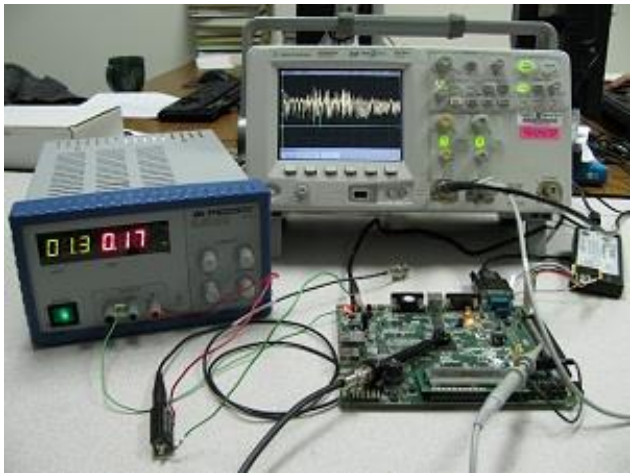
Leakage-Resilient Symmetric Cryptography
(Overview of the ERC Project CRASH, part II)



François-Xavier Standaert

UCL Crypto Group, Belgium

INDOCRYPT, December 2016

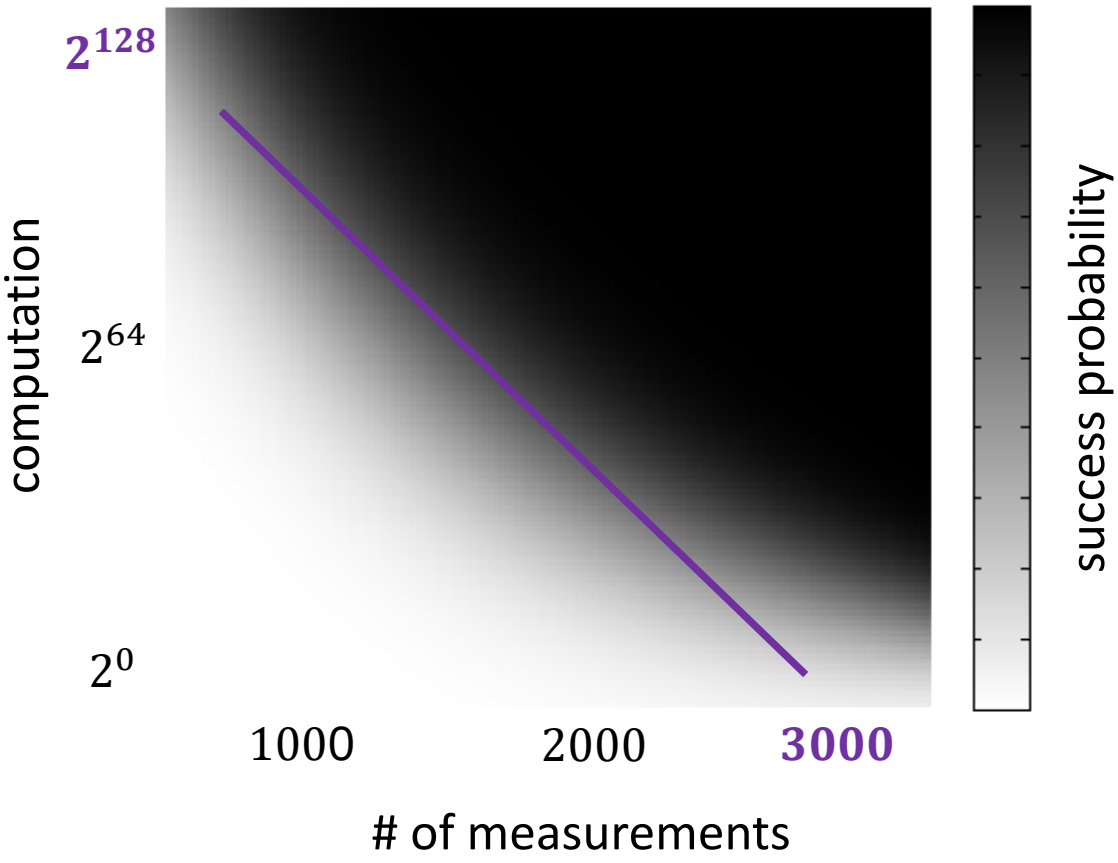
Outline

- Introduction
- Natural PRGs/PRFs and separation result
- PRGs & theoretical challenges
  - Ensuring independence
  - Bounding the leakage
  - The simulatable leakage attempt
- Protocols & practical challenges
  - Authentication & encryption
  - Authenticated encryption
  - Initialization issue
- Summary and conclusions

Outline

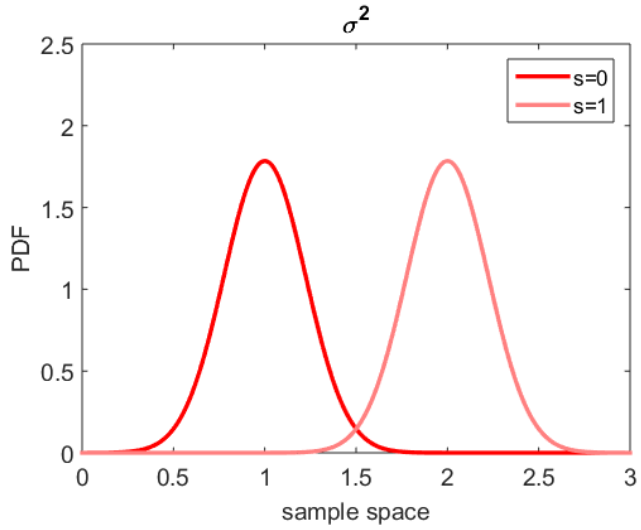
- Introduction
- Natural PRGs/PRFs and separation result
- PRGs & theoretical challenges
  - Ensuring independence
  - Bounding the leakage
  - The simulatable leakage attempt
- Protocols & practical challenges
  - Authentication & encryption
  - Authenticated encryption
  - Initialization issue
- Summary and conclusions

Side-channel attacks

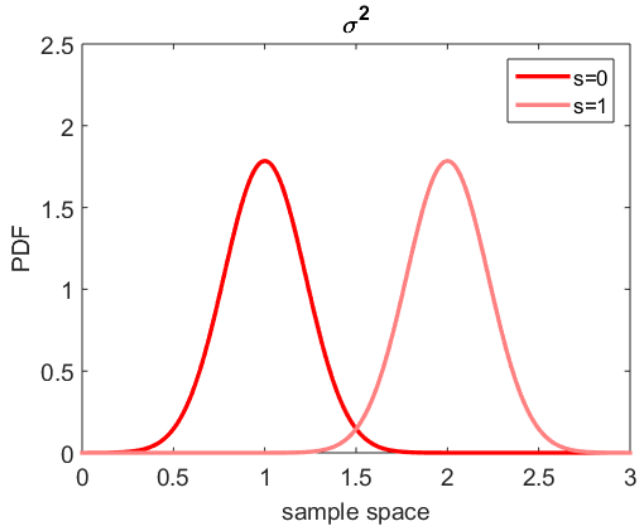


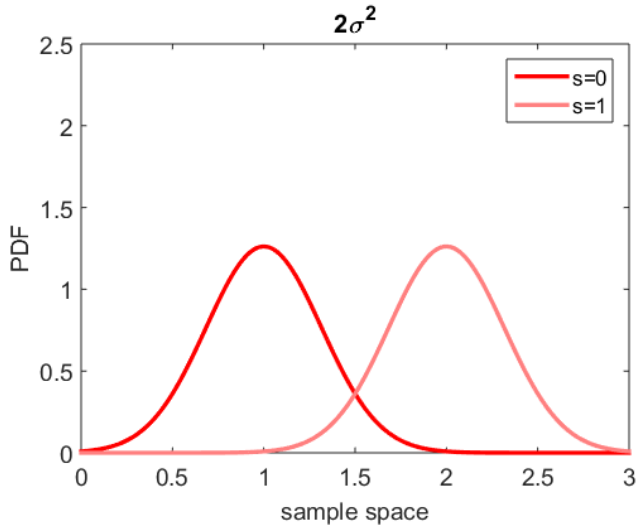
- ≈ physical attack that decreases security exponentially in the # of measurements

Noise (hardware countermeasures)



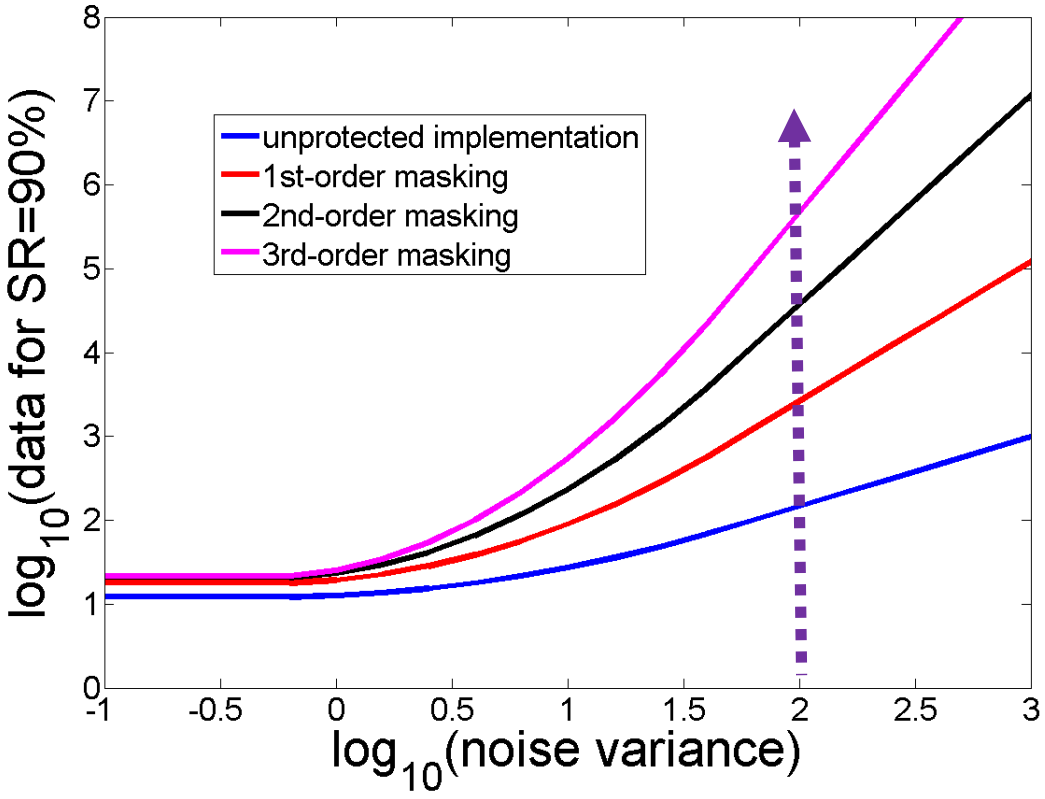
Noise (hardware countermeasures)





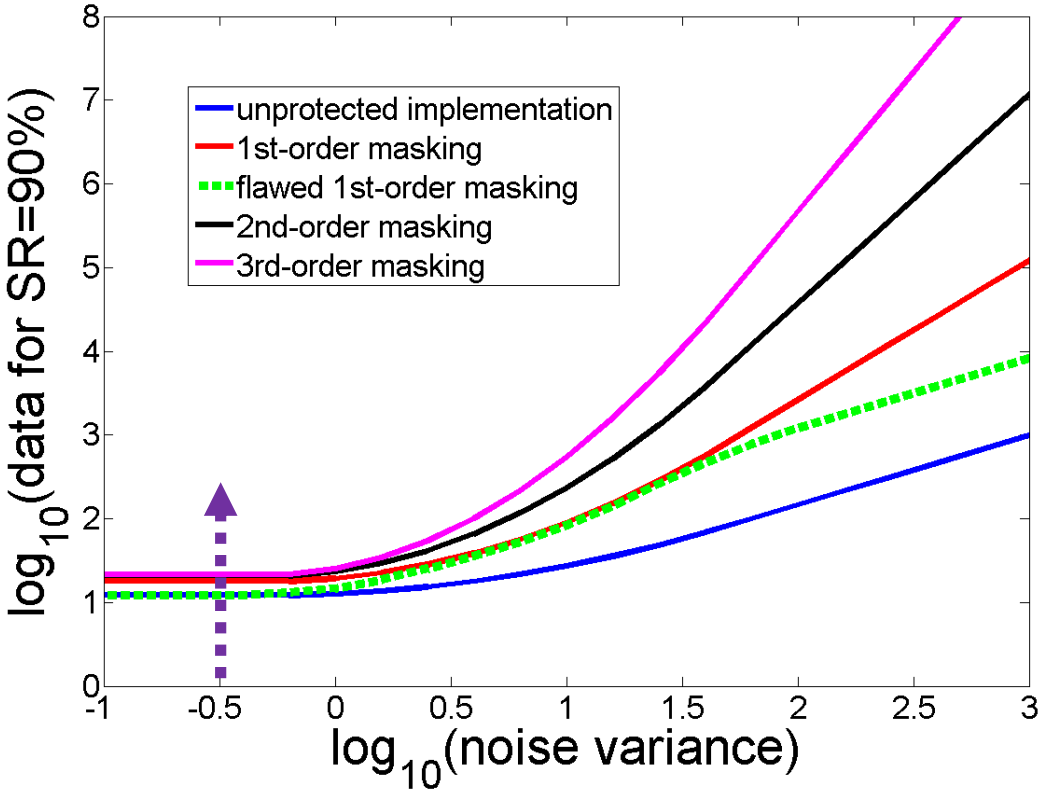
- Additive noise $\approx$ cost $\times$ 2 $\Rightarrow$ security $\times$ 2
  $\Rightarrow$ not a good (crypto) security parameter

Masking (noise amplification)



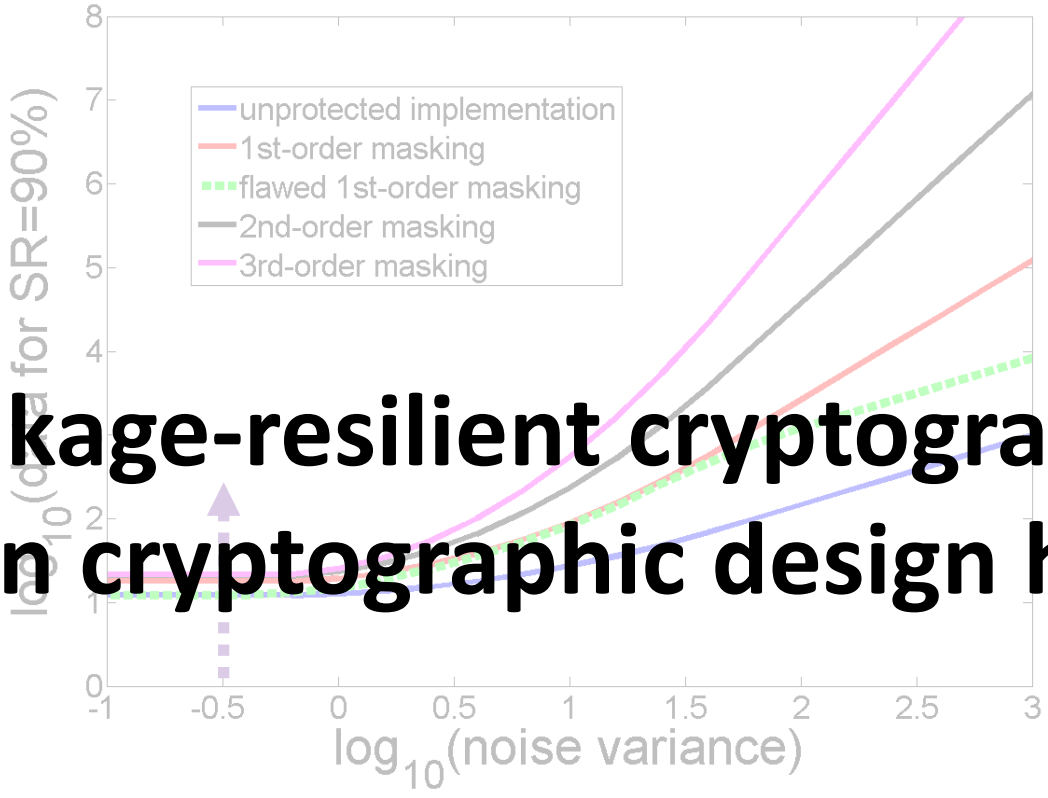
- ≈ secret sharing allows increasing security exponentially in the # of shares (*d*)

Masking limitations



- Problem: masking is hard to implement (noise & independence) and is **<u>expensive</u>** (cost > d^2)

Masking limitations



Leakage-resilient cryptography
≈ can cryptographic design help?

- Problem: masking is hard to implement (noise & independence) and is **expensive** (cost > d^2)

Seed results

- Micali & Reyzin 2004
  - *Physically observable cryptography*
  - « Only computation leaks » assumption
    - Used in all following works
  - Indistinguishability ≠ unpredictability (with L)
    - Impact for encryption & authentication

Seed results

- Micali & Reyzin 2004
  - *Physically observable cryptography*
  - « Only computation leaks » assumption
    - Used in all following works
  - Indistinguishability ≠ unpredictability (with L)
    - Impact for encryption & authentication

- Dziembowski & Pietrzak 2008
  - *Leakage-resilient cryptography*
    - First (nearly) practical stream cipher construction analyzed in a formal model

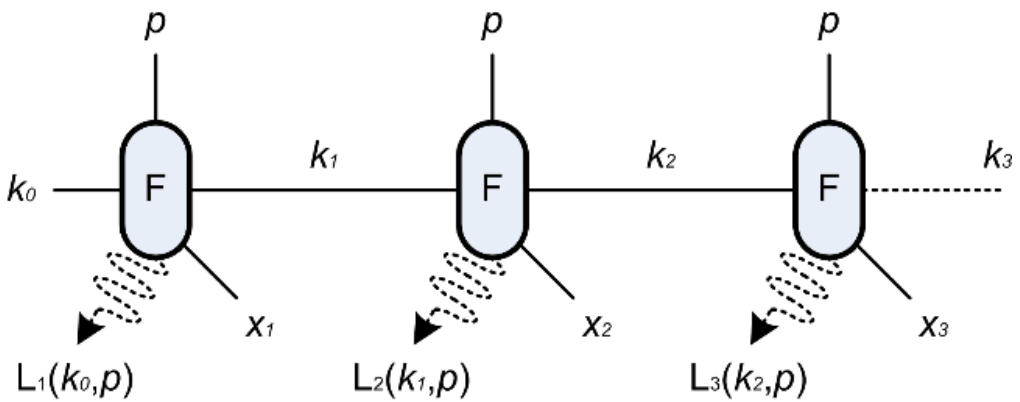
Outline

- Introduction
- Natural PRGs/PRFs and separation result
- PRGs & theoretical challenges
  - Ensuring independence
  - Bounding the leakage ≈ noise
  - The simulatable leakage attempt
- Protocols & practical challenges
  - Authentication & encryption
  - Authenticated encryption
  - Initialization issue
- Summary and conclusions

Outline

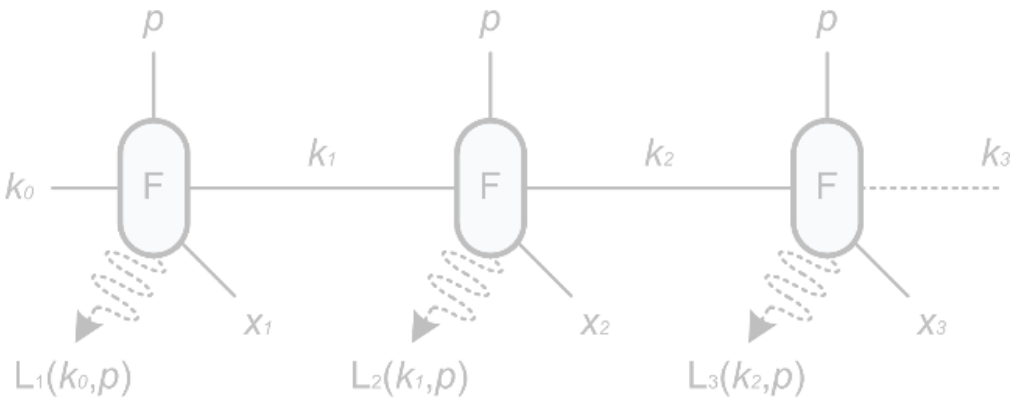
- Introduction
- Natural PRGs/PRFs and separation result
- PRGs & theoretical challenges
  - Ensuring independence
  - Bounding the leakage
  - The simulatable leakage attempt
- Protocols & practical challenges
  - Authentication & encryption
  - Authenticated encryption
  - Initialization issue
- Summary and conclusions

- Most natural construction: forward-secure PRG



Stateful PRGs

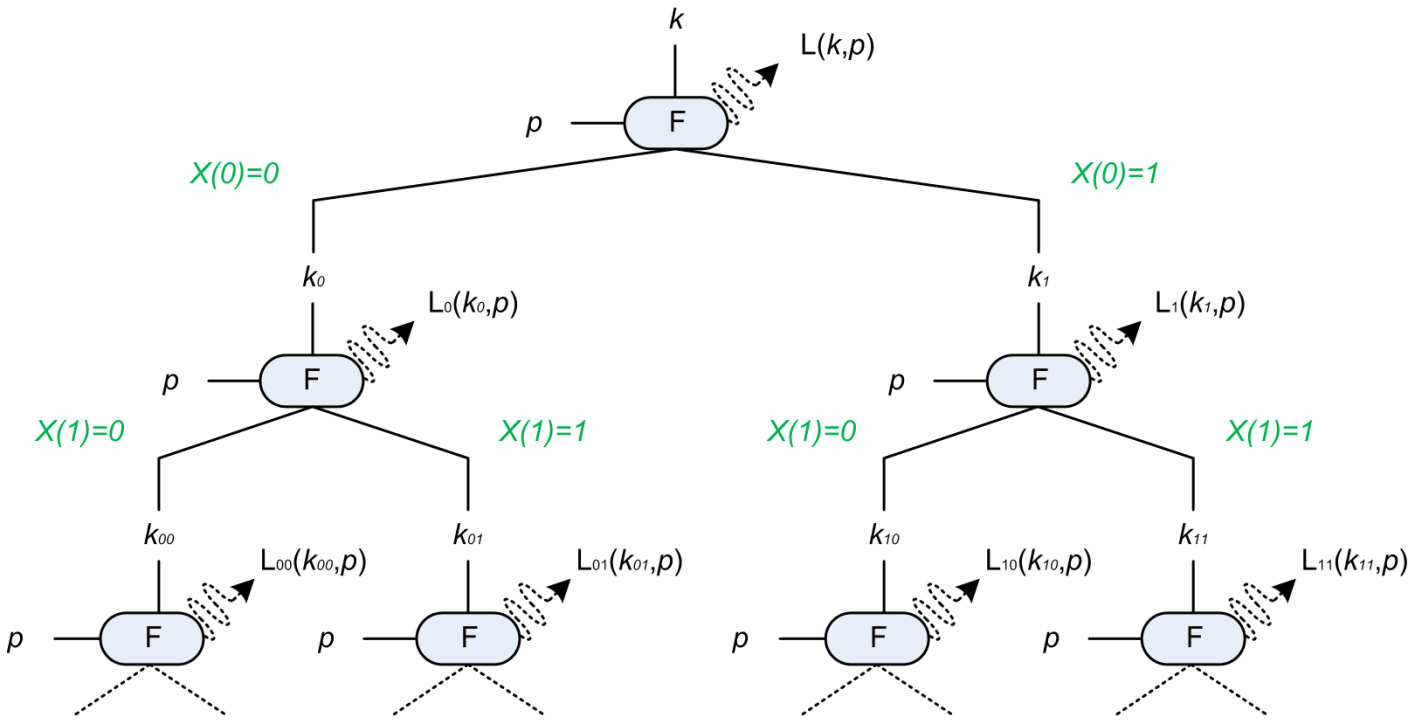
- Most natural construction: forward-secure PRG



- Re-keying impact: bounds the number of (noisy) measurements per key (*prevents averaging*)

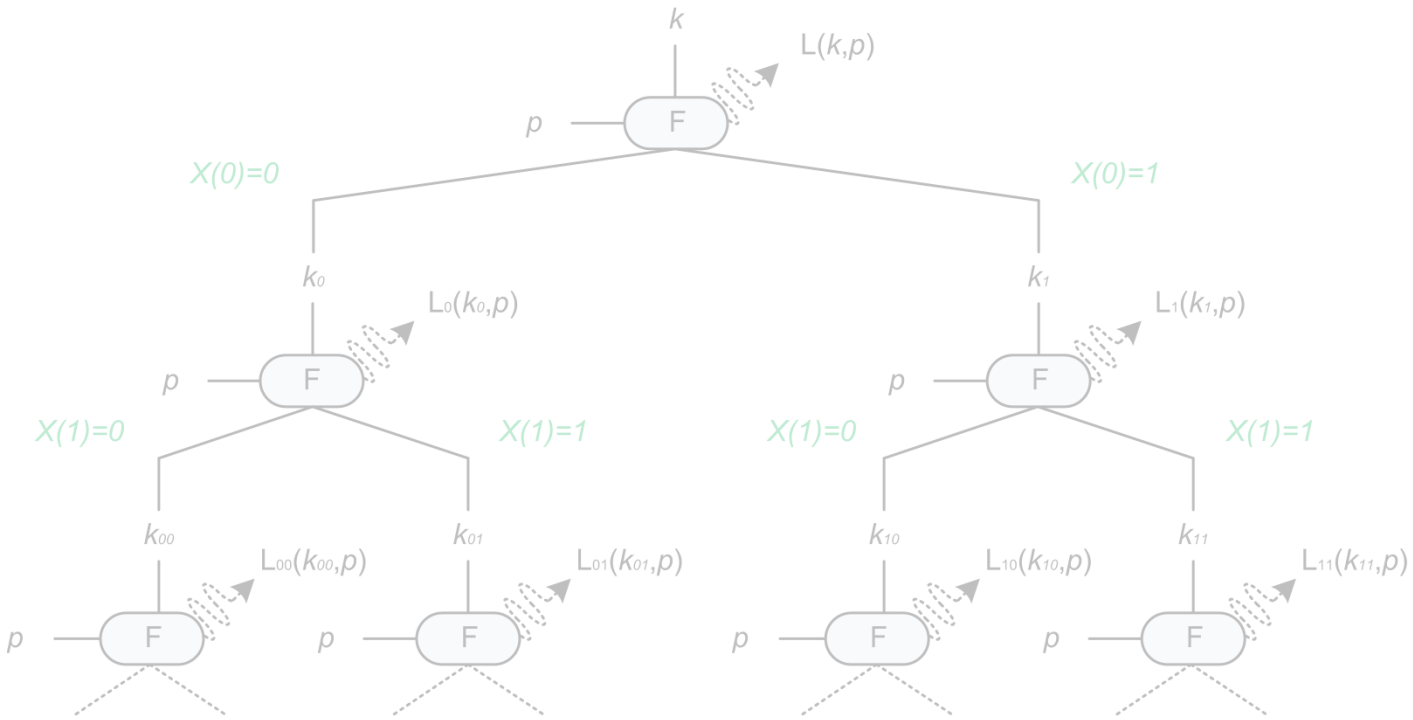
Stateless PRFs (or PRPs)

- Most natural construction: GGM tree



Stateless PRFs (or PRPs)

- Most natural construction: GGM tree

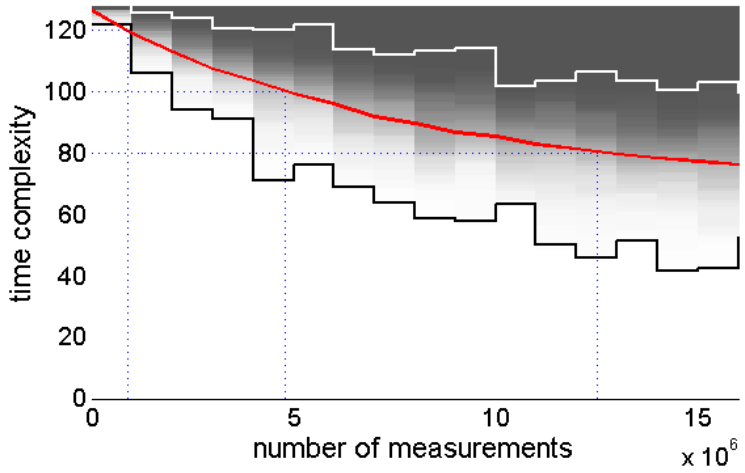


- Re-keying impact: bounds the number of noise-free observations per key (*allows averaging*)

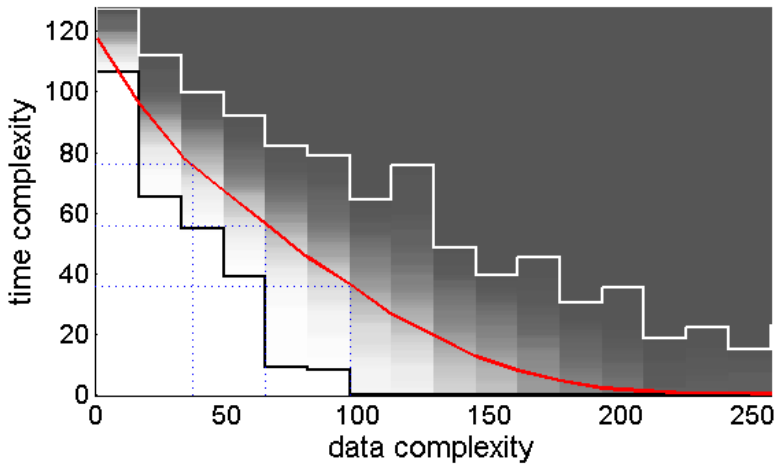
The stateful / stateless separation

- Key recovery security (standard SCA):

PRG



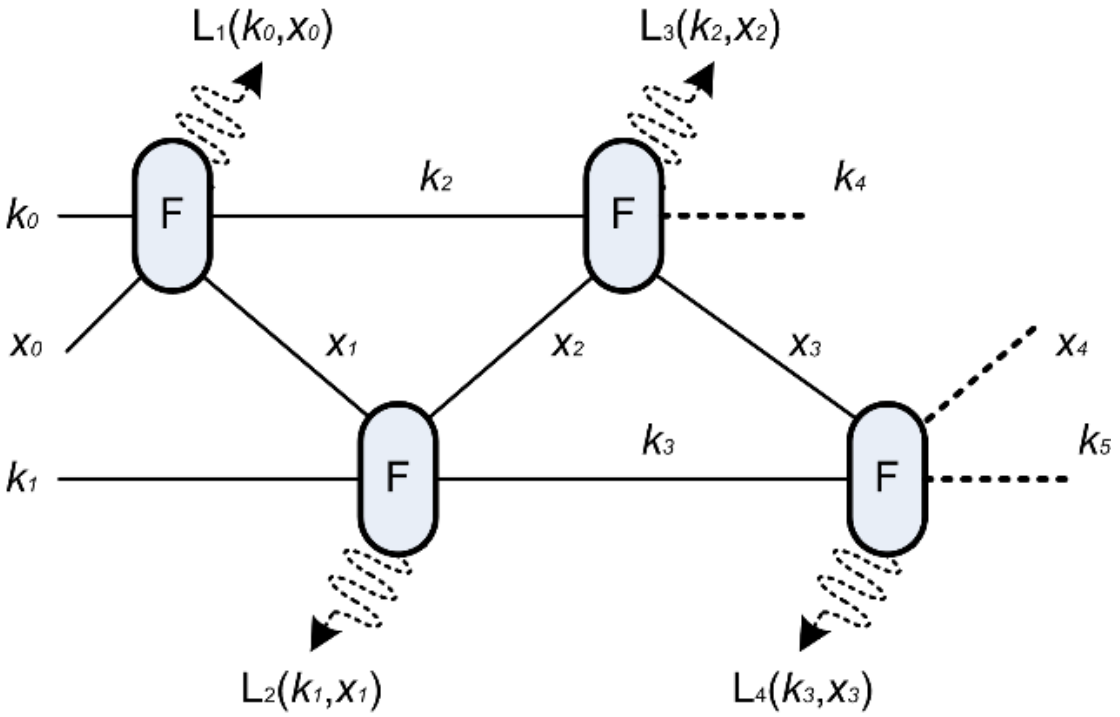
PRF



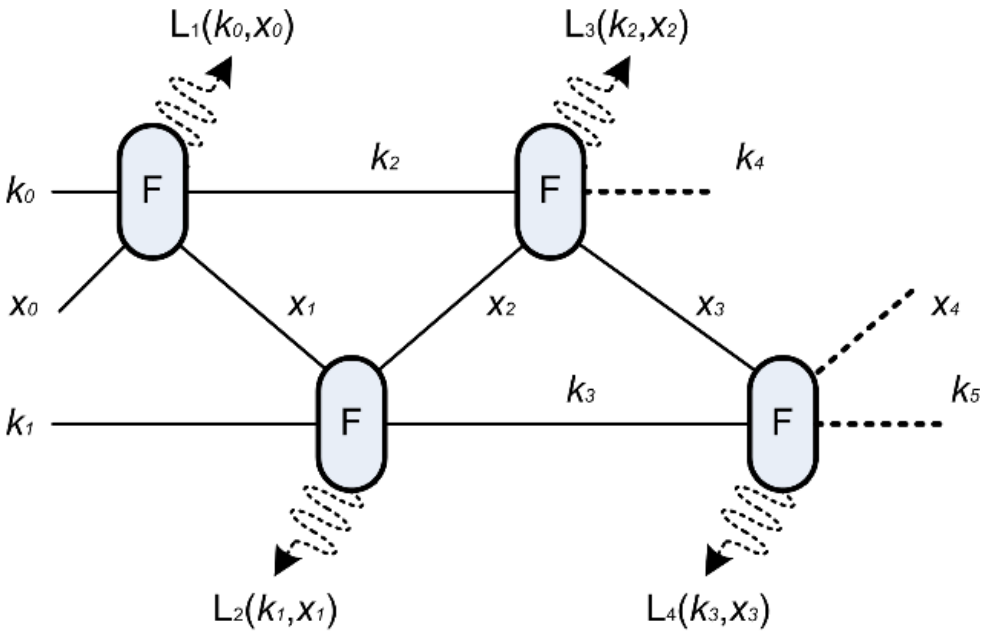
- « Bounded security » for the PRG only
  - Despite proofs being similar (i.e., assumption issue)

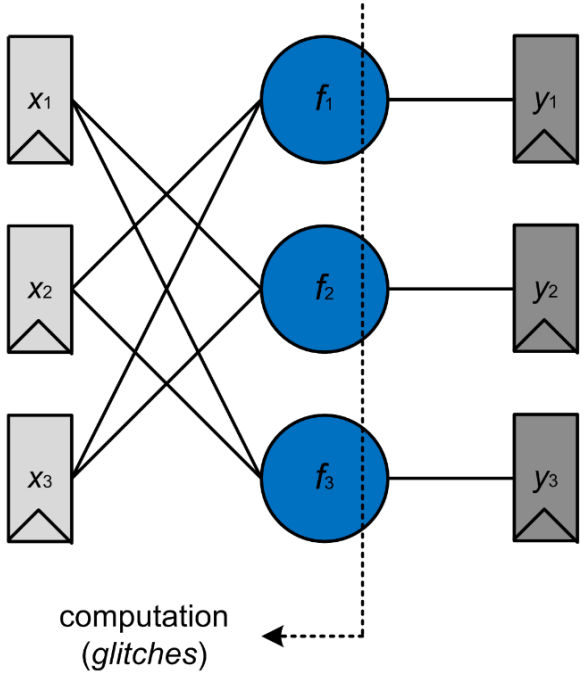
Outline

- Introduction
- Natural PRGs/PRFs and separation result
- PRGs & theoretical challenges
  - Ensuring independence
  - Bounding the leakage
  - The simulatable leakage attempt
- Protocols & practical challenges
  - Authentication & encryption
  - Authenticated encryption
  - Initialization issue
- Summary and conclusions

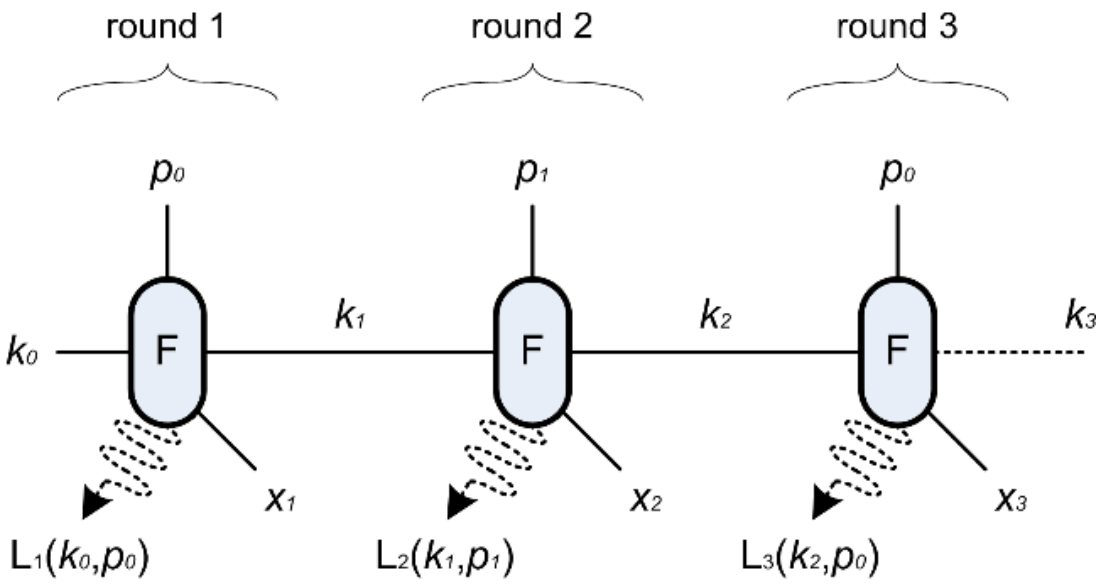


- L modeled as a polytime function => alternating structure prevents « precomputation attack »

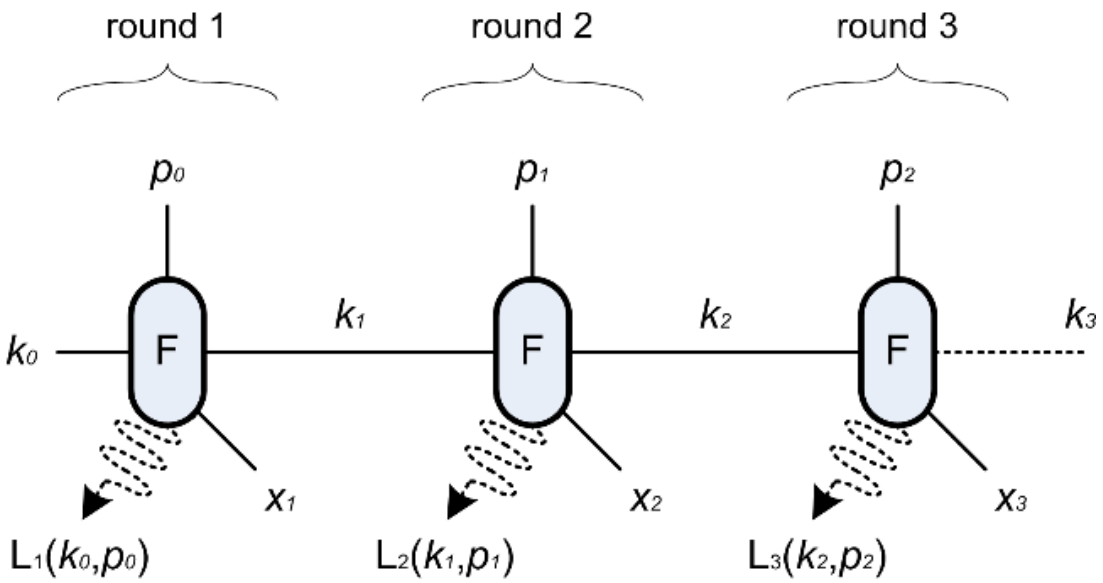




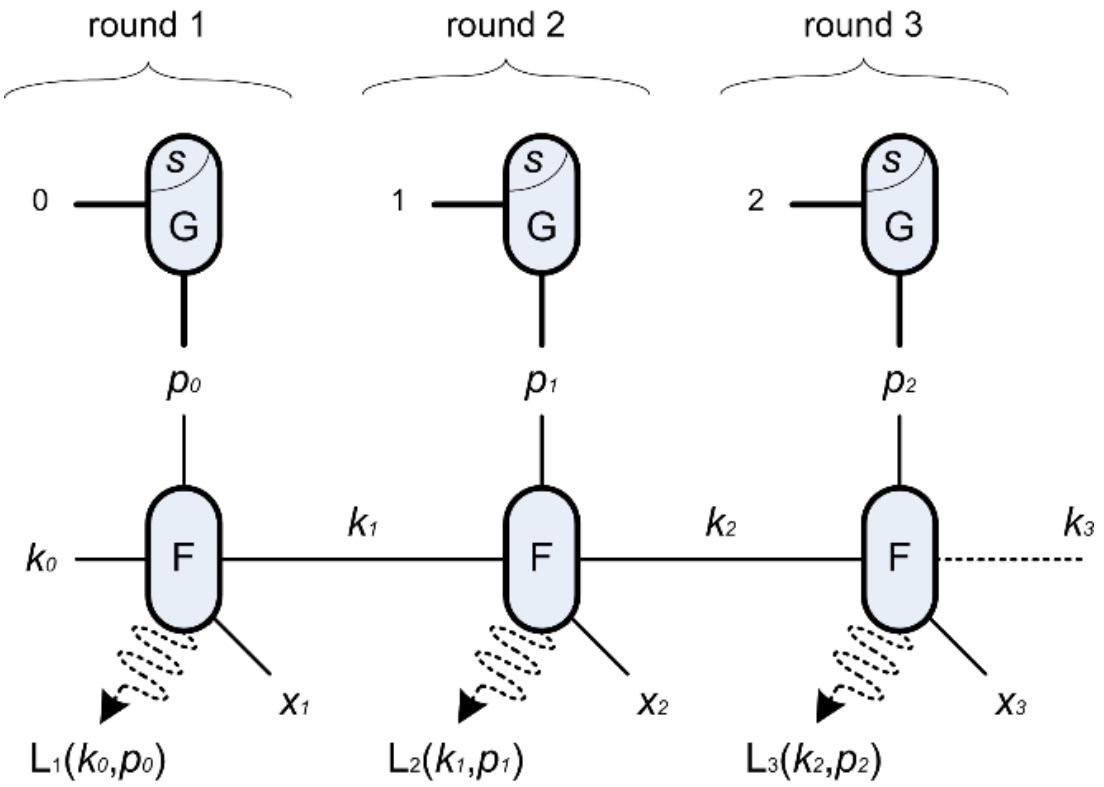
- Note: looks artificial but is in fact funnily similar to the idea of threshold implementations



- Alternating randomness (to save key material)
  - Unfortunately not sufficient (CHES 2012)…

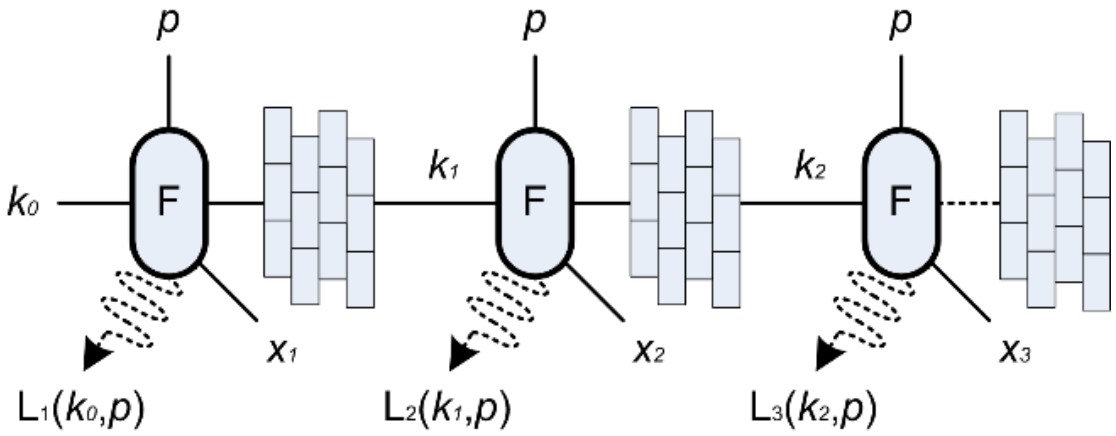


- Fresh randomness in each round
  - Sound but expensive (generated after L)



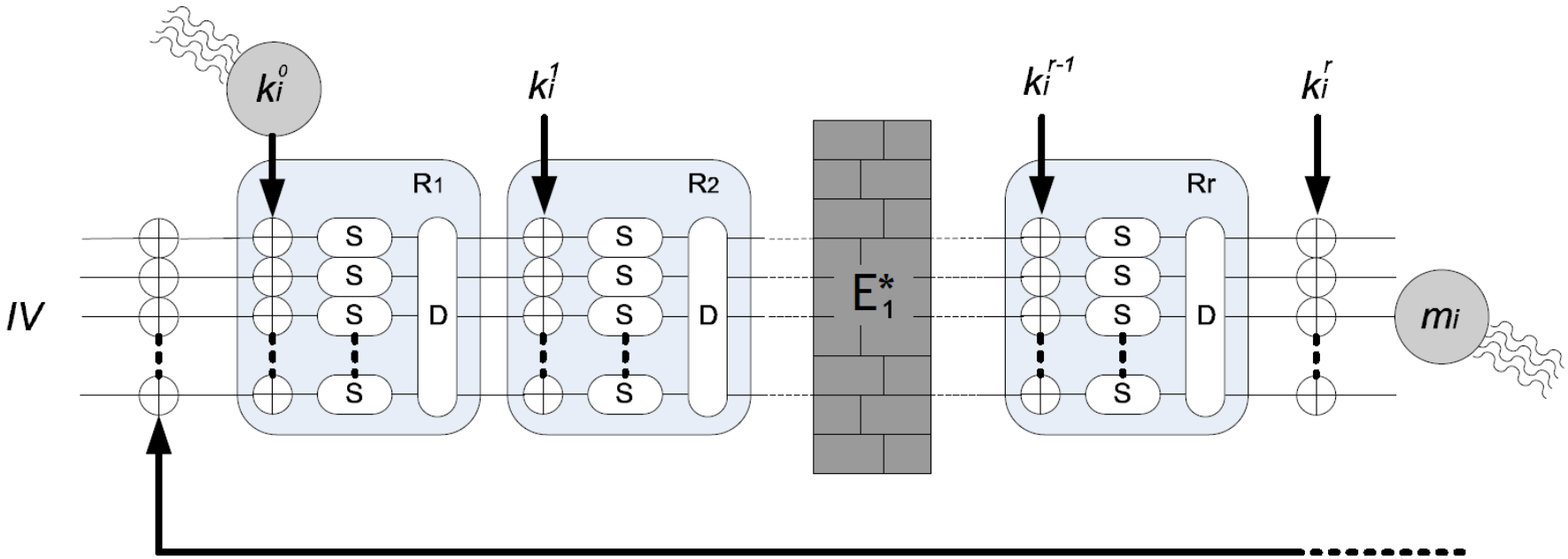
- Public randomness generated from a PRG
  - (Non quantitative) proof in MiniCrypt

CCS 2010 again (I)



- Most natural construction proven under a (non standard) random oracle assumption
  - L cannot query the random oracle

CCS 2010 again (II)

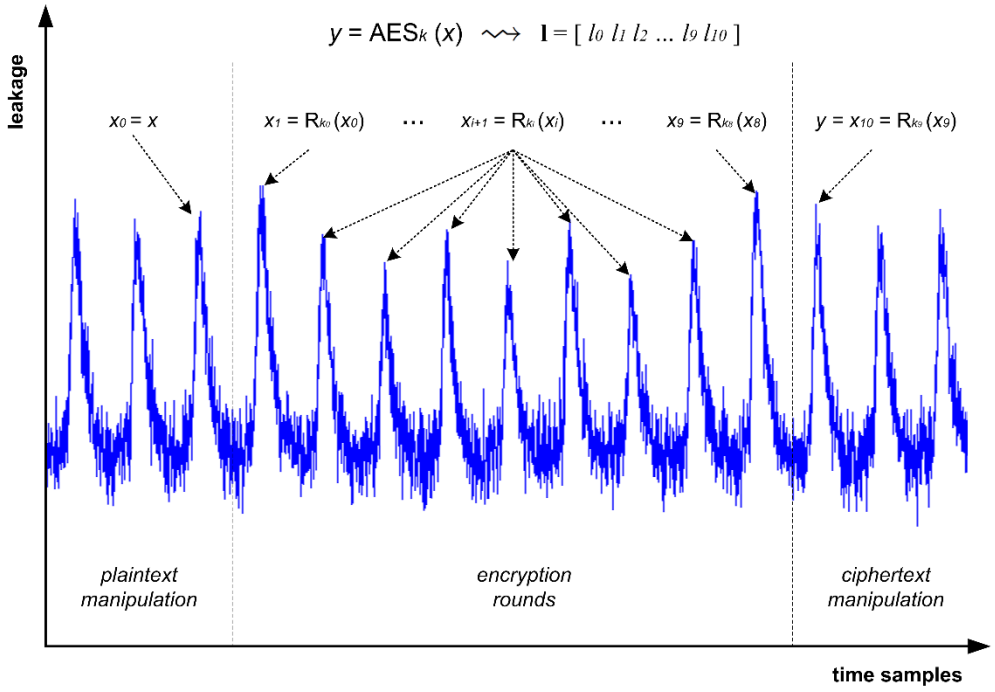


- ≈ formalization of early re-keying attempts
  - e.g., ASIACCS 2008: internal wall within AES
  - e.g., early patents in the field from CRI
  - (*Where it was already clear that init. is challenging*!)

Outline

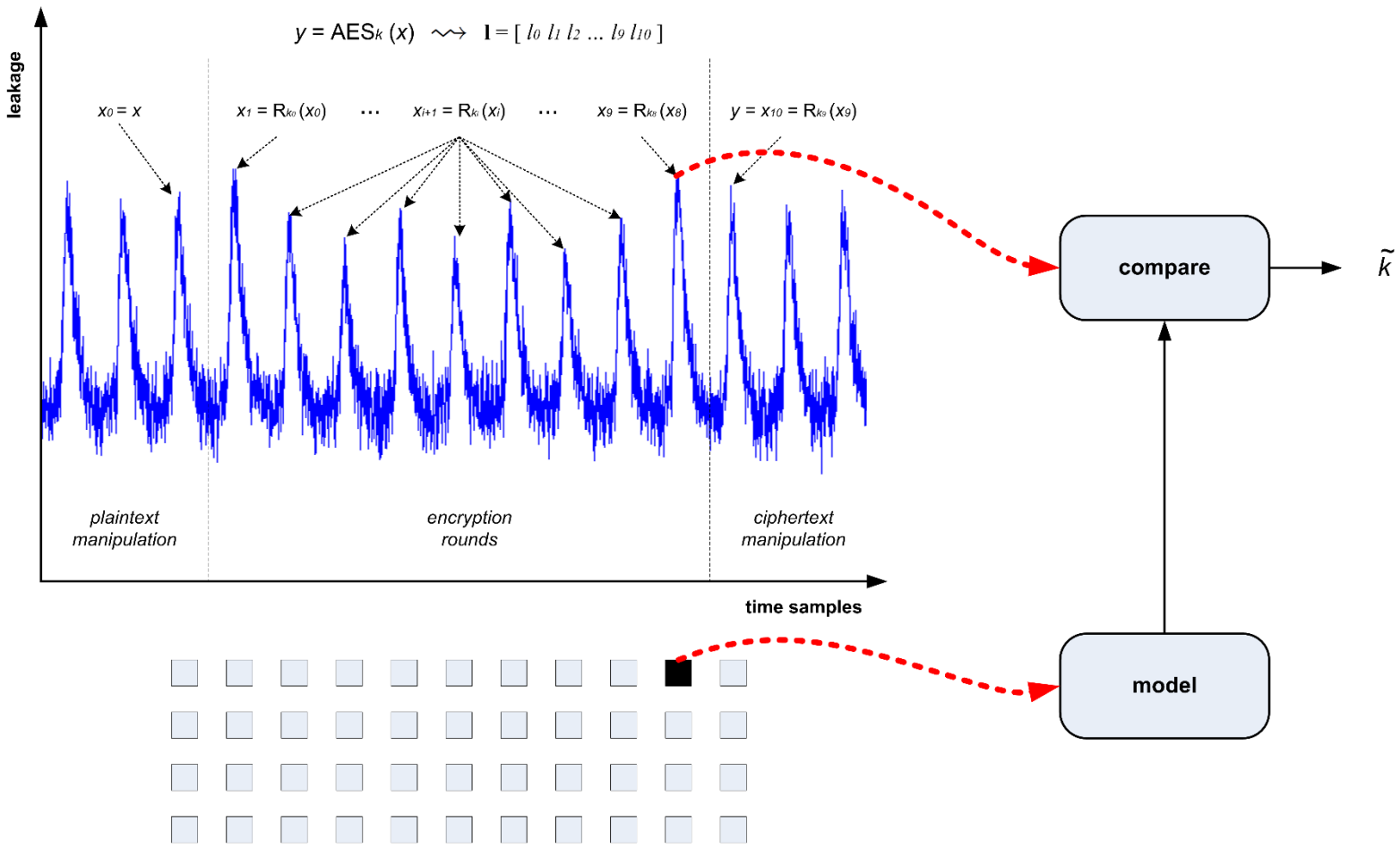
- Introduction
- Natural PRGs/PRFs and separation result
- PRGs & theoretical challenges
  - Ensuring independence
  - Bounding the leakage
  - The simulatable leakage attempt
- Protocols & practical challenges
  - Authentication & encryption
  - Authenticated encryption
  - Initialization issue
- Summary and conclusions

Bounded range



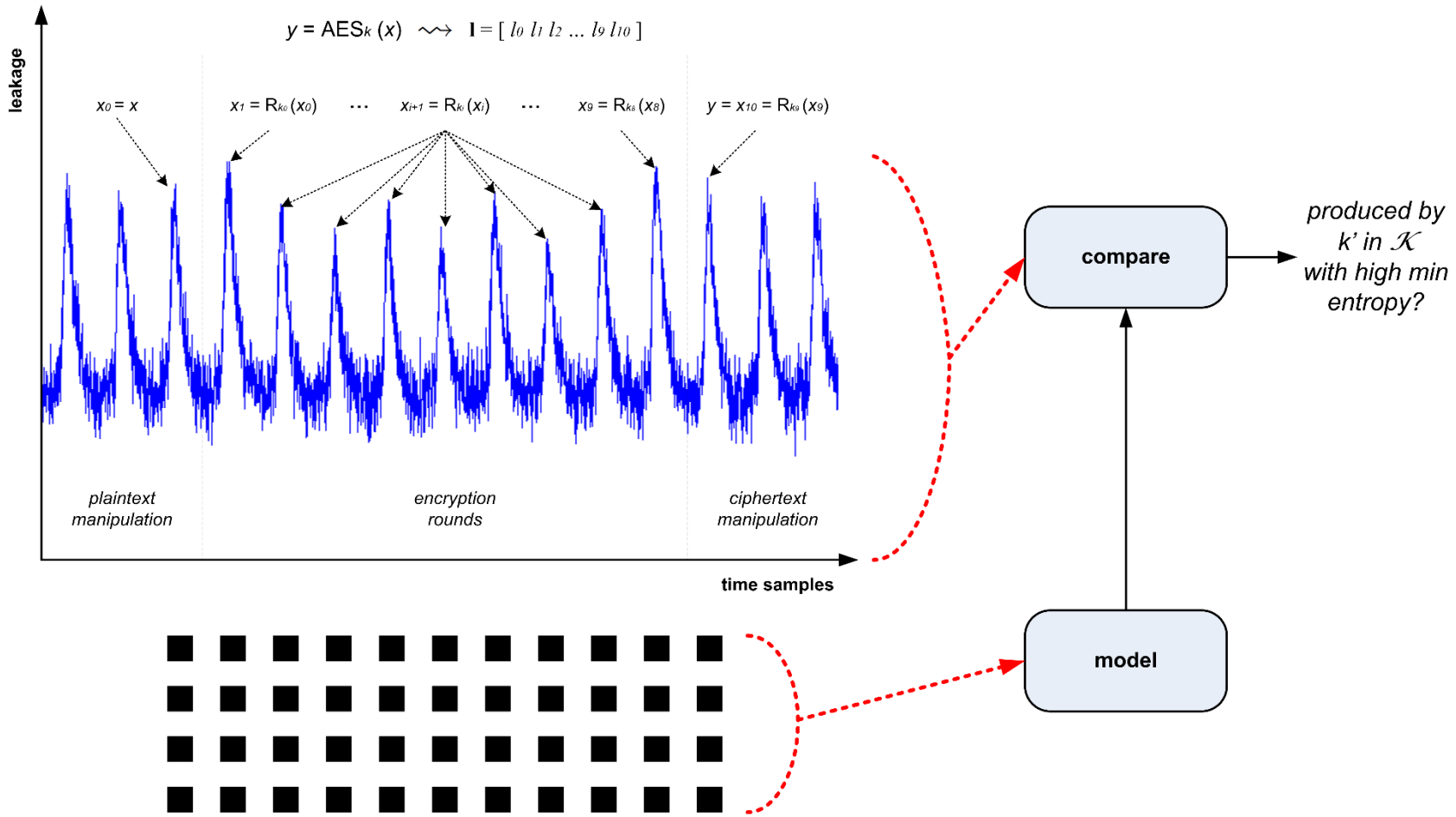
- Unrealistic: leakages ≈ Gbytes of data

Security against DPA



- Not sufficient to prove anything

Key has high HILL pseudoentropy



- Hard to guarantee (indistinguishability-based)

Wrapping up

- Finding sound ways to guarantee independence between multiple PRG rounds and to bound their leakage is notorioulsy difficult (!)

Wrapping up

- Finding sound ways to guarantee independence between multiple PRG rounds and to bound their leakage is notorioulsy difficult (!)
- No perfectly satisfying solution so far
  - e.g., assuming L polytime is not realistic but no other restrictions seem to work
  - ∃ a gap between what proofs require and what engineers can guarantee (evaluate)

Wrapping up

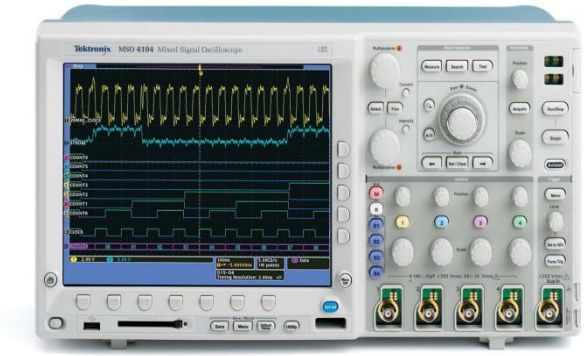
- Finding sound ways to guarantee independence between multiple PRG rounds and to bound their leakage is notorioulsy difficult (!)
- No perfectly satisfying solution so far
  - e.g., assuming L polytime is not realistic but no other restrictions seem to work
  - ∃ a gap between what proofs require and what engineers can guarantee (evaluate)
  - *Independent of concrete security (!)*

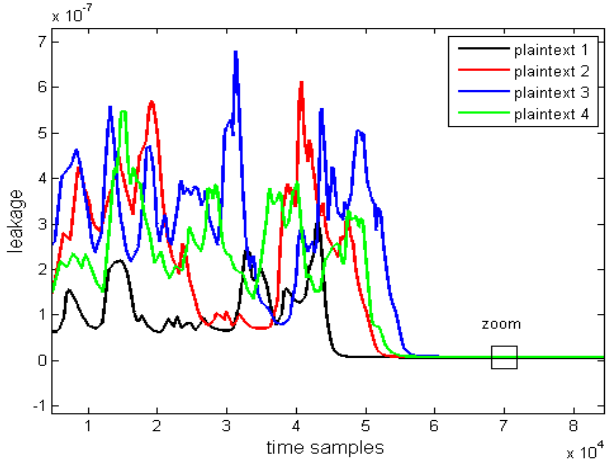
Outline

- Introduction
- Natural PRGs/PRFs and separation result
- PRGs & theoretical challenges
  - Ensuring independence
  - Bounding the leakage
  - The simulatable leakage attempt
- Protocols & practical challenges
  - Authentication & encryption
  - Authenticated encryption
  - Initialization issue
- Summary and conclusions

Looking for physical assumptions

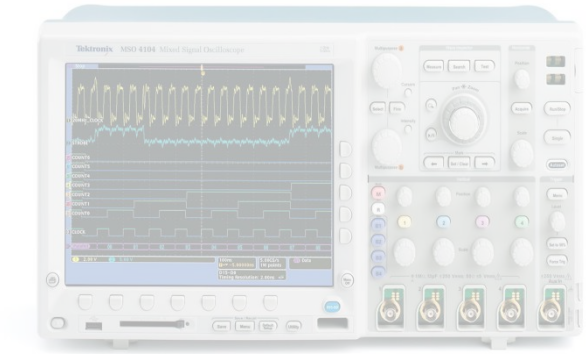
- Main issue: leakage function is hard to model
  - It solves Maxwell's equations
  - But circuits give immediate solutions

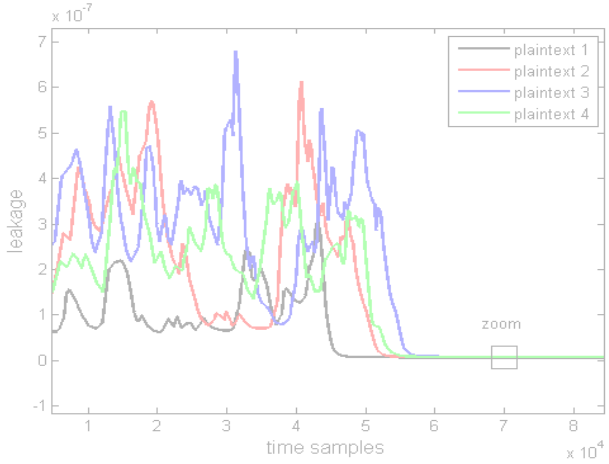




Looking for physical assumptions

- Main issue: leakage function is hard to model
  - It solves Maxwell's equations
  - But circuits give immediate solutions

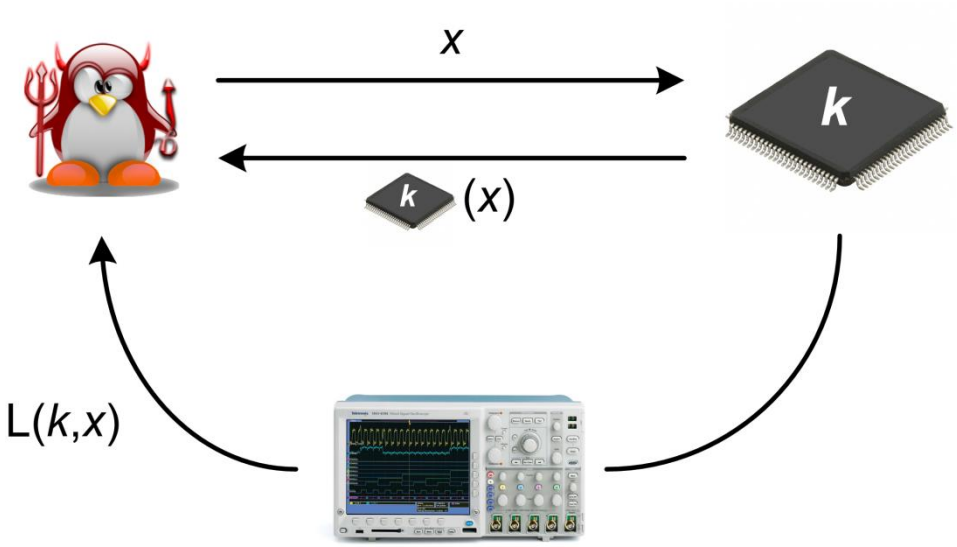




⇒ Just don't model it!

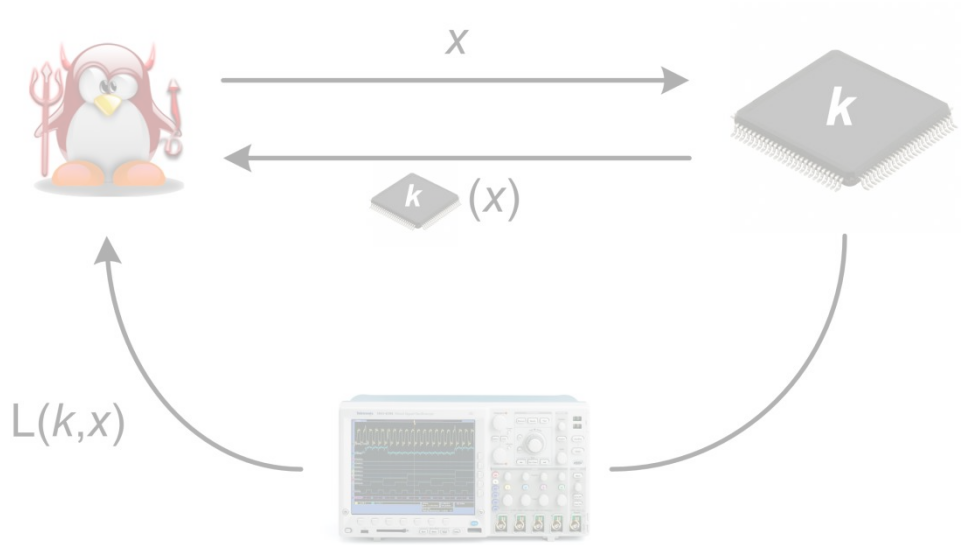
Our setting (Crypto 2013)

(a) Give public I/O access to device & setup



Our setting (Crypto 2013)

(a) Give public I/O access to device & setup



(b) Assume L(*k*,*x*) can be simulated

- Using the same HW as the target
- But without knowing the secret key *k*!

More formally

( , ) has simulatable leakages if ∃ S^{L} such that the bit *b* in the following game is hard to guess

Game *q*-sim(Adv,  , S^{L},*b*) with *k*,*k** uniformly random		
q queries	response if *b*=0	response if *b*=1
Enc(*x*)	 (*x*), S^{L}(*k*,*x*,  (*x*))	 (*x*), S^{L}(*k**,*x*,  (*x*))
1 query	response if *b*=0	response if *b*=1
Gen(*x*)	S^{L}(*z*,*x*,k)	S^{L}(*z*,*x*,k*)

More formally

( , ) has simulatable leakages if ∃ S^{L} such that the bit *b* in the following game is hard to guess

Game *q*-sim(Adv,  , S^{L},*b*) with *k*,*k** uniformly random		
q queries	response if *b*=0	response if *b*=1
Enc(*x*)	 (*x*), S^{L}(*k*,*x*, (*x*))	 (*x*), S^{L}(*k**,*x*, (*x*))
1 query	response if *b*=0	response if *b*=1
Gen(*x*)	S^{L}(*z*,*x*,*k*)	S^{L}(*z*,*x*,*k**)

- With $S^{L}(k,x,$  $(x)) \stackrel{def}{=} L(k,x)$ (*makes our results dependent only on the number of calls to* S^{L})

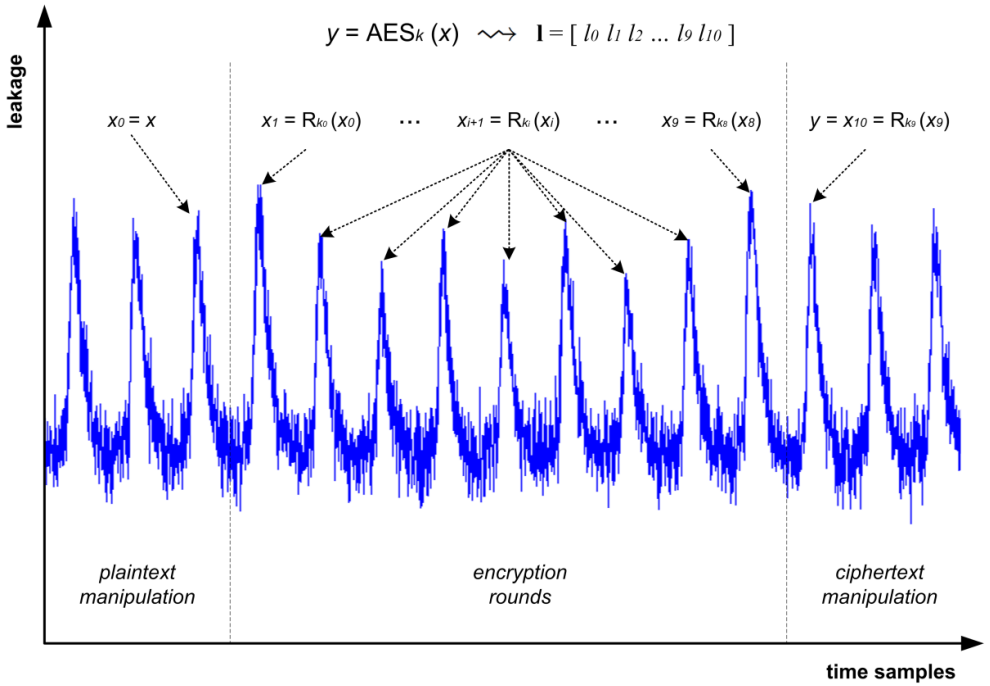
Block cipher leakage simulator

- Let L(k,x) = $l^p(k,x)||l^c(k,$ 

$(x))$
  - l^p corresponds to the first rounds of 

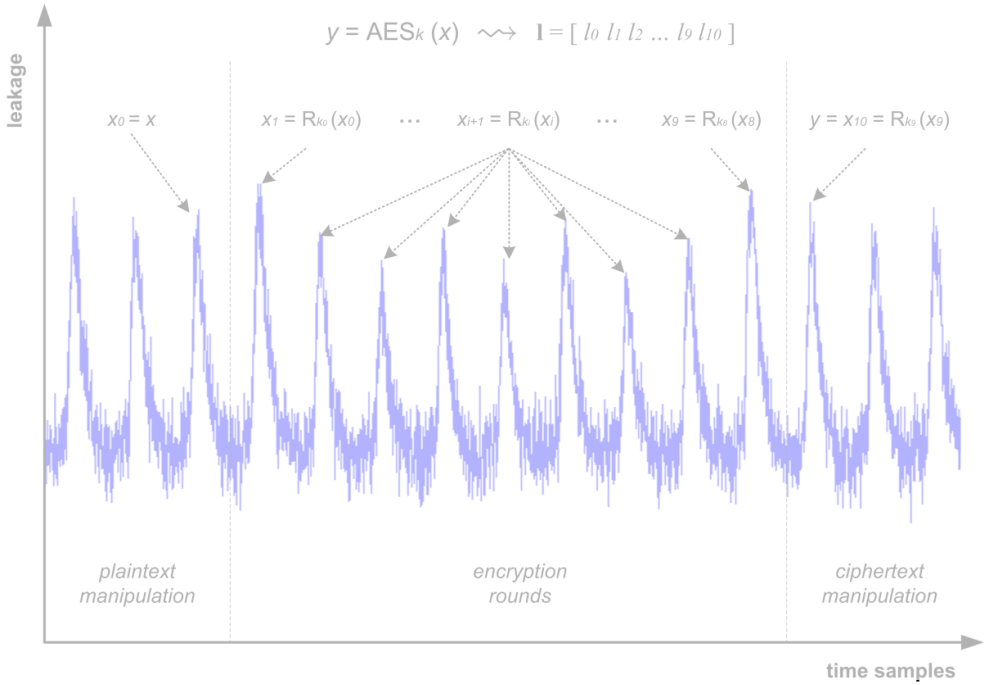
  - l^c corresponds to the last rounds of 

- e.g.,



Block cipher leakage simulator

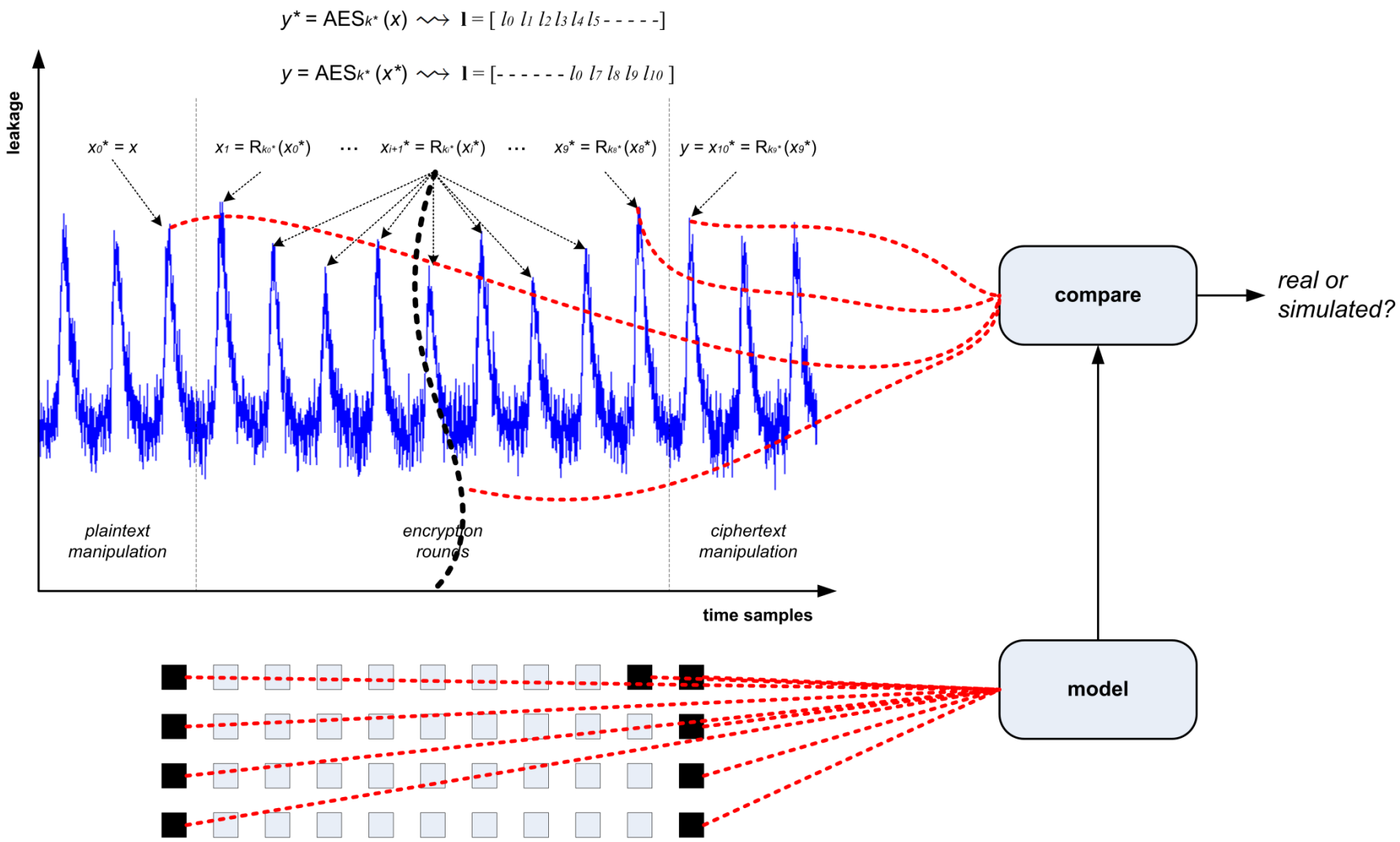
- Let L(k,x) = $l^p(k,x)||l^c(k,$  $(x))$
  - l^p corresponds to the first rounds of 
  - l^c corresponds to the last rounds of 
- e.g.,



⇒ Instantiate S^L(k,x,y) = $l^p(k,x)||\ l^c(k,y)$

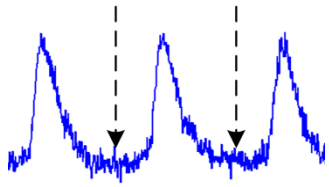
Why would this work?

Simulatable leakages ≈ DPA + I/O's leakages



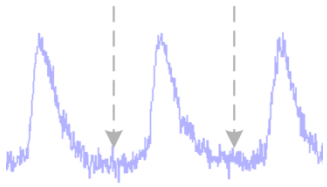
Summarizing

a. Attacks against *q*-sim. exploit the same leakages as DPA if the traces are consistent with the I/O's
   - *this is exactly what the simulator does*
b. Additionally needs concatenation
   - *OK if* ∃ *leakage samples without interest:*



Summarizing

a. Attacks against *q*-sim. exploit the same leakages as DPA if the traces are consistent with the I/O's
   - *this is exactly what the simulator does*
b. Additionally needs concatenation
   - *OK if ∃ leakage samples without interest:*

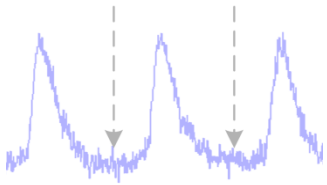


c. *q*-sim. at least easier to guarantee than H^{HILL}

Summarizing

a. Attacks against *q*-sim. exploit the same leakages as DPA if the traces are consistent with the I/O's
   - *this is exactly what the simulator does*

b. Additionally needs concatenation
   - *OK if ∃ leakage samples without interest:*



c. *q*-sim. at least easier to guarantee than H^{HILL}

d. Engineering challenges

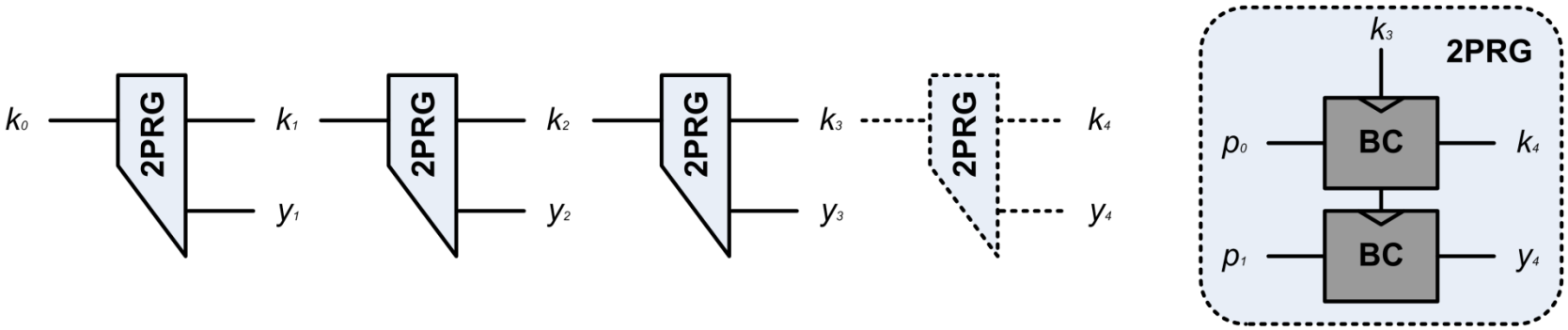
(constructive) Design alternative S^{L} instances

(constructive) Given S^{L}, design  with *q*-sim. leakages

(destructive) Given S^{L} and  , break the *q*-sim. game

First instances falsified by Galea et al. (cfr. end of talk if time allows)

Most natural construction

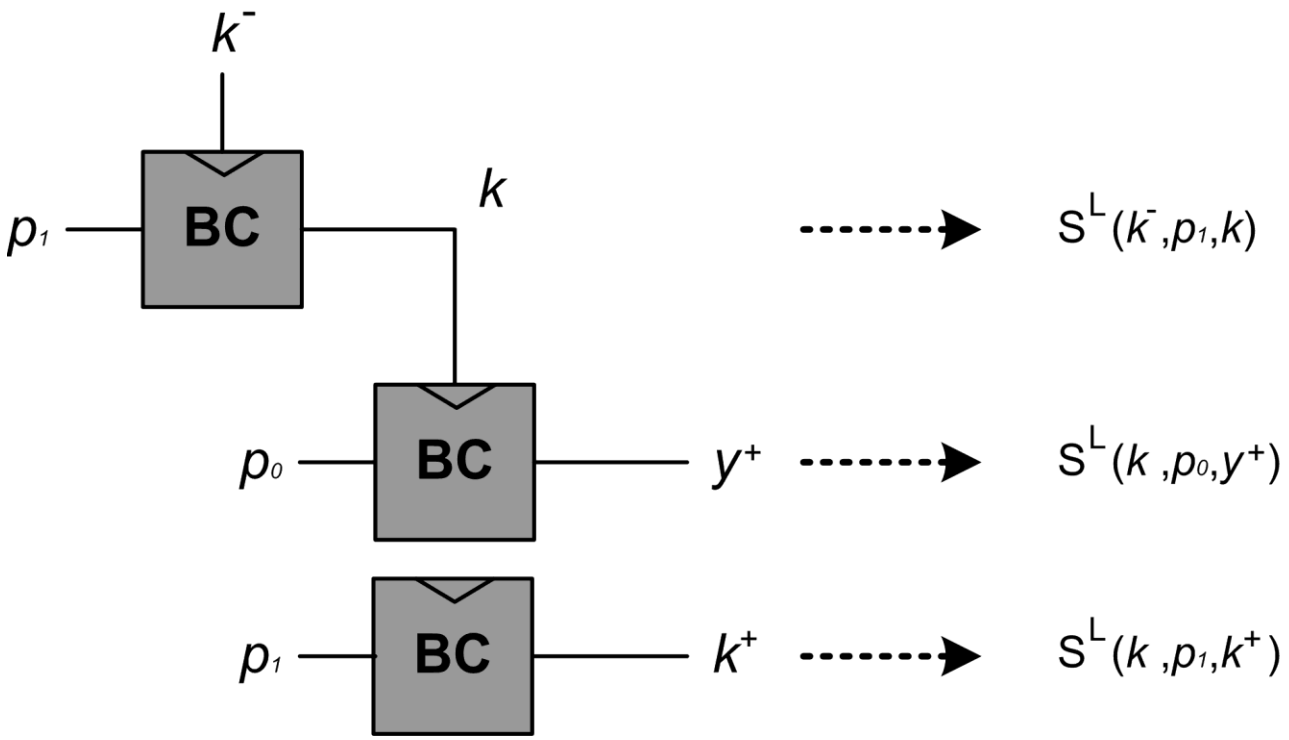


- Goal: remain secure after $\approx 10^6$ runs
- While relying on q-sim. for q=2

- Proving it was surprisingly difficult so far
  - (see slides 9 to 19 of this talk)

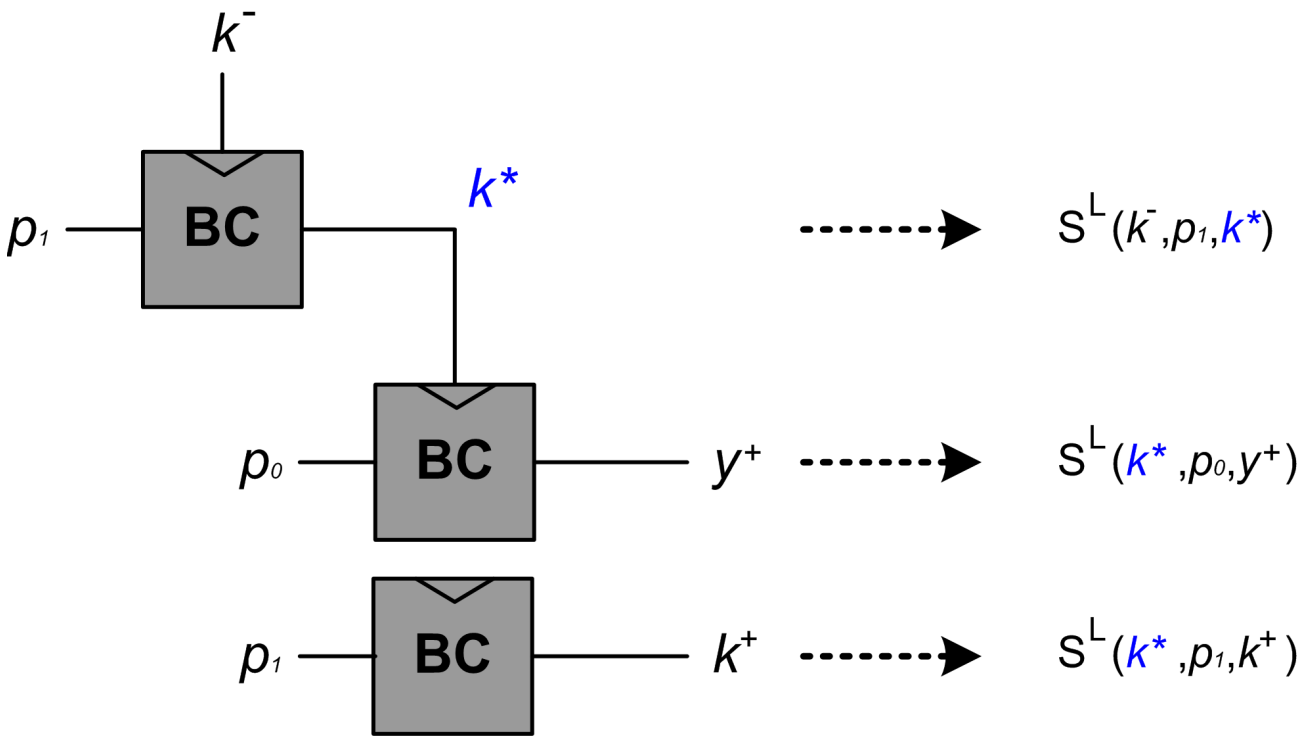
Proof idea #1: replacing lemma

Original view



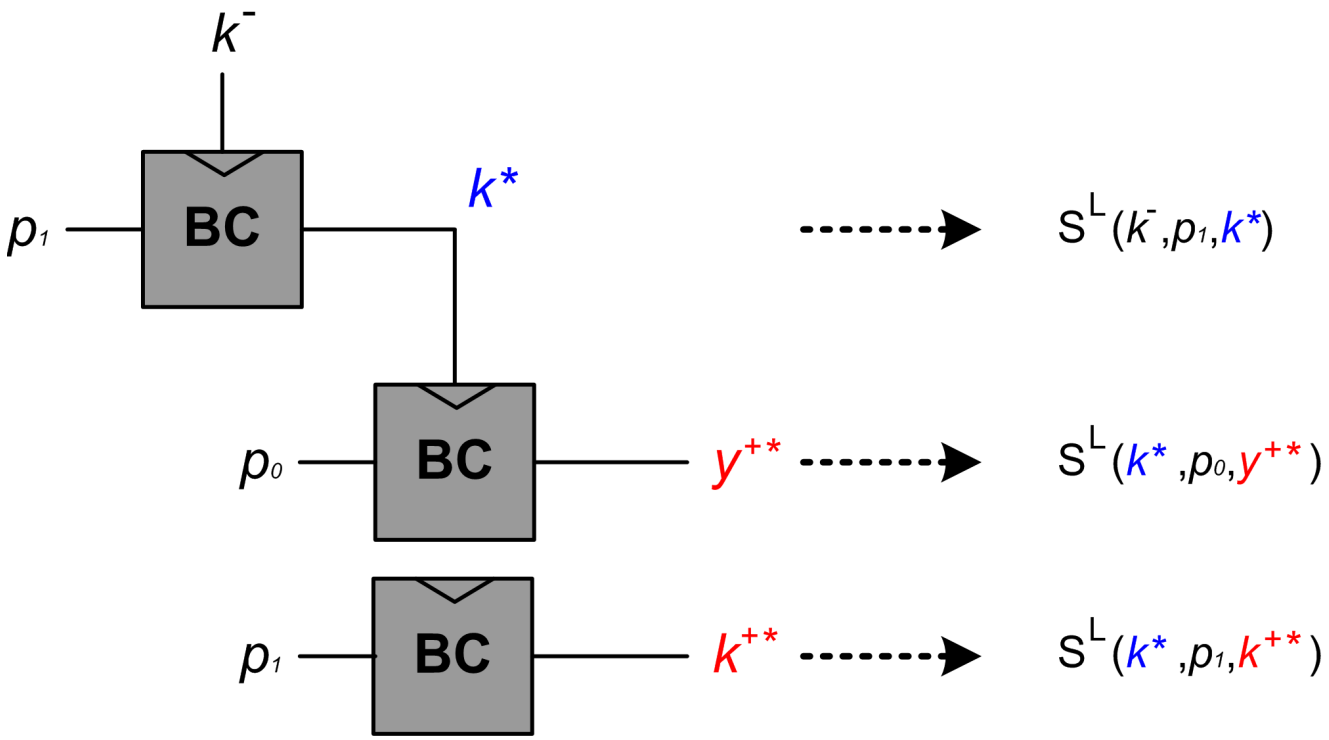
Proof idea #1: replacing lemma

a. Exploit the 2-sim. leakages assumption



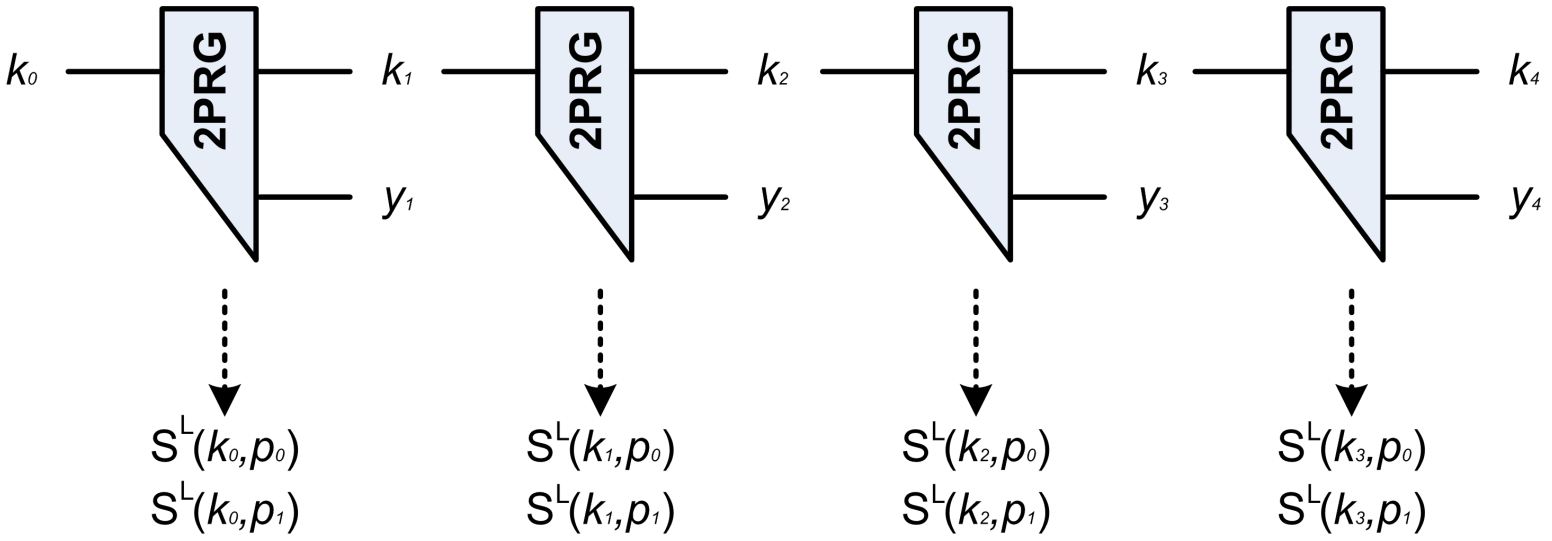
Proof idea #1: replacing lemma

b. Exploit the BC ≈ PRF assumption



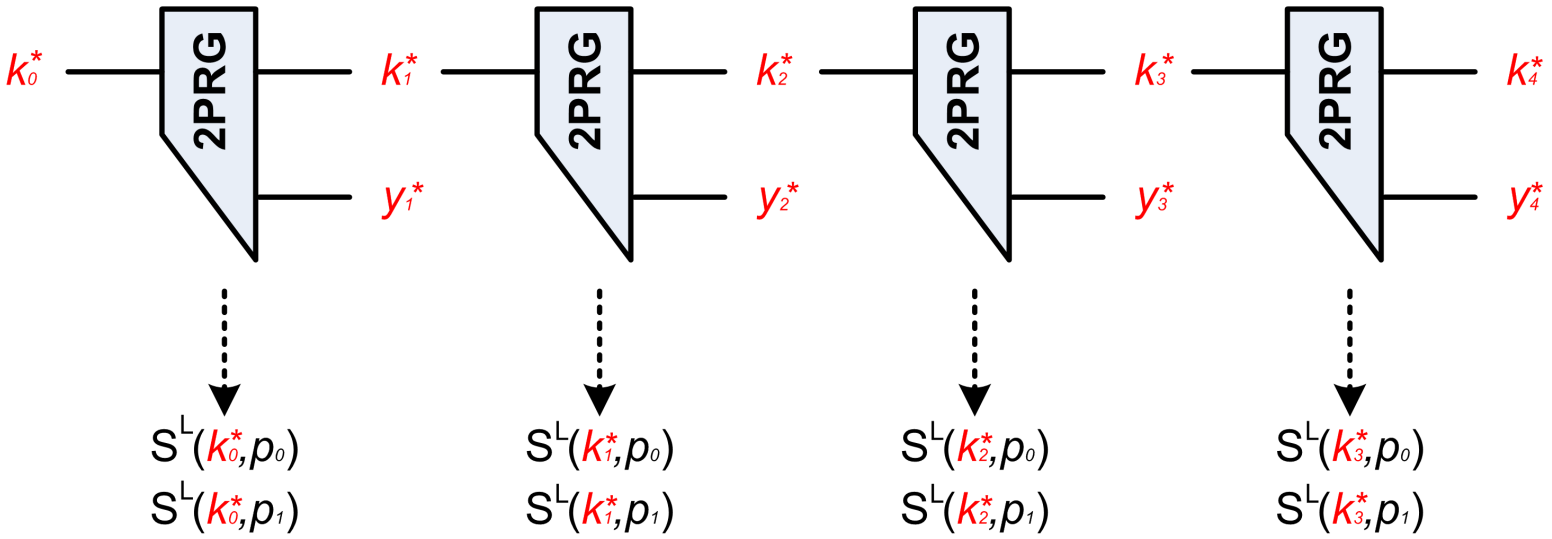
Proof idea #2: extend (hybrid argument)

Original view



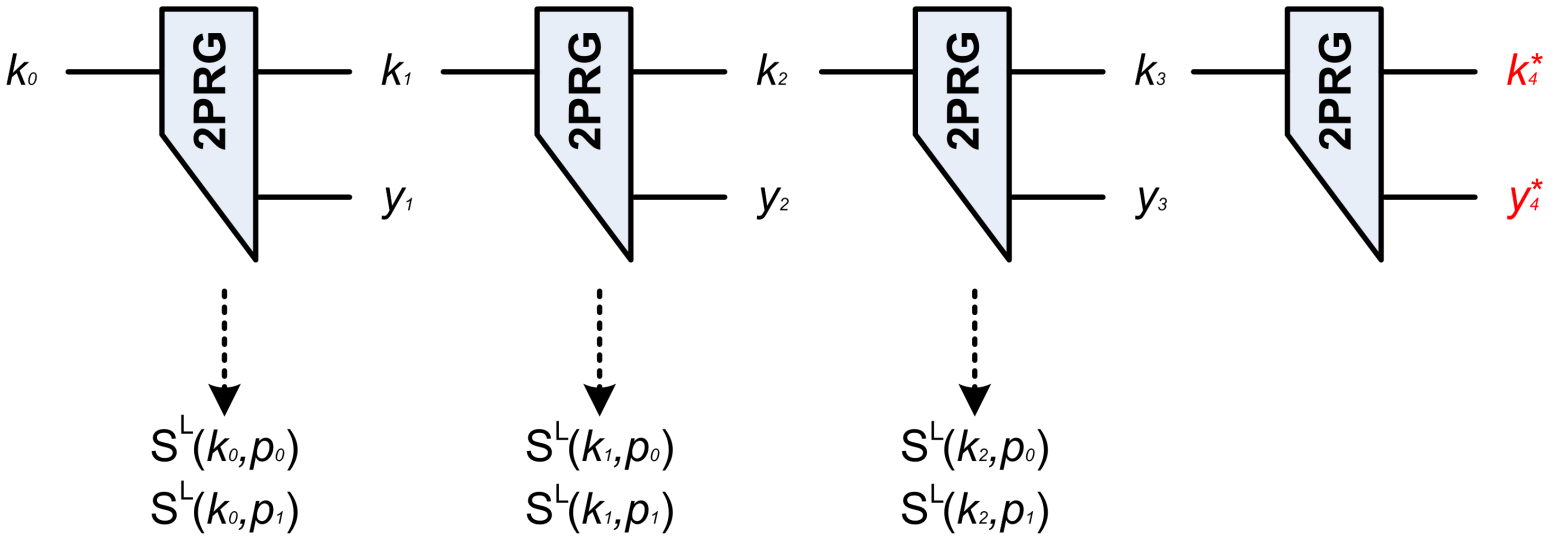
Proof idea #2: extend (hybrid argument)

a. Completely random view (l=4 calls to S^L)



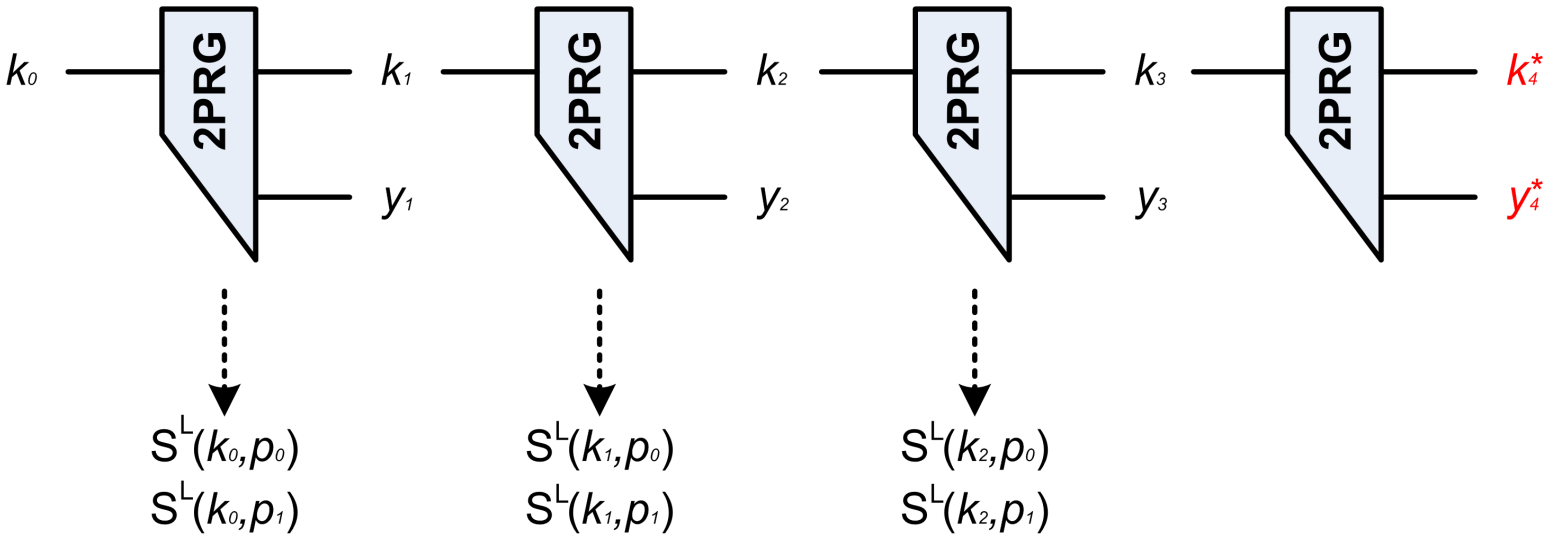
Proof idea #2: extend (hybrid argument)

b. Real view with random y_4 (l=4 calls to S^L)



Proof idea #2: extend (hybrid argument)

b. Real view with random y_4 (l=4 calls to S^L)



Theorem: $y_l \approx U_n$ given $y_1,\ldots,y_{l-1}$,L(k_0),L(k_{l-2}) if BC is a PRF and has 2-simulatable leakages

(*with security degradation proportional to 2l*)

Outline

- Introduction
- Natural PRGs/PRFs and separation result
- PRGs & theoretical challenges
  - Ensuring independence
  - Bounding the leakage
  - The simulatable leakage attempt
- Protocols & practical challenges
  - Authentication & encryption
  - Authenticated encryption
  - Initialization issue
- Summary and conclusions

Pragmatic view

- A call to a stateless primitive is always needed
  - For initialization / randomization
  - For authentication and encryption

Pragmatic view

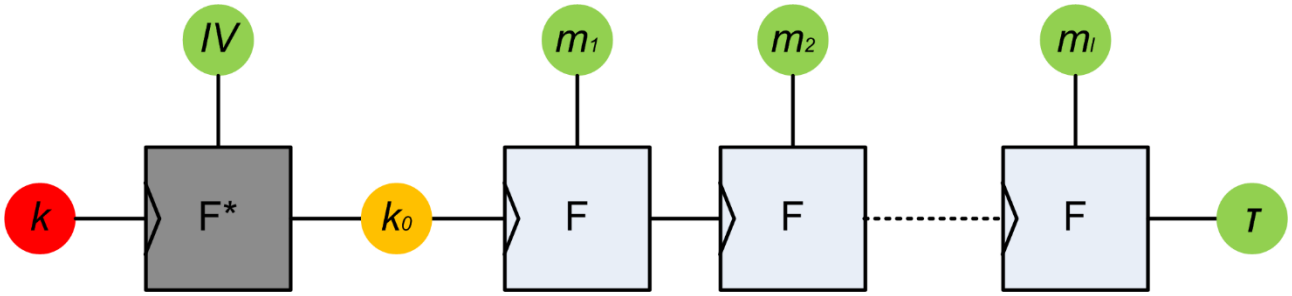
- A call to a stateless primitive is always needed
  - For initialization / randomization
  - For authentication and encryption

- But we can try to encrypt large messages with a single call to this (more expensive) primitive

Pragmatic view

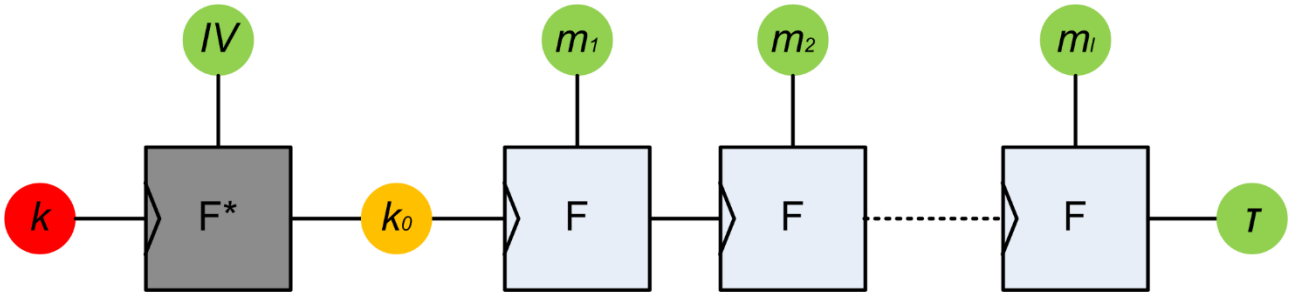
- A call to a stateless primitive is always needed
  - For initialization / randomization
  - For authentication and encryption
- But we can try to encrypt large messages with a single call to this (more expensive) primitive
- And to use leakage-resilience otherwise
  - i.e., use stateful primitives whenever possible
  - And assume one call to a leak-free PRF

Example I: authentication



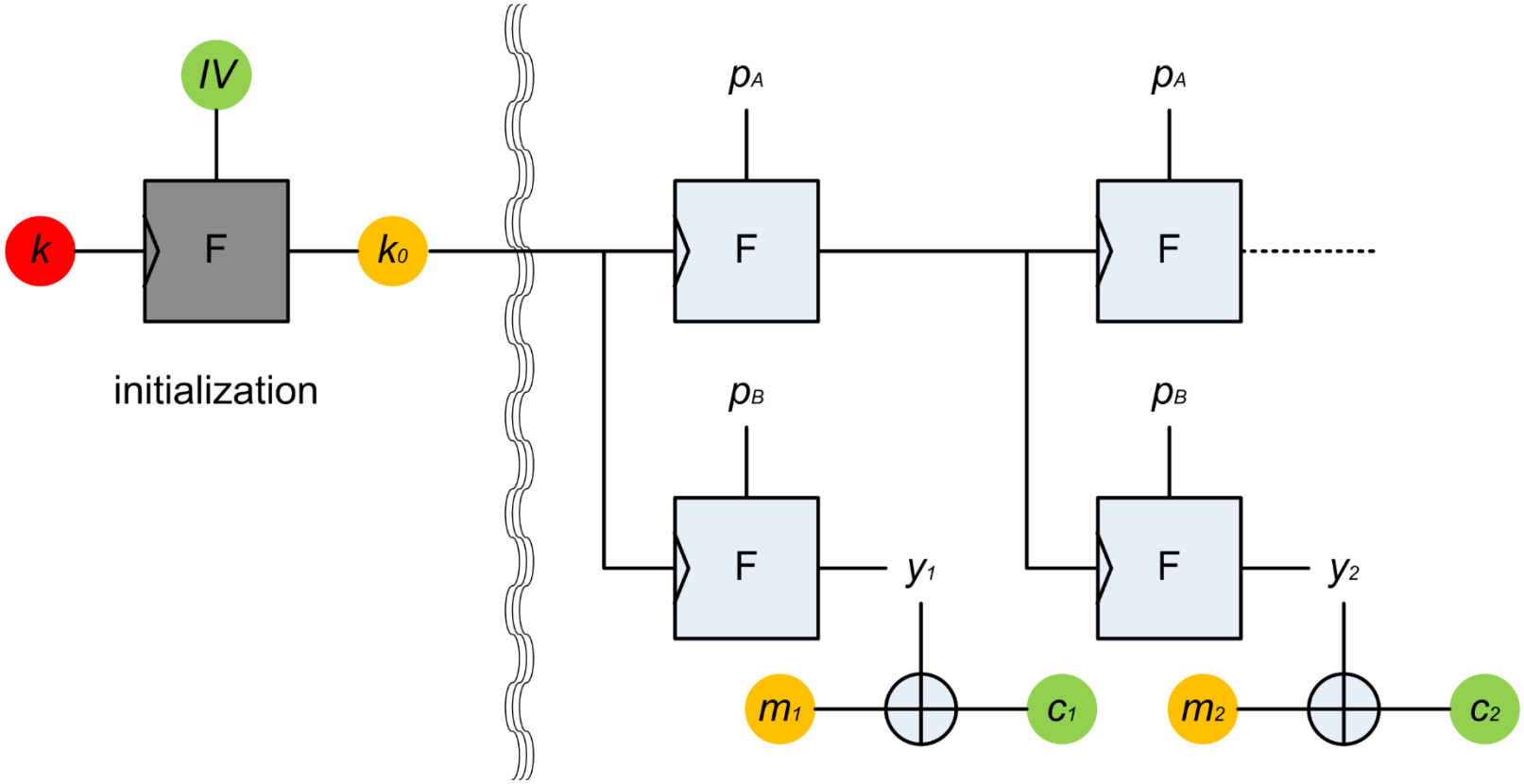
- Green: public value, orange: ephemeral secret, red: long-term secret (protected with leak-free F*)

Example I: authentication



- Green: public value, orange: ephemeral secret, red: long-term secret (protected with leak-free F*)
- τ unforgeable even with leakage (during enc.)
- Security of 1-block $\approx$ security of l-blocks
- & high concrete security levels expected
  - Because it is an unpredictability game

Example II: encryption



- Similar reduction but lower concrete security
  - Because it is an indistinguishability game

Encryption: definition issue

- Conceptual problem: distinguishing is always easy if L is given in the challenge phase

Encryption: definition issue

- Conceptual problem: distinguishing is always easy if L is given in the challenge phase
- Theoretical approach: exclude L in the challenge phase (which is not justified in practice)

Encryption: definition issue

- Conceptual problem: distinguishing is always easy if L is given in the challenge phase
- Theoretical approach: exclude L in the challenge phase (which is not justified in practice)
- Our (pragmatic) approach: admit semantic security is impossible. Leakage will always allow distinguishing plaintexts/ciphertexts!

Encryption: definition issue

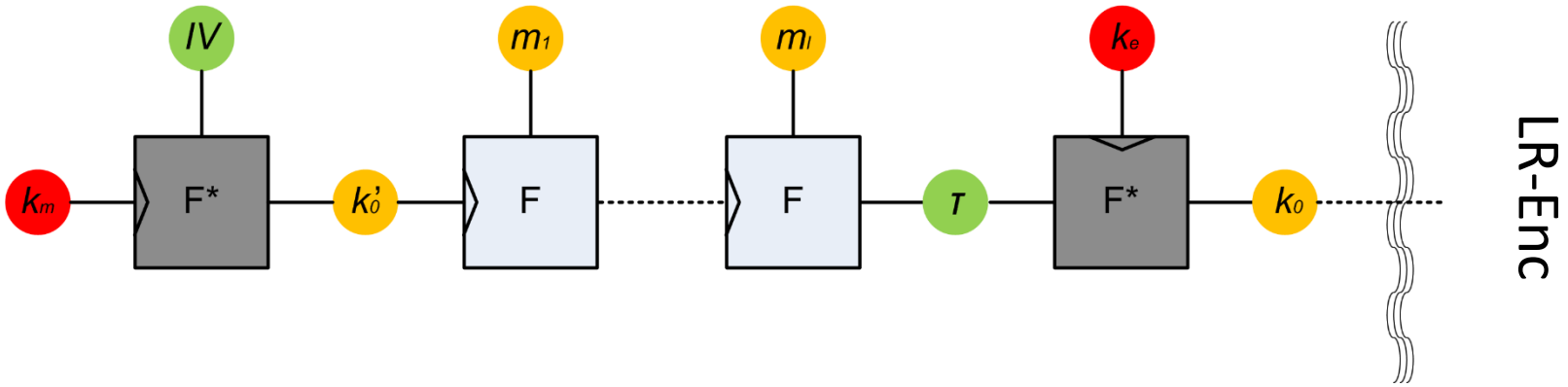
- Conceptual problem: distinguishing is always easy if L is given in the challenge phase
- Theoretical approach: exclude L in the challenge phase (which is not justified in practice)
- Our (pragmatic) approach: admit semantic security is impossible. Leakage will always allow distinguishing plaintexts/ciphertexts!

- CPA security reduction: security of R rounds reduces to security of 1 round (independent of what we can actualy achieve for 1 round)
  - *See our CCS 2015 paper for the details*

Outline

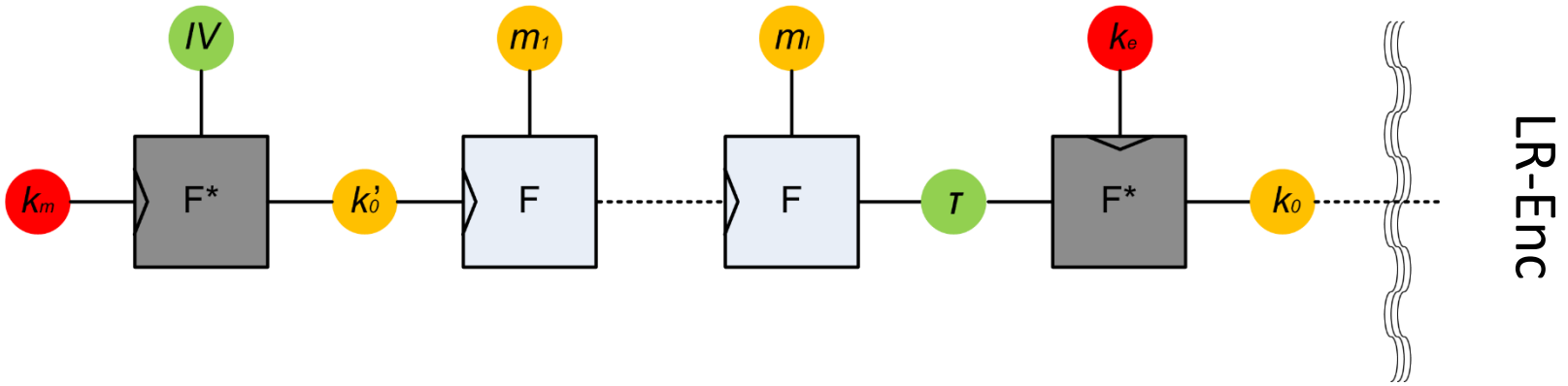
- Introduction
- Natural PRGs/PRFs and separation result
- PRGs & theoretical challenges
  - Ensuring independence
  - Bounding the leakage
  - The simulatable leakage attempt
- Protocols & practical challenges
  - Authentication & encryption
  - Authenticated encryption
  - Initialization issue
- Summary and conclusions

Composing LR-MAC & LR-Enc



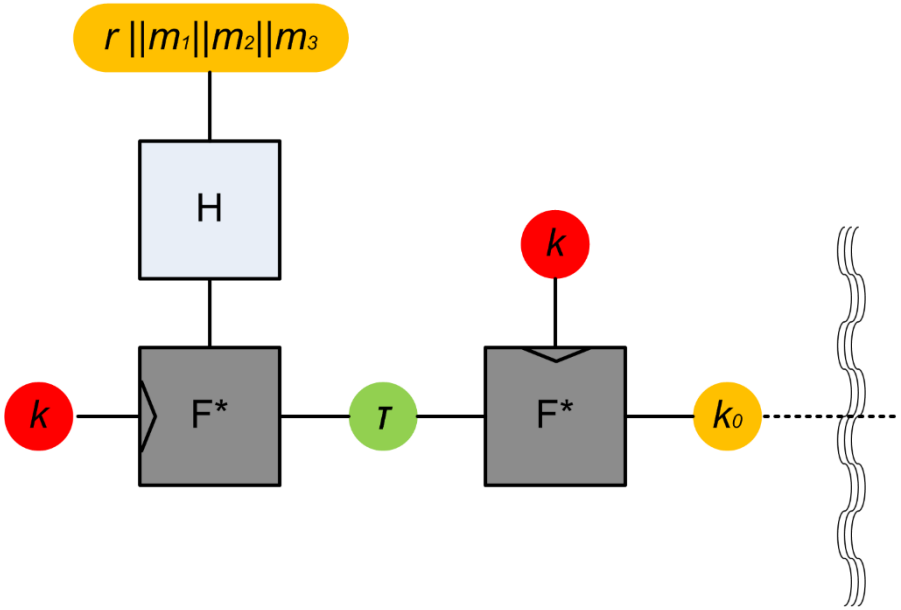
- OK without misuse

Composing LR-MAC & LR-Enc



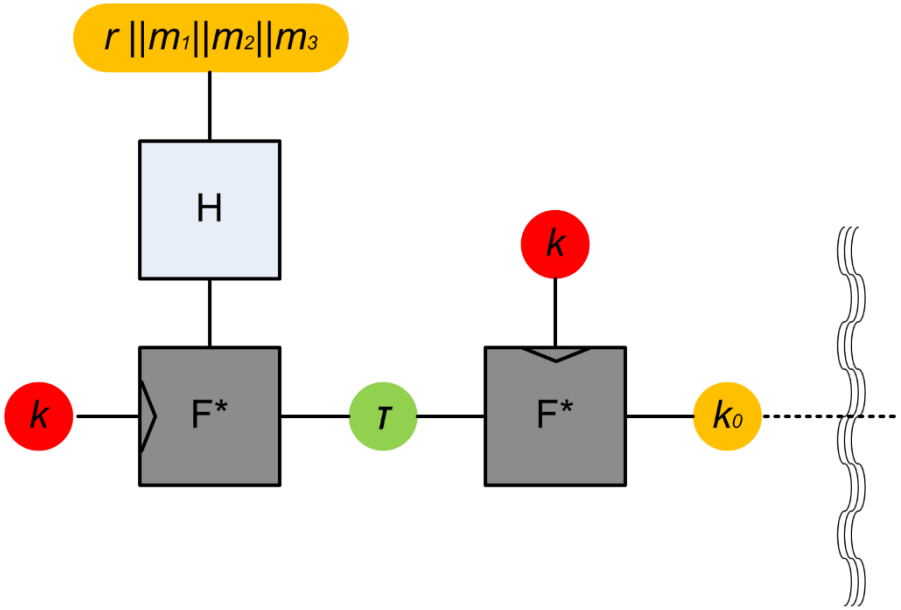
- OK without misuse, *forgery* attacks with misuse:
  - Fix IV and τ, get $k_0{}'$ via DPA, pick m', for $i = 1: l-1$ compute $k_i' = F_{k_{i-1}'}(m_i)$ and finally adjust the last message block $m_l = F^{-1}_{k_{l-1}'}(\tau)$

An improved solution



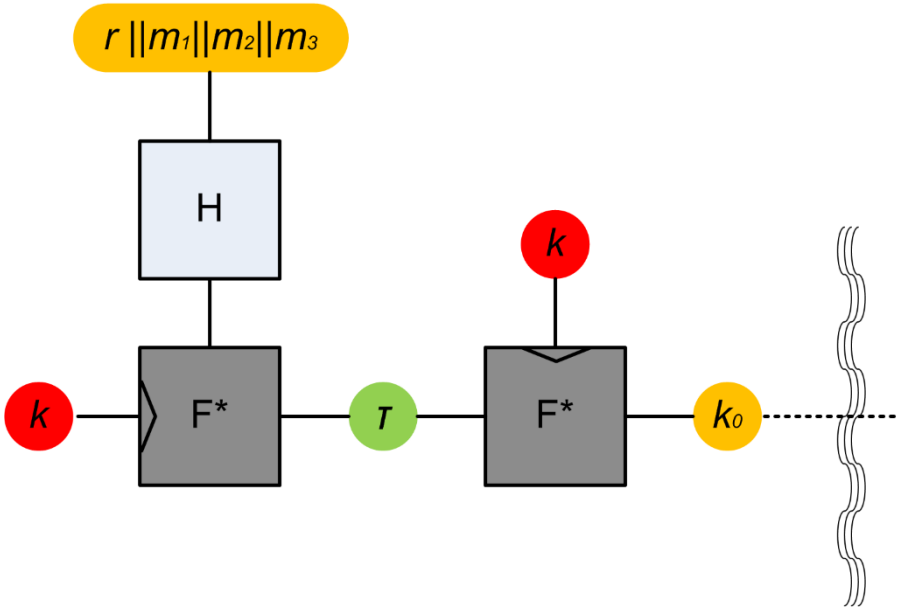
- Digest (i.e., hash), Tag and Encrypt (DTE)
  - Prevents the previous forgery attack
  - Encrypts the randomness (for CPA security)

An improved solution



- Digest (i.e., hash), Tag and Encrypt (DTE)
  - Still not fully misuse-resistant with leakage
    - *Probably impossible in the symmetric setting*

An improved solution



- Digest (i.e., hash), Tag and Encrypt (DTE)
  - Ciphertext Integrity with Misuse & Leakage
    - *Best achievable in the symmetric setting?*

Outline

- Introduction
- Natural PRGs/PRFs and separation result
- PRGs & theoretical challenges
  - Ensuring independence
  - Bounding the leakage
  - The simulatable leakage attempt
- Protocols & practical challenges
  - Authentication & encryption
  - Authenticated encryption
  - Initialization issue
- Summary and conclusions

How to instantiate F*?

- Masking (⇒ bitslice ciphers)

Vincent Grosso, Gaëtan Leurent, François-Xavier Standaert, Kerem Varici: *LS-Designs: Bitslice Encryption for Efficient Masked Software Implementations*. FSE 2014: 18-37. Gilles Barthe, François Dupressoir, Sebastian Faust, Benjamin Gregoire, François-Xavier Standaert, Pierre-Yves Strub, *Parallel Implementations of Masking Schemes and the Bounded Moment Leakage Model*, IACR e-Print 2016/912

How to instantiate F*?

- Masking (⇒ bitslice ciphers)

  Vincent Grosso, Gaëtan Leurent, François-Xavier Standaert, Kerem Varici: *LS-Designs: Bitslice Encryption for Efficient Masked Software Implementations*. FSE 2014: 18-37. Gilles Barthe, François Dupressoir, Sebastian Faust, Benjamin Gregoire, François-Xavier Standaert, Pierre-Yves Strub, *Parallel Implementations of Masking Schemes and the Bounded Moment Leakage Model*, IACR e-Print 2016/912

- PRFs with non-standard assumptions

  Marcel Medwed, François-Xavier Standaert, Antoine Joux: *Towards Super-Exponential Side-Channel Security with Efficient Leakage-Resilient PRFs*. CHES 2012: 193-212. Marcel Medwed, François-Xavier Standaert, Ventzi Nikov, Martin Feldhofer, *Unknown-Input Attacks in the Parallel Setting: Improving the Security and Performances of the CHES 2012 Leakage-Resilient PRF*, ASIACRYPT 2016: 602-623

How to instantiate F*?

- Masking (⇒ bitslice ciphers)

  Vincent Grosso, Gaëtan Leurent, François-Xavier Standaert, Kerem Varici: *LS-Designs: Bitslice Encryption for Efficient Masked Software Implementations*. FSE 2014: 18-37. Gilles Barthe, François Dupressoir, Sebastian Faust, Benjamin Gregoire, François-Xavier Standaert, Pierre-Yves Strub, *Parallel Implementations of Masking Schemes and the Bounded Moment Leakage Model*, IACR e-Print 2016/912

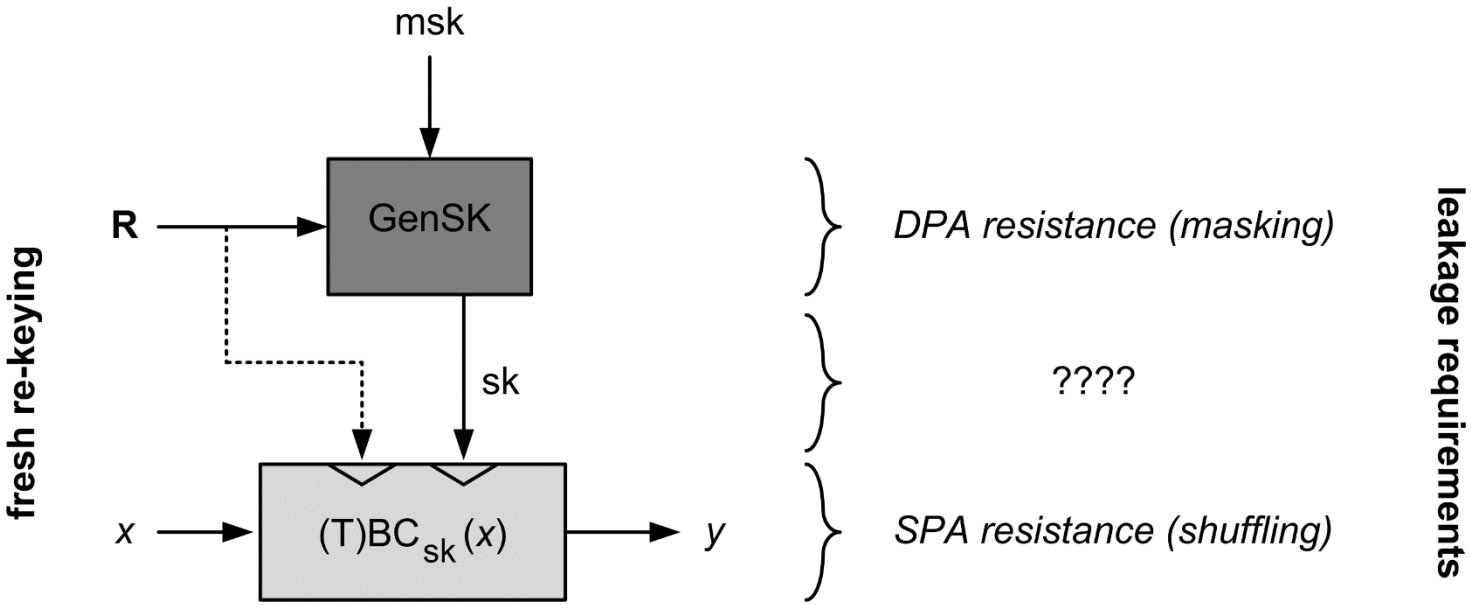
- PRFs with non-standard assumptions

  Marcel Medwed, François-Xavier Standaert, Antoine Joux: *Towards Super-Exponential Side-Channel Security with Efficient Leakage-Resilient PRFs*. CHES 2012: 193-212. Marcel Medwed, François-Xavier Standaert, Ventzi Nikov, Martin Feldhofer, *Unknown-Input Attacks in the Parallel Setting: Improving the Security and Performances of the CHES 2012 Leakage-Resilient PRF*, ASIACRYPT 2016: 602-623

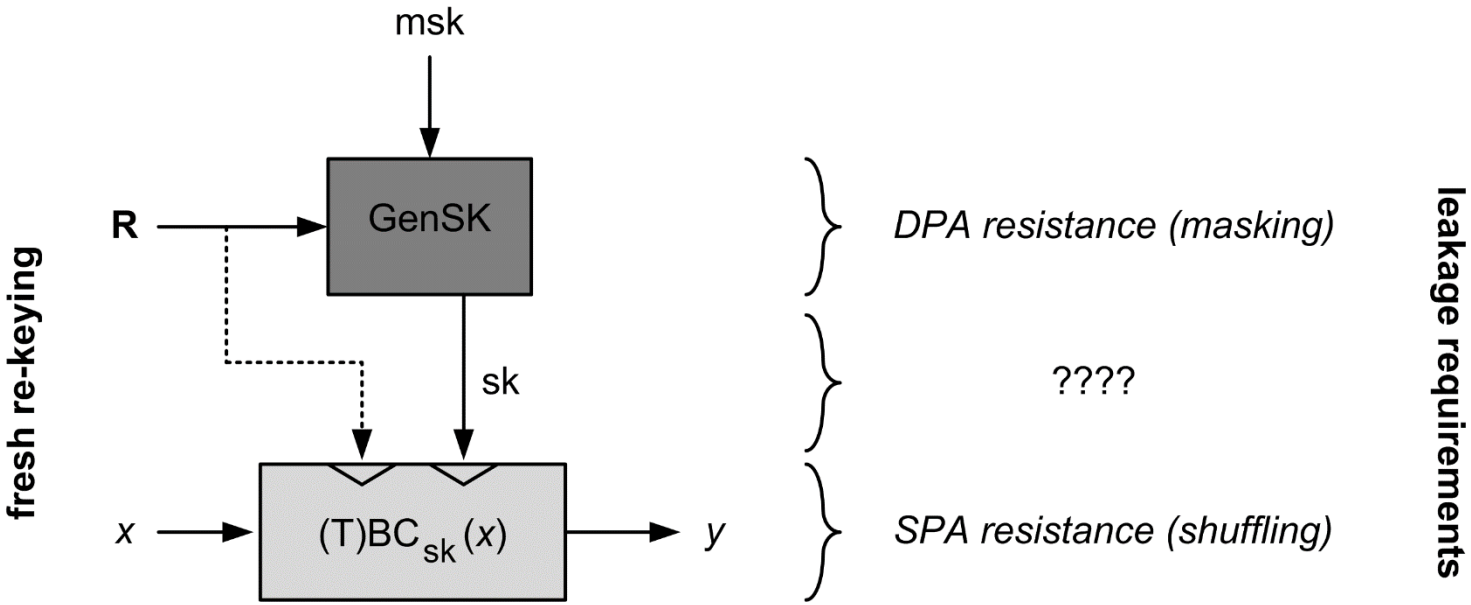
- Key homomorphism & fresh re-keying

  Christoph Dobraunig, François Koeune, Stefan Mangard, Florian Mendel, François-Xavier Standaert: *Towards Fresh and Hybrid Re-Keying Schemes with Beyond Birthday Security*. CARDIS 2015: 225-241

A recent proposal (Crypto 2016)

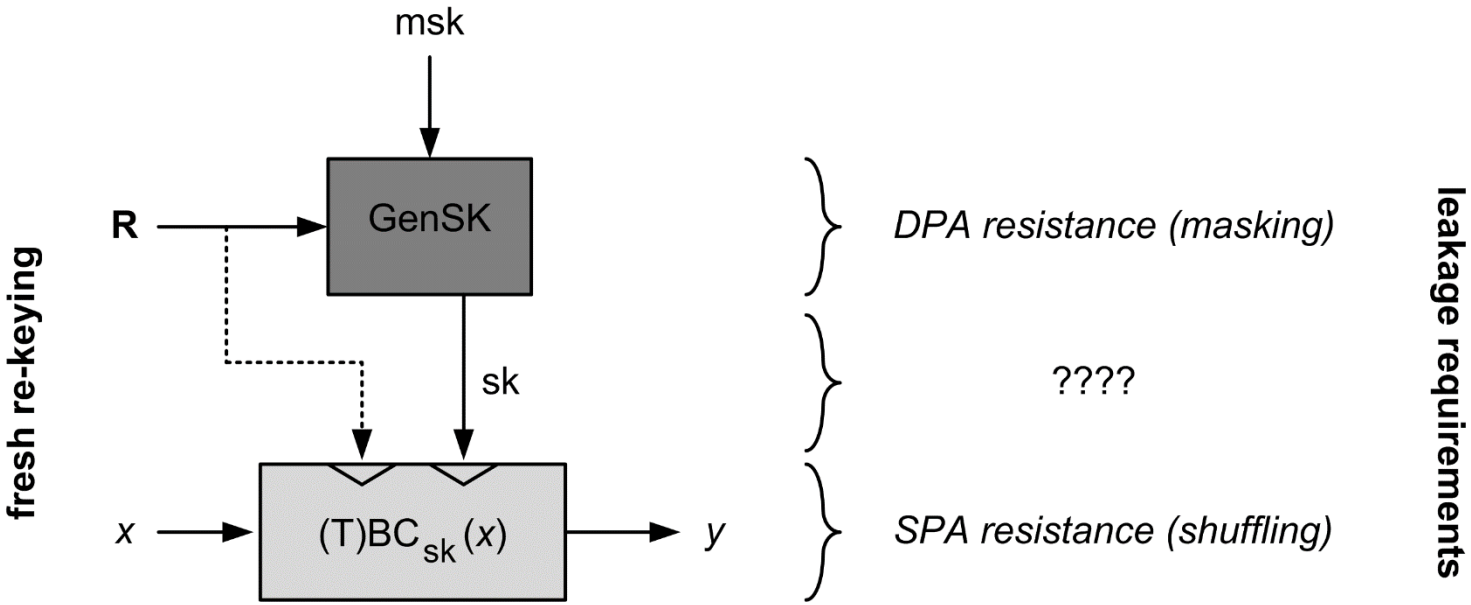


A recent proposal (Crypto 2016)



- Cryptographically strong re-keying function
  - $\mathrm{sk} =< \mathbf{R}, \mathrm{msk} >= \sum(< \mathbf{R}, \mathrm{msk}_i >)$

A recent proposal (Crypto 2016)



- Cryptographically strong re-keying function
  - $\mathrm{sk} =< \mathbf{R}, \mathrm{msk} >= \sum(< \mathbf{R}, \mathrm{msk}_i >)$
- Security based on hard lattice problems
- Simple & efficient: all computations in Z_{2^m}

Outline

- Introduction
- Natural PRGs/PRFs and separation result
- PRGs & theoretical challenges
  - Ensuring independence
  - Bounding the leakage
  - The simulatable leakage attempt
- Protocols & practical challenges
  - Authentication & encryption
  - Authenticated encryption
  - Initialization issue
- Summary and conclusions

Conclusions

- Concretely, leakage-resilience is effective and efficient for stateful primitives such as PRGs

Conclusions

- Concretely, leakage-resilience is effective and efficient for stateful primitives such as PRGs
- Protection of stateless primitives such as PRFs and PRPs is much more challenging

Conclusions

- Concretely, leakage-resilience is effective and efficient for stateful primitives such as PRGs
- Protection of stateless primitives such as PRFs and PRPs is much more challenging

- Pragmatic solution: minimize the number of (leak-free) stateless primitives in leakage-resilient encryption and authentication

Open problems

- Sound (empirically falsifiable) assumptions
  - e.g. new instances of leakage simulators

Open problems

- Sound (empirically falsifiable) assumptions
  - e.g. new instances of leakage simulators

- Can we better formalize CPA security with L?

Open problems

- Sound (empirically falsifiable) assumptions
  - e.g. new instances of leakage simulators
- Can we better formalize CPA security with L?
- Leakage-resilient decryption & tag verification
  - Excluded from the analysis so far
  - Mostly because of IV control by the Adv.

And more generally

- Tools, formal methods, automation, …

And more generally

- Tools, formal methods, automation, …
- Design against physical defaults
  - Independence issues (glitches, transitions, …)

And more generally

- Tools, formal methods, automation, …
- Design against physical defaults
  - Independence issues (glitches, transitions, …)
- Advanced masking schemes
  - e.g., inner product based (beyond probing security)

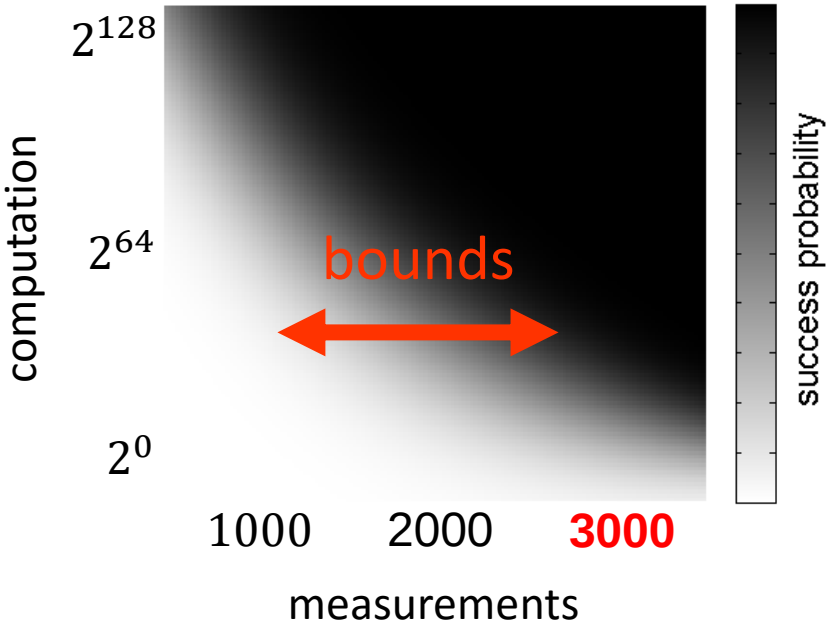
And more generally

- Tools, formal methods, automation, …
- Design against physical defaults
  - Independence issues (glitches, transitions, …)
- Advanced masking schemes
  - e.g., inner product based (beyond probing security)
- Security without obscurity
  - Needed for high security design/evaluation

Security evaluation tools

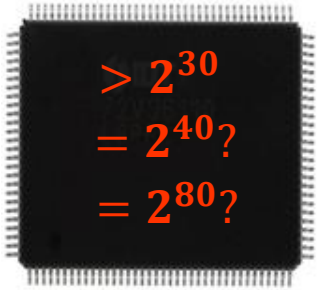
standard practice

attack-based evaluations

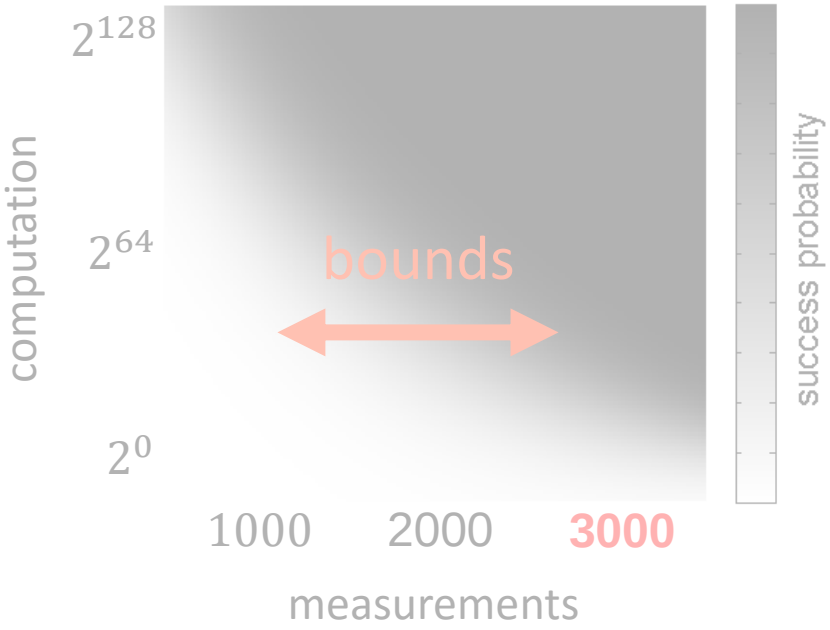


Security evaluation tools

standard practice



attack-based evaluations

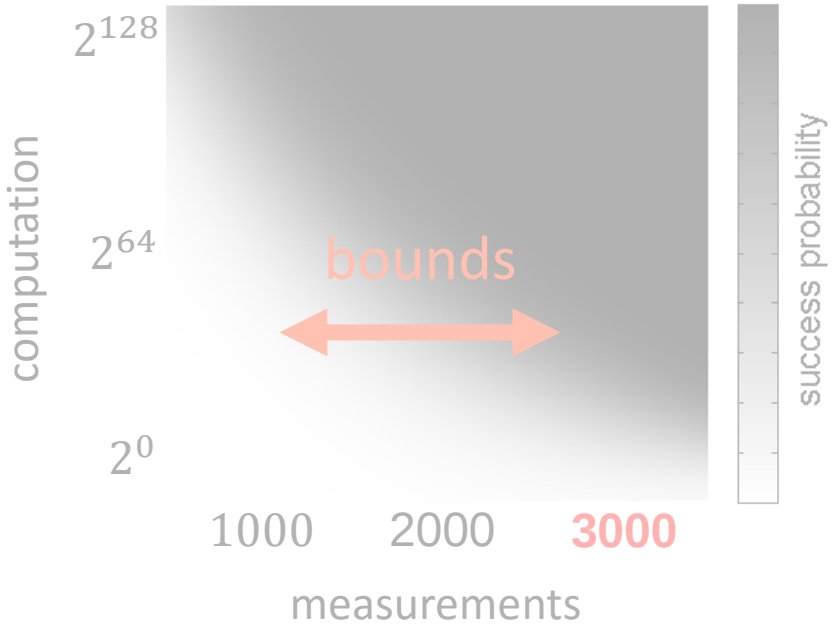


standard practice

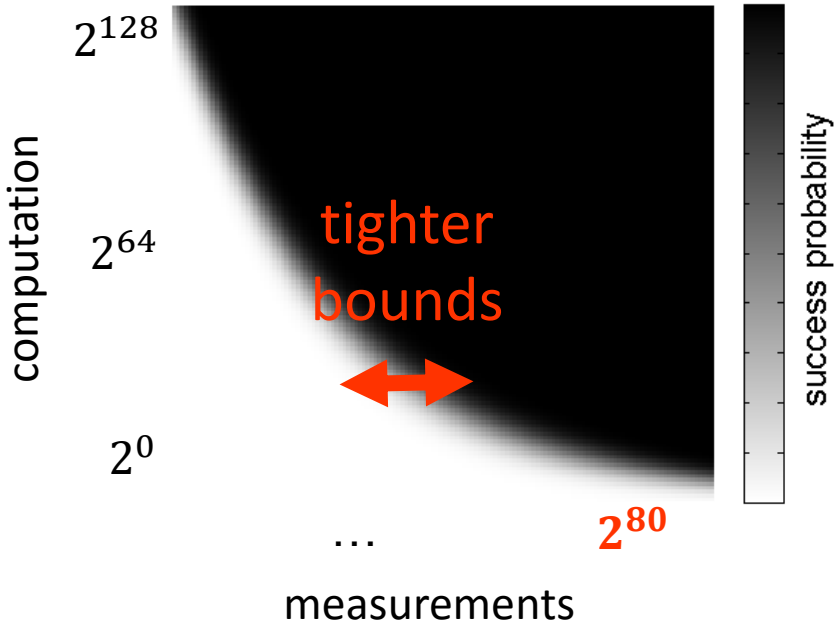
transparency

helps evaluations

attack-based evaluations



proof-based evaluations



THANKS

http://perso.uclouvain.be/fstandae/

http://perso.uclouvain.be/fstandae/PUBLIS/184.pdf

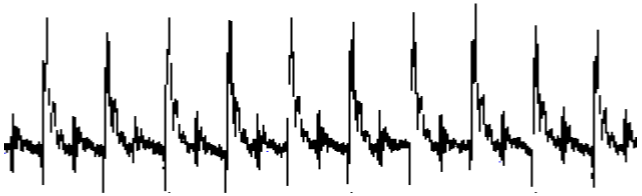
Additional slides

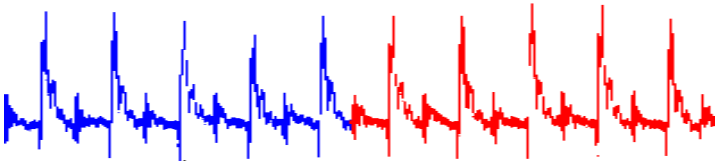
(leakage simulators & the Bristol distinguisher)

Background

- Split & Concatenate Simulator (CRYPTO 2013)

$$\mathrm{L}(x, k, y) \approx \mathrm{L}(x, \tilde{k}, y^*) \| \mathrm{L}(x^*, \tilde{k}, y)$$





Background

- Split & Concatenate Simulator (CRYPTO 2013)

$$\mathrm{L}(x, k, y) \approx \mathrm{L}(x, \tilde{k}, y^*) || \mathrm{L}(x^*, \tilde{k}, y)$$

- Longo Galea et al (ASIACRYPT 2014): ∃ correlation between samples *within* real traces (e.g. $\rho > 0.5$)
  … that are significantly reduced in simulated ones
  ⇒ Allows distinguishing!

Background

- Split & Concatenate Simulator (CRYPTO 2013)

$$\mathrm{L}(x, k, y) \approx \mathrm{L}(x, \tilde{k}, y^*) || \mathrm{L}(x^*, \tilde{k}, y)$$

- Longo Galea et al (ASIACRYPT 2014): ∃ correlation between samples *within* real traces (e.g. $\rho > 0.5$) … that are significantly reduced in simulated ones

⇒ Allows distinguishing!

- Proposed solution: very noisy implementations, *but it scales badly*: noise arbitrarily reduced with averaging

Background

- Split & Concatenate Simulator (CRYPTO 2013)

$$\mathrm{L}(x, k, y) \approx \mathrm{L}(x, \tilde{k}, y^*) || \mathrm{L}(x^*, \tilde{k}, y)$$

- Longo Galea et al (ASIACRYPT 2014): ∃ correlation between samples *within* real traces (e.g. $\rho > 0.5$) … that are significantly reduced in simulated ones
  ⇒ Allows distinguishing!
- Proposed solution: very noisy implementations, *but it scales badly*: noise arbitrarily reduced with averaging

Can we do better?

Origin of the intra-trace correlation

- Algorithmic? Unlikely: $\rho\big(x, \mathrm{Sbox}(x)\big) \ll 0.5$

Origin of the intra-trace correlation

- Algorithmic? Unlikely: $\rho\left(x, \mathrm{Sbox}(x)\right) \ll 0.5$
- Physical then ⇒ let's use a simple physical model

Origin of the intra-trace correlation

- Algorithmic? Unlikely: $\rho\big(x, \mathrm{Sbox}(x)\big) \ll 0.5$
- Physical then ⇒ let's use a simple physical model

$$\mathrm{L}(x,k,y) = \underbrace{\delta(x,k,y)}_{\text{signal}} + \underbrace{N}_{\text{noise}}$$

Origin of the intra-trace correlation

- Algorithmic? Unlikely: $\rho\big(x, \mathrm{Sbox}(x)\big) \ll 0.5$
- Physical then ⇒ let's use a simple physical model

$$\mathrm{L}(x,k,y) = \underbrace{\delta(x,k,y)}_{\text{signal}} + \underbrace{N}_{\text{noise}}$$

⇒ Does the correlation come from signal or noise?

Origin of the intra-trace correlation

- Algorithmic? Unlikely: $\rho\big(x, \mathrm{Sbox}(x)\big) \ll 0.5$
- Physical then ⇒ let's use a simple physical model

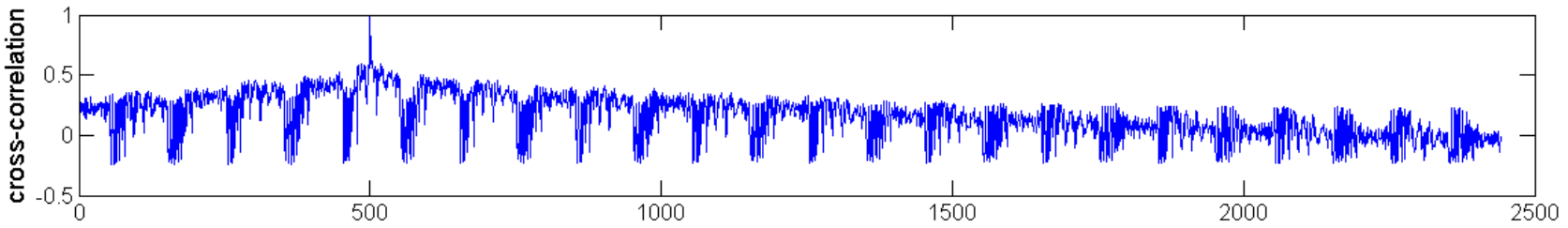
$$\mathrm{L}(x, k, y) = \underbrace{\delta(x, k, y)}_{\text{signal}} + \underbrace{N}_{\text{noise}}$$

⇒ Does the correlation come from signal or noise?

- In particular for *large parallel implementations* (since we know 8-bit AES implementations can be broken in one trace anyway – see SASCA paper)

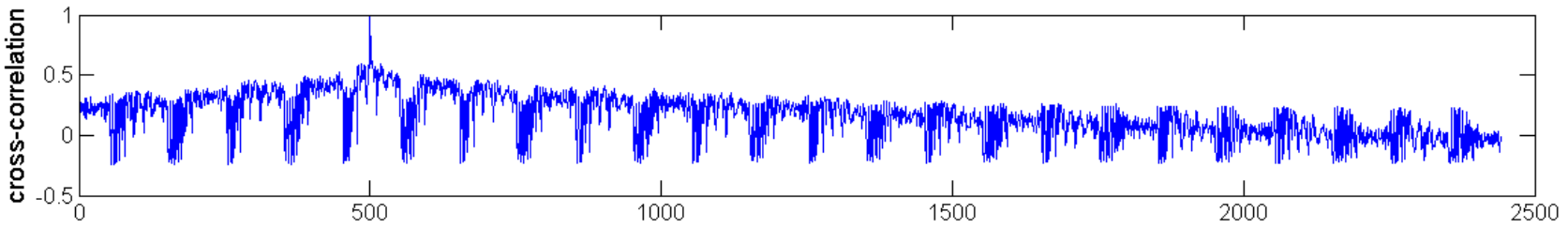
Repeating experiments with a 65nm ASIC

- Intra-trace correlation (real traces, sample 500)

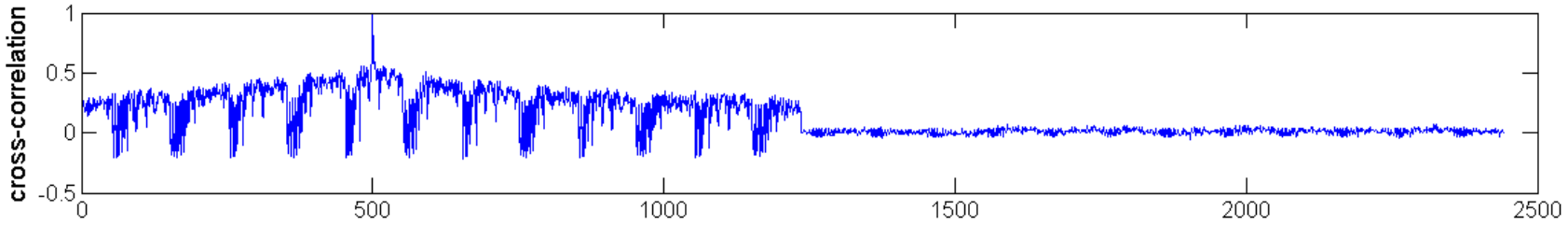


Repeating experiments with a 65nm ASIC

- Intra-trace correlation (real traces, sample 500)

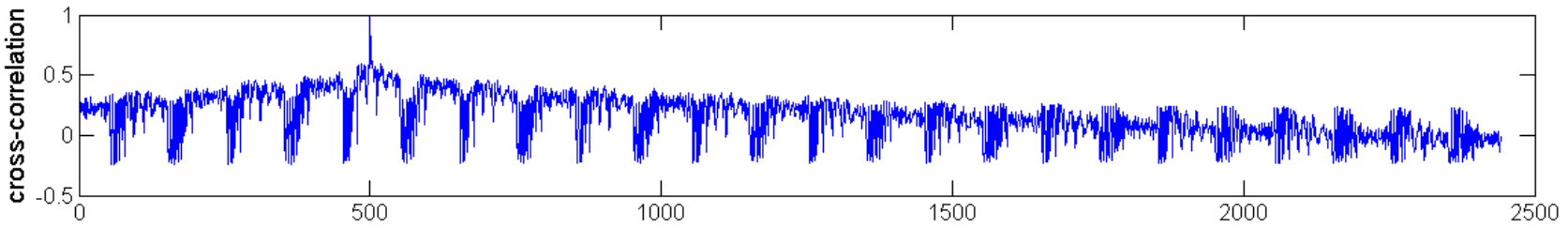


- Same, with simulated traces $\mathrm{L}(x, \tilde{k}, y^*)||\mathrm{L}(x^*, \tilde{k}, y)$

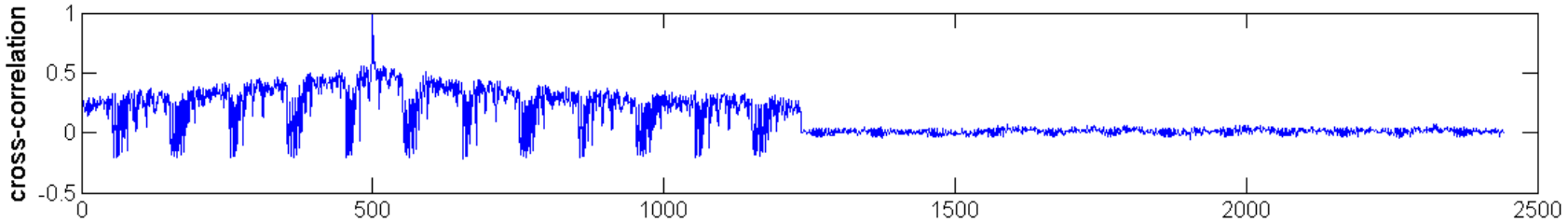


Repeating experiments with a 65nm ASIC

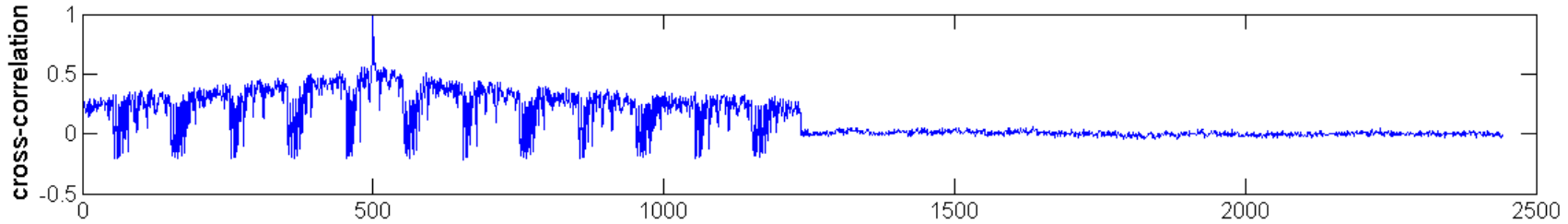
- Intra-trace correlation (real traces, sample 500)



- Same, with simulated traces $\mathrm{L}(x, \tilde{k}, y^*)||\mathrm{L}(x^*, \tilde{k}, y)$

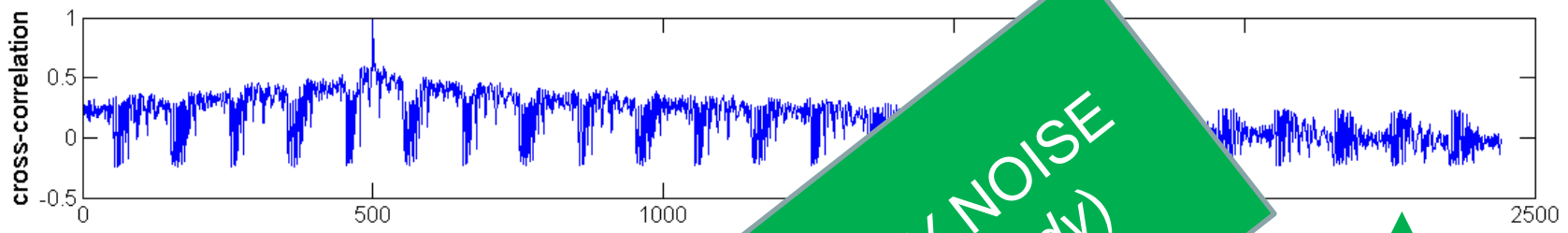


- & fake simulated traces $\delta(x, k, y) + N_1||\delta(x, k, y) + N_2$

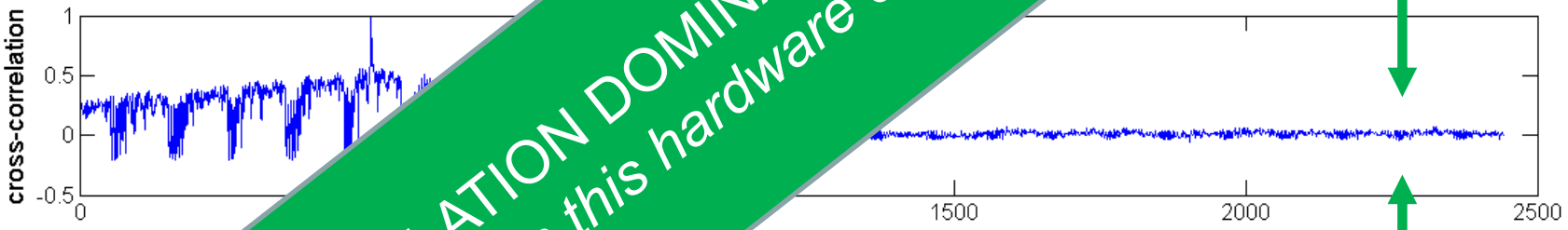


Repeating experiments with a 65nm ASIC

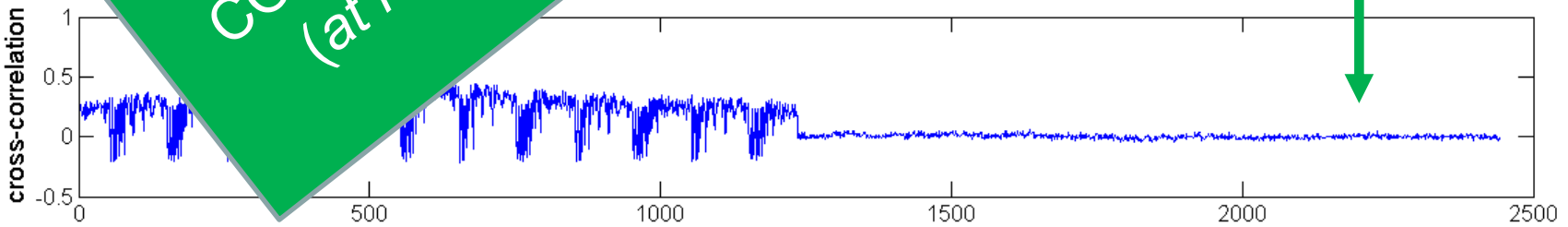
- Intra-trace correlation (real traces, sample 500)



- Same, with simulat… $y^*)||\mathrm{L}(x^*, \tilde{k}, y)$



- & fak… …es $\delta(x,k,y) + N_1||\delta(x,k,y) + N_2$



CORRELATION DOMINATED BY NOISE
(at least in this hardware case study)

A first improvement

- Sliding simulator

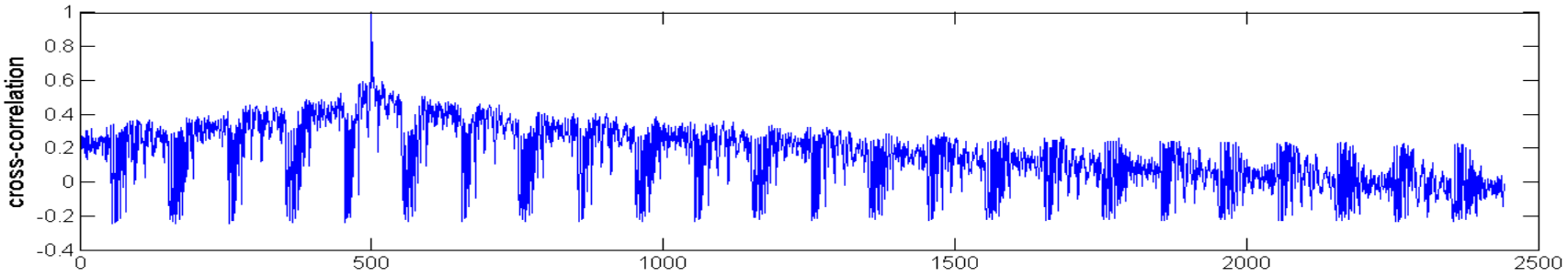
$$L(x, \tilde{k}, y^*) \cdot$$  $$+ \; L(x^*, \tilde{k}, y) \cdot$$ 

A first improvement

- Sliding simulator

$$L(x, \tilde{k}, y^*) \cdot \quad + \quad L(x^*, \tilde{k}, y) \cdot$$

- Real traces

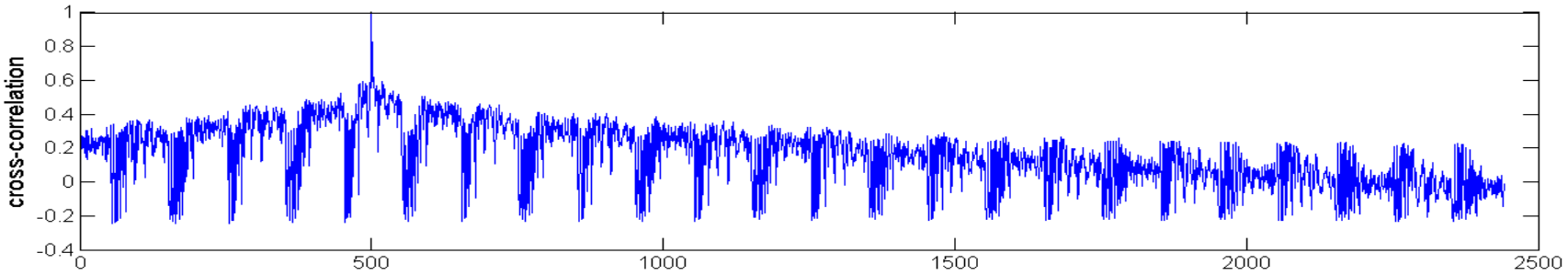


A first improvement

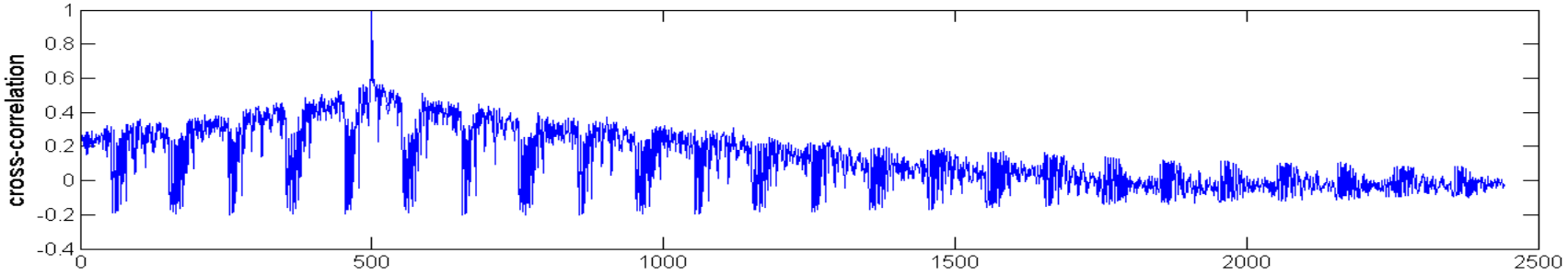
- Sliding simulator

$$L(x, \tilde{k}, y^*) \cdot \quad + \quad L(x^*, \tilde{k}, y) \cdot$$

- Real traces



- Simulated traces

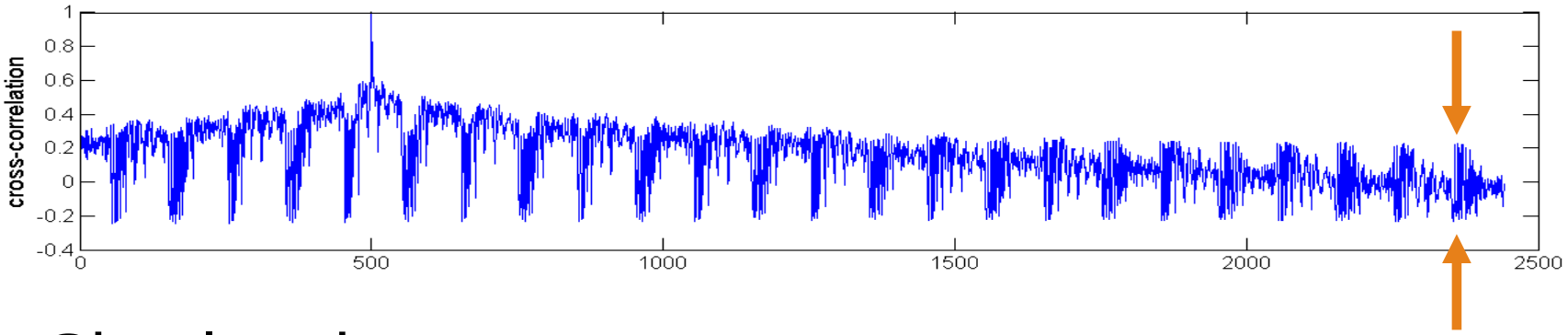


A first improvement

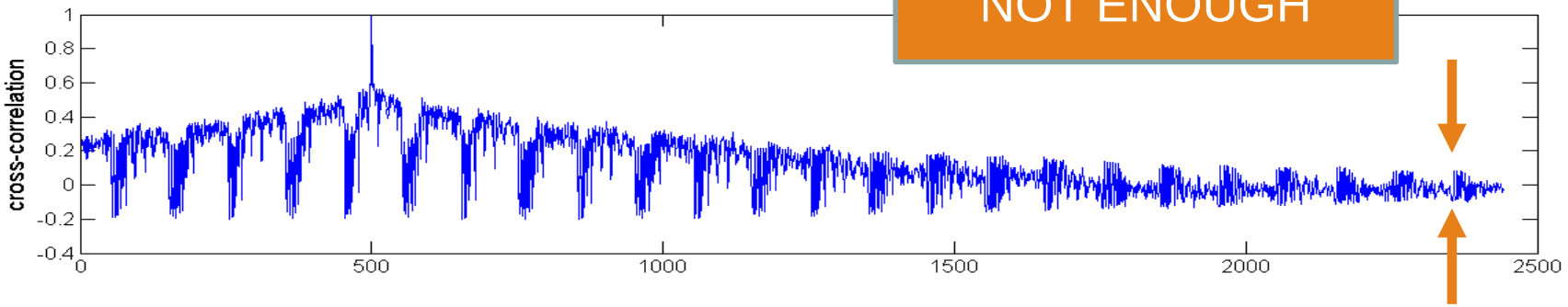
- Sliding simulator

$$L(x, \tilde{k}, y^*) \cdot \quad + \quad L(x^*, \tilde{k}, y) \cdot$$

- Real traces



- Simulated traces



Another idea: separate signal and noise

- Sliding signal + noise simulator

$\hat{\delta}(x, \tilde{k}, y^*) \cdot$  + $\hat{\delta}(x^*, \tilde{k}, y) \cdot$  + N

Another idea: separate signal and noise

- Sliding signal + noise simulator

$\hat{\delta}(x, \tilde{k}, y^*) \cdot$  + $\hat{\delta}(x^*, \tilde{k}, y) \cdot$  + N

avg. trace

avg. trace – single trace

Another idea: separate signal and noise

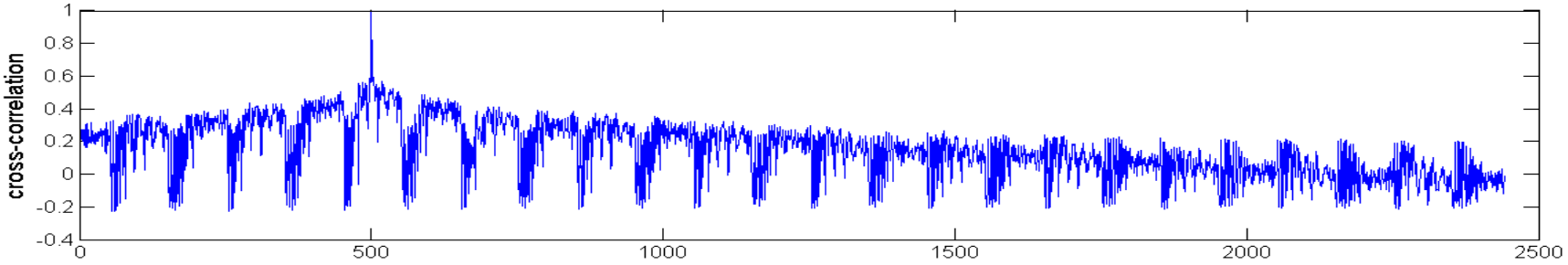
- Sliding signal + noise simulator

$\hat{\delta}(x, \tilde{k}, y^*) \cdot$  + $\hat{\delta}(x^*, \tilde{k}, y) \cdot$  + N

avg. trace

avg. trace – single trace

- Real traces



Another idea: separate signal and noise

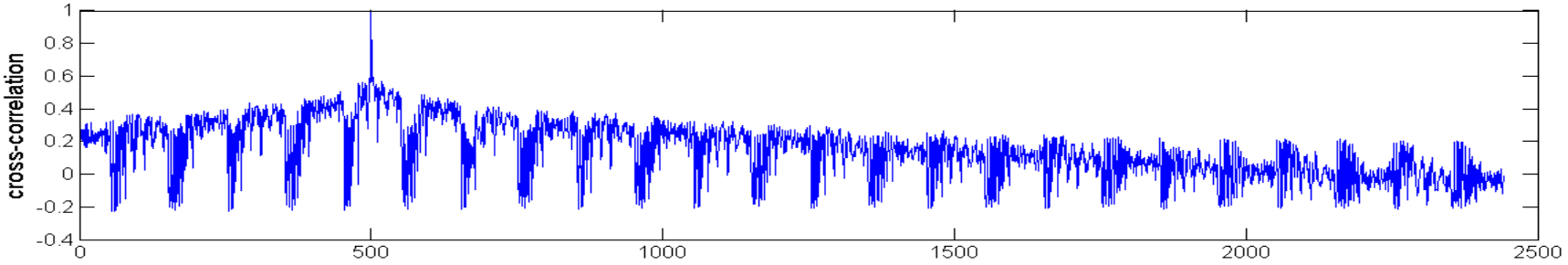
- Sliding signal + noise simulator

$$\hat{\delta}(x, \tilde{k}, y^*) \cdot \blacktriangleleft + \hat{\delta}(x^*, \tilde{k}, y) \cdot \blacktriangleleft + N$$

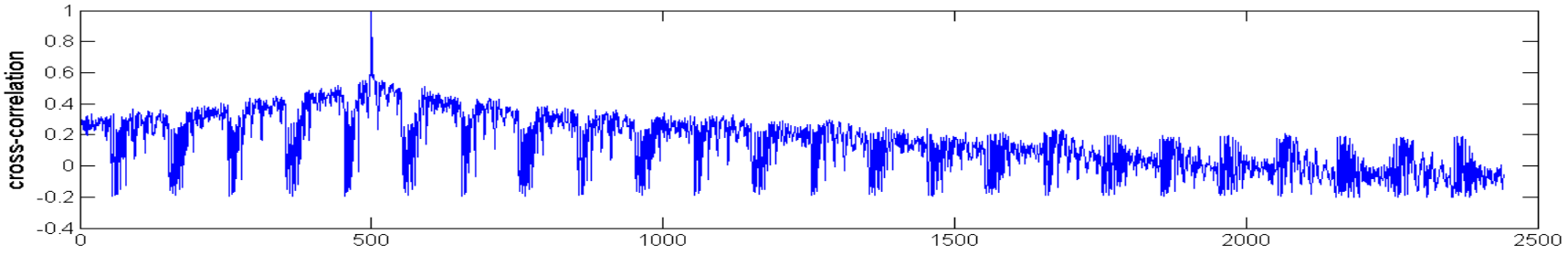
avg. trace

avg. trace – single trace

- Real traces



- Simulated traces



Another idea: separate signal and noise

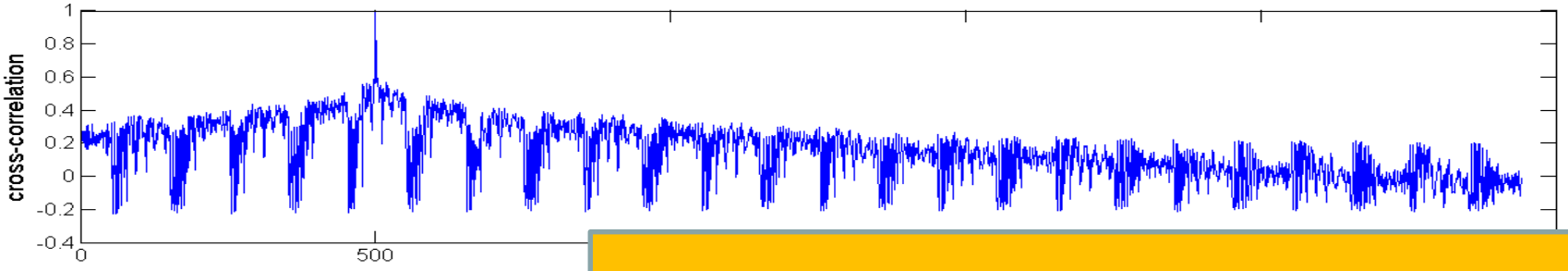
- Sliding signal + noise simulator

$$\hat{\delta}(x, \tilde{k}, y^*) \cdot \blacktriangleleft + \hat{\delta}(x^*, \tilde{k}, y) \cdot \blacktriangleright + N$$

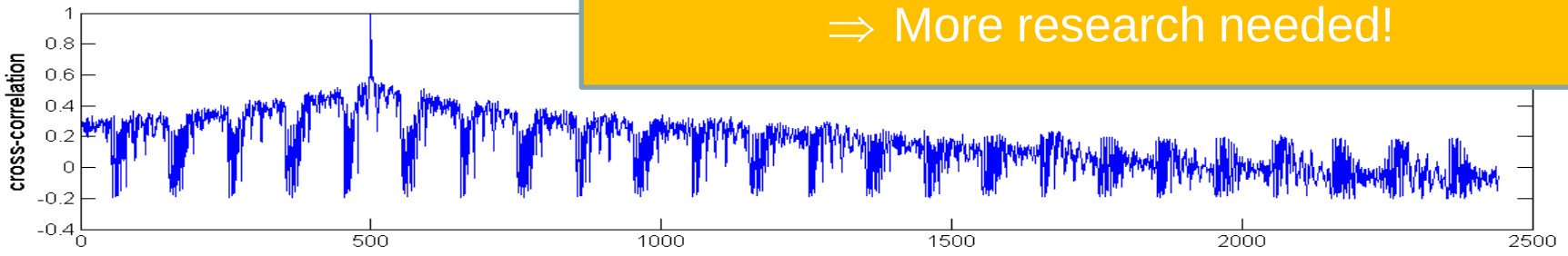
avg. trace

avg. trace – single trace

- Real traces



- Simulated traces



LOOKS BETTER

(but probably not enough for low-freq. events)

⇒ More research needed!