Challenges in Leakage-Resilient Symmetric Cryptography

Krzysztof Pietrzak

ECRYPT II Workshop on Physical Attacks, Graz, November 28, 2012

Provable Security

1. Define "Breaking the Cryptosystem".

2. Construct Cryptosystem.
3. Prove Cryptosystem Secure.

Provable Security

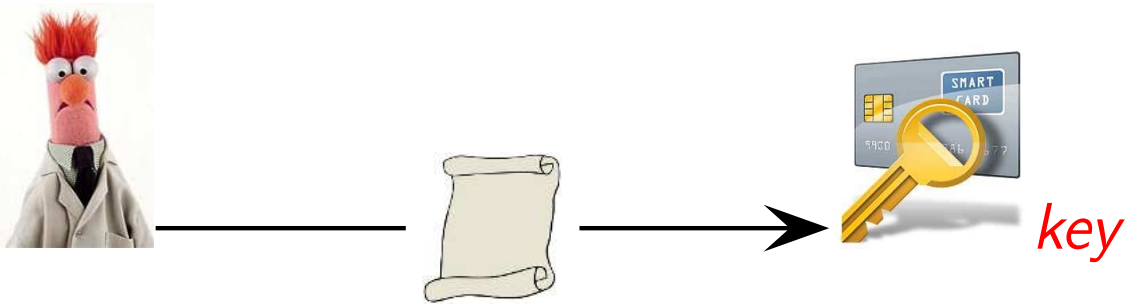
1. Define "Breaking the Cryptosystem".
   Example: Digital Signatures

key

2. Construct Cryptosystem.
3. Prove Cryptosystem Secure.

Provable Security

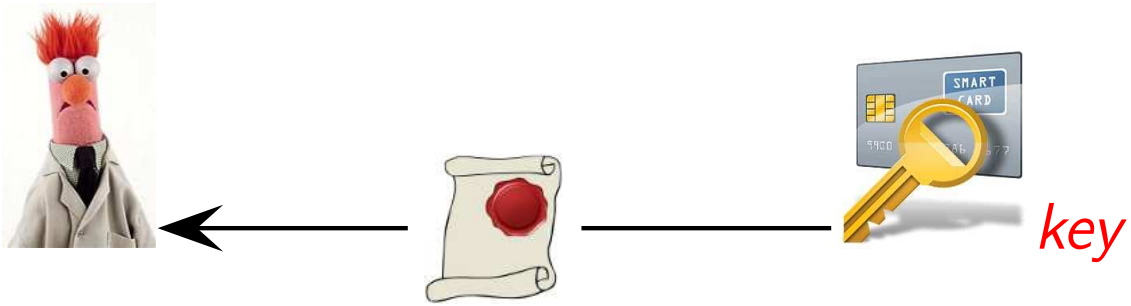
1. Define "Breaking the Cryptosystem".
   Example: Digital Signatures



2. Construct Cryptosystem.
3. Prove Cryptosystem Secure.

Provable Security

1. Define "Breaking the Cryptosystem".
   Example: Digital Signatures



2. Construct Cryptosystem.
3. Prove Cryptosystem Secure.

Provable Security

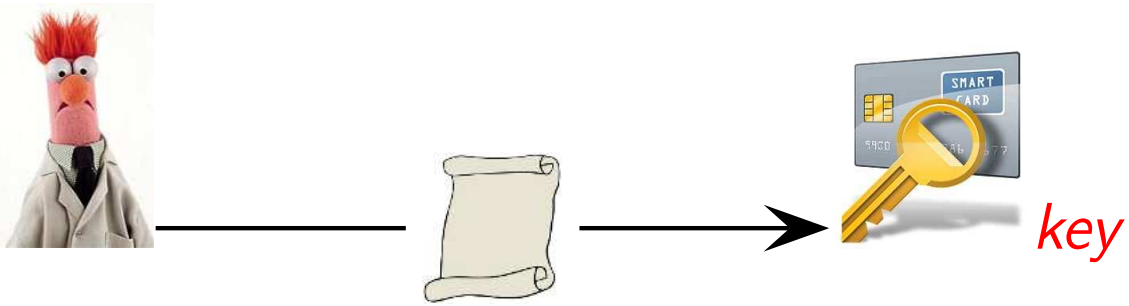
1. Define "Breaking the Cryptosystem".
   Example: Digital Signatures

key

2. Construct Cryptosystem.
3. Prove Cryptosystem Secure.

Provable Security

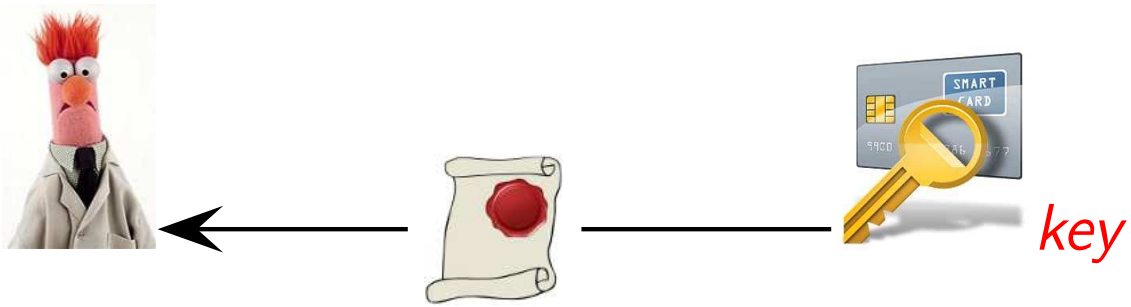
1. Define "Breaking the Cryptosystem".
   Example: Digital Signatures



2. Construct Cryptosystem.
3. Prove Cryptosystem Secure.

Provable Security

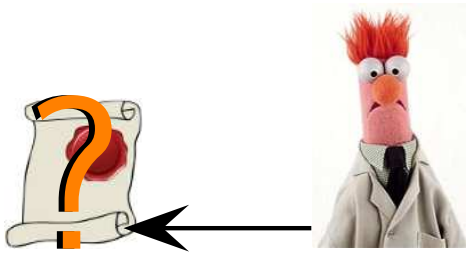
1. Define "Breaking the Cryptosystem".
   Example: Digital Signatures



2. Construct Cryptosystem.
3. Prove Cryptosystem Secure.

Provable Security

1. Define "Breaking the Cryptosystem".
   Example: Digital Signatures



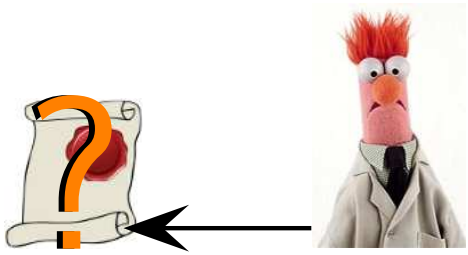
key

 breaks scheme if  is a valid signature for a new message.

2. Construct Cryptosystem.
3. Prove Cryptosystem Secure.

Provable Security

1. Define "Breaking the Cryptosystem".
   Example: Digital Signatures



key

 breaks scheme if  is a valid signature for a new message.

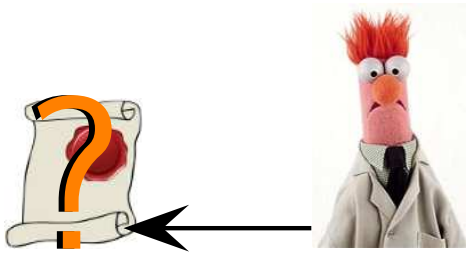
2. Construct Cryptosystem.
3. Prove Cryptosystem Secure.

Theorem

No efficient adversary who breaks the scheme exists

Provable Security

1. Define "Breaking the Cryptosystem".
   Example: Digital Signatures



key

 breaks scheme if  is a valid signature for a new message.

2. Construct Cryptosystem.
3. Prove Cryptosystem Secure.

Theorem

No efficient adversary who breaks the scheme exists if (factoring, SVP,...) is hard.

- *Provably secure cryptosystems get broken in practice.*

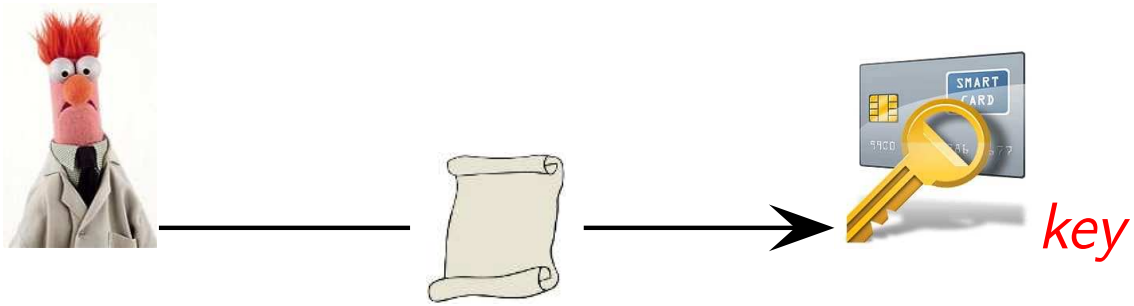
- *Provably secure cryptosystems get broken in practice.*
- *Problem: adversaries outside the anticipated model.*

- *Provably secure cryptosystems get broken in practice.*
- *Problem: adversaries outside the anticipated model.*

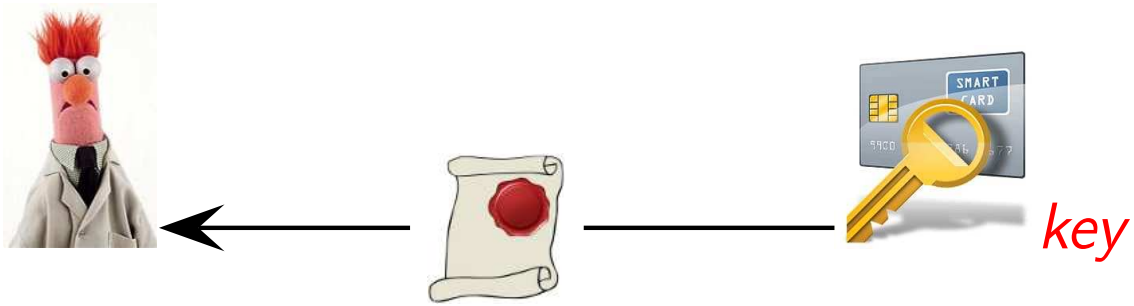
Black-Box Security Models vs. Reality

key

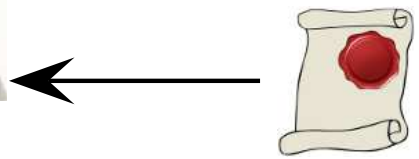
Black-Box Security Models vs. Reality



Black-Box Security Models vs. Reality



Black-Box Security Models vs. Reality



key

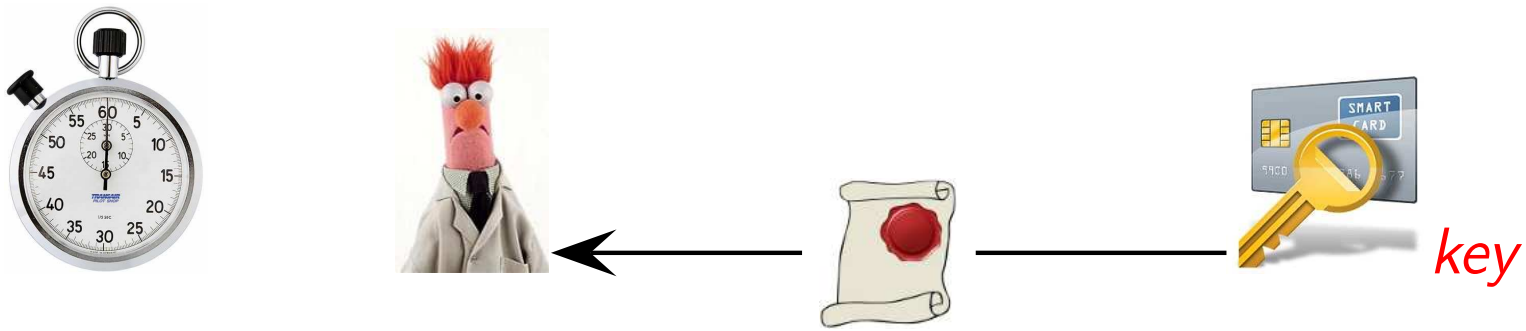
- E.g. can measure time to compute  .

Black-Box Security Models vs. Reality

key

- E.g. can measure time to compute  .
- breaks RSA on smart cards [Kocher'95]

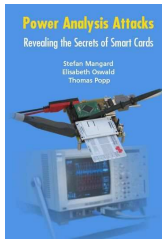
Black-Box Security Models vs. Reality



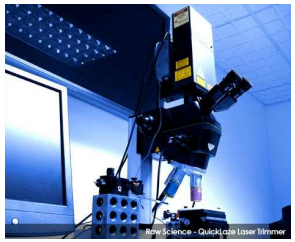
- E.g. can measure time to compute  .
- breaks RSA on smart cards [Kocher'95]

Side-Channel Attack: Cryptanalytic attack exploring information leaked from a physical implementation of a cryptosystem.

- power analysis

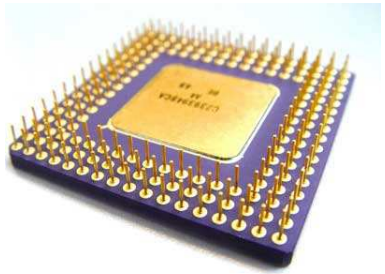


- probing attacks

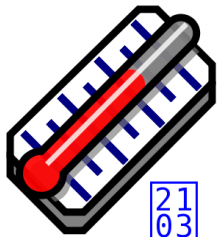


- cold-boot attacks

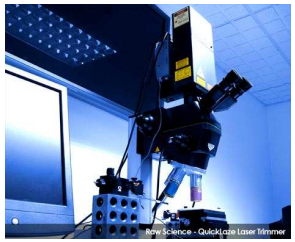
- cache attacks



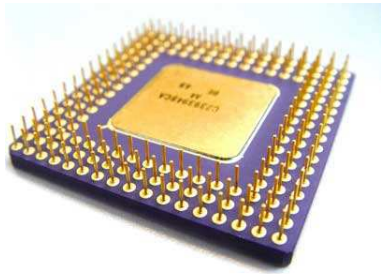
- radiation, sound, heat,. . .



- power analysis
  [Eisenbarth et al. CRYPTO'08]
  break wireless car keys
- probing attacks



- cold-boot attacks
  [Halderman et al. USENIX'08]
  break disc-encryption schemes
- cache attacks
  [Ristenpart et al. CCS'09]
  break cloud computing
- radiation, sound, heat,...



amazon web services™

The Rise of Side-Channel Attacks

- Became major threat in the last few decades.

The Rise of Side-Channel Attacks

- Became major threat in the last few decades.
  - Ubiquitous computing: Light-weight crypto-devices are susceptible to side-channel attacks.

The Rise of Side-Channel Attacks

- Became major threat in the last few decades.
  - Ubiquitous computing: Light-weight crypto-devices are susceptible to side-channel attacks.
  - Provable security: Side-channels became the weakest link.

Side-channels are a physical phenomenon, how could theoretical cryptography be of help?

Side-channels are a physical phenomenon, how could theoretical cryptography be of help?

- Reductions in the context of side-channel attacks [MicRey'04]
- Construct schemes that remain provably secure in the presence of leakage.

Leakage models: one-time vs. continuous

key

one-time

f

key

continuous

Leakage models: one-time vs. continuous

key

one-time

$f(key)$

key

continuous

Leakage models: one-time vs. continuous

key

one-time

key

continuous

Leakage models: one-time vs. continuous

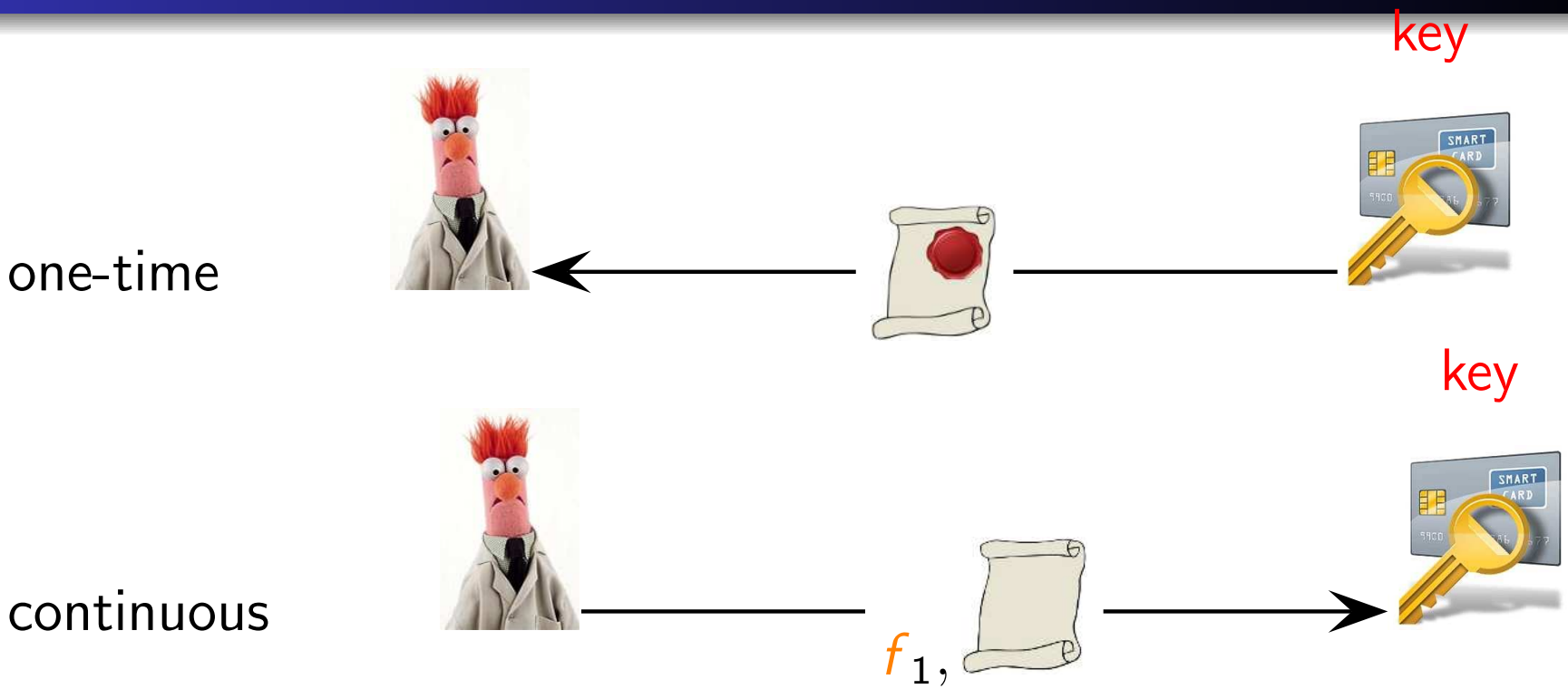
one-time

key

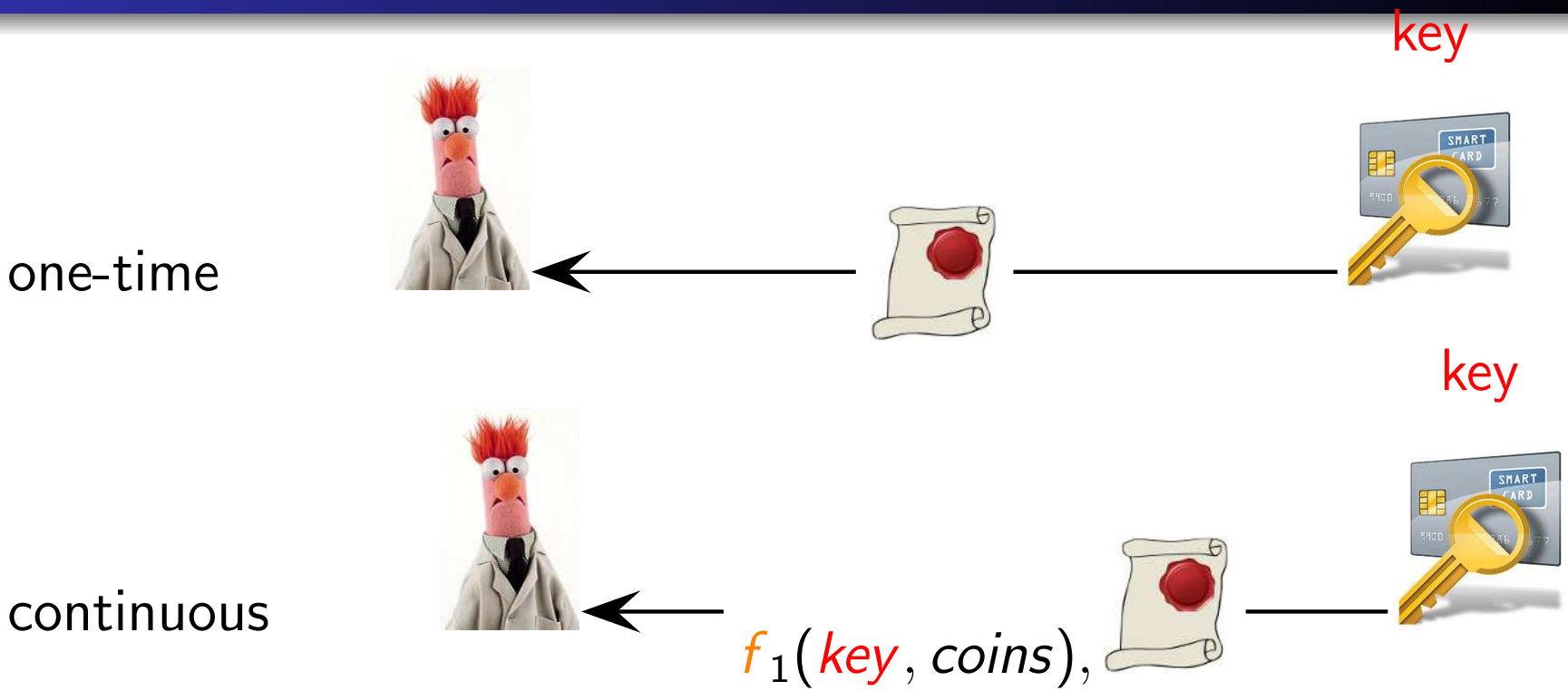
continuous

key

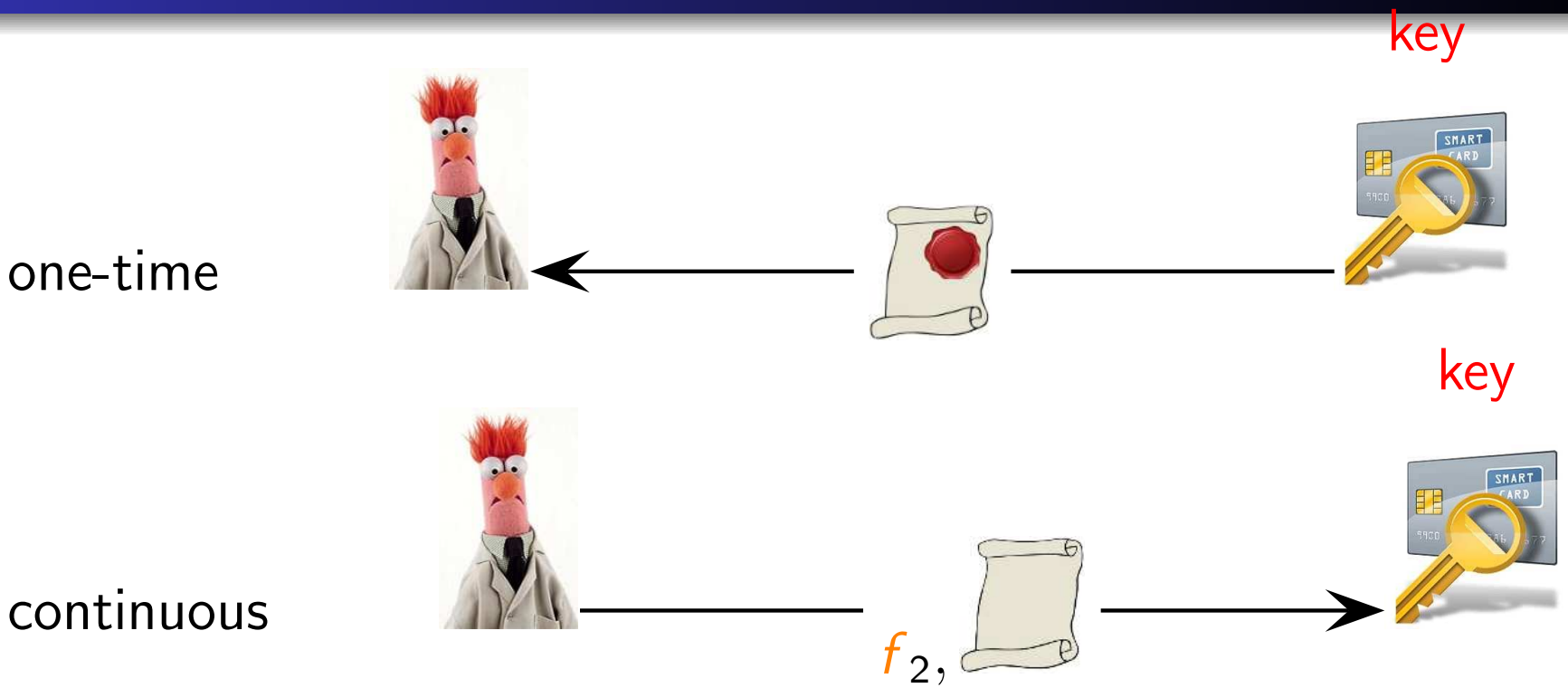
Leakage models: one-time vs. continuous



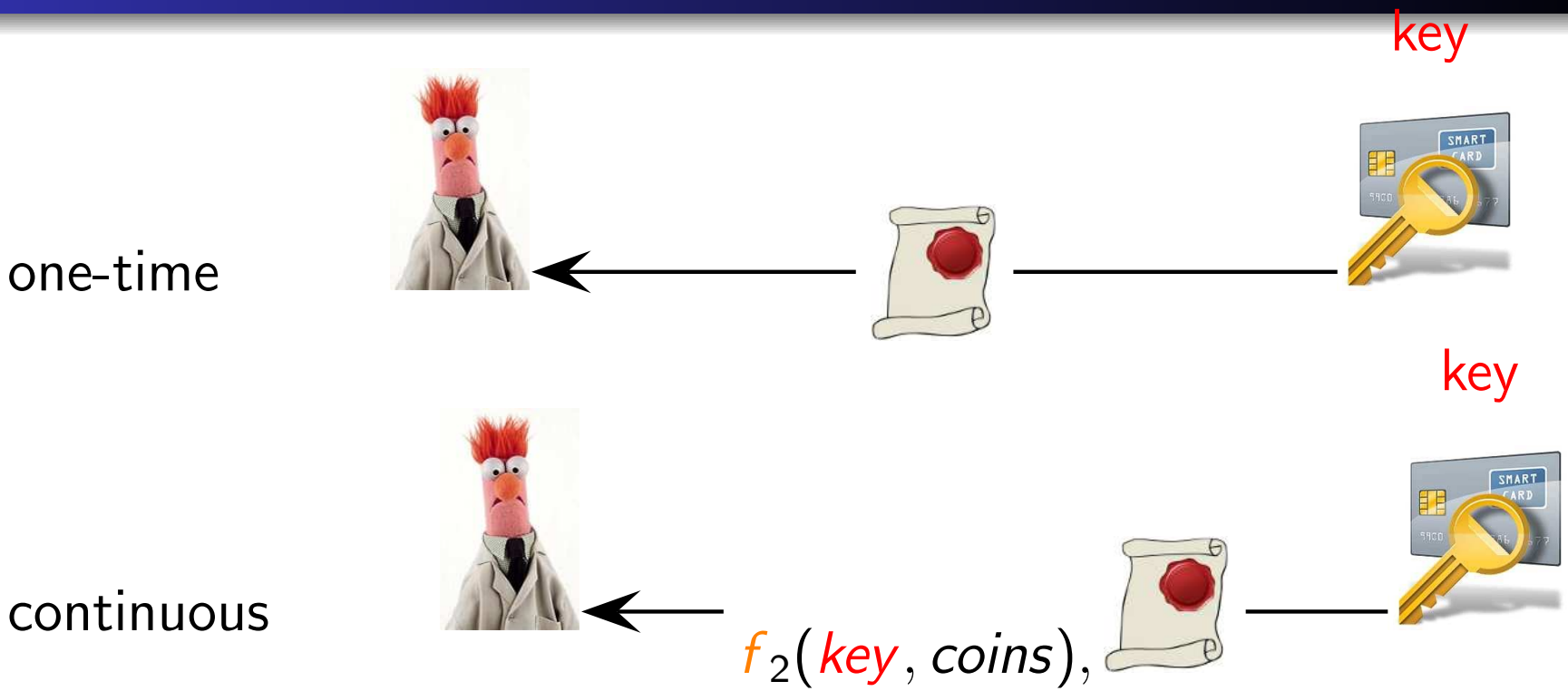
Leakage models: one-time vs. continuous



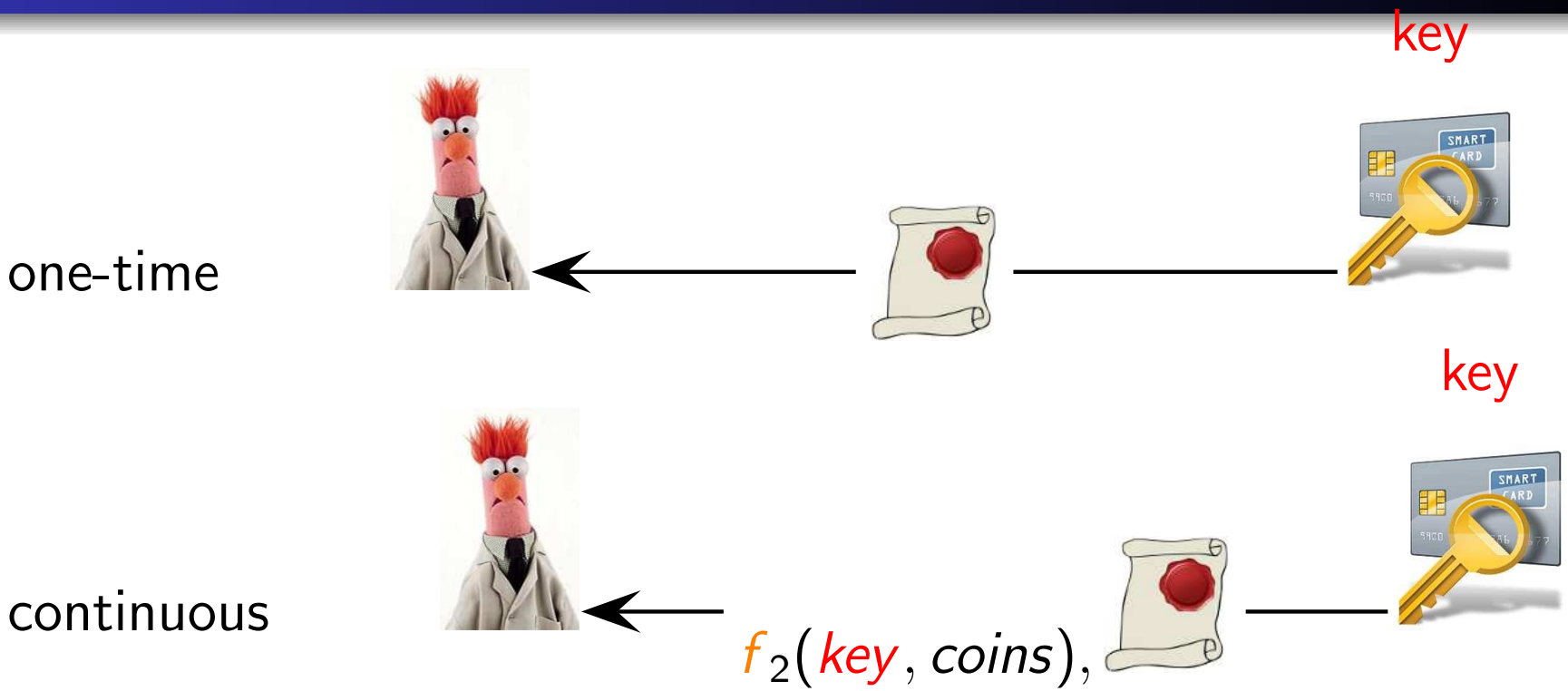
Leakage models: one-time vs. continuous



Leakage models: one-time vs. continuous



Leakage models: one-time vs. continuous



- Most side-channels like timing,power,. . . are continuous. Notable exception cold-boot.
- Security against continuous leakage is *much* harder to achieve. E.g. requires key-refreshing.
- Intermediate "Floppy model".

Leakage models: dedicated vs. general

dedicated leakage functions

f models a particular side-channel

timing: Make running time independent of input.

probing: Private Circuits ([Ishai,Sahai,Wagner Crypto'03])

Leakage models: dedicated vs. general

dedicated leakage functions

f models a particular side-channel

timing: Make running time independent of input.

probing: Private Circuits ([Ishai,Sahai,Wagner Crypto'03])

general leakage functions

bounded: $f(key)$ has length $\ell \ll |key|$ bits.

entropic: Entropy of key decreases by at most ℓ given $f(key)$.

auxiliary input: Computationally hard to compute key given $f(key)$.

One-Time Bounded/Entropic leakage

$key \in \{0,1\}^n$. Adv choses f and gets $f(key)$.

1. Bounded leakage: f must satisfy $|f(key)| = \ell \ll n$.
2. Entropic leakage: f must satisfy $H_\infty(key|f(key)) \geq n - \ell$.

- Maurer's bounded storage model, privacy amplification,...
- Intrusion resilience [Dzi'06,CDDLLW'07,...] (symmetric)
- Memory attacks [AGV'09,NaoSeg'09,...] (public-key)

Continuous Bounded/Entropic leakage

- key_i state after i'th invocation of the scheme.
- $key_i^+ \subseteq key_i$ touched in i'th invocation.

Before i'th invocation, Adv chooses $f(.)$ with range $\{0,1\}^\ell$ and gets

- $f(key_i^+)$ (Leakage-Resilient Cryptography [DziPie08],...)
- $f(key_i)$ (Continuous Memory Attacks [DHLW12],...)

Public-key

Signatures: [AWD09, KV09, FKPR10, DHLW10, BKKV10, BSW11,...]
Public key encryption: [AGV09, NS09, DHLW10, BKKV10, BSW11,...]
Identity based encryption: [DHLW10, CDRW10, LRW11,...]
Multiparty Computation: [FRRTV10, GR10, JV10,...]
Zero Knowledge: [GJS11,...]

Secret-key

Stream-Ciphers: [DP08, Pie09, YSPY10,YS12,...]
Pseudorandom Functions/Permutations: [DP10, FPS11,MSJ12,...]

Compilers

[ISW03,FRRTV10,GolRot12,...]

Public-key

Signatures: [AWD09, KV09, FKPR10, DHLW10, BKKV10, BSW11,...]
Public key encryption: [AGV09, NS09, DHLW10, BKKV10, BSW11,...]
Identity based encryption: [DHLW10, CDRW10, LRW11,...]
Multiparty Computation: [FRRTV10, GR10, JV10,...]
Zero Knowledge: [GJS11,...]

Secret-key

Stream-Ciphers: [DP08, Pie09, YSPY10,YS12,...]
Pseudorandom Functions/Permutations: [DP10, FPS11,MSJ12,...]

Compilers

[ISW03,FRRTV10,GolRot12,...]

3 Principles

- Share Secret: Blinding
- Evolve Secret: Stream-Ciphers
- Hide Secret: For every *pk* many *sk* (HPS,Σ-Protocols)

Leakage-Resilient Stream-Ciphers

weak PRFs

Definition Weak PRF

$\mathcal{F} : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^m$ is a (s, ϵ, q) secure weak PRF if no adversary of size s can distinguish the following distributions advantage ϵ

$$(X_1, Y_1), \ldots, (X_q, Y_q) \qquad (X_1, Z_1), \ldots, (X_q, Z_q)$$

where X_i, Z_i are uniform and $Y_i = \mathcal{F}(K, X_i)$ for a random K.

weak PRFs

Definition Weak PRF

$\mathcal{F} : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^m$ is a (s, ϵ, q) secure weak PRF if no adversary of size s can distinguish the following distributions advantage ϵ

$$(X_1, Y_1), \ldots, (X_q, Y_q) \qquad (X_1, Z_1), \ldots, (X_q, Z_q)$$

where X_i, Z_i are uniform and $Y_i = \mathcal{F}(K, X_i)$ for a random K.

Definition Min-Entropy

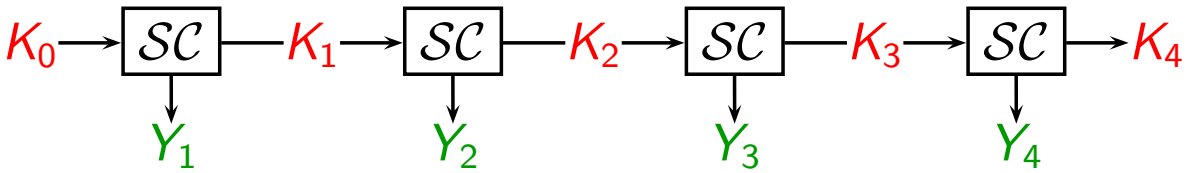
X has min entropy m if $\Pr[X = x] \leq 2^{-m}$ for all x.

If $K \in \{0,1\}^k$ is uniform and $|f(K)| = \lambda$, then K has $k - \lambda$ bits min-entropy given $f(K)$.

Stream-Cipher

Definition (Stream-Cipher)

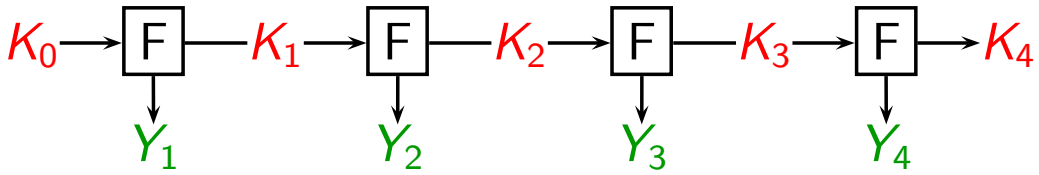
- A function $\mathcal{SC} : \{0,1\}^\kappa \to \{0,1\}^n \times \{0,1\}^\kappa$ is a stream-cipher if for random K_0 the output $Y_1, Y_2, \ldots$ (where $(K_i, Y_i) = \mathcal{SC}(K_{i-1})$) is pseudorandom



Stream-Cipher

Definition (Stream-Cipher)

- A function $\mathcal{SC} : \{0,1\}^\kappa \to \{0,1\}^n \times \{0,1\}^\kappa$ is a stream-cipher if for random K_0 the output $Y_1, Y_2, \ldots$ (where $(K_i, Y_i) = \mathcal{SC}(K_{i-1})$) is pseudorandom

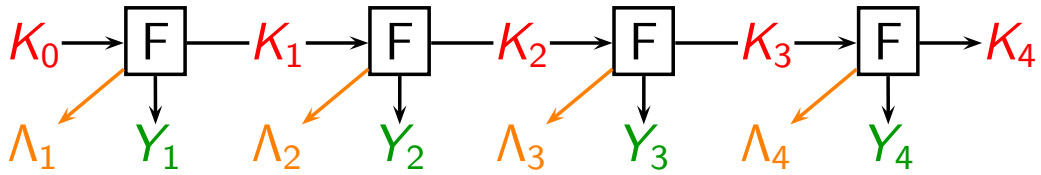


Can use any pseudorandom generator F

Stream-Cipher

Definition (Stream-Cipher)

- A function $\mathcal{SC} : \{0,1\}^\kappa \to \{0,1\}^n \times \{0,1\}^\kappa$ is a stream-cipher if for random K_0 the output $Y_1, Y_2, \ldots$ (where $(K_i, Y_i) = \mathcal{SC}(K_{i-1})$) is pseudorandom

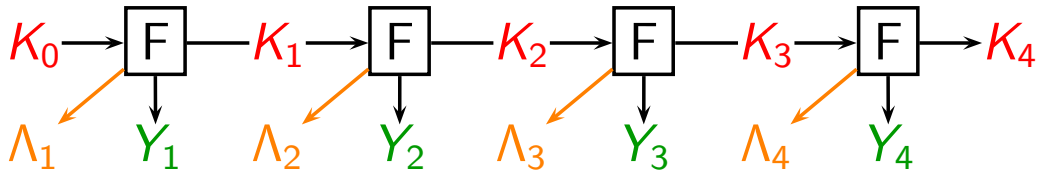


Can use any pseudorandom generator F

Stream-Cipher

Definition (Stream-Cipher)

- A function $\mathcal{SC} : \{0,1\}^\kappa \to \{0,1\}^n \times \{0,1\}^\kappa$ is a stream-cipher if for random K_0 the output $Y_1, Y_2, \ldots$ (where $(K_i, Y_i) = \mathcal{SC}(K_{i-1})$) is pseudorandom



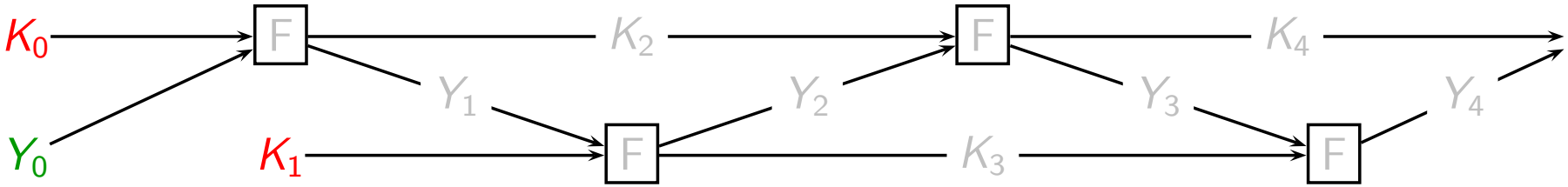
Can use any pseudorandom generator F

But not leakage resilient even for $\lambda = 1$*: For* $t = |K| + 1$*, define*

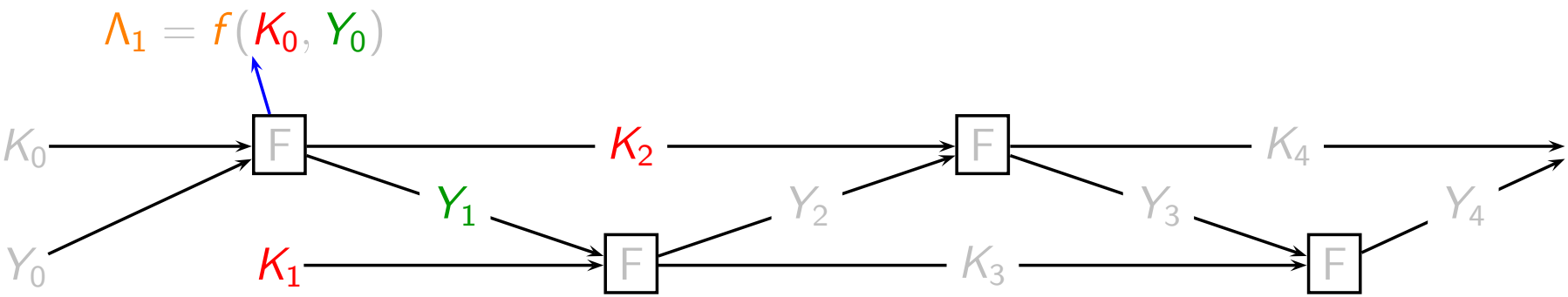
$$\Lambda_i = f(K_{i-1}) \stackrel{\text{def}}{=} \; i\text{'th bit of } K_t.$$

After t *rounds leaked entire* K_t*.*

A Leakage Resilient Stream-Cipher from any wPRF

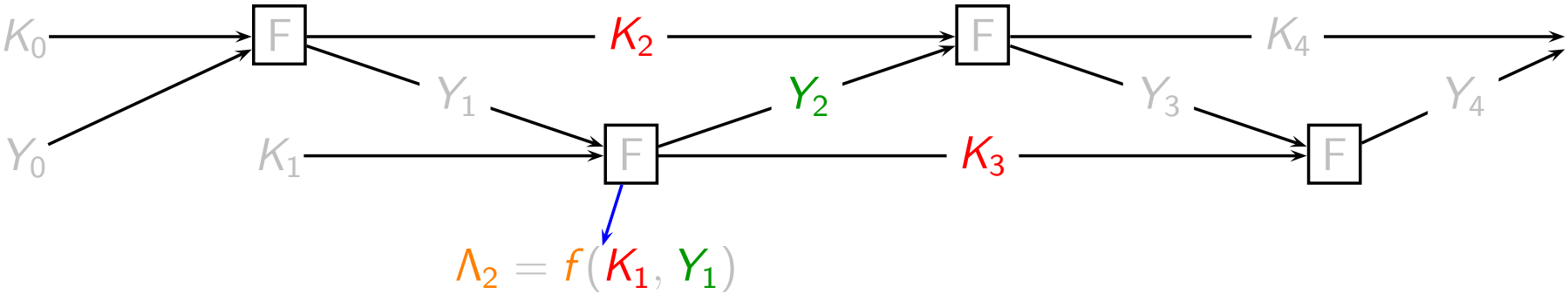


A Leakage Resilient Stream-Cipher from any wPRF

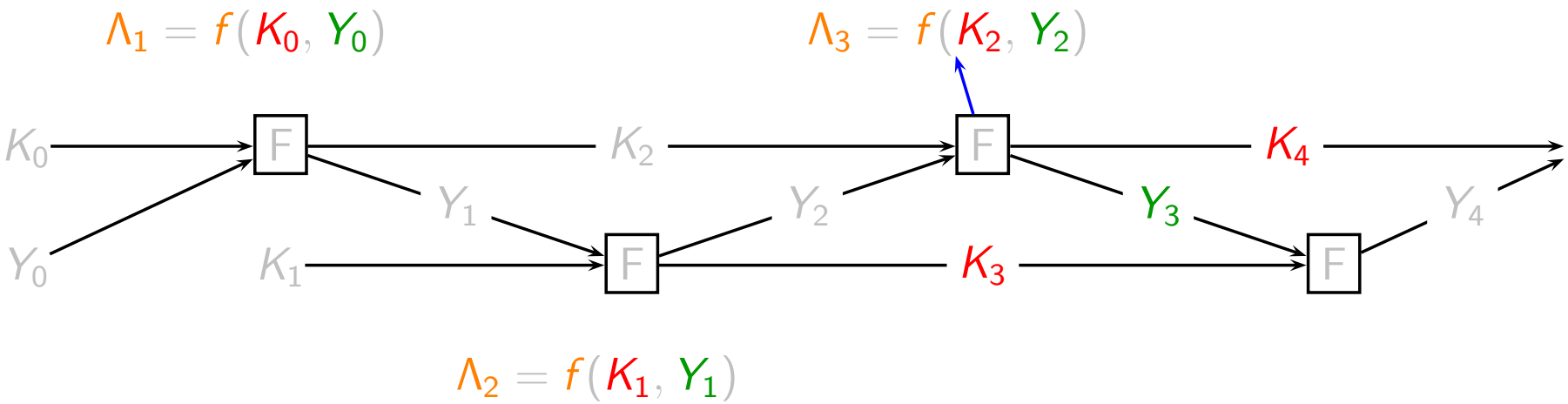


A Leakage Resilient Stream-Cipher from any wPRF

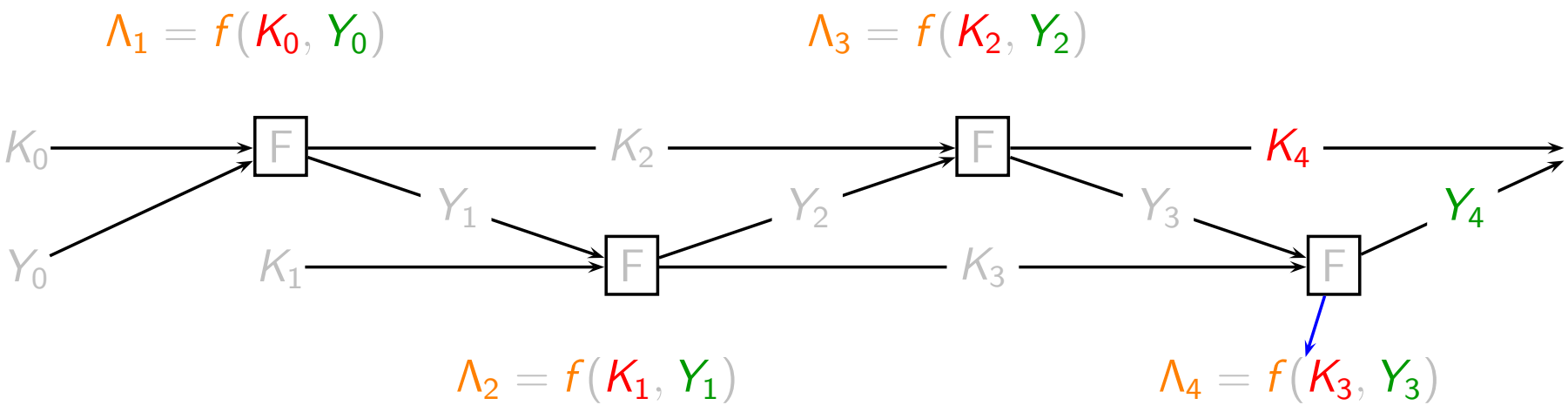
$\Lambda_1 = f(K_0, Y_0)$



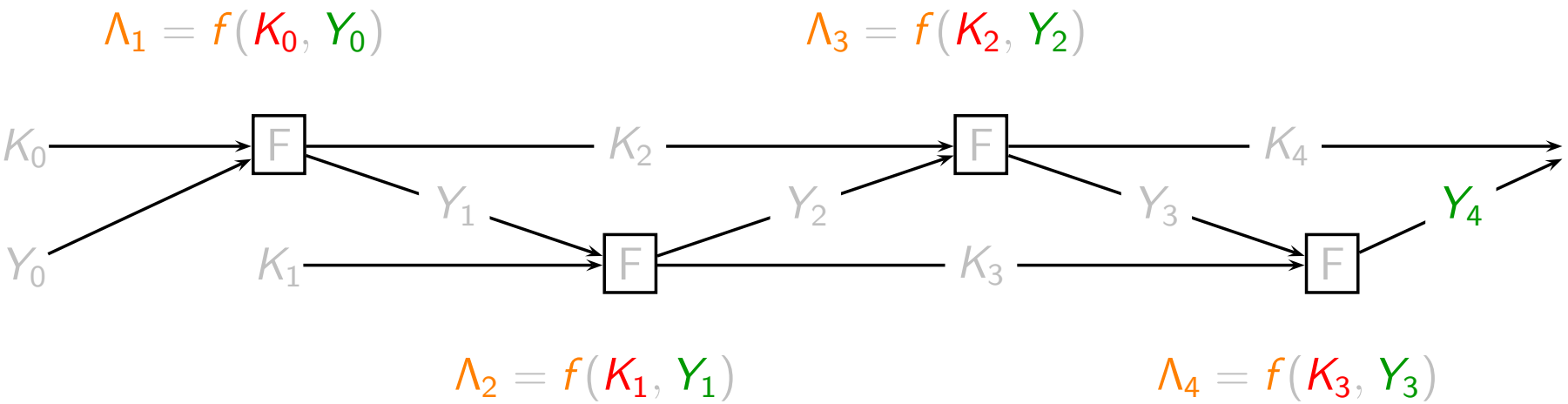
A Leakage Resilient Stream-Cipher from any wPRF



A Leakage Resilient Stream-Cipher from any wPRF



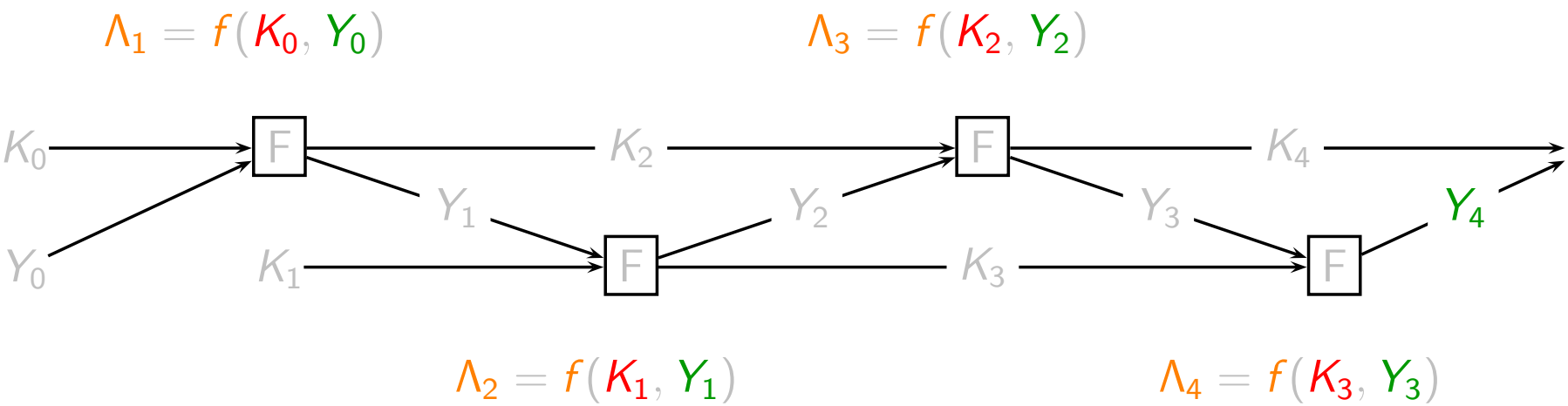
A Leakage Resilient Stream-Cipher from any wPRF



Theorem ([P'09])

If F *is a wPRF then the above is a leakage-resilient stream-cipher:*
Given $Y_0, \ldots, Y_i$ *and* $\Lambda_1, \ldots, \Lambda_i$ *the* $Y_{i+1}, Y_{i+2}, \ldots$ *is pseudorandom.*

A Leakage Resilient Stream-Cipher from any wPRF



Theorem ([P'09])

If F *is a wPRF then the above is a leakage-resilient stream-cipher:*
Given $Y_0, \ldots, Y_i$ *and* $\Lambda_1, \ldots, \Lambda_i$ *the* $Y_{i+1}, Y_{i+2}, \ldots$ *is pseudorandom.*

- Leakage function $f(K_i, Y_i) \to \Lambda_i$ can't compute $K_{i+2}, K_{i+3}, \ldots$.

Quantitative bound in [P'09] is nowhere practical.

(s, ϵ) secure wPRF gave (s', ϵ') secure stream cipher where

$$\epsilon' \approx \epsilon^{1/12} \qquad s' \approx s \cdot \epsilon^2$$

As $\log(s/\epsilon) \leq$ key length, require wPRF with key length $\gg 1000$ to get meaningful bounds.

Quantitative bound in [P'09] is nowhere practical.

(s, ϵ) secure wPRF gave (s', ϵ') secure stream cipher where

$$\epsilon' \approx \epsilon^{1/12} \qquad s' \approx s \cdot \epsilon^2$$

As $\log(s/\epsilon) \leq$ key length, require wPRF with key length $\gg 1000$ to get meaningful bounds.

With two recent results we can give a meaningful bound for keys of length 256.

- Overcoming weak expectations. [DodisYu 2012]
- How to fake auxiliary input. [JetchevP 2012]

Overcoming Weak Expectations

Theorem [DodisYu 2012] (improving [P'09])

If $\mathcal{F}$ is a $(\epsilon, 2s, 2q)$ secure wPRF, then it is a

$$(\sqrt{2^{\lambda}\epsilon}, s, q)$$

secure wPRF if the key $K \in \{0,1\}^k$ comes from any distribution with $k - \lambda$ bits of min-entropy.

Overcoming Weak Expectations

Theorem [DodisYu 2012] (improving [P'09])

If $\mathcal{F}$ is a $(\epsilon, 2s, 2q)$ secure wPRF, then it is a

$$(\sqrt{2^{\lambda}\epsilon}, s, q)$$

secure wPRF if the key $K \in \{0,1\}^k$ comes from any distribution with $k - \lambda$ bits of min-entropy.

Every weak PRF is one-time bounded leakage-resilient!

Every also holds for entropic leakage (if leakage function is efficient).

How to fake auxiliary input

Theorem [JetchevP 2012]

- Consider any joint distribution $(X, A) \in \mathcal{X} \times \{0,1\}^{\lambda}$.

How to fake auxiliary input

Theorem [JetchevP 2012]

- Consider any joint distribution $(X, A) \in \mathcal{X} \times \{0,1\}^{\lambda}$.
- Let $\mathcal{D}$ be a class distinguishers, say circuits of size $s = 2^{80}$

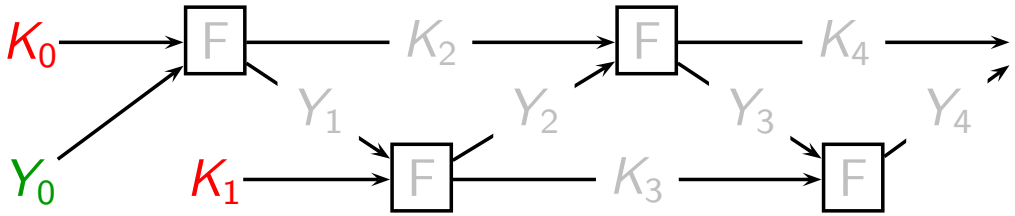
How to fake auxiliary input

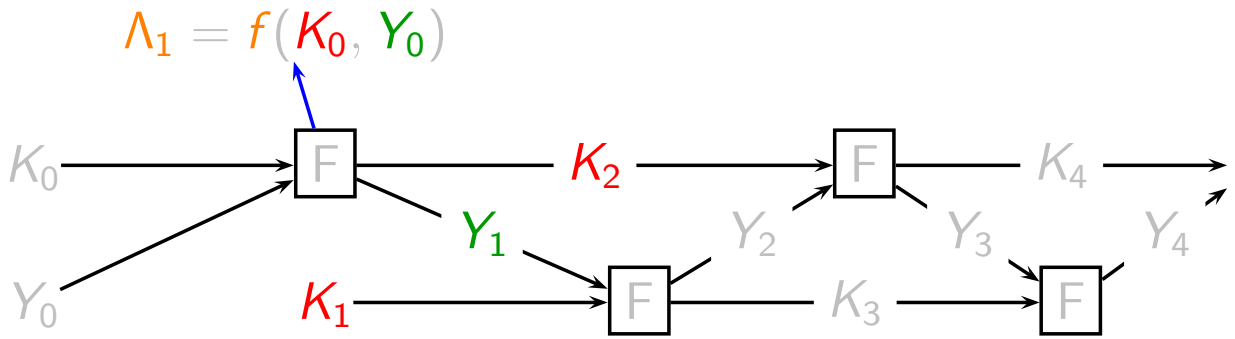
Theorem [JetchevP 2012]

- Consider any joint distribution $(X, A) \in \mathcal{X} \times \{0,1\}^\lambda$.
- Let $\mathcal{D}$ be a class distinguishers, say circuits of size $s = 2^{80}$
- There exists an efficient simulator $h : \mathcal{X} \to \{0,1\}^\lambda$ that
  - Fools every D in $\mathcal{D}$

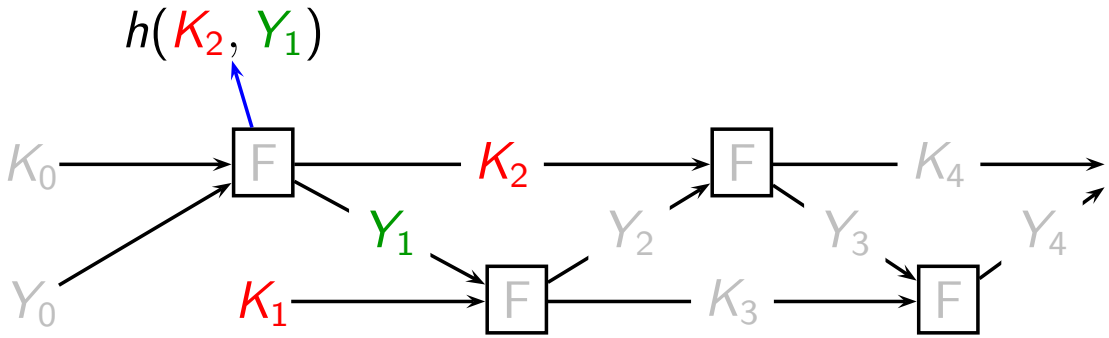
$$\forall D \in \mathcal{D} \ : \ |\Pr[D(X, A) = 1] - \Pr[D(X, h(X)) = 1]| \leq \epsilon$$

  - h is of size $s \cdot 2^{3\lambda}/\epsilon^2$.

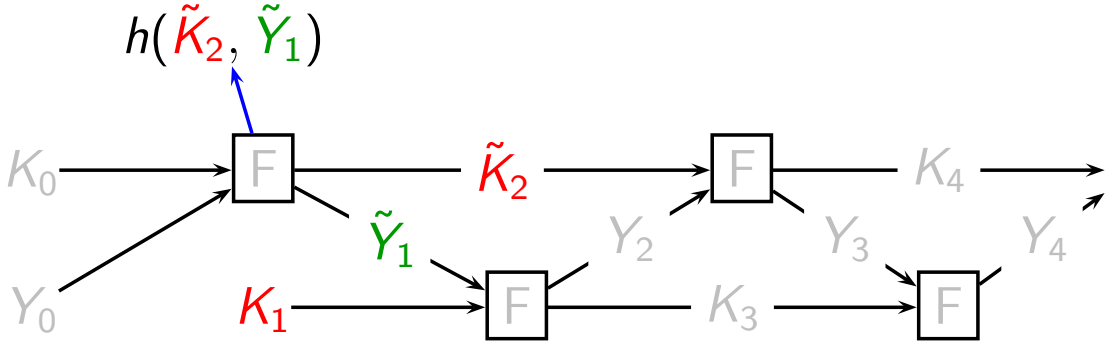

K_0
F
K_2
F
K_4
Y_1
Y_2
Y_3
Y_4
Y_0
K_1
F
K_3
F



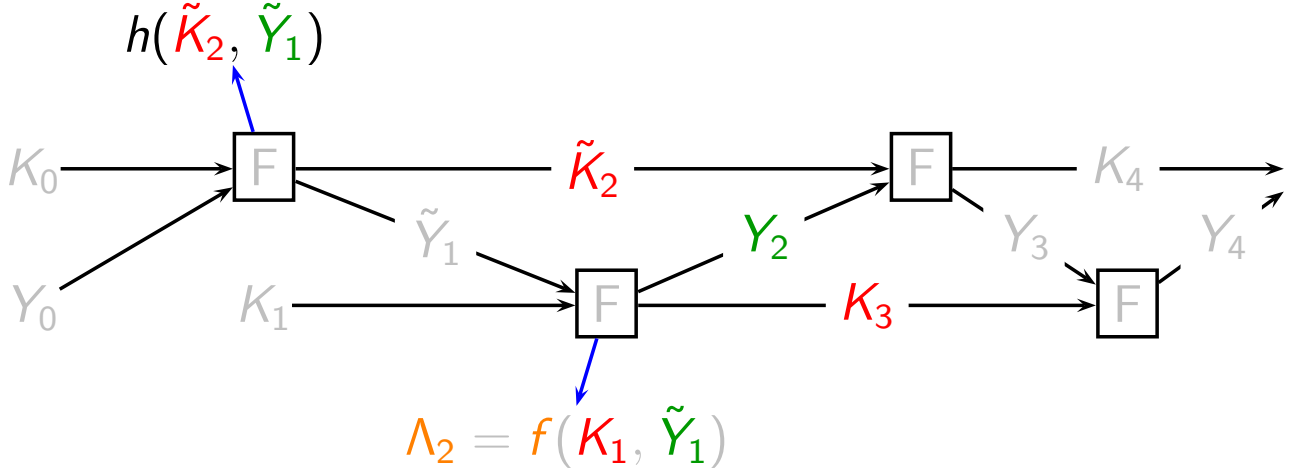
- Replace Λ_1 with "fake" $h(K_2, Y_1)$.



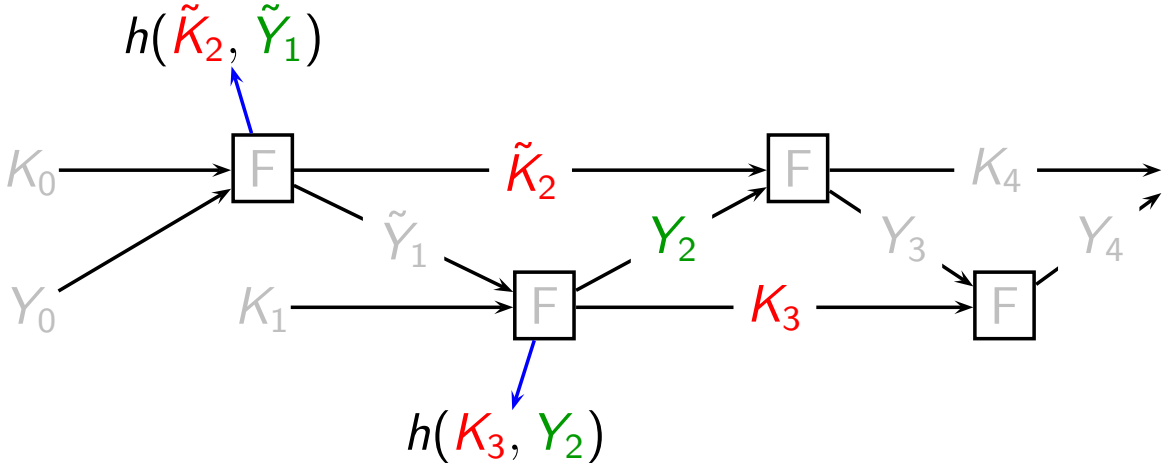
- Replace Λ_1 with "fake" $h(K_2, Y_1)$.
- Replace (Y_1, K_2) with uniformly random $(\tilde{Y}_1, \tilde{K}_2)$



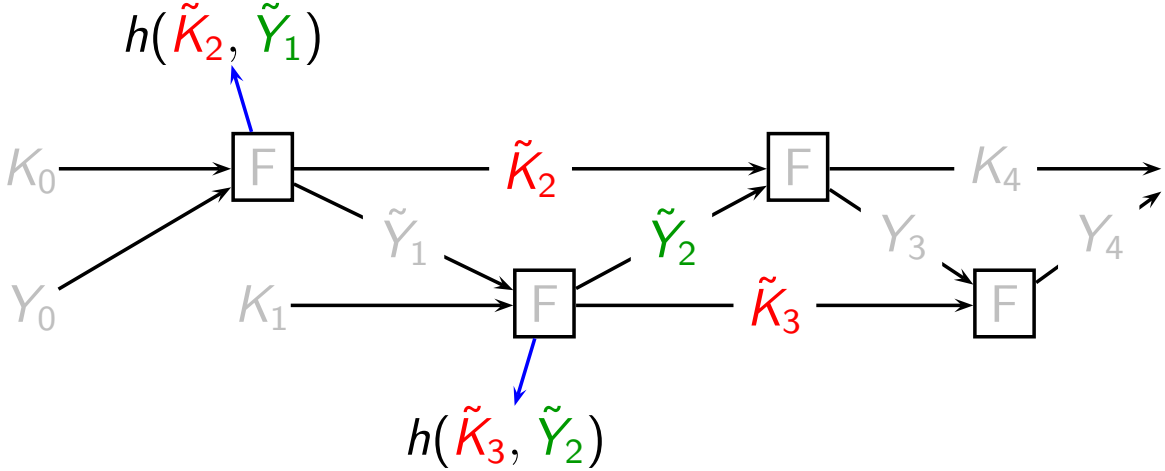
- Replace Λ_1 with "fake" $h(K_2, Y_1)$.
- Replace (Y_1, K_2) with uniformly random $(\tilde{Y}_1, \tilde{K}_2)$
- ...



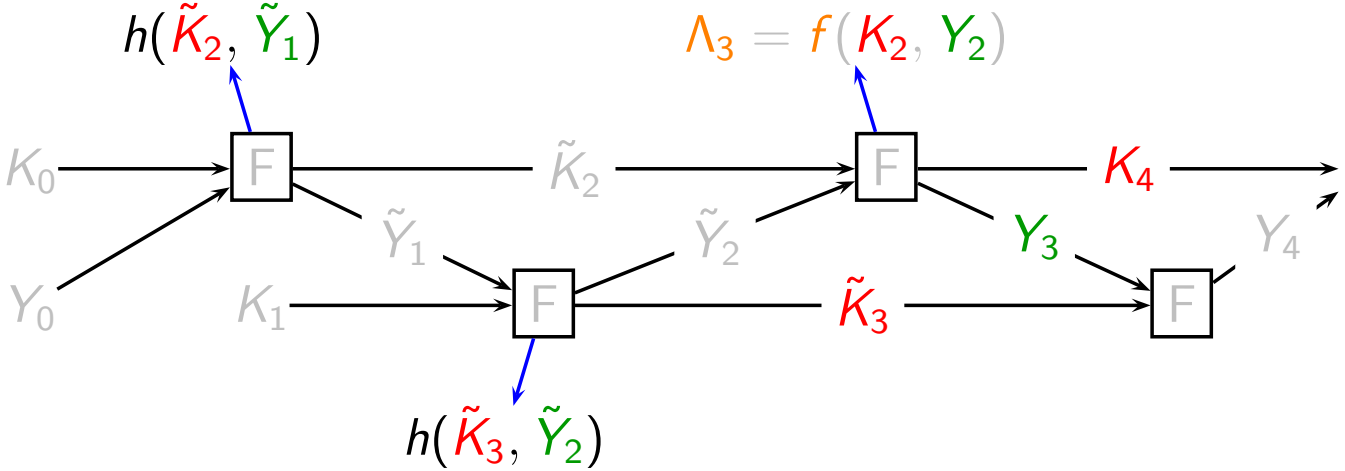
- Replace Λ_1 with "fake" $h(K_2, Y_1)$.
- Replace (Y_1, K_2) with uniformly random $(\tilde{Y}_1, \tilde{K}_2)$
- ...



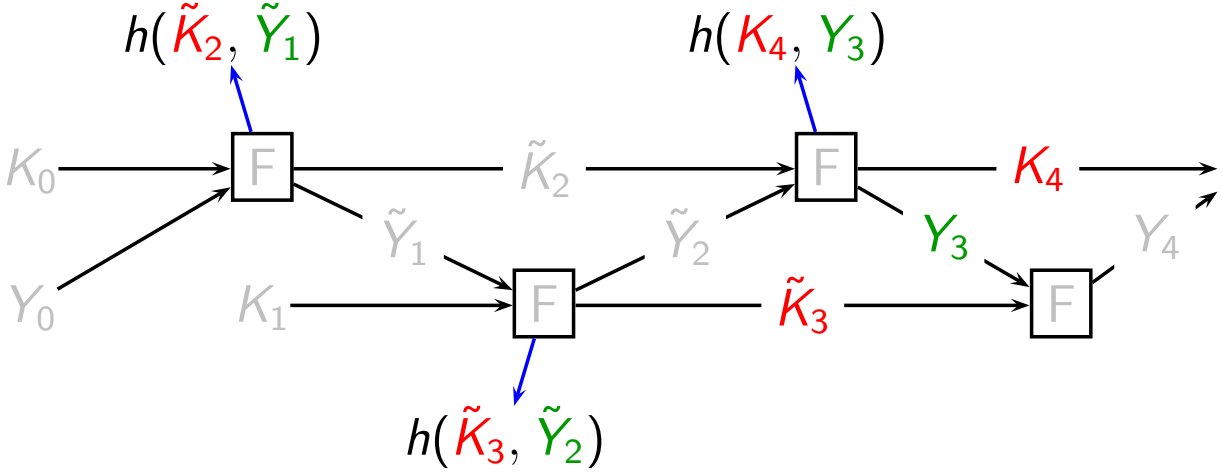
- Replace Λ_1 with "fake" $h(K_2, Y_1)$.
- Replace (Y_1, K_2) with uniformly random $(\tilde{Y}_1, \tilde{K}_2)$
- ...



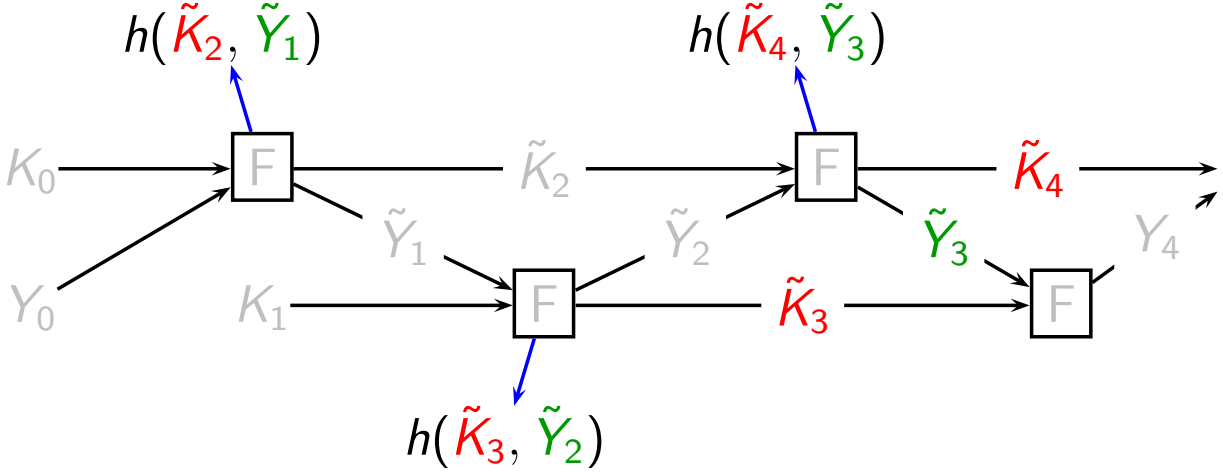
- Replace Λ_1 with "fake" $h(K_2, Y_1)$.
- Replace (Y_1, K_2) with uniformly random $(\tilde{Y}_1, \tilde{K}_2)$
- ...



- Replace Λ_1 with "fake" $h(K_2, Y_1)$.
- Replace (Y_1, K_2) with uniformly random $(\tilde{Y}_1, \tilde{K}_2)$
- ...



- Replace Λ_1 with "fake" $h(K_2, Y_1)$.
- Replace (Y_1, K_2) with uniformly random $(\tilde{Y}_1, \tilde{K}_2)$
- ...



- Replace Λ_1 with "fake" $h(K_2, Y_1)$.
- Replace (Y_1, K_2) with uniformly random $(\tilde{Y}_1, \tilde{K}_2)$
- ...

λ : # of bits leaked per round. q : # of blocks output.

Lemma ([JetPie'12])

If F *is a* $(\epsilon_F, s_F, 2)$*-secure weak PRF the this is a* $(\epsilon', s', q, \lambda)$*-secure leakage resilient stream cipher where*

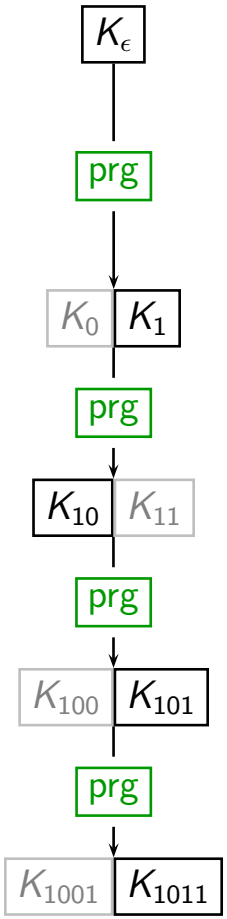
$$\epsilon' = 4q\sqrt{\epsilon_F 2^{\lambda}} \qquad s' = \Theta(1) \cdot \frac{s_F \epsilon'^2}{2^{3\lambda}}$$

$$q = 2^{20} \ , \ \lambda = 10 \ , \ \epsilon_F = 2^{-100} \ , \ s_F = 2^{154} \quad (s_F/\epsilon_F = 2^{256})$$

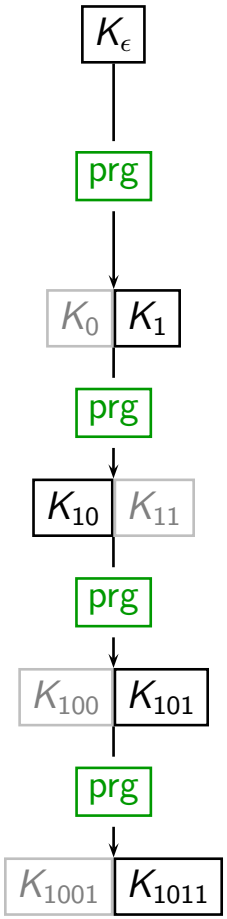
$$\epsilon' = 2^{-23} \qquad s' = 2^{78}$$

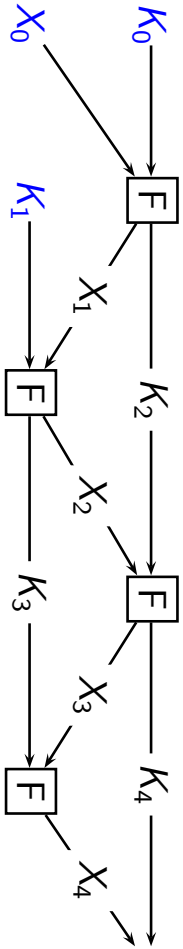
Leakage-Resilient PRFs

GGM

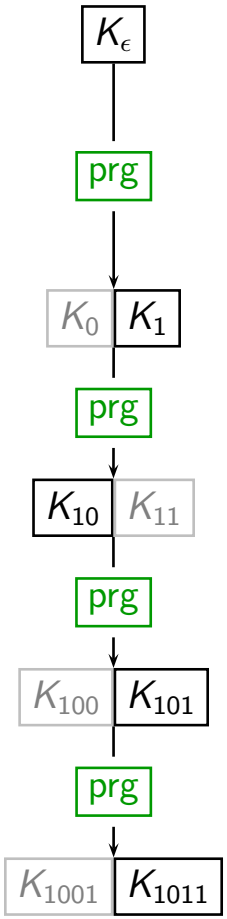


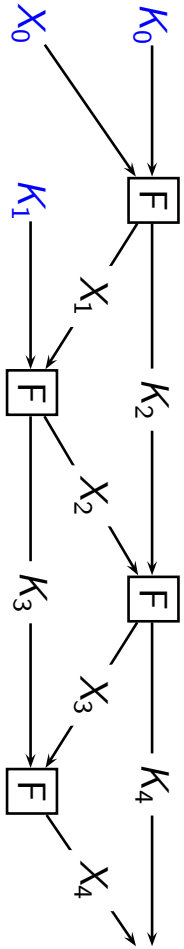
GGM + LR-SC

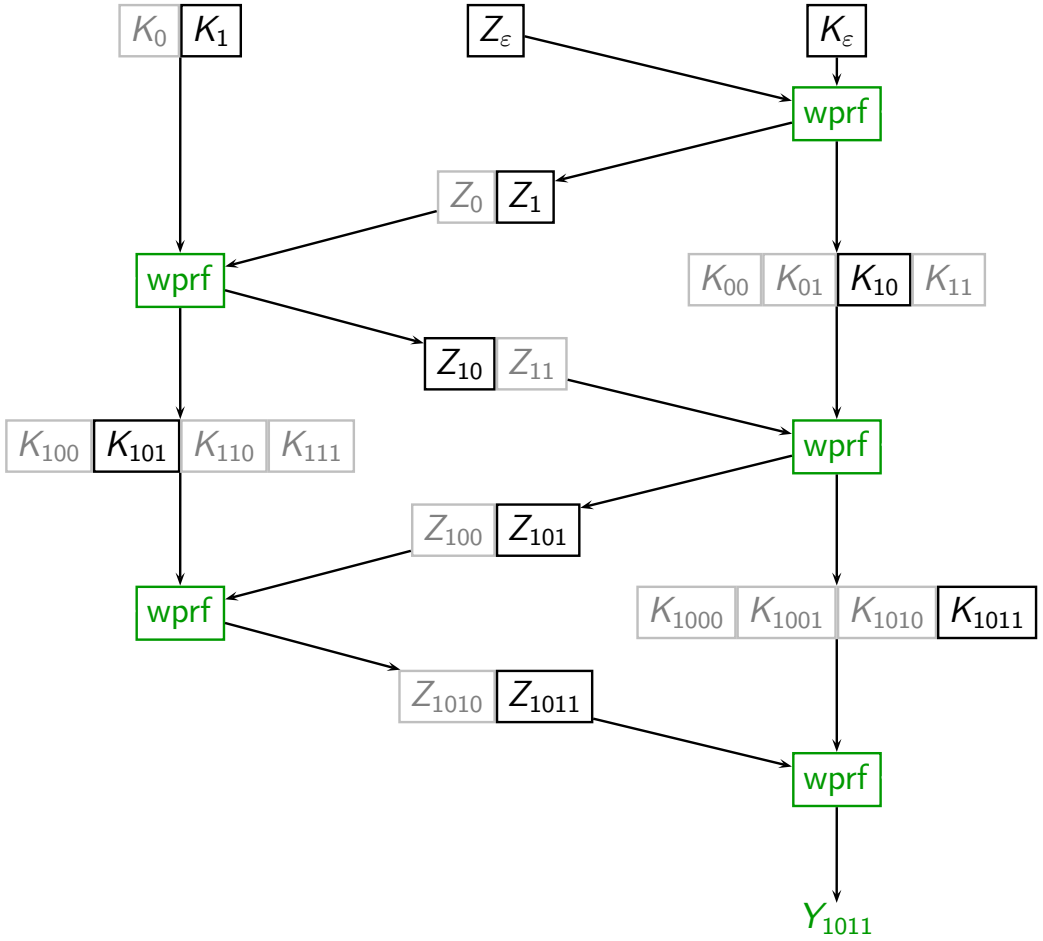

K_ϵ
prg
K_0 K_1
prg
K_{10} K_{11}
prg
K_{100} K_{101}
prg
K_{1001} K_{1011}


X_0
K_0
K_1
F
X_1
F
K_2
X_2
K_3
F
X_3
F
K_4
X_4

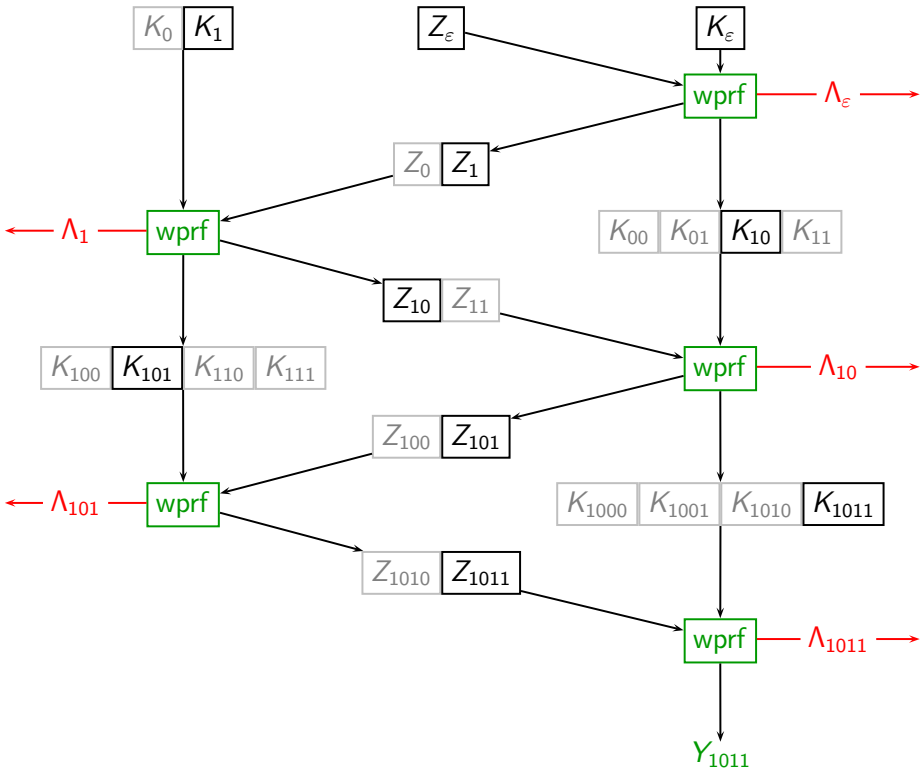
GGM + LR-SC $\Rightarrow$ LR-PRF







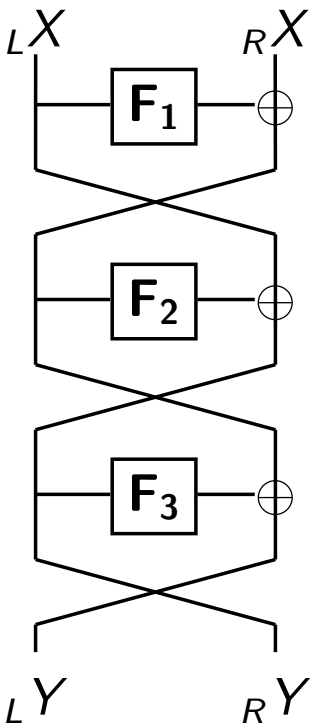
LR-PRF



1. Granular Leakage.
2. Non-adaptive leakage.
3. 1. & 2. allow static key!
4. Inefficient construction.

Leakage-Resilient PRPs

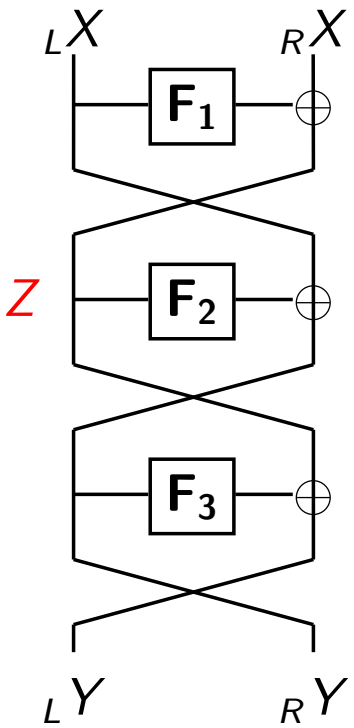
Indistinguishability & Indifferentiabiliyt



Theorem ([LubyRackoff'88])

3-round Feistel instantiated with PRFs is a PRP.

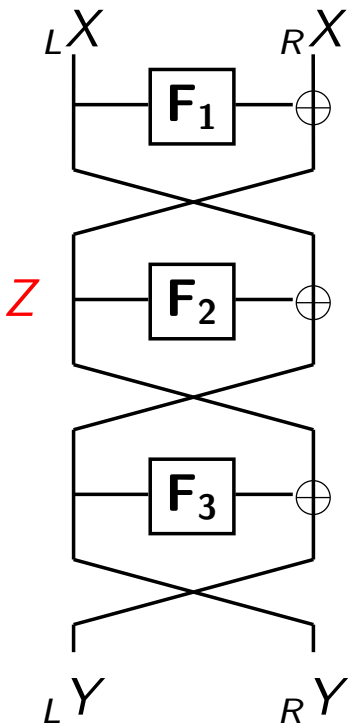
Indistinguishability & Indifferentiabiliyt



Theorem ([LubyRackoff'88])

3-round Feistel instantiated with PRFs is a PRP.

Indistinguishability & Indifferentiabiliyt



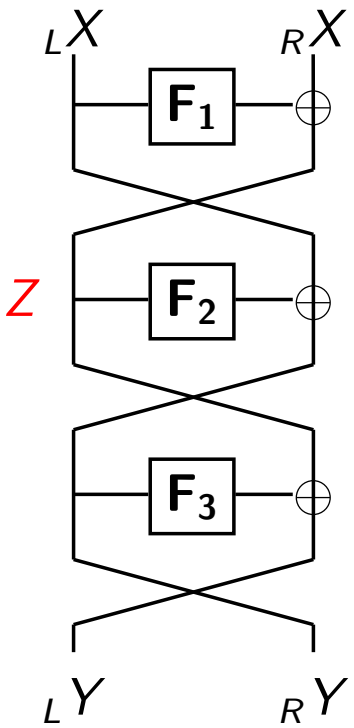
Theorem ([LubyRackoff'88])

3-*round Feistel instantiated with PRFs is a PRP.*

Theorem ([HolensteinKuenzlerTessaro'11])

18-*round Feistel instantiated with URFs is indifferentiable from a URP.*

Indistinguishability & Indifferentiabiliyt



Theorem ([LubyRackoff'88])

3-round Feistel instantiated with PRFs is a PRP.

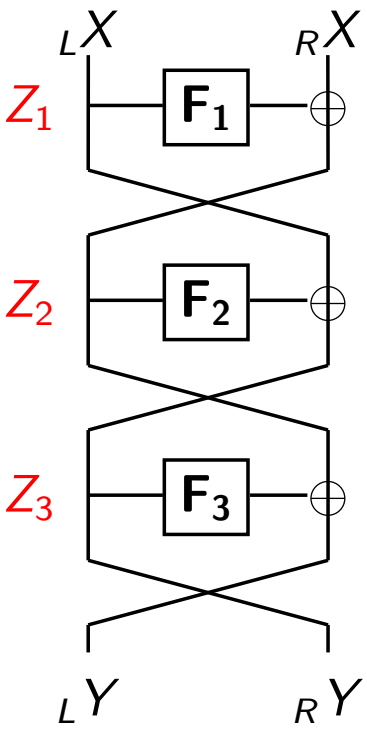
Theorem ([HolensteinKuenzlerTessaro'11])

18-round Feistel instantiated with URFs is indifferentiable from a URP.

Theorem ([DodisP'10])

r-round Feistel instantiated with leakage-resilient PRFs is a secure leakage-resilient super PRP for q-query distinguishers satisfying $q \leq 1.38^{r/2-1}$.

Side-Channel Attacks on Feistel

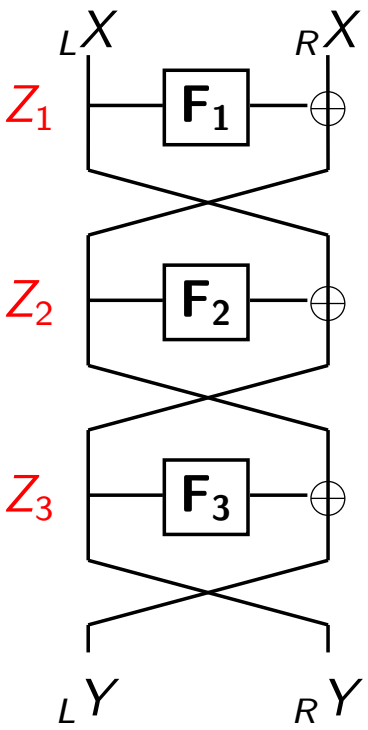


Ψ_r : r-round Feistel instantiated with uniformly random functions $\{0,1\}^n \to \{0,1\}^n$.

Theorem ([DodisP'10])

Can invert Ψ_r on any value Y making $4n^r$ forward queries.
If given $|Z_1|_1, \ldots, |Z_n|_1$ with every query.

Side-Channel Attacks on Feistel



Ψ_r : r-round Feistel instantiated with uniformly random functions $\{0,1\}^n \to \{0,1\}^n$.

Theorem ([DodisP'10])

Can invert Ψ_r on any value Y making $4n^r$ forward queries.
If given $|Z_1|_1, \ldots, |Z_n|_1$ with every query.

- Works for other leakages (than Hamming weight) of the Z_i's.

getting LR PRFs is hard
what to do?

- Use algebraic PRFs, e.g. $f(x) = g^{a_0 \prod_{x_i=1} a_i}$ [NaorReingold'97]. Can use blinding to protect.

- Use algebraic PRFs, e.g. $f(x) = g^{a_0 \prod_{x_i=1} a_i}$ [NaorReingold'97]. Can use blinding to protect.
- Avoid PRFs! Use algebraic MACs [DodKilPieWic'12] like LaPiN

A Proposal: LaPiN [HeyKilLyuPaaP FSE'12]

Ring [1] $\mathsf{R} = \mathcal{F}_2[X]/(f)$

Prover		Verifier
	$\xleftarrow{c}$	Random challenge $c \in \{0,1\}^{80}$
Chose $r, e \in \mathsf{R}$		
$z = r \cdot (k \cdot \pi(c) + \hat{k}) + e \in \mathsf{R}$	$\xrightarrow{r,z}$	$\hat{e} = z - r \cdot (k \cdot \pi(c) + \hat{k})$
		Accept if $\hat{e}$ is a small element in ring R.

- Key are two ring elements $k, \hat{k}$ (621 bits each)
- Share $k = k_0 \cdot k_1, \hat{k} = \hat{k}_0 \cdot \hat{k}_1$
- Run protocol using $(k_i, \hat{k}_i)$ for $i \in \{0,1\}$, combine at the end.
- Occasionally refresh $k_0 \leftarrow k_0 \cdot z$, $k_1 \leftarrow k_1 \cdot z^{-1}$.

[1] $f(X) = (X^{127} + X^8 + X^7 + X^3 + 1)(X^{126} + X^9 + X^6 + X^5 + 1)(X^{125} + X^9 + X^7 + X^4 + 1)(X^{122} + X^7 + X^4 + X^3 + 1)(X^{121} + X^8 + X^5 + X^1 + 1)$

Auxiliary Input Security?

Auxiliary Input vs Bounded Leakage: A Conjecture

Adversary gets

Bounded Leakage: $f(key)$, $|f(key)| \leq \ell$.

Auxiliary Input: $f(key)$, key is hard to compute given $f(key)$.

Auxiliary Input vs Bounded Leakage: A Conjecture

Adversary gets

Bounded Leakage: $f(key)$, $|f(key)| \leq \ell$.

Auxiliary Input: $f(key)$, key is hard to compute given $f(key)$.

Is Aux. Input really stronger than bounded leakage *in practice*?

Does there exist a *natural* scheme that is secure against bounded leakage, but not auxiliary input (which does not trivially contradict the bounded leakage bound)?

Auxiliary Input vs Bounded Leakage: A Conjecture

Adversary gets

Bounded Leakage: $f(key)$, $|f(key)| \leq \ell$.

Auxiliary Input: $f(key)$, key is hard to compute given $f(key)$.

Is Aux. Input really stronger than bounded leakage *in practice*?

Does there exist a *natural* scheme that is secure against bounded leakage, but not auxiliary input (which does not trivially contradict the bounded leakage bound)?

RO analogy

Does there exist a *natural* scheme that is secure in the random oracle model, but not if the RO is replaced with, say SHA3.

Questions?