

# RFID Distance Bounding Multistate Enhancement

Gildas Avoine<sup>1</sup> Christian Floerkemeier<sup>2</sup> Benjamin Martin<sup>1</sup>

<sup>1</sup>Université catholique de Louvain  
Belgium

<sup>2</sup>Massachusetts Institute of Technology  
USA



# Outline

- 1 RFID Primer
- 2 Relay attacks and counter-measures
- 3 Our proposal
- 4 Conclusion

# A quick introduction to RFID

## The name

RFID stands for Radio-Frequency IDentification

## The parties

There are two parties involved :

- A reader : the verifier
- A tag : the prover

## The specificities of this technology

- The communication field
- The tag computation capability

# A quick introduction to RFID

## Focus on the tag

### Applications

From pet identification to passports.

- Banking
- Access control
- Event ticketing...

### Tags HF

We mainly focus on these tags which are very commonly used.

- They can do cryptographic computations
- Frequency: 13.56 MHz
- Emission Range: few decimeters

# Authentication in RFID

ISO 9798-2



secret  $x$

computes  $E_x(r_V, V)$



secret  $x$

generates  $r_V$

$\xleftarrow{r_V}$

$\xrightarrow{E_x(r_V, V)}$

# Authentication in RFID

ISO 9798-2



secret  $x$

computes  $E_x(r_V, V)$



secret  $x$   
generates  $r_V$

$\xleftarrow{r_V}$

$\xrightarrow{E_x(r_V, V)}$

# Authentication in RFID

ISO 9798-2



secret  $x$

computes  $E_x(r_V, V)$



secret  $x$   
generates  $r_V$

$\xleftarrow{r_V}$

$\xrightarrow{E_x(r_V, V)}$

# Authentication in RFID

ISO 9798-2



secret  $x$

computes  $E_x(r_V, V)$



secret  $x$   
generates  $r_V$

$\xleftarrow{r_V}$

$\xrightarrow{E_x(r_V, V)}$

# Authentication in RFID

ISO 9798-2



secret  $x$

computes  $E_x(r_V, V)$



secret  $x$

generates  $r_V$

$\xleftarrow{r_V}$

$\xrightarrow{E_x(r_V, V)}$

# Authentication in RFID

ISO 9798-2

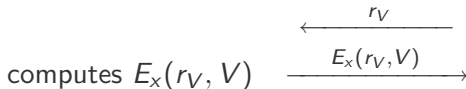


secret  $x$



secret  $x$

generates  $r_V$



## Assumption

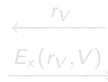
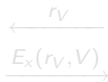
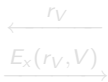
The prover is near to the verifier during the execution of the protocol

# Relay attack

Mafia fraud



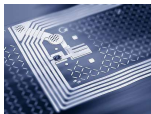
$E_x(r_V, V)$



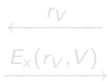
generates  $r_V$

# Relay attack

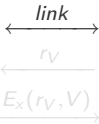
Mafia fraud



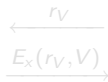
$E_x(r_V, V)$



$\tilde{V}$



$\tilde{P}$



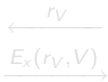
generates  $r_V$

# Relay attack

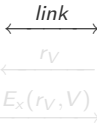
Mafia fraud



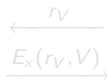
$E_x(r_V, V)$



$\tilde{V}$



$\tilde{P}$



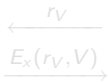
generates  $r_V$

# Relay attack

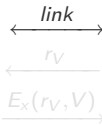
Mafia fraud



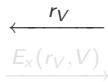
$E_x(r_V, V)$



$\tilde{V}$



$\tilde{P}$



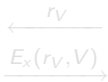
generates  $r_V$

# Relay attack

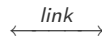
Mafia fraud



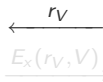
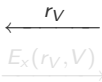
$E_x(r_V, V)$



$\tilde{V}$



$\tilde{P}$



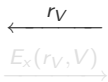
generates  $r_V$

# Relay attack

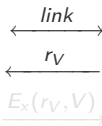
Mafia fraud



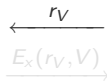
$E_x(r_V, V)$



$\tilde{V}$



$\tilde{P}$



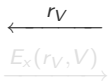
generates  $r_V$

# Relay attack

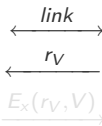
Mafia fraud



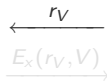
$E_x(r_V, V)$



$\tilde{V}$



$\tilde{P}$



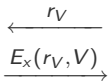
generates  $r_V$

# Relay attack

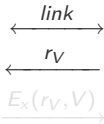
Mafia fraud



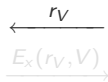
$E_x(r_V, V)$



$\tilde{V}$



$\tilde{P}$



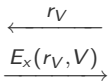
generates  $r_V$

# Relay attack

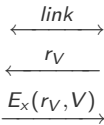
Mafia fraud



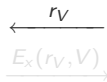
$E_x(r_V, V)$



$\tilde{V}$



$\tilde{P}$



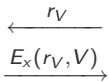
generates  $r_V$

# Relay attack

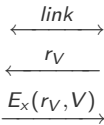
Mafia fraud



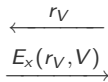
$E_x(r_V, V)$



$\tilde{V}$



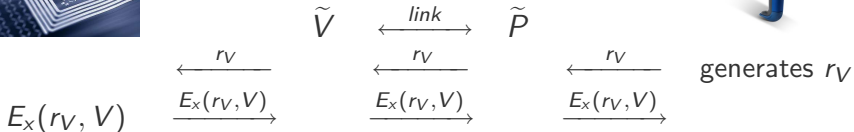
$\tilde{P}$



generates  $r_V$

# Relay attack

Mafia fraud

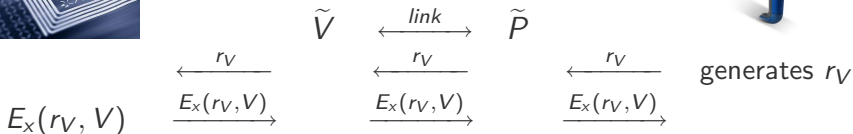


How to defeat the attack?

The verifier measures the distance between him and the prover.

# Relay attack

Mafia fraud



How to defeat the attack ?

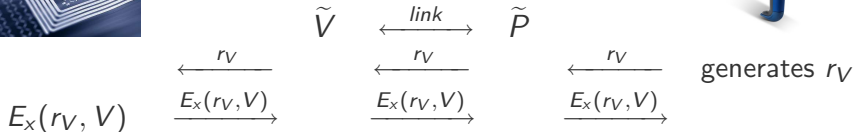
The verifier measures the distance between him and the prover.

How to measure the distance ?

The verifier measures of the time spent during a exchange between him and the prover.

# Relay attack

Mafia fraud



How to defeat the attack ?

The verifier measures the distance between him and the prover.

How to measure the distance ?

The verifier measures of the time spent during a exchange between him and the prover.

Our contribution

A generic method which improves the published protocols

# Hancke and Khun 2005

## The protocol

$P$

secret  $x$

$V$

secret  $x$

slow phase

fast phase

# Hancke and Khun 2005

The protocol

$P$   
secret  $x$

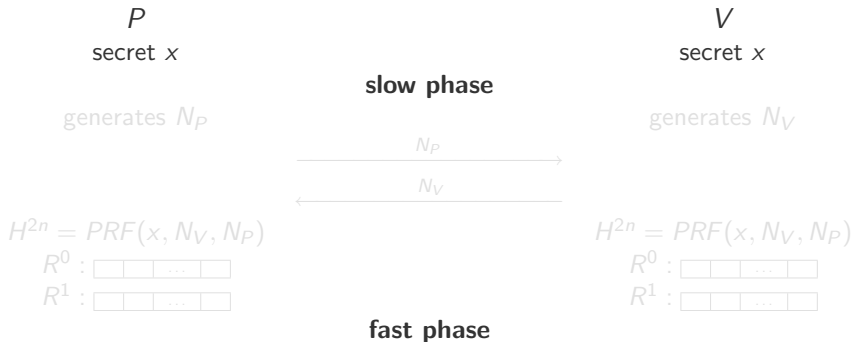
$V$   
secret  $x$

**slow phase**

**fast phase**

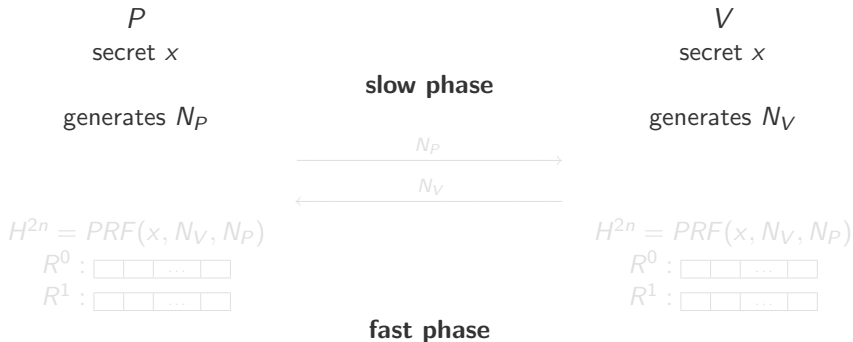
# Hancke and Khun 2005

## The protocol



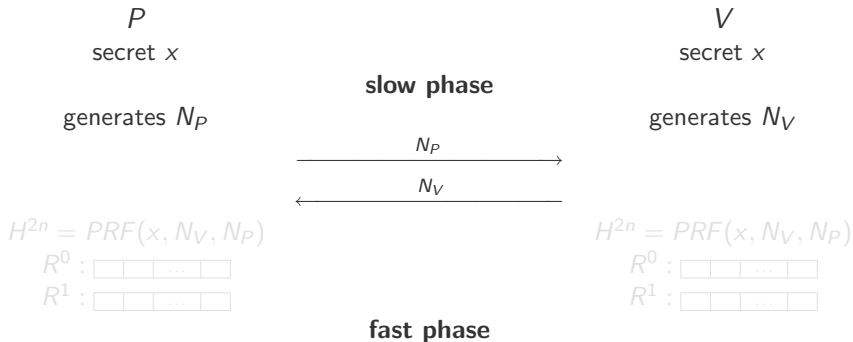
# Hancke and Khun 2005

## The protocol



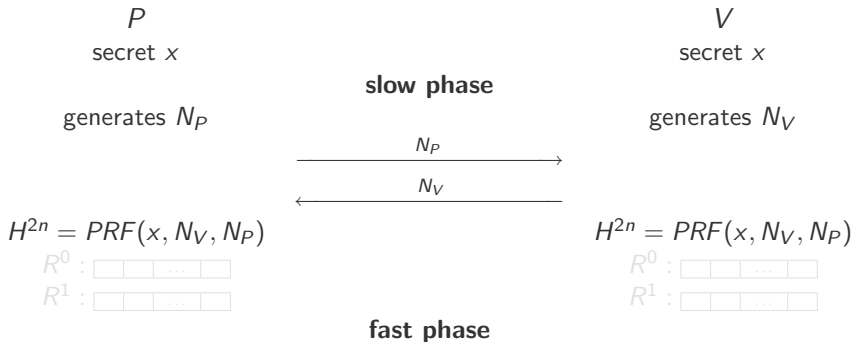
# Hancke and Khun 2005

## The protocol



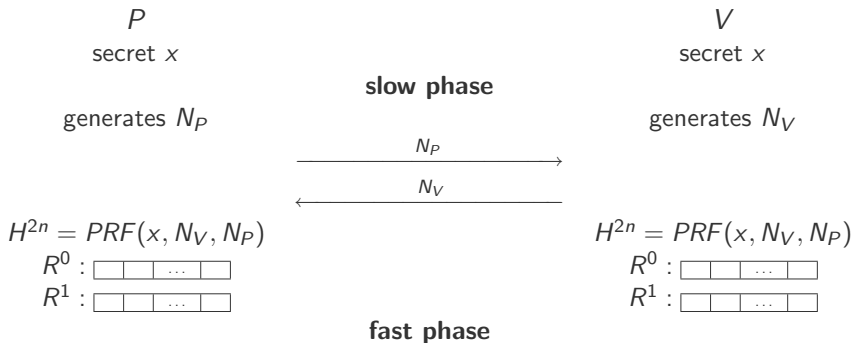
# Hancke and Khun 2005

## The protocol



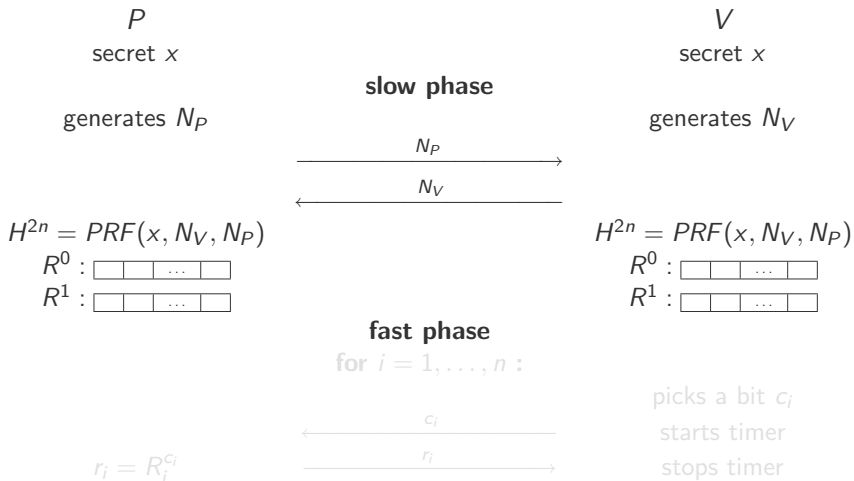
# Hancke and Khun 2005

## The protocol



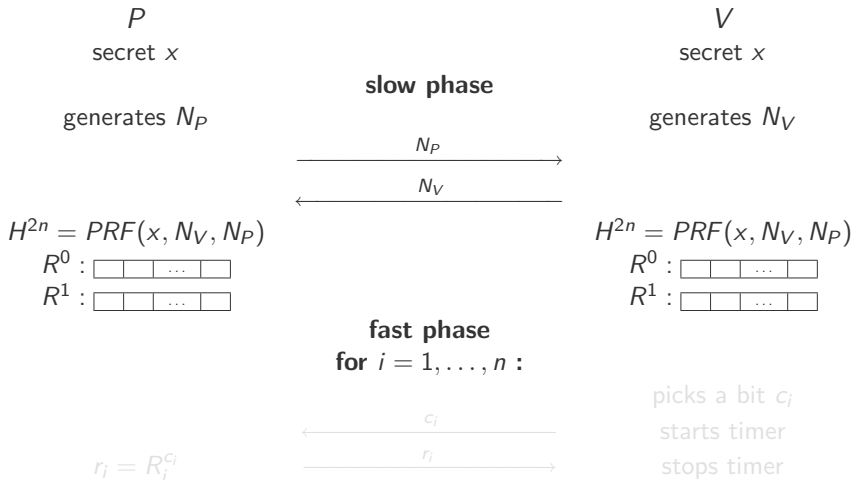
# Hancke and Khun 2005

## The protocol



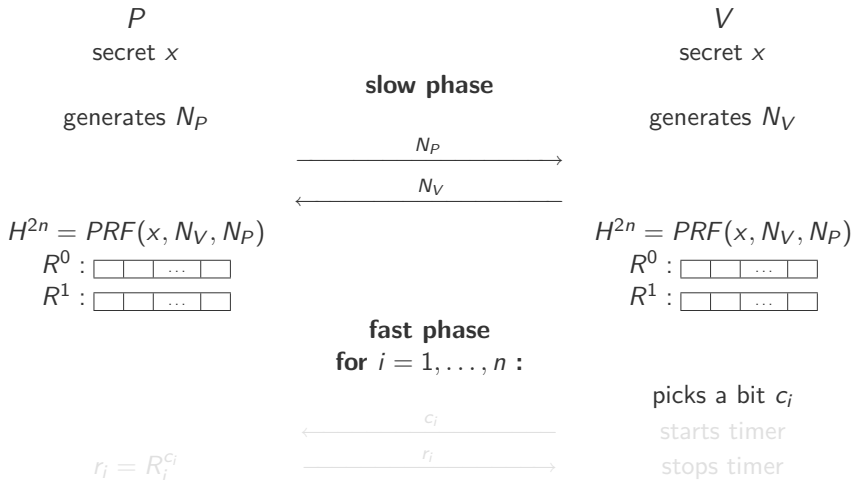
# Hancke and Khun 2005

## The protocol



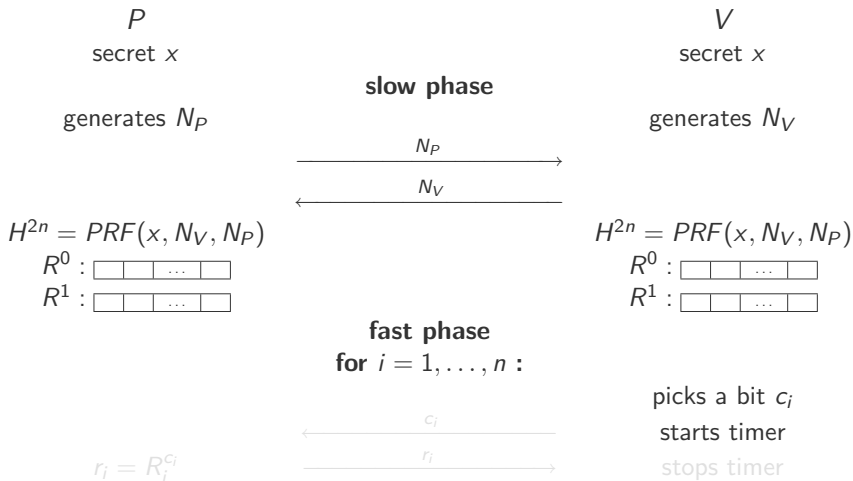
# Hancke and Khun 2005

## The protocol



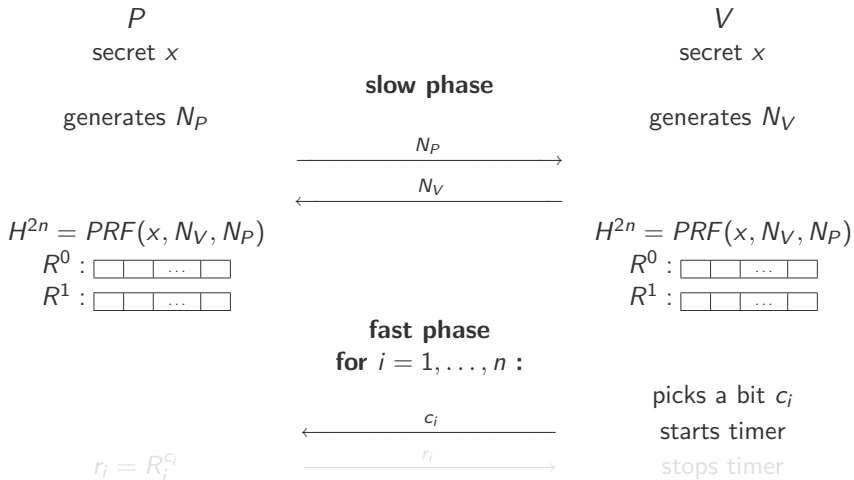
# Hancke and Khun 2005

## The protocol



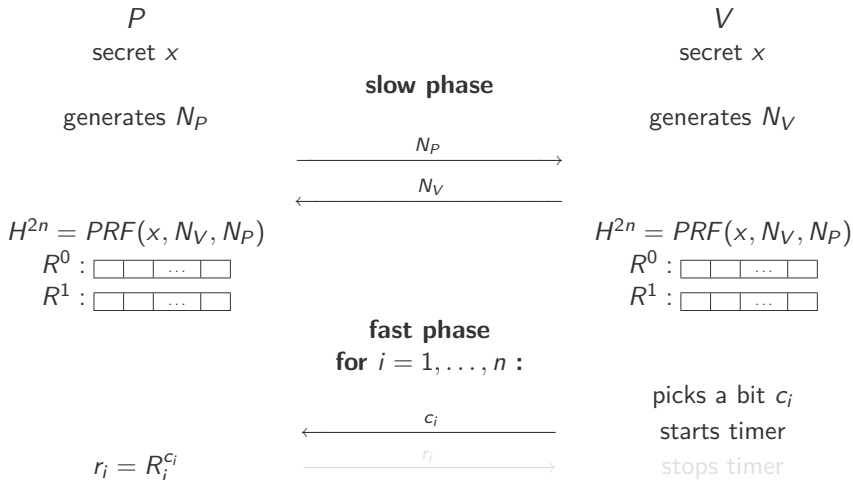
# Hancke and Khun 2005

## The protocol



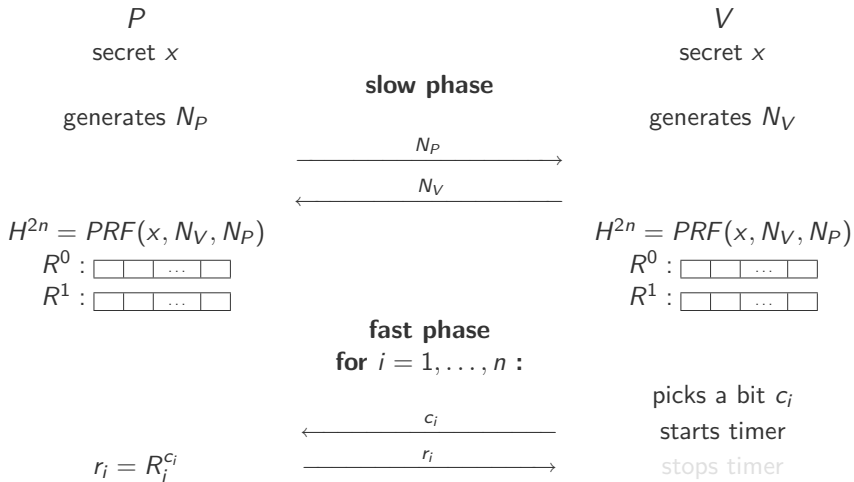
# Hancke and Khun 2005

## The protocol



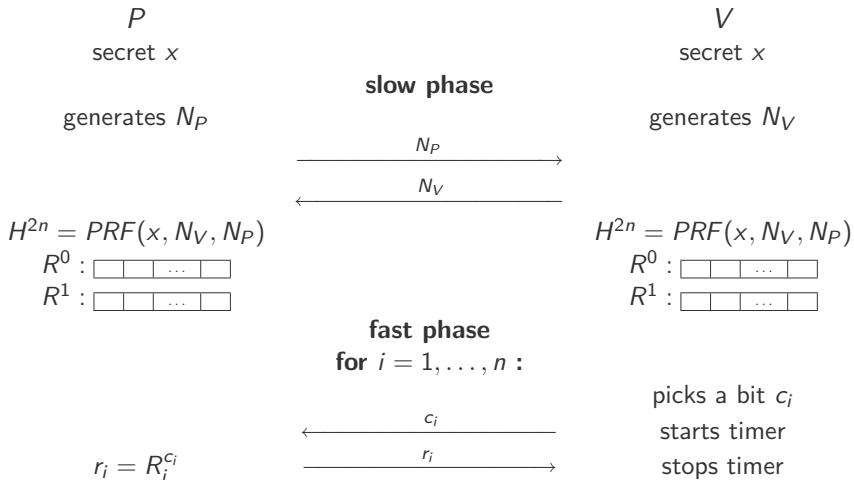
# Hancke and Khun 2005

## The protocol



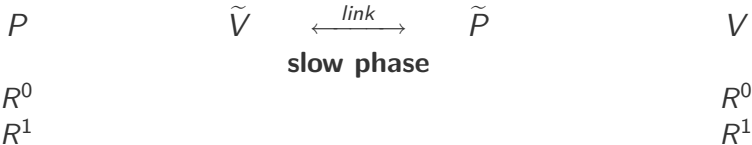
# Hancke and Khun 2005

## The protocol

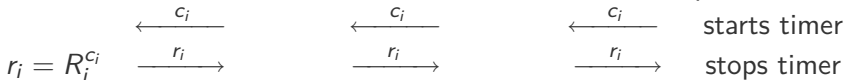


# Hancke and Khun 2005

## Protocol analysis

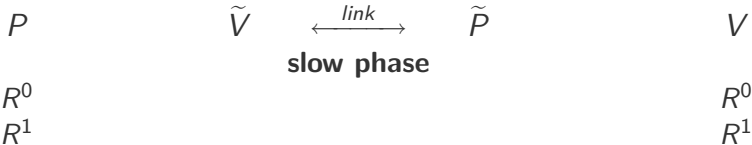


### fast phase

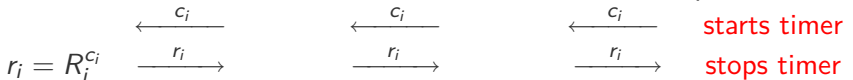


# Hancke and Khun 2005

## Protocol analysis



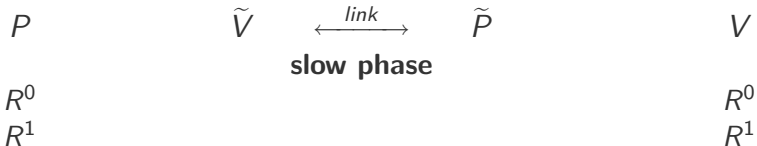
### fast phase



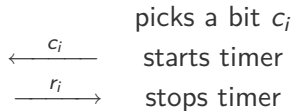


# Hancke and Khun 2005

## Protocol analysis



## fast phase

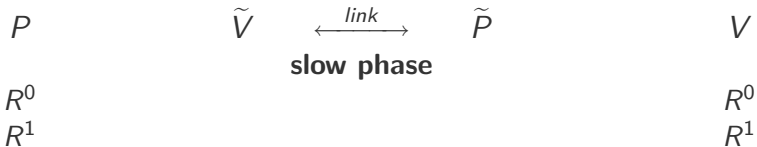


## Adversary strategy

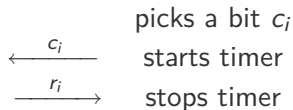
- Without asking strategy

# Hancke and Khun 2005

## Protocol analysis



## fast phase

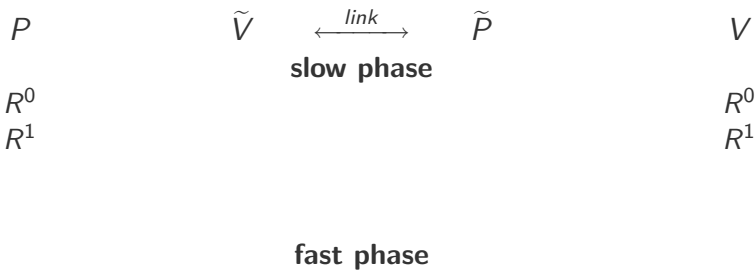


## Adversary strategy

- Without asking strategy :  $\frac{1}{2}$

# Hancke and Khun 2005

## Protocol analysis

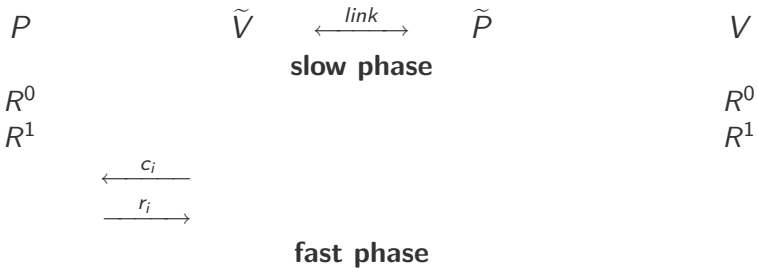


### Adversary strategy

- Asking strategy

# Hancke and Khun 2005

## Protocol analysis

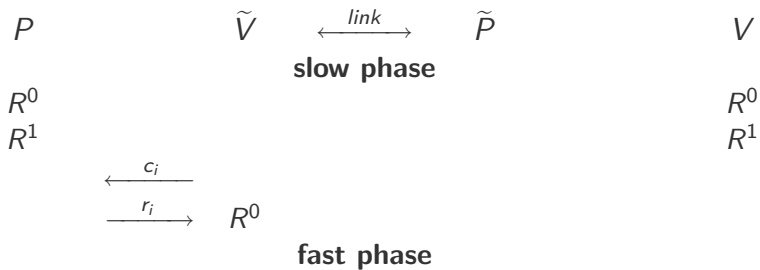


### Adversary strategy

- Asking strategy

# Hancke and Khun 2005

## Protocol analysis

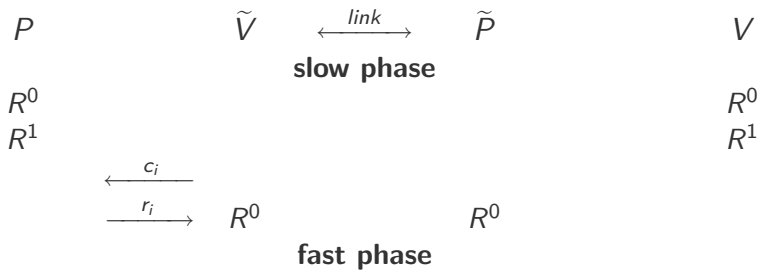


### Adversary strategy

- Asking strategy

# Hancke and Khun 2005

## Protocol analysis

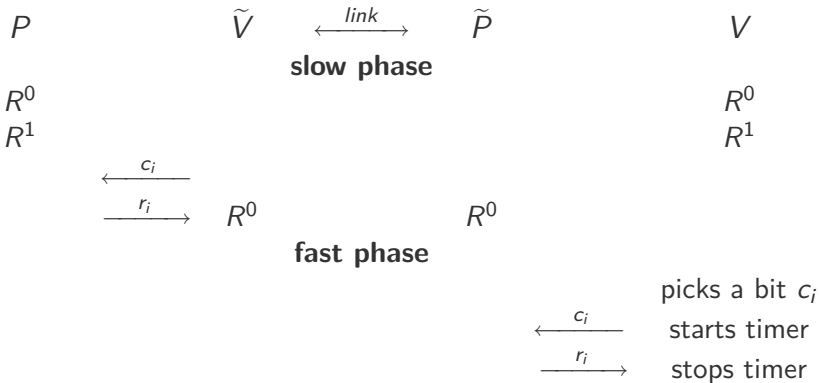


### Adversary strategy

- Asking strategy

# Hancke and Khun 2005

## Protocol analysis

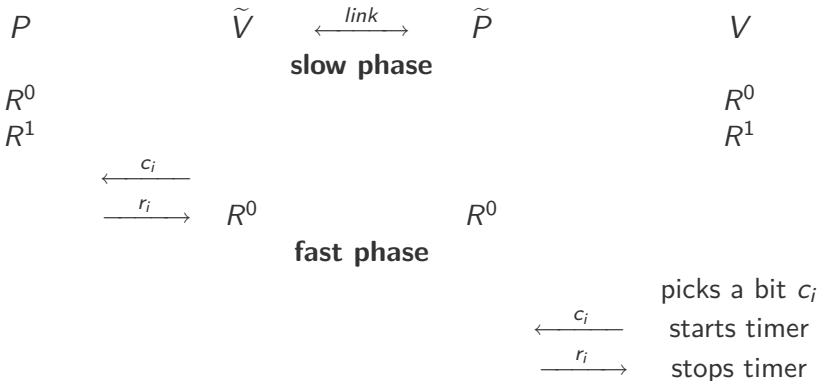


### Adversary strategy

- Asking strategy

# Hancke and Khun 2005

## Protocol analysis



### Adversary strategy

- Asking strategy :  $\frac{1}{2} \cdot 1 + \frac{1}{2} \cdot \frac{1}{2} = \frac{3}{4}$

# Hancke and Khun 2005

## Protocol analysis

### Adversary strategy

- Without asking strategy :  $\frac{1}{2}$
- Asking strategy :  $\frac{1}{2} \cdot 1 + \frac{1}{2} \cdot \frac{1}{2} = \frac{3}{4}$

# Hancke and Khun 2005

## Protocol analysis

### Adversary strategy

- Without asking strategy :  $\frac{1}{2}$
- Asking strategy :  $\frac{1}{2} \cdot 1 + \frac{1}{2} \cdot \frac{1}{2} = \frac{3}{4}$

### Adversary success probability

The adversary chooses the asking strategy, and succeeds with probability :

$$P_{HK} = \left(\frac{3}{4}\right)^n$$

# Munilla and Peinado 2008

## Protocol analysis

### New concept

- *Void challenge*

# Munilla and Peinado 2008

## Protocol analysis

### New concept

- *Void challenge*

### Improvement

- An additional register
- A final message

# Munilla and Peinado 2008

## Protocol analysis

### New concept

- *Void challenge*

### Improvement

- An additional register
- A final message

### Adversary strategy

$p_f$  : Probability that a full challenge appears.

Without asking strategy :  $p_{noask} = \left(1 - \frac{p_f}{2}\right)^n$ .

Asking strategy :  $p_{ask} = \left(p_f \cdot \frac{3}{4}\right)^n$ .

# Munilla and Peinado 2008

## Protocol analysis

### Adversary strategy

$p_f$  : Probability that a full challenge appears.

Without asking strategy :  $p_{noask} = \left(1 - \frac{p_f}{2}\right)^n$ .

Asking strategy :  $p_{ask} = \left(p_f \cdot \frac{3}{4}\right)^n$ .

### Success probability

With  $p_f = \frac{3}{4}$ , the adversary chooses the no asking strategy :

$$P_{MP} = \left(\frac{5}{8}\right)^n$$

# A first proposal : MUSE-3 HK

## MUltiState Enhancement

### Assumption

Same as in Munilla and Peinado's protocol : the verifier and the prover have the capability to send a third state.

# A first proposal : MUSE-3 HK

## MUltiState Enhancement

### Assumption

Same as in Munilla and Peinado's protocol : the verifier and the prover have the capability to send a third state.

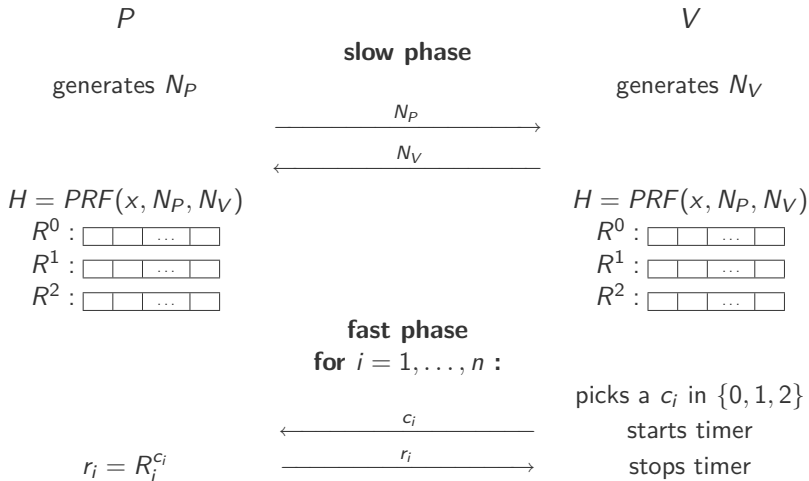
### Our Improvement

The MUltiState Enhancement is a generic method, where we consider the three states in the same manner.

These three states are called the *3-symbols*.

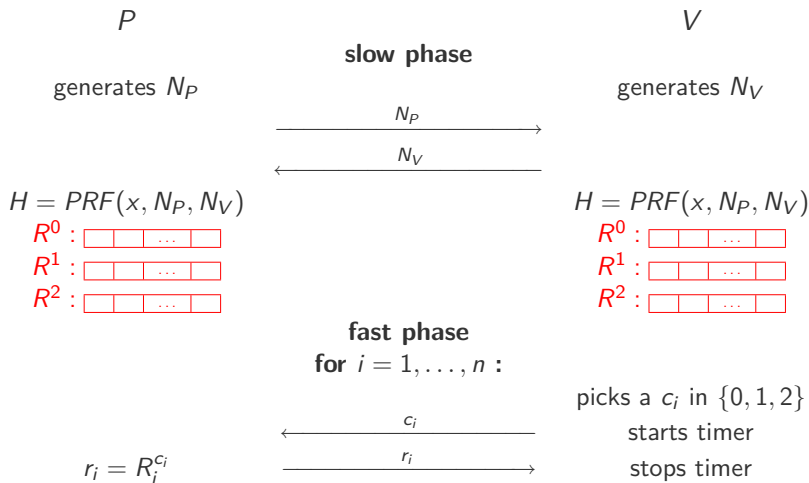
# A first proposal : MUSE-3 HK

## The protocol



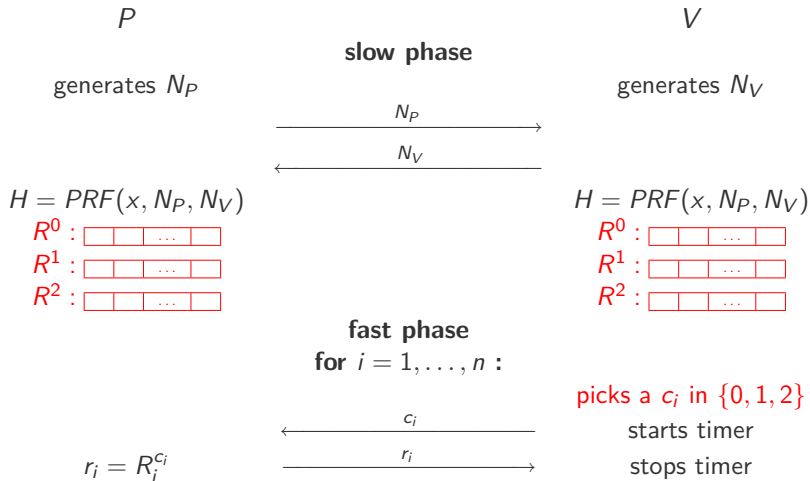
# A first proposal : MUSE-3 HK

## The protocol



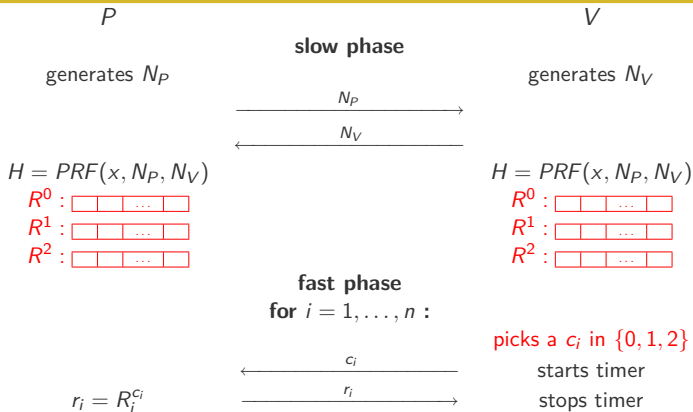
# A first proposal : MUSE-3 HK

## The protocol



# A first proposal : MUSE-3 HK

## The protocol

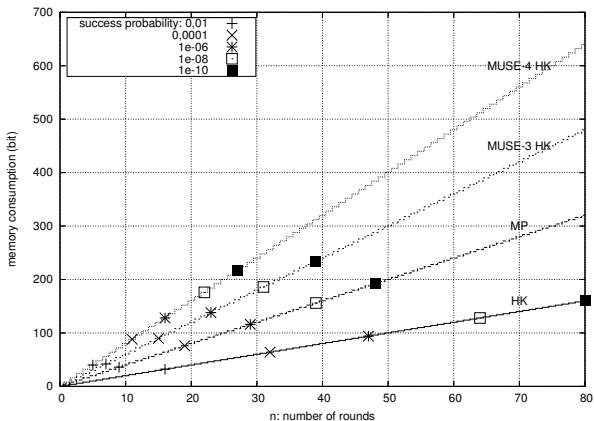


## Adversary success probability

At each rounds  $\frac{1}{3} \cdot 1 + \frac{2}{3} \cdot \frac{1}{3} = \frac{5}{9}$ .

# Our second proposal : MUSE-4 HK

## Comparison



Security :  $10^{-4}$

	Mem	$n$
HK	64	32
MP	76	19
MUSE-3 HK	90	15
MUSE-4 HK	88	11

# Conclusion

## Our contributions

- Based on Munilla and Peinado's assumptions, a third state at the lowest layer, we design a better improvement of Hancke and Kuhn's protocol.
- For theoretical purpose, we generalize this approach and study its behavior regarding the numbers of rounds, the memory consumption, and the security.

## Further Work

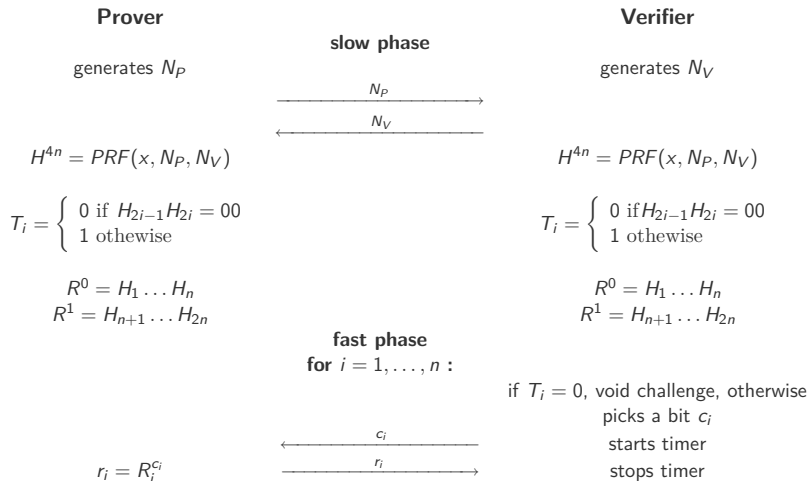
- To look deeper on the implementability of MUSE- $p$  HK.
- To study the behavior of MUSE- $p$  HK regarding the other relay attacks.

# Conclusion

Any questions?

# Munilla and Peinado 2008

## The protocol



# The generalization : MUSE- $p$ HK

## Generation of the registers

### Few mathematics

To generate one  $p$ -symbol, one needs  $\lceil \log_2(p) \rceil$  bits.

$\mathcal{A} \subseteq \mathbb{F}_{2^{\lceil \log_2(p) \rceil}}$  such that  $\#\mathcal{A} = p$ .

$q$  the probability of picking an element of  $\mathcal{A}$  is equal to  $\frac{p}{2^{\lceil \log_2(p) \rceil}}$ .

$A_i$  the event of picking an element of  $\mathcal{A}$ ,  $P(A_i) = (1 - q)^{i-1} \cdot q$ .

Expectation of  $P(A_i)$  is equal to  $\frac{1}{q}$ .

### result

To fill up a registers of  $n$   $p$ -symbol, one needs :

$$n \cdot \lceil \log_2(p) \rceil \cdot \frac{2^{\lceil \log_2(p) \rceil}}{p}$$

# The generalization : MUSE- $p$ HK

## Memory consumption

