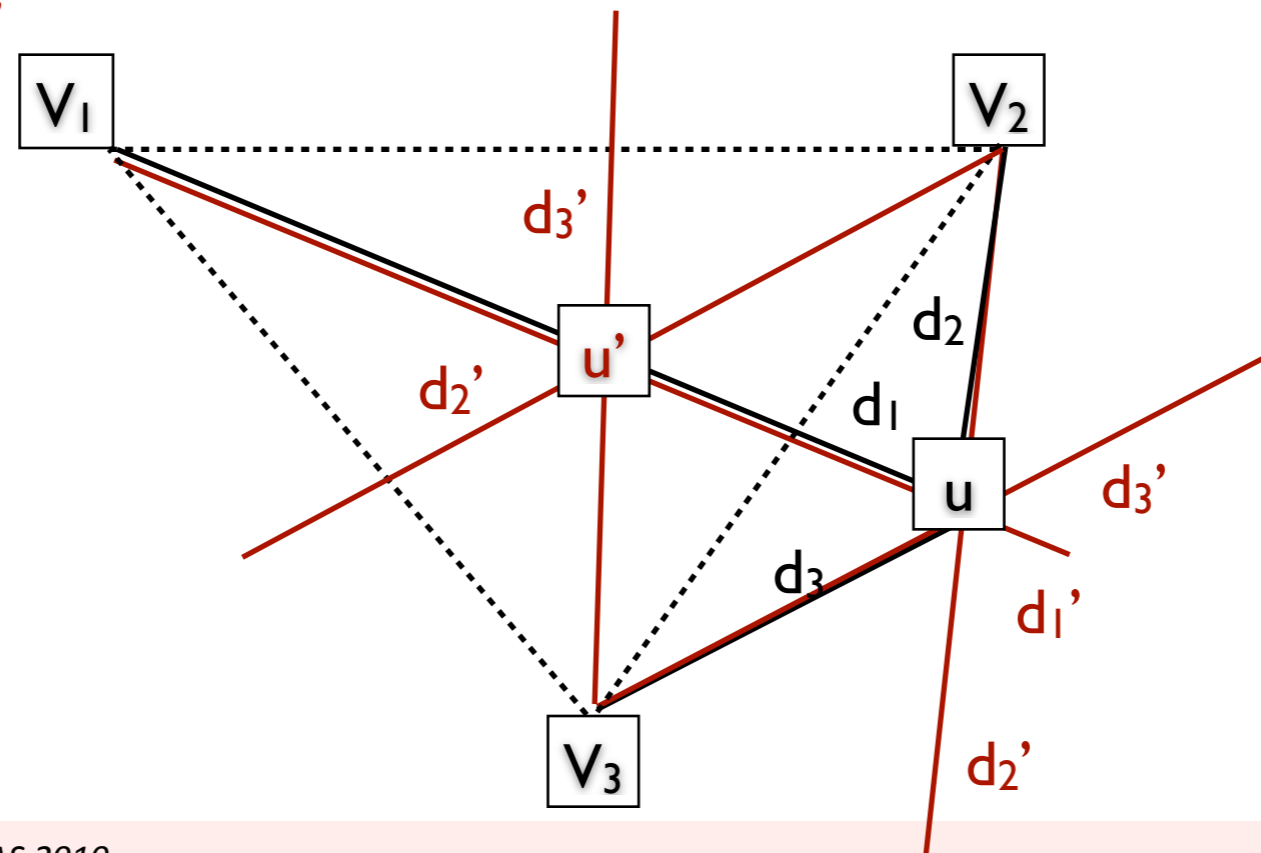$P \rightarrow P' \Rightarrow d2' < d2$

Verifiable Multilateration

The algorithm and the errors:

- Need to be careful how the position is computed!

- Example: *Minimum Mean Square Estimate (MMSE)*

Let $f_i(x'_u, y'_u) = db_i - \sqrt{(x_i - x'_u)^2 + (y_i - y'_u)^2}$

> The position of u is obtained by minimizing
> $F(x'_u, y'_u) = \sum_{v_i \in \mathcal{T}} f_i^2(x'_u, y'_u)$
> over all estimates of u

- *Attack:*

ETH Zürich

Verifiable Multilateration

The algorithm and the errors:

- Need to be careful how the position is computed!

- Example: *Minimum Mean Square Estimate (MMSE)*

Let $f_i(x'_u, y'_u) = db_i - \sqrt{(x_i - x'_u)^2 + (y_i - y'_u)^2}$

> The position of u is obtained by minimizing
> $F(x'_u, y'_u) = \sum_{v_i \in \mathcal{T}} f_i^2(x'_u, y'_u)$
> over all estimates of u

- *Attack:*

ETH *Zürich*

Verifiable Multilateration

The algorithm and the errors:

- Need to be careful how the position is computed!

- Example: *Minimum Mean Square Estimate (MMSE)*

Let $f_i(x'_u, y'_u) = db_i - \sqrt{(x_i - x'_u)^2 + (y_i - y'_u)^2}$

The position of u is obtained by minimizing
$F(x'_u, y'_u) = \sum_{v_i \in \mathcal{T}} f_i^2(x'_u, y'_u)$
over all estimates of u

- *Attack:*

ETH Zürich

Verifiable Multilateration

Verifiable Multilateration Algorithm

$\mathcal{T} = \emptyset$; set of verification triangles enclosing u

$\mathcal{V} = \{v_1, ..., v_n\}$; set of verifiers in the power range of u

1 *For all* $v_i \in \mathcal{V}$, perform distance bounding
  from v_i to u and obtain db_i

2 With all $v_i \in \mathcal{V}$, compute the estimate (x'_u, y'_u) of the position
  by MMSE

3 *If for all* $v_i \in \mathcal{V}$, $\left| db_i - \sqrt{(x_i - x'_u)^2 + (y_i - y'_u)^2} \right| \le \delta$ *then*
  for all $(v_i, v_j, v_k) \in \mathcal{V}^3$, *if* $(x'_u, y'_u) \in \triangle(v_i, v_j, v_k)$
  then $\mathcal{T} = \mathcal{T} \cup (v_i, v_j, v_k)$
  if $|\mathcal{T}| > 0$ *then* position is accepted and $x_u = x'_u$, $y_u = y'_u$
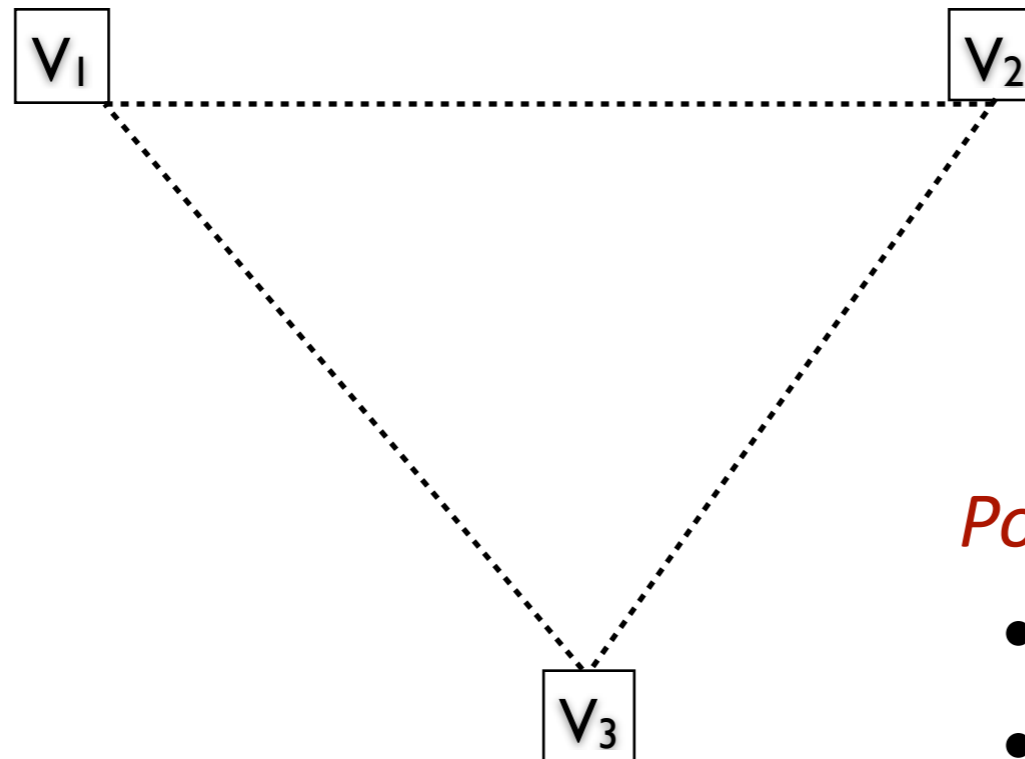 *else* the position is rejected
  else the position is rejected
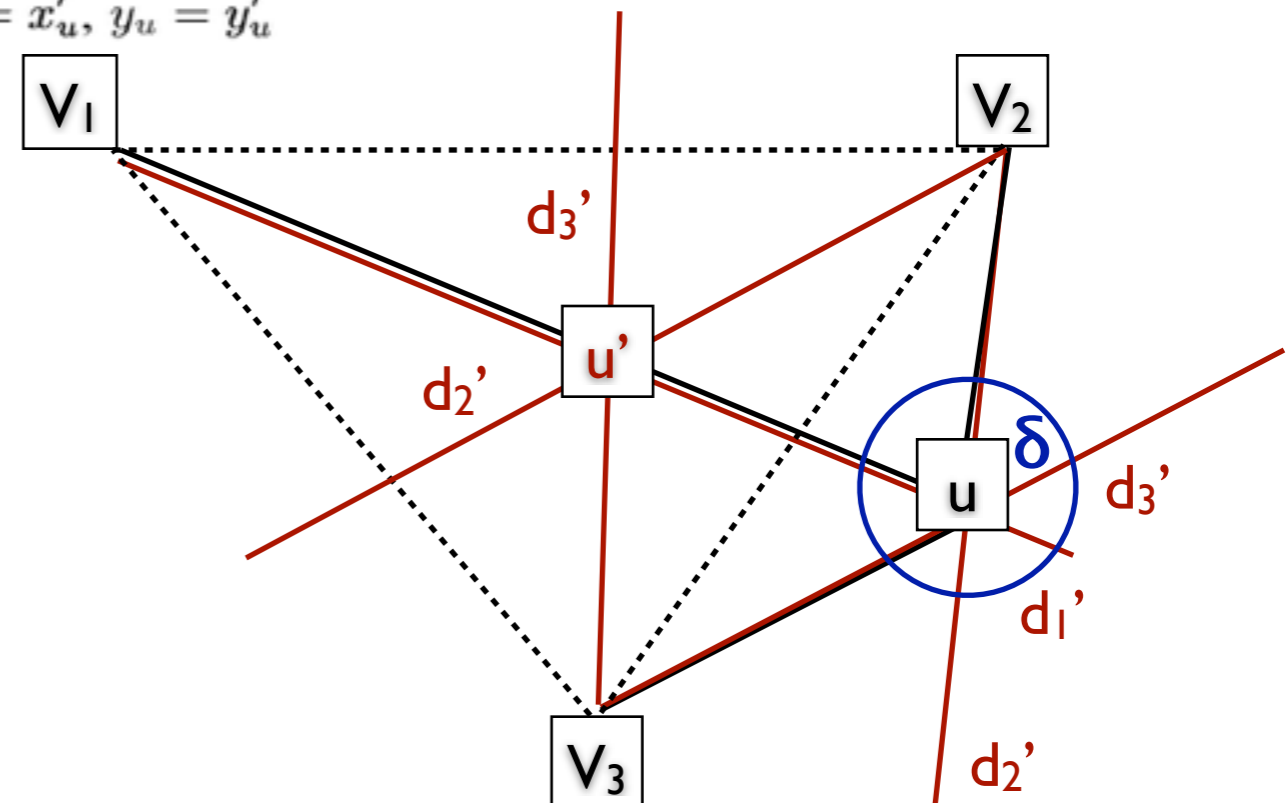
ETH Zürich

Verifiable Multilateration

Collusion attacks (only with untrusted prover under location verification)
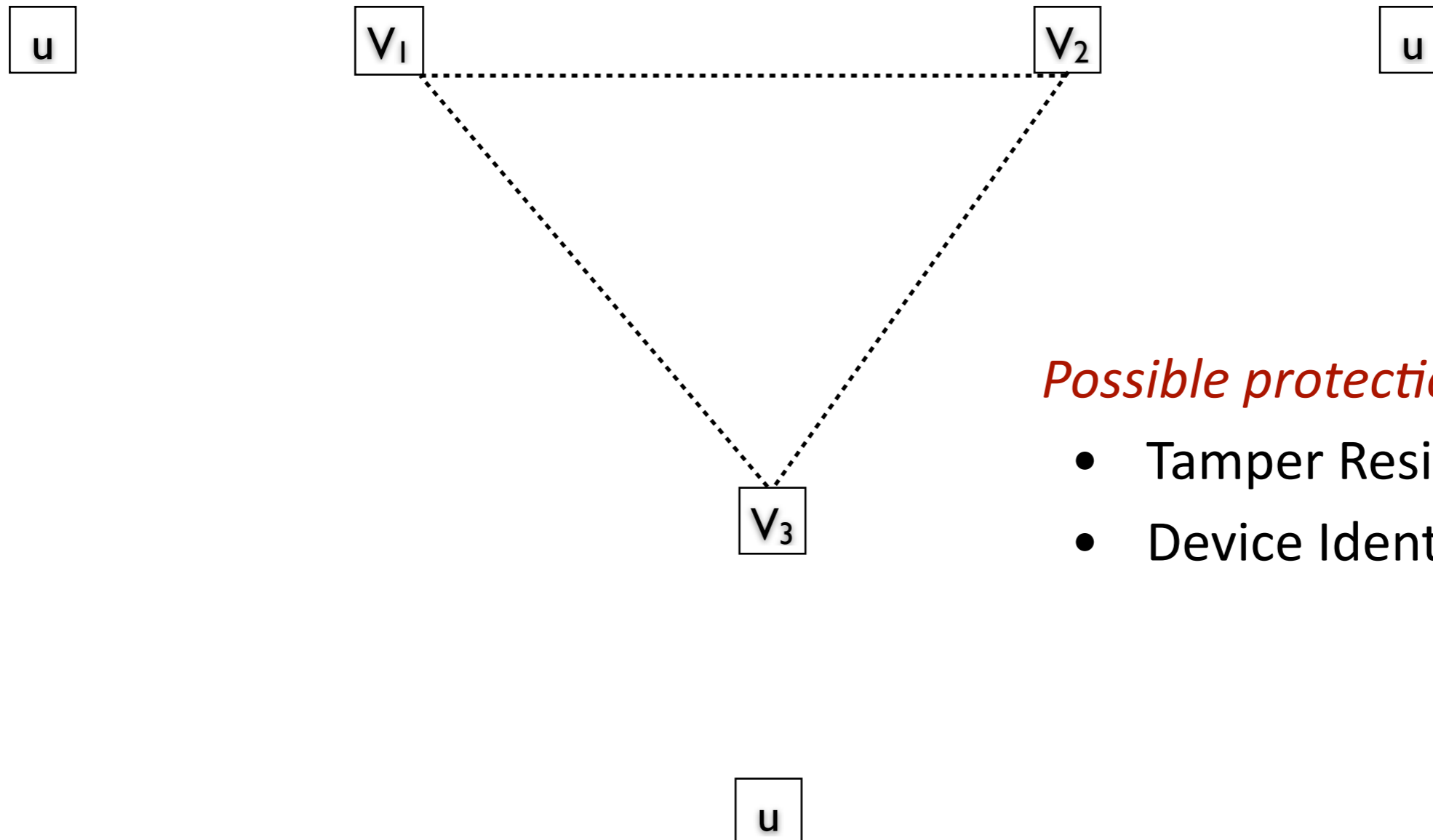
- *Attack:*

Possible protections:

- Tamper Resistance
- Device Identification

Verifiable Multilateration

Collusion attacks (only with untrusted prover under location verification)
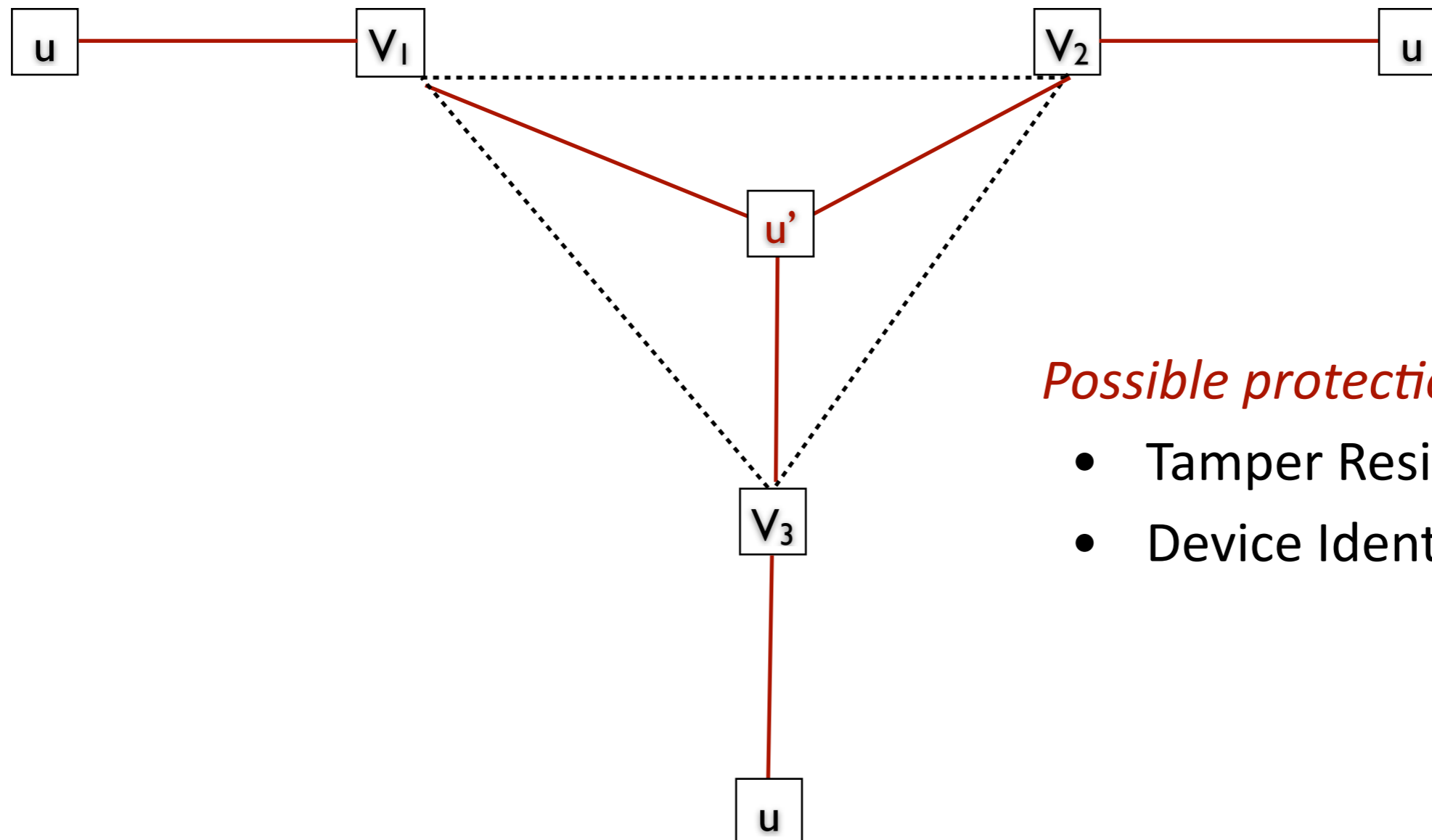
- *Attack:*

u

V_1 - V_2 u

V_3

Possible protections:

- Tamper Resistance
- Device Identification

u

ETH Zürich

Verifiable Multilateration

Collusion attacks (only with untrusted prover under location verification)

- *Attack:*

Possible protections:

- Tamper Resistance
- Device Identification

ETH Zürich

Verifiable Multilateration

Collusion attacks (only with untrusted prover under location verification)

- *Attack:*
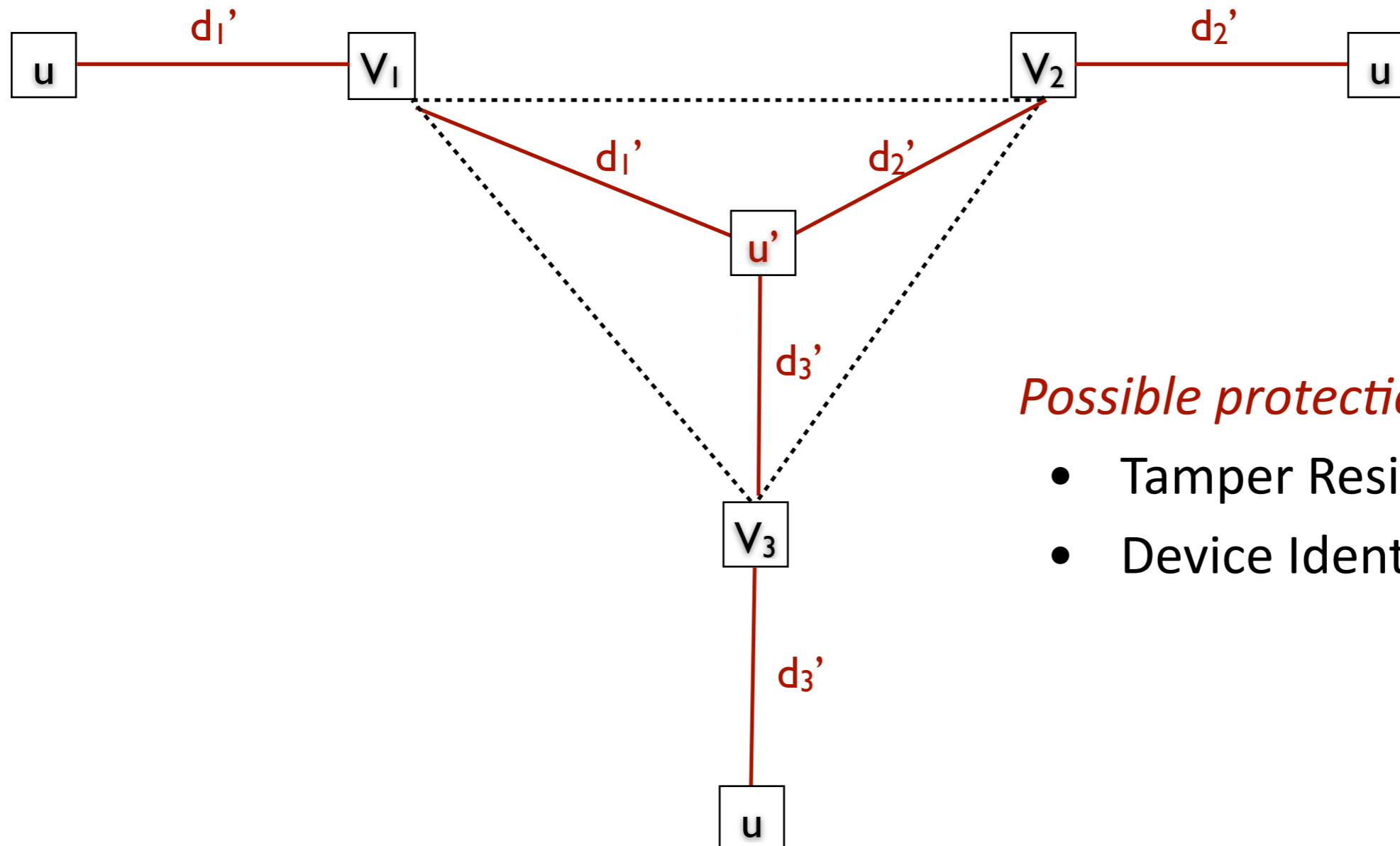
Possible protections:

- Tamper Resistance
- Device Identification

Location Verification using
Hidden and Mobile Stations *(Verifiers)*

The basic idea:

- *If the prover does not know where the verifiers are, it doesn't know how to cheat.*
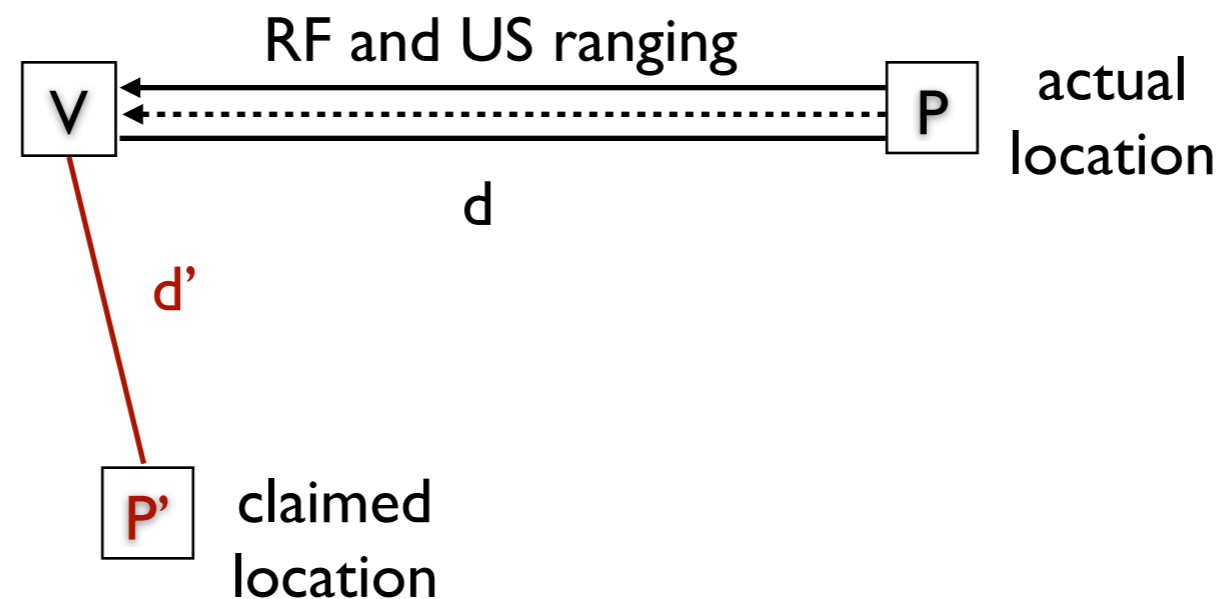
RF and US ranging

$\boxed{\text{V}}$

$\boxed{\text{P}}$ actual location

d

d'

claimed location

p(successful cheating) = p(d-d' ≤ Δ)

where Δ is the ranging/localization accuracy

ETH Zürich

Location Verification using
Hidden and Mobile Stations *(Verifiers)*
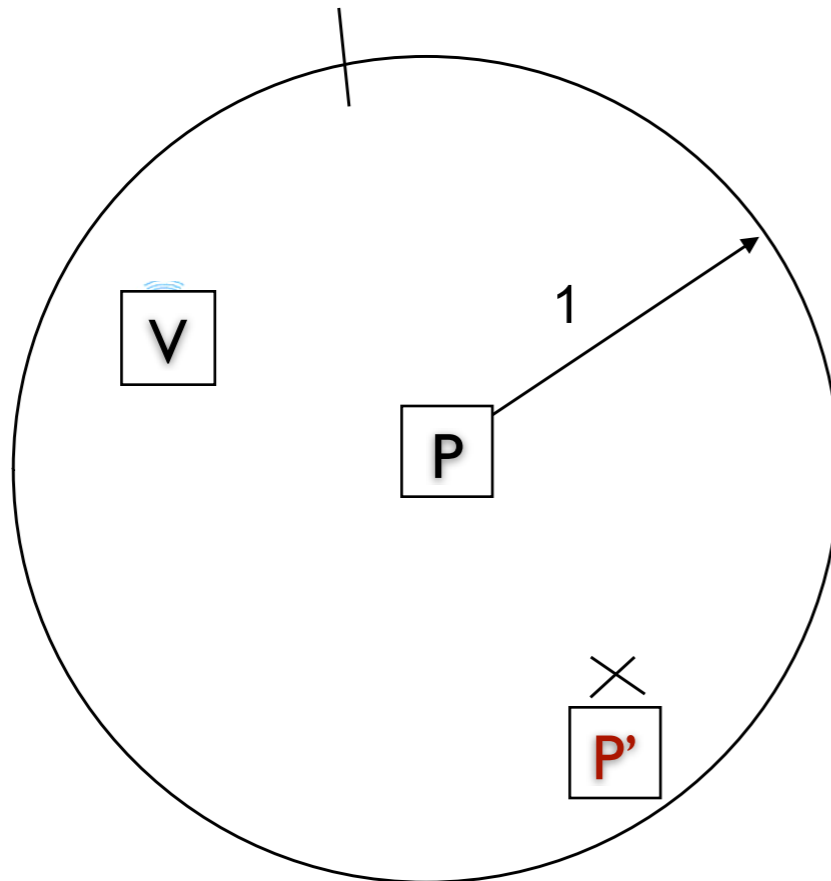
The basic idea:

- *If the prover does not know where the verifiers are, it doesn't know how to cheat.*
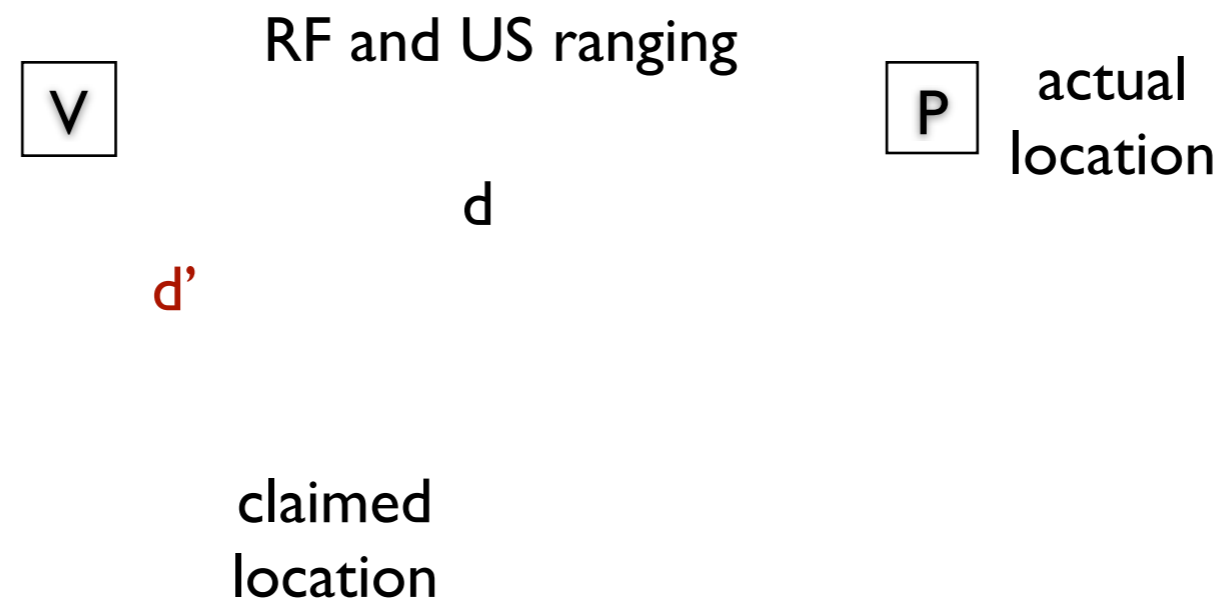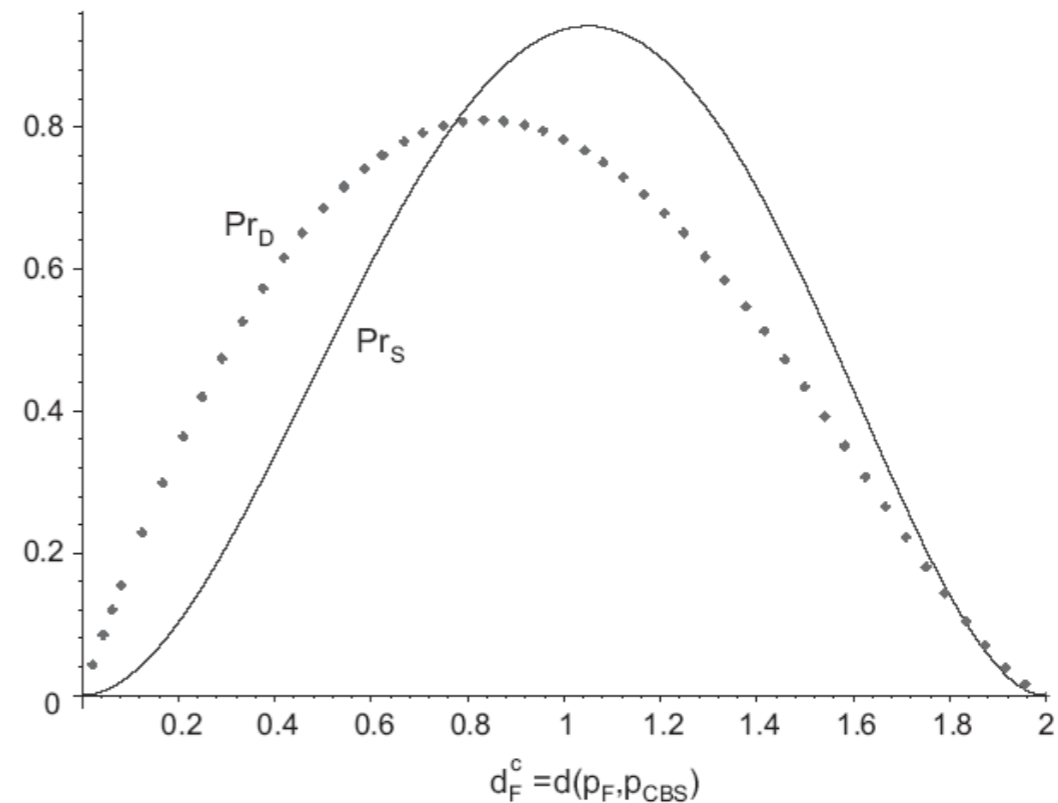

$$p(successful\ cheating) = p(d\text{-}d' \leq \Delta)$$

where Δ is the ranging/localization accuracy

Location Verification using
Hidden and Mobile Stations *(Verifiers)*

Observation 1:

not all distances are equally likely

- Not all locations are equally easy to fake (center is the 'easiest').

- *Problems if the attacker knows where verifiers cannot be.*

ETH Zürich

Summary (on secure localization)

Main ideas

- Use time as a side-channel (e.g., distance bounding)
- Use hidden verifier locations
- Use spread spectrum communication (hide the signals such that they cannot be manipulated - in time)

- References:
  - Verifiable Multilateration:
    S. Capkun, J. P. Hubaux, Secure positioning in wireless networks, IEEE Journal on Selected Areas in Communications: Special Issue on Security in Wireless Ad Hoc Networks, February 2006.

  - http://www.syssec.ethz.ch/research/spot

ETH Zürich

Summary

- Secure Localization / Location Verification is a fascinating area
- Brings up interesting interactions between logical and physical layer
- New challenges in formal protocol analysis

- Can be used for Secure Localization and Location Verification
- Numerous Applications
  - Physical and Logical Access Control, Anti-Spoofing, Protection of Networking Functions, ...