Survey on Different Leakage Models

Krzysztof Pietrzak

crypto in the clouds, MIT Boston, Aug. 4th 2009
Provable security is a big success story. Last 30 years: Strong security notions & matching constructions for all important primitives.

Security notions (mostly from mid 80ies) consider security game where cryptosystem is an idealized black-box.

This notion do not capture “physical attacks” that became more relevant in the last 1-2 decades.

- **Side-channell attacks** are a thread to leight-weight devices (RFIDs, smart-cards).
- **Malware attacks** (viruses, Trojans) are a thread for heighly connected (i.e. over the Internet).
Proveably Secure!
Proveably Secure!

Side-Channel Attacks

Malware
viruses/trojans

Side-Channel Attacks

Proveably Secure!

Krzysztof Pietrzak
Survey on Different Leakage Models
In the physical world the adversary can attack an implementation.
In the physical world the adversary can attack an implementation.

Possibly can extract information $\Lambda$. 
In the physical world the adversary can attack an implementation.

Possibly can extract information $\Lambda$.

To get secure implementations we need security notions which take leakage $\Lambda$ into account.
In the physical world the adversary can attack an implementation. Possibly can extract information $\Lambda$. To get secure implementations we need security notions which take leakage $\Lambda$ into account. Leakage $\Lambda = f(S, R, X)$ is a function of secret state $S$, input $X$ and random coins $R$. 
Outline of this talk

Part 1: Particular leakage functions.
- Exposure Resilience (against cold-boot attacks).
- Private Circuits (against probing attacks).

Part 2: General leakage functions.
- Memory attacks, aka. bounded leakage.
- Auxiliary Input.
- Bounded leakage & Auxiliary Input are incomparable.

Part 3: Unbounded leakage.
- Bounded-Retrieval Model (against malware).
- Leakage-Resilience (against side-channel attacks).
- Extensions/Restrictions of leakage-resilience.
Not in this talk

- Anything on **active attacks** (fault attacks, tamper-resistance).
- **Proactive-Security, Forward-Security, Intrusion-Resilience, Crypto without “perfect shredding”** [CEGL08], one-time programs [GTKR08],.
- **1000+ papers from a practical perspective** (e.g. anything from CHES).
Part 1: Particular leakage functions

- Exposure Resilience (against cold-boot attacks).
- Private Circuits (against probing attacks).
cold-boot attacks

...the attack relies on the data remanence property of DRAM and SRAM to retrieve memory contents which remain readable in the seconds to minutes after power has been removed.

1 J. Alex Halderman, Seth D. Schoen, Nadia Heninger, William Clarkson, William Paul, Joseph A. Calandrino, Ariel J. Feldman, Jacob Appelbaum, and Edward W. Felten
Lest We Remember: Cold Boot Attacks on Encryption Keys.
In *USENIX* 2008.
Exposure-Resilient Cryptography


- Leakage $\Lambda = f(M)$ are some bits of memory $M$. 

---

Krzysztof Pietrzak  Survey on Different Leakage Models
Exposure-Resilient Cryptography

1. B. Chor, O. Goldreich, J. Håstad, J. Friedman, S. Rudich, R. Smolensky
   The Bit Extraction Problem of $t$-Resilient Functions
   In *FOCS* 1985.

   Exposure-resilient functions and all-or-nothing transforms
   In *EUROCRYPT* 2000.

- Leakage $\Lambda = f(M)$ are some bits of memory $M$.
- Don’t keep secret $S$ in plain on memory but encode using
  “$t$-resilient function” $g$

  $$EN\text{C}(S) = [R, g(R) \oplus S] \quad R \text{ random}$$

  $g(.)$ is $t$-resilient means $g(R)$ is uniform even when given
  $t$ bits of $R$. 
Y. Ishai, A. Sahai, and D. Wagner.
Private Circuits: Securing Hardware against Probing Attacks
In CRYPTO 2003.

Leakage $\Lambda = f(S)$ are the values carried by any $t$ wires of a circuit $C(S)$. 
Y. Ishai, A. Sahai, and D. Wagner.
Private Circuits: Securing Hardware against Probing Attacks
In *CRYPTO* 2003.

- Leakage $\Lambda = f(S)$ are the values carried by any $t$ wires of a circuit $C(S)$.
- For any $t \in \mathbb{N}$, show how to transform any circuit $C(.)$ into a circuit $C_t(.)$ such that
  1. $\forall S : C_t(S) = C(S)$
  2. Value on any $t$ wires of $C_t(S)$ are independent of $S$. 
Y. Ishai, A. Sahai, and D. Wagner.
Private Circuits: Securing Hardware against Probing Attacks
In CRYPTO 2003.

Leakage $\Lambda = f(S)$ are the values carried by any $t$ wires of a circuit $C(S)$.

For any $t \in \mathbb{N}$, show how to transform any circuit $C(.)$ into a circuit $C_t(.)$ such that

1. $\forall S : C_t(S) = C(S)$
2. Value on any $t$ wires of $C_t(S)$ are independent of $S$.

Uses techniques from general multiparty computation. Big blowup, $|C_t| \approx t^2|C|$.
Part 2: General leakage functions

- Memory attacks, aka. bounded leakage.
- Auxiliary Input.
- Bounded leakage & Auxiliary Input are incomparable.
Leakage $\Lambda = f(S)$ where

$$f : \{0, 1\}^* \rightarrow \{0, 1\}^\lambda$$

can be any (adversarially chosen) function with bounded range (to $\lambda \in \mathbb{N}$ bits).

Can extend any standard security notion (ind-CPA/CCA, unforgeability) by additionally assuming that the adversary gets leakage $\Lambda$. 

Example: Signatures

Observation (Every sig. scheme is secure against memory attacks with security loss exponential in $\lambda$.)

If signature scheme $\text{Sig}$ cannot be forged with advantage $\epsilon$, then it cannot be forged with advantage $\epsilon \cdot 2^\lambda$ in a $\lambda$-memory attack (by adversaries of the same size).

- Similar results hold for encryption schemes, weak PRFs, but not for PRFs, PRGs.
- Is the exponential loss necessary? Yes in general.
- Next slide: *particular* constructions of signature/encryption schemes where the security does not degrade with $\lambda$ (and $\lambda$ can be as big as a constant fraction of the key-length).
1. Adi Akavia, Shafi Goldwasser, Vinod Vaikuntanathan
   Simultaneous Hardcore Bits and Cryptography against Memory Attacks
   *TCC’09*

2. M. Naor, G. Segev
   Public-Key Cryptosystems Resilient to Key Leakage
   *Crypto’09*.

3. J. Katz, V. Vaikuntanathan
   Signature schemes with bounded leakage resilience
   *Eprint 2009/220*.

4. Joel Alwen and Yevgeniy Dodis and Daniel Wichs
   Public Key Cryptography in the Bounded Retrieval Model and Security Against Side-Channel Attacks
   *Crypto’09*
Let \( \text{Sig} \) be a signature scheme constructed via Fiat-Shamir transform from a witness-indistinguishable \( \Sigma \)-protocol where each \( pk \) corresponds to exponentially many (say \( 2^m \)) different \( sk \) (e.g. Okamoto).

**Theorem (KV09,ADW09 informal)**

*If \( \text{Sig} \) cannot be forged with advantage \( \epsilon \), then it cannot be forged with advantage \( \epsilon \) even in \( \lambda \)-memory attack where \( \lambda \) is almost \( m \).*

Can choose \( m \) as large as \( (1 - \delta)|sk| \) for any \( \delta > 0 \). Then no exponential degradation in security (in fact, no degradation at all) even if almost all the key is leaked.
Leakage $\Lambda = f(S)$ where $f : \{0, 1\}^* \rightarrow \{0, 1\}^\lambda$ can be any function that is exponentially hard to invert.

$$\exists \alpha > 0 \ \forall \text{PPT } A \ \exists n' \ \forall n > n' : \Pr_{x \leftarrow \{0,1\}^n} [A(f(x)) = x] \leq 2^{-\alpha \cdot n}$$

1. Yevgeniy Dodis, Yael Tauman Kalai and Shachar Lovett
   On Cryptography with Auxiliary Input
   *STOC*’09

2. Y. Kalai and V. Vaikuntanathan
   Public-key Encryption Schemes with Auxiliary Inputs and Applications
Leakage $\Lambda = f(S)$ where $f : \{0, 1\}^* \rightarrow \{0, 1\}^\lambda$ can be any function that is exponentially hard to invert.

$$\exists \alpha > 0 \ \forall \text{PPT} \ A \ \exists n' \ \forall n > n' : \Pr_{x \leftarrow \{0,1\}^n} [A(f(x)) = x] \leq 2^{-\alpha \cdot n}$$

1. Yevgeniy Dodis, Yael Tauman Kalai and Shachar Lovett
   On Cryptography with Auxiliary Input
   *STOC’09*

2. Y. Kalai and V. Vaikuntanathan
   Public-key Encryption Schemes with Auxiliary Inputs and Applications

Learing Paritiy with noise: $\{A, Ax + e\} \equiv \{A, U\}$
$A \in_R \{0, 1\}^{t \times n}$, $x \in_R \{0, 1\}^n$ and $e$ is a error vector.
Leakage $\Lambda = f(S)$ where $f : \{0, 1\}^* \rightarrow \{0, 1\}^\lambda$ can be any function that is exponentially hard to invert

$$\exists \alpha > 0 \ \forall \text{PPT} \ A \ \exists n' \ \forall n > n' : \Pr_{x \leftarrow \{0,1\}^n} [A(f(x)) = x] \leq 2^{-\alpha \cdot n}$$

1. Yevgeniy Dodis, Yael Tauman Kalai and Shachar Lovett
   On Cryptography with Auxiliary Input
   \textit{STOC'09}

2. Y. Kalai and V. Vaikuntanathan
   Public-key Encryption Schemes with Auxiliary Inputs and Applications

Learning Parity with noise: $\{f(x), A, Ax + e\} \equiv \{f(x), A, U\}$

$A \in_R \{0, 1\}^{t \times n}, x \in_R \{0, 1\}^n$ and $e$ is a error vector.
Bounded leakage vs. Auxiliary input

- Let $f : \{0, 1\}^n \rightarrow \{0, 1\}^\lambda$ (with say $\lambda = n/2$) be any bounded leakage-function.
- Then $f$ is exponentially hard to invert:
  \[
  \Pr_{x \in \{0,1\}^n} \left[ A(f(x)) = x \right] \leq 2^{-H_\infty(x|f(x))} \leq 2^{\lambda-n} = 2^{-n/2}
  \]
Let $f : \{0, 1\}^n \rightarrow \{0, 1\}^\lambda$ (with say $\lambda = n/2$) be any bounded leakage-function.

Then $f$ is exponentially hard to invert

$$\Pr_{x \in \{0,1\}^n} [A(f(x)) = x] \leq 2^{-H_\infty(x|f(x))} \leq 2^{\lambda-n} = 2^{-n/2}$$

What is actually required is not $|f(x)| = \lambda$, but

$$H_\infty(x|f(x)) \geq n - \lambda$$
Let \( f : \{0, 1\}^n \rightarrow \{0, 1\}^\lambda \) (with say \( \lambda = n/2 \)) be any bounded leakage-function.

Then \( f \) is exponentially hard to invert

\[
\Pr_{x \in \{0,1\}^n}[A(f(x)) = x] \leq 2^{-H_\infty(x|f(x))} \leq 2^{\lambda-n} = 2^{-n/2}
\]

What is actually required is not \( |f(x)| = \lambda \), but

\( H_\infty(x|f(x)) \geq n - \lambda \)

And in fact only a computational version of this

\( H_{s,\epsilon}^{HILL}(x|f(x)) \geq n - \lambda \)

**Definition (HILL pseudoentropy [HåstadILL99],[BarakSW03])**

\( X \) has \( HILL \) pseudoentropy \( k \), denoted \( H_{\epsilon,s}^{HILL}(X) \geq k \), if \( \exists Y \) s.t. \( H_\infty(Y) \geq k \) and no \( A \) of size \( s \) can distinguish \( X \) from \( Y \) with advantage \( \epsilon \).
So what we have to compare are $f(.)$ (and say $\lambda = n/2$) where

1. $H_\infty(x|f(x)) \geq n - \lambda$
2. $\Pr_{x \in R\{0,1\}^n}[A(f(x)) = x] \leq 2^{-\alpha n}$ for some $\alpha > 0$
So what we have to compare are \( f(.) \) (and say \( \lambda = n/2 \)) where

1. \( H_{\infty}(x|f(x)) \geq n - \lambda \)
2. \( \Pr_{x \in \{0,1\}^n}[A(f(x)) = x] \leq 2^{-\alpha n} \) for some \( \alpha > 0 \)

Assume \( f(.) \) is an exponentially hard to invert one-way permutation, i.e.

\[
\Pr[A(f(x)) = x] \leq 2^{-\alpha n}
\]

but

\[
H_{\infty}(x|f(x)) = 0
\]
So what we have to compare are $f(.)$ (and say $\lambda = n/2$) where

1. $H_\infty(x|f(x)) \geq n - \lambda$
2. $\Pr_{x \in \{0,1\}^n}[A(f(x)) = x] \leq 2^{-\alpha n}$ for some $\alpha > 0$

Assume $f(.)$ is an exponentially hard to invert one-way permutation, i.e.

$$\Pr[A(f(x)) = x] \leq 2^{-\alpha n}$$

but

$$H_\infty(x|f(x)) = 0$$

So satisfies aux. input (2) but not bounded leakage (1).
So what we have to compare are $f(.)$ (and say $\lambda = n/2$) where

1. $H_\infty(x|f(x)) \geq n - \lambda$
2. $\Pr_{x \in \mathbb{R}\{0,1\}^n}[A(f(x)) = x] \leq 2^{-\alpha n}$ for some $\alpha > 0$

Let $\phi : \{0,1\}^n \rightarrow \{0,1\}^n \cup \bot$

- $\Pr[\phi(x) = x] = 2^{-n^{0.5}}$
- $\Pr[\phi(x) = \bot] = 1 - 2^{-n^{0.5}}$

- $\exists A : \Pr[A(\phi(x)) = x] = 2^{-n^{0.5}} \gg 2^{-\alpha n}$
- $H_\infty(x|\phi(x)) = n$ (with prob. $1 - 2^{-n^{0.5}}$).

So satisfies bounded leakage (1) but not aux. input (2).
Bounded-leakage & Auxiliary input are incompareable
Part 3: Unbounded leakage

- Bounded-Retrieval Model (against malware).
- Leakage-Resilience (against side-channel attacks).
- Bounded-leakage vs. Auxiliary-input against side-channel attacks.
Challenge: protect against malware that (temporarily) controls your computer on which a secret key $sk$ is stored.
Challange: protect against malware that (temporarily) controls your computer on which a secret key $sk$ is stored.

Bounded Retrieval Model: malware has complete control over the computer but can only send out a bounded amount of information (1GB say).
Bounded-Retrieval Model [D06,CLW06, ...]

- Idea, make $sk$ huge (2GB say) and design a scheme that remains secure even when $f(sk)$ is leaked for any $f$ where $|f(sk)| \leq 1\text{GB}$.
- The efficiency of the scheme should only depend on some security parameter $n$ but not on $|sk|$. So can’t simply use schemes secure against memory attacks with huge keys.
Side-Channel attacks

$X_i \rightarrow S_{i-1}$
Side-Channel attacks

\[ X_i \quad Y_i \quad S_i \]
Side-Channel attacks

- $X_i$ 
- $Y_i$ 
- $\Lambda_i$ 
- $S_i$

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

Security against \( \lambda \)-memory attacks insufficient as
\[ |\Lambda_1| + |\Lambda_2| + \ldots \] can be arbitrary large.
Adversary measures leakage $\Lambda_1, \Lambda_2, \ldots$ on each invocation.

Security against $\lambda$-memory attacks insufficient as $|\Lambda_1| + |\Lambda_2| + \ldots$ can be arbitrary large.

Bounded-retrieval model inconvenient (huge keys) and a priori bound on queries.
Side-Channels

- electromagnetic radiation [QuisquaterS01]
- power consumption [KocherJJ99]
- running-time [Kocher96]
- sound [ShamirTromer]

people.csail.mit.edu/tromer/acoustic
Most general leakage model:

\[ \Lambda_i = f(X_i, R_i, S_{i-1}) \]

where \( f(\cdot) \) is an adaptively, adversarially chosen function.

- The \( i \)th input \( X_i \)
- The random coins \( R_i \) used during the \( i \)th invocation.
- The secret internal state \( S_{i-1} \).
Most general leakage model:

\[ \Lambda_i = f(X_i, R_i, S_{i-1}) \]

where \( f(.) \) is an adaptively, adversarially chosen function.

- The \( i \)th input \( X_i \)
- The random coins \( R_i \) used during the \( i \)th invocation.
- The secret internal state \( S_{i-1} \).

Note that \( f(.) \) can compute all internal variables (including the output \( Y_i \) and the state \( S_i \)) from its input.
Most general leakage model:

\[ \Lambda_i = f(X_i, R_i, S_{i-1}) \]

where \( f(. \) is an adaptively, adversarially chosen function.

- The \( i \)th input \( X_i \)
- The random coins \( R_i \) used during the \( i \)th invocation.
- The secret internal state \( S_{i-1} \).

Note that \( f(.) \) can compute all internal variables (including the output \( Y_i \) and the state \( S_i \)) from its input.

This model is clearly too strong, e.g. we can leak the entire internal state \( \Lambda_1 = S_0 \).
Most general leakage model:

\[ \Lambda_i = f(X_i, R_i, S_{i-1}) \]

where \( f(\cdot) \) is an adaptively, adversarially chosen function.
- The \( i \)th input \( X_i \)
- The random coins \( R_i \) used during the \( i \)th invocation.
- The secret internal state \( S_{i-1} \).

Note that \( f(\cdot) \) can compute all internal variables (including the output \( Y_i \) and the state \( S_i \)) from its input.

This model is clearly too strong, e.g. we can leak the entire internal state \( \Lambda_1 = S_0 \).

We must add restrictions on \( f(\cdot) \) which should be

1. **sufficient**: allow for actual leakage-resilient constructions.
2. **general**: should cover almost all side-channel attacks.
Restricting the leakage function $\Lambda_i = f(X_i, R_i, S_{i-1})$

1. **Bounded leakage**: $|\Lambda_i| = \lambda$ for a leakage parameter $\lambda \ll |S|$. 
Modelling Generic Side-Channel Attacks

Restricting the leakage function $\Lambda_i = f(X_i, R_i, S_{i-1})$

1. **Bounded leakage:** $|\Lambda_i| = \lambda$ for a leakage parameter $\lambda \ll |S|$.

2. **Efficient:** $f(.)$ must be efficient [MR03 Ax5].
Restricting the leakage function \( \Lambda_i = f(X_i, R_i, S_{i-1}) \)

1. **Bounded leakage:** \(|\Lambda_i| = \lambda\) for a leakage parameter \(\lambda \ll |S|\).
2. **Efficient:** \(f(.)\) must be efficient [MR03 Ax5].
3. **Only computation leaks information** [MR03 Ax1]:

   \[
   \Lambda_i = f(X_i, R_i, S_{i-1}^+) 
   \]

   where \(S_{i-1}^+ \subseteq S_{i-1}\) is the part of the state that is accessed on the \(i\)th invocation.
Side-Channel Countermeasures Design Process

Currently (exaggerated)

1. Implement primitive.
2. Find a side-channel attack.
3. Find & implement a fix.
Side-Channel Countermeasures Design Process

Currently (exaggerated)

1. Implement primitive.
2. Find a side-channel attack.
3. Find & implement a fix.

Using Leakage-Resilience

1. Consider a general class $\mathcal{F}$ of leakage-functions (cf. next slide).
2. Cryptography: Design a primitive and prove it’s secure against side-channels in $\mathcal{F}$.
3. Engineering: Design hardware whose leakage is in $\mathcal{F}$. 
Side-Channel Countermeasures Design Process

Currently (exaggerated)

1. Implement primitive.
2. Find a side-channel attack.
3. Find & implement a fix.

Using Leakage-Resilience

1. Consider a general class $\mathcal{F}$ of leakage-functions (cf. next slide).
2. Cryptography: Design a primitive and prove it’s secure against side-channels in $\mathcal{F}$.
3. Engineering: Design hardware whose leakage is in $\mathcal{F}$.

Advantage: modular and you can blame someone if it fails.
1. S. Dziembowski and K. P.
   Leakage-Resilient Cryptography (Stream-Cipher in standard model)
   FOCS’08

2. K. P.
   A Leakage-Resilient Mode of Operation
   EUROCRYPT’09

3. E. Kiltz and K. P.
   How to Secure ElGamal against Side-Channel Attacks (PKE in
   generic group model)
   manuscript’09

4. S. Faust, E. Kiltz, K. P and G. Rothblum
   Leakage-Resilient Signatures (standard model)
   eprint 2009/282
Leakage-Resilient Cryptography bibliography

1. S. Dziembowski and K. P.
   Leakage-Resilient Cryptography (Stream-Cipher in standard model)
   \textit{FOCS’08}

2. K. P.
   A Leakage-Resilient Mode of Operation
   \textit{EUROCRYPT’09}

3. E. Kiltz and K. P.
   How to Secure ElGamal against Side-Channel Attacks (PKE in generic group model)
   \textit{manuscript’09}

4. S. Faust, E. Kiltz, K. P and G. Rothblum
   Leakage-Resilient Signatures (standard model)
   \textit{eprint 2009/282}

Open problems: LR block-cipher? LR PKE in standard model? generic compiler (à la private circuits)?
Leakage-resilient primitives are inherently stateful. LR achieved either by key-evolution [1,2,4] or secret-sharing [3].
Leakage-Resilient Signatures

Tree based signatures: use signature-scheme $\text{SIG} = (\text{KG}, \text{Sign}, \text{Vfy})$ that can sign up to 3 messages in a tree mode to get a scheme $\text{SIG}^*$. 

\[
\begin{align*}
(pk_0, sk_0) \quad & \cdots \quad \phi_0 = \text{Sign}(sk_0, 0pk_0) \quad \phi_1 \quad \cdots \quad (pk_1, sk_1) \\
(pk_0, sk_0) \quad & \cdots \quad \phi_{00} \quad \phi_{01} \quad \cdots \quad (pk_{00}, sk_{00}) \quad (pk_{01}, sk_{01}) \quad \phi_{10} \quad \phi_{11} \quad \cdots \quad (pk_{10}, sk_{10}) \quad (pk_{11}, sk_{11})
\end{align*}
\]
Leakage-Resilient Signatures

\[ (pk_\varepsilon, sk_\varepsilon) \ldots \overset{\varepsilon}{\cdots} \]

\[ \phi_0 = \text{Sign}(sk_\varepsilon, 0pk_0) \]

\[ (pk_0, sk_0) \ldots \overset{0}{\cdots} \]

\[ \phi_1 \]

\[ (pk_1, sk_1) \ldots \overset{1}{\cdots} \]

**Theorem**

*If SIG is secure against \( \lambda \)-memory attacks, then SIG* is leakage-resilient where one can leak up to \( \lambda/3 \) bits per invocation.*

- Tree based signatures: use signature-scheme 
  \[ \text{SIG} = (KG, \text{Sign}, \text{Vfy}) \] 
  that can sign up to 3 messages in a tree mode to get a scheme SIG*.
A Leakage-Resilient Mode of Operation [P09]

- \( F : \{0, 1\}^\kappa \times \{0, 1\}^n \rightarrow \{0, 1\}^{\kappa+n} \)
- Secret key is \( K_0, K_1, X_0 \), output is \( X_1, X_2, \ldots \)
- \( i \)'th round: \((K_{i+2}, K_{i+1}) \leftarrow F(K_i, X_i).\)

**Theorem**

This is a leakage resilient stream-cipher if instantiated with any weak PRF \( F \).

- Simpler & more efficient than [Dziembowski-P FOCS’08] where we used PRGs & Extractors.
Adversary can choose leakage functions $f_1, f_2, \ldots$ \textit{adaptively}. In a weaker non-adaptive model (i.e. $f_1 = f_2 = \ldots$) much more seems possible [SPYQYO09]
Adversary can choose leakage functions $f_1, f_2, \ldots$ \textit{adaptively}. In a weaker non-adaptive model (i.e. $f_1 = f_2 = \ldots$) much more seems possible [SPYQYO09]

Potentially can get rid of the “noly computation leaks information” assumption by

1. Low complexity leakage functions.
2. Randomness gates.
Adversary can choose leakage functions $f_1, f_2, \ldots$ *adaptively*. In a weaker non-adaptive model (i.e. $f_1 = f_2 = \ldots$) much more seems possible [SPYQY09]

Potentially can get rid of the “only computation leaks information” assumption by

1. Low complexity leakage functions.
2. Randomness gates.

Next Slide: Leakage-resilience can be seen as a continuous version of security against memory attacks. Can also consider a continuous version of security against auxiliary input.
In side-channel attacks one often measures *lots* of data from which only few bits $X$, $|X| \ll |S|$ are extracted and kept for further analysis.
Bounded Leakage vs. Auxiliary Input in Side-Channel attacks

1. In side-channel attacks one often measures *lots* of data from which only few bits $X$, $|X| \ll |S|$ are extracted and kept for further analysis.

2. If $|X| \leq \lambda$ and system is $\lambda$-leakage resilient we’re fine.
Bounded Leakage vs. Auxiliary Input in Side-Channel attacks

1. In side-channel attacks one often measures *lots* of data from which only few bits $X, |X| \ll |S|$ are extracted and kept for further analysis.

2. If $|X| \leq \lambda$ and system is $\lambda$-leakage resilient we’re fine.

3. Concievable that there are attacks which extract more than $|X| > |S|$ bits per invocation.
In side-channel attacks one often measures *lots* of data from which only few bits $X, |X| \ll |S|$ are extracted and kept for further analysis.

If $|X| \leq \lambda$ and system is $\lambda$-leakage resilient we’re fine.

Conceivable that there are attacks which extract more than $|X| > |S|$ bits per invocation.

Intuitively, we don’t need that $\ll |S|$ bits leak, but only that one can’t compute $S$ from.
In side-channel attacks one often measures *lots* of data from which only few bits $X, |X| \ll |S|$ are extracted and kept for further analysis.

If $|X| \leq \lambda$ and system is $\lambda$-leakage resilient we’re fine.

Conceivable that there are attacks which extract more than $|X| > |S|$ bits per invocation.

Intuitively, we don’t need that $\ll |S|$ bits leak, but only that one can’t compute $S$ from them.

Could instead require that for all efficient $A$

$$\Pr[A(X) = S] \leq 2^{-\alpha n}$$
Questions?