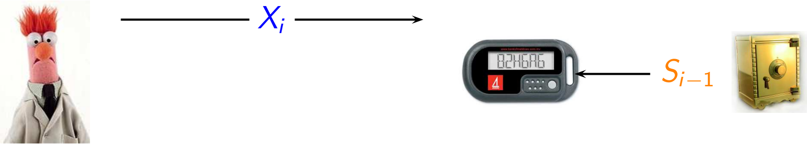
On leakage-resilient pseudorandom functions

Krzysztof Pietrzak

CWI Centrum Wiskunde & Informatica

crypto in the clouds workshop, MIT, Aug.3-5 2009

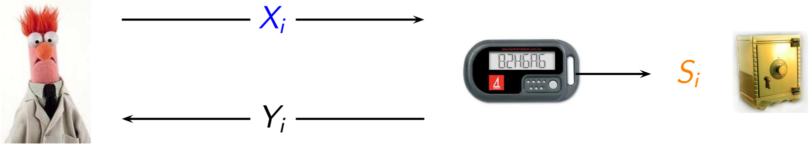
Side-Channel attacks



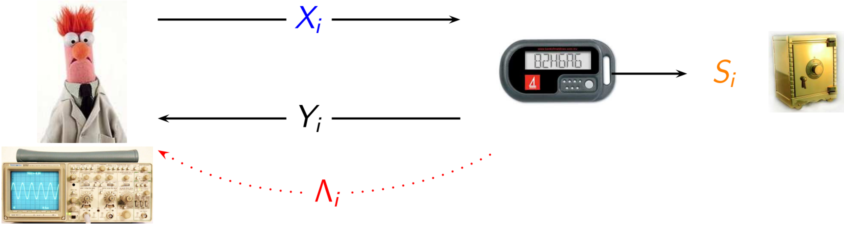
X_i

S_{i-1}

Side-Channel attacks



Side-Channel attacks



- Adversary measures leakage $\Lambda_1, \Lambda_2, \ldots$ on each invocation.

Leakage-Resilience [DP08]

Leakage $\Lambda_i = f_i(X_i, S_{i-1})$. Leakage function f_i adaptively chosen before ith invocation, under following restrictions

Leakage-Resilience [DP08]

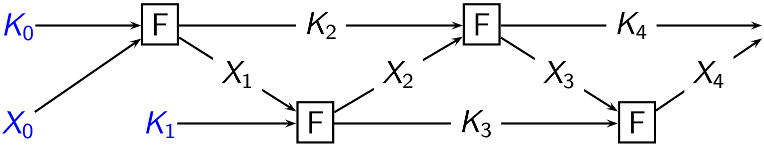
Leakage $\Lambda_i = f_i(X_i, S_{i-1})$. Leakage function f_i adaptively chosen before ith invocation, under following restrictions

- Bounded leakage: $|\Lambda_i| = \lambda$ for some $\lambda \ll |S|$.
- Efficient: $f_i(.)$ must be efficient [MR03 Ax5].
- Only computation leaks information [MR03 Ax1]: $\Lambda_i = f_i(X_i, S^+_{i-1})$ $S^+_{i-1} \subseteq S_{i-1}$ is state that is *accessed* during ith invocation.

Leakage-Resilience [DP08]

Leakage $\Lambda_i = f_i(X_i, S_{i-1})$. Leakage function f_i adaptively chosen before ith invocation, under following restrictions

- Bounded leakage: $|\Lambda_i| = \lambda$ for some $\lambda \ll |S|$.
- Efficient: $f_i(.)$ must be efficient [MR03 Ax5].
- Only computation leaks information [MR03 Ax1]: $\Lambda_i = f_i(X_i, S^+_{i-1})$ $S^+_{i-1} \subseteq S_{i-1}$ is state that is *accessed* during ith invocation.



Leakage-Resilience [DP08]

Leakage $\Lambda_i = f_i(X_i, S_{i-1})$. Leakage function f_i adaptively chosen before ith invocation, under following restrictions

- Bounded leakage: $|\Lambda_i| = \lambda$ for some $\lambda \ll |S|$.
- Efficient: $f_i(.)$ must be efficient [MR03 Ax5].
- Only computation leaks information [MR03 Ax1]: $\Lambda_i = f_i(X_i, S^+_{i-1})$ $S^+_{i-1} \subseteq S_{i-1}$ is state that is *accessed* during ith invocation.

Impossible to get *stateless* primitives. Open how to get PRFs (block-cipher). Recently (Standaert et al. eprint 2009/341) proposed the following additional restriction.

Leakage-Resilience [DP08]

Leakage $\Lambda_i = f_i(X_i, S_{i-1})$. Leakage function f_i adaptively chosen before ith invocation, under following restrictions

- Bounded leakage: $|\Lambda_i| = \lambda$ for some $\lambda \ll |S|$.
- Efficient: $f_i(.)$ must be efficient [MR03 Ax5].
- Only computation leaks information [MR03 Ax1]: $\Lambda_i = f_i(X_i, S^+_{i-1})$ $S^+_{i-1} \subseteq S_{i-1}$ is state that is *accessed* during ith invocation.

Impossible to get *stateless* primitives. Open how to get PRFs (block-cipher). Recently (Standaert et al. eprint 2009/341) proposed the following additional restriction.

Leakage-Resilience [DP08]

Leakage $\Lambda_i = f_i(X_i, S_{i-1})$. Leakage function f_i adaptively chosen before ith invocation, under following restrictions

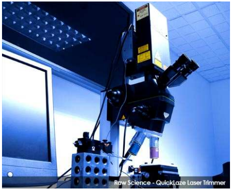
- Bounded leakage: $|\Lambda_i| = \lambda$ for some $\lambda \ll |S|$.
- Efficient: $f_i(.)$ must be efficient [MR03 Ax5].
- Only computation leaks information [MR03 Ax1]: $\Lambda_i = f_i(X_i, S_{i-1}^+)$ $S_{i-1}^+ \subseteq S_{i-1}$ is state that is *accessed* during ith invocation.

Impossible to get *stateless* primitives. Open how to get PRFs (block-cipher). Recently (Standaert et al. eprint 2009/341) proposed the following additional restriction.

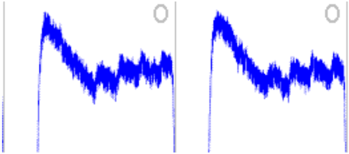
- Non-adaptive leakage function [MR03 Ax4.($\neg$Ax3.)]: For some fixed $f(.)$
$$f_i(.) = f(.)$$
- Partition an invocation into > 1 parts and assume each part leaks independently.

On non-adaptive leakage

Non-adaptive leakage does not protect against probing



Good enough against most side-channels like power-analysis, timing, electromagnetic radiation...



Definition of leakage-resilient PRF

1. Security definition for PRFs is indistinguishability: Adversary $\mathcal{A}$ gets access to either $F(K,.)$ (for random K) or a URF $R(.)$ and must guess which is the case.

Definition of leakage-resilient PRF

1. Security definition for PRFs is indistinguishability: Adversary $\mathcal{A}$ gets access to either $F(K,.)$ (for random K) or a URF $R(.)$ and must guess which is the case.
2. Generalize to leakage setting: $\mathcal{A}$ gets either

$$F(K,.) + \mathrm{leakage} \qquad \text{or} \qquad R(.) + \mathrm{leakage}$$

But what should leakage be? We could use a simulation based definition [HMR08], but that's not what we do.

Definition of leakage-resilient PRF

1. Security definition for PRFs is indistinguishability: Adversary $\mathcal{A}$ gets access to either $F(K,.)$ (for random K) or a URF $R(.)$ and must guess which is the case.
2. Generalize to leakage setting: $\mathcal{A}$ gets either

   $$F(K,.) + \text{leakage} \qquad \text{or} \qquad R(.) + \text{leakage}$$

   But what should leakage be? We could use a simulation based definition [HMR08], but that's not what we do.
3. We let $\mathcal{A}$ query $\quad F(K,.) + \text{leakage}$

   and then $F(K,.)$ must look pseudorandom on all inputs that $\mathcal{A}$ did not yet query (without further leakage).

Definition of leakage-resilient PRF

1. Security definition for PRFs is indistinguishability: Adversary $\mathcal{A}$ gets access to either $F(K,.)$ (for random K) or a URF $R(.)$ and must guess which is the case.
2. Generalize to leakage setting: $\mathcal{A}$ gets either

   $$F(K,.) + \text{leakage} \qquad \text{or} \qquad R(.) + \text{leakage}$$

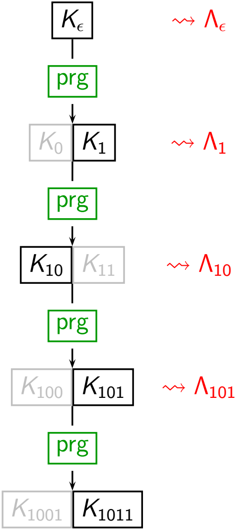
   But what should leakage be? We could use a simulation based definition [HMR08], but that's not what we do.
3. We let $\mathcal{A}$ query $F(K,.) + \text{leakage}$

   and then $F(K,.)$ must look pseudorandom on all inputs that $\mathcal{A}$ did not yet query (without further leakage).
4. Alternative, $\mathcal{A}$ gets

   $$F(K,.) + \text{leakage} \qquad \text{or} \qquad R(.) + \text{leakage}$$

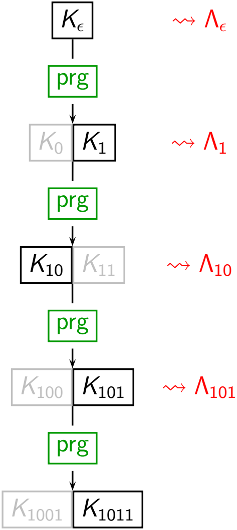
   leakage does not contain the leakage of the last "step".

The GGM construction


K_ϵ
$\rightsquigarrow \Lambda_\epsilon$
prg
K_0
K_1
$\rightsquigarrow \Lambda_1$
prg
K_{10}
K_{11}
$\rightsquigarrow \Lambda_{10}$
prg
K_{100}
K_{101}
$\rightsquigarrow \Lambda_{101}$
prg
K_{1001}
K_{1011}

- GGM84: PRF $F : \{0,1\}^n \times \{0,1\}^n \rightarrow \{0,1\}^m$ from PRG $\{0,1\}^n \rightarrow \{0,1\}^{2n}$.
- $F_{K_\epsilon}(s) = K_s$ where $K_{s0} \| K_{s1} = \mathrm{prg}(K_s)$.

The GGM construction



K_ϵ $\rightsquigarrow \Lambda_\epsilon$

prg

K_0 K_1 $\rightsquigarrow \Lambda_1$

prg

K_{10} K_{11} $\rightsquigarrow \Lambda_{10}$

prg

K_{100} K_{101} $\rightsquigarrow \Lambda_{101}$

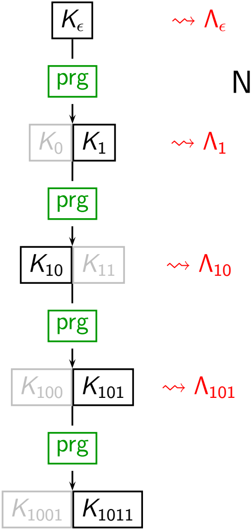
prg

K_{1001} K_{1011}

Leakage:

- Not leakage resilient (can learn λ different bits of K_ϵ with each invocation).

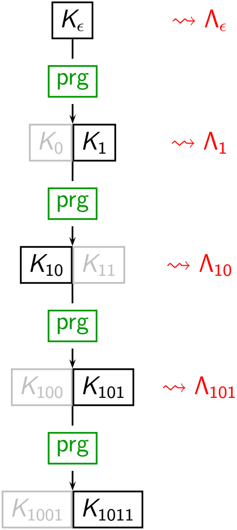
The GGM construction


K_ϵ
$\rightsquigarrow \Lambda_\epsilon$
prg
K_0
K_1
$\rightsquigarrow \Lambda_1$
prg
K_{10}
K_{11}
$\rightsquigarrow \Lambda_{10}$
prg
K_{100}
K_{101}
$\rightsquigarrow \Lambda_{101}$
prg
K_{1001}
K_{1011}

Non adaptive leakage:

- Fixed leakage function $f : \{0,1\}^n \rightarrow \{0,1\}^\lambda$.
- "Only computation leaks information axiom": each invocation of the PRG leaks independently.
- On query $F_{K_\epsilon}(s)$ leaks $\Lambda_{s'} = f(K_{s'})$ for every prefix s' of s.
- Additional restrictions in [SPYQYO09]
  - prg is a random oracle.
  - f may not query the RO.

The GGM construction

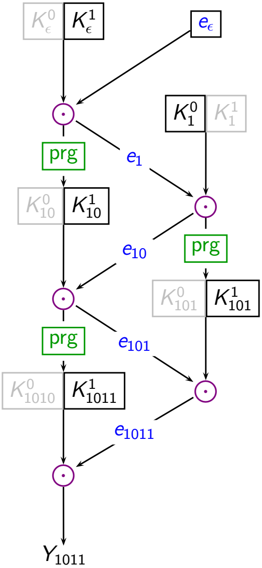

K_ϵ
$\rightsquigarrow \Lambda_\epsilon$
prg
K_0
K_1
$\rightsquigarrow \Lambda_1$
prg
K_{10}
K_{11}
$\rightsquigarrow \Lambda_{10}$
prg
K_{100}
K_{101}
$\rightsquigarrow \Lambda_{101}$
prg
K_{1001}
K_{1011}

- This talk: PRF secure against non-adaptive leakage in the standard model, i.e. avoid assumptions:

1. prg is a random oracle.
2. f may not query the RO.

- (1) is used to argue that $prg(K_s)$ is uniform even given $f(K_s)$.
- (2) is used to avoid "pre-computation": $f(K_s)$ is independent of $f(K_{s\|t})$ for any $t \neq \emptyset$.

Leakage-Resilient PRF 1st Construction

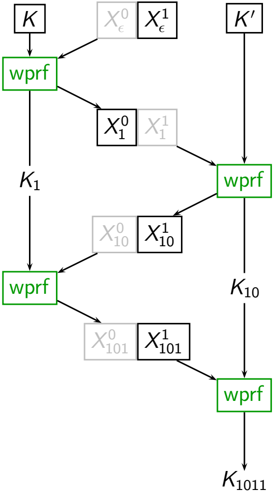


- prg : $[n] \rightarrow [2k]$
- Strong extractor $\odot$: $[s] \times [k] \rightarrow [n]$
- Similar to leakage-resilient stream-cipher form Dziembowski-P (FOCS'08)

PRF secure against non-adaptive leakage

- $F : [2k + s] \times [m] \rightarrow [n]$
- $F_{K_\epsilon^0, K_\epsilon^1, e_\epsilon}(1011) = Y_{1011}$

Leakage-Resilient PRF 2nd Construction

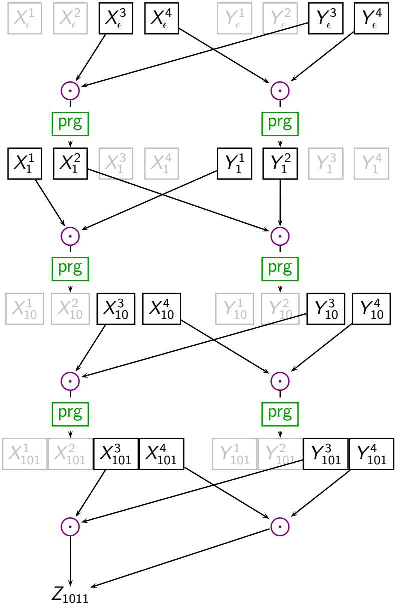


- weak PRF wprf : $[k] \times [2n] \rightarrow [k + 2n]$
- Similar to leakage-resilient mode of operation form Eurocrypt'09

PRF secure against non-adaptive leakage

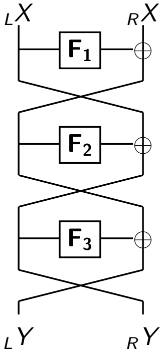
- $F : [2k + s] \times [m] \rightarrow [n]$
- $F_{K,K',X_\epsilon^0,X_\epsilon^1}(1011) = K_{1011}$

Leakage-Resilient PRF 3rd Construction


X_ϵ^1 X_ϵ^2 X_ϵ^3 X_ϵ^4
Y_ϵ^1 Y_ϵ^2 Y_ϵ^3 Y_ϵ^4
prg
prg
X_1^1 X_1^2 X_1^3 X_1^4
Y_1^1 Y_1^2 Y_1^3 Y_1^4
prg
prg
X_{10}^1 X_{10}^2 X_{10}^3 X_{10}^4
Y_{10}^1 Y_{10}^2 Y_{10}^3 Y_{10}^4
prg
prg
X_{101}^1 X_{101}^2 X_{101}^3 X_{101}^4
Y_{101}^1 Y_{101}^2 Y_{101}^3 Y_{101}^4
Z_{1011}

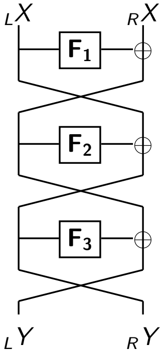
- prg : $[n] \rightarrow [4k]$
- strong 2-source extractor
  $\odot : [k] \times [k] \rightarrow [n]$
- $F_{X_\epsilon^1,\ldots,X_\epsilon^4,Y_\epsilon^1,\ldots,Y_\epsilon^4}(1011) = Z_{1011}$

PRP secure against non-adaptive leakage



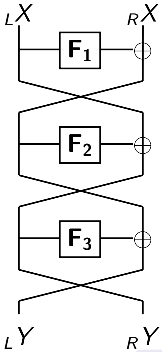
- [SPYQYO09]: "*Eventually, using our PRF in the standard Feistel network of Luby and Rackoff, we can build leakage resilient PRPs.*"

PRP secure against non-adaptive leakage



- [SPYQYO09]: "*Eventually, using our PRF in the standard Feistel network of Luby and Rackoff, we can build leakage resilient PRPs.*"
- Unfortunately that's wrong: from the leakage of $F_2(A)$ and $F_2(B)$ one can determine the length of the common prefix of A and B.

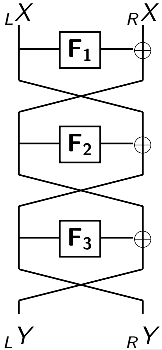
PRP secure against non-adaptive leakage



- [SPYQYO09]: "*Eventually, using our PRF in the standard Feistel network of Luby and Rackoff, we can build leakage resilient PRPs.*"
- Unfortunately that's wrong: from the leakage of $F_2(A)$ and $F_2(B)$ one can determine the length of the common prefix of A and B.

Reductions in the leakage setting are tricky [MR03] and standard cryptographic reductions often fail in this setting.

PRP secure against non-adaptive leakage



- [SPYQYO09]: "*Eventually, using our PRF in the standard Feistel network of Luby and Rackoff, we can build leakage resilient PRPs.*"
- Unfortunately that's wrong: from the leakage of $F_2(A)$ and $F_2(B)$ one can determine the length of the common prefix of A and B.

Reductions in the leakage setting are tricky [MR03] and standard cryptographic reductions often fail in this setting.

- But "indifferentiability like" reductions [MRH04,CDMP05,DP07,CPS08] seems enough for non-adaptive leakage-resilience!

Some Open Problems

Some Open Problems

- (adaptive) leakage-resilient PRF (under standard assumption / under minicrypt assumption).

Some Open Problems

- (adaptive) leakage-resilient PRF (under standard assumption / under minicrypt assumption).
  - Any (adaptive) leakage-resilient PRF must be stateful.

Some Open Problems

- (adaptive) leakage-resilient PRF (under standard assumption / under minicrypt assumption).
    - Any (adaptive) leakage-resilient PRF must be stateful.
    - In [KP09] we construct a leakage-resilient weak PRF in the generic group model or making a somewhat falsifiable [Naor'03] conjecture (which quantifies over all leakage functions).

Some Open Problems

- (adaptive) leakage-resilient PRF (under standard assumption / under minicrypt assumption).
    - Any (adaptive) leakage-resilient PRF must be stateful.
    - In [KP09] we construct a leakage-resilient weak PRF in the generic group model or making a somewhat falsifiable [Naor'03] conjecture (which quantifies over all leakage functions).
- Public-Key encryption secure against *non-adaptive* leakage in standard model?

Questions?