On the Connection between Leakage Tolerance and Adaptive Security

Daniele Venturi

Aarhus University

PKC 2013—Nara

Joint work with Jesper Buus Nielsen and Angela Zottarel

Secure Message Transmission

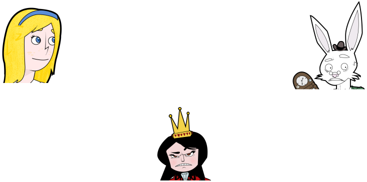
- Secret communication (in a world where public-key crypto exists)

Secure Message Transmission

- Secret communication (in a world where public-key crypto exists)

real world

$m \in \mathcal{M}$

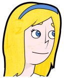


Secure Message Transmission

- Secret communication (in a world where public-key crypto exists)

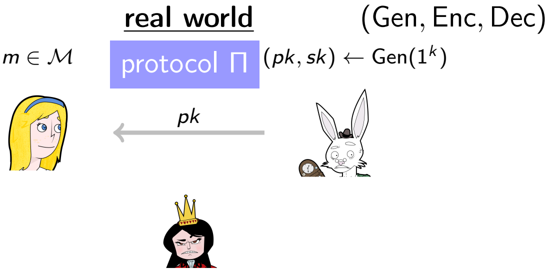
real world $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$

$m \in \mathcal{M}$ protocol Π



Secure Message Transmission

- Secret communication (in a world where public-key crypto exists)

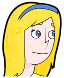


Secure Message Transmission

- Secret communication (in a world where public-key crypto exists)

real world $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$

$m \in \mathcal{M}$ protocol Π $(pk, sk) \leftarrow \mathsf{Gen}(1^k)$



pk ←

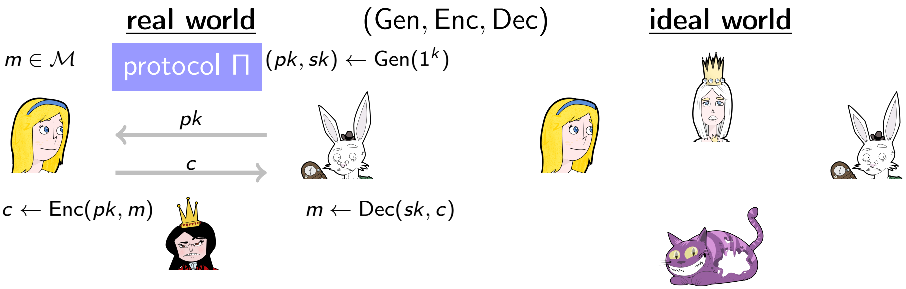
c →

$c \leftarrow \mathsf{Enc}(pk, m)$

$m \leftarrow \mathsf{Dec}(sk, c)$

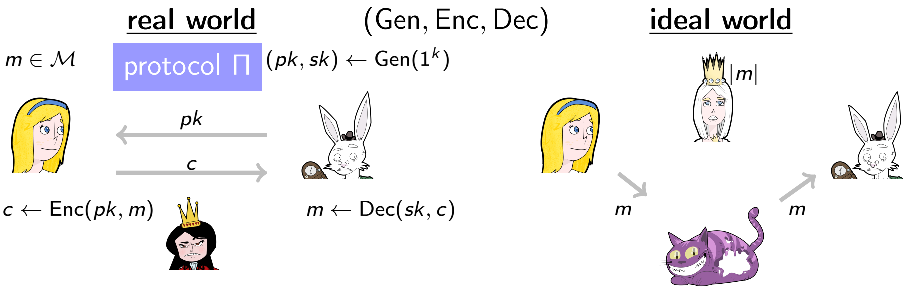
Secure Message Transmission

- Secret communication (in a world where public-key crypto exists)



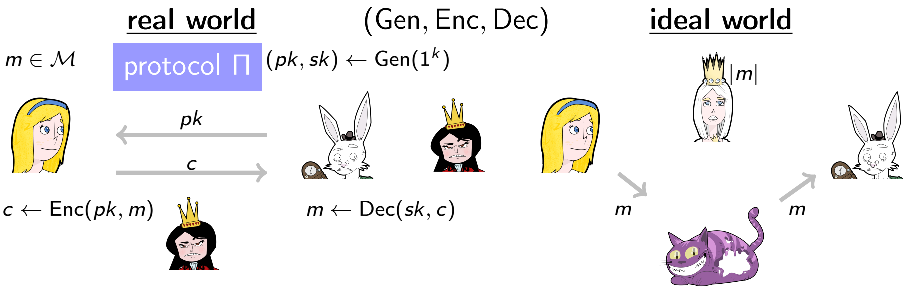
Secure Message Transmission

- Secret communication (in a world where public-key crypto exists)



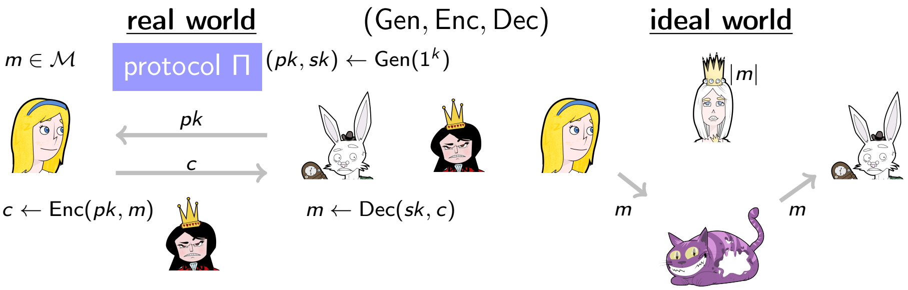
Secure Message Transmission

- Secret communication (in a world where public-key crypto exists)



Secure Message Transmission

- Secret communication (in a world where public-key crypto exists)



Leakage leakage leakage leakage leakage leakage...

- It turns out  can gain partial information on the state of uncorrupted players in a number of ways:

Leakage leakage leakage leakage leakage leakage...

- It turns out  can gain partial information on the state of uncorrupted players in a number of ways:

(Gen, Enc, Dec)

real world

$m \in \mathcal{M}$

protocol Π

Leakage leakage leakage leakage leakage leakage...

- It turns out  can gain partial information on the state of uncorrupted players in a number of ways:

(Gen, Enc, Dec)

real world

$m \in \mathcal{M}$ protocol Π $(pk, sk) \to \mathsf{Gen}(1^k; r_G)$

$\xleftarrow{pk}$

$\xrightarrow{c}$

$c = \mathsf{Enc}(pk, m; r_E)$ $m = \mathsf{Dec}(sk, c; r_D)$

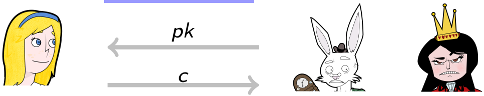
Leakage leakage leakage leakage leakage leakage...

- It turns out  can gain partial information on the state of uncorrupted players in a number of ways:

(Gen, Enc, Dec)

real world

$m \in \mathcal{M}$ protocol Π $(pk, sk) \to \mathsf{Gen}(1^k; r_G)$



$c = \mathsf{Enc}(pk, m; r_E)$ $m = \mathsf{Dec}(sk, c; r_D)$

$\sigma_A = (m, r_E)$ $\sigma_B = (m, sk, r_G, r_D)$

Leakage leakage leakage leakage leakage leakage...

- It turns out  can gain partial information on the state of uncorrupted players in a number of ways:

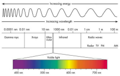
(Gen, Enc, Dec)

real world

$m \in \mathcal{M}$ protocol Π $(pk, sk) \to \mathsf{Gen}(1^k; r_G)$ leakage

pk ←

c →



$c = \mathsf{Enc}(pk, m; r_E)$ $m = \mathsf{Dec}(sk, c; r_D)$

$\sigma_A = (m, r_E)$ $\sigma_B = (m, sk, r_G, r_D)$

Leakage leakage leakage leakage leakage leakage...

- It turns out  can gain partial information on the state of uncorrupted players in a number of ways:

(Gen, Enc, Dec)

real world

$m \in \mathcal{M}$ | protocol Π | $(pk, sk) \rightarrow \text{Gen}(1^k; r_G)$ | leakage

$\xleftarrow{pk}$

$\xrightarrow{c}$

$c = \text{Enc}(pk, m; r_E)$

$\sigma_A = (m, r_E)$

$m = \text{Dec}(sk, c; r_D)$

$\sigma_B = (m, sk, r_G, r_D)$

- Even partial leakage on σ_A or σ_B sufficient to put security of the scheme under attack on edge

Modeling leakage: simulation-based approach

- In the UC framework  could have a very hard life

Modeling leakage: simulation-based approach

- In the UC framework  could have a very hard life
  - e.g., semantic security impossible for a single bit of leakage on m

Modeling leakage: simulation-based approach

- In the UC framework  could have a very hard life
  - e.g., semantic security impossible for a single bit of leakage on m
- Natural fix: Allow to leak on the ideal state

Modeling leakage: simulation-based approach

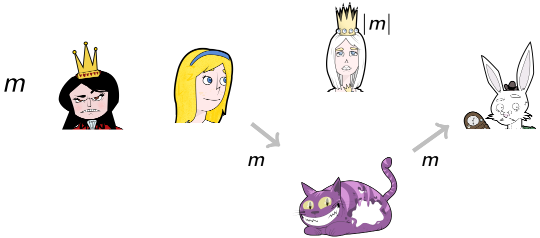
- In the UC framework  could have a very hard life
  - e.g., semantic security impossible for a single bit of leakage on m
- Natural fix: Allow to leak on the ideal state

ideal world

Modeling leakage: simulation-based approach

- In the UC framework  could have a very hard life
  - e.g., semantic security impossible for a single bit of leakage on m
- Natural fix: Allow to leak on the ideal state

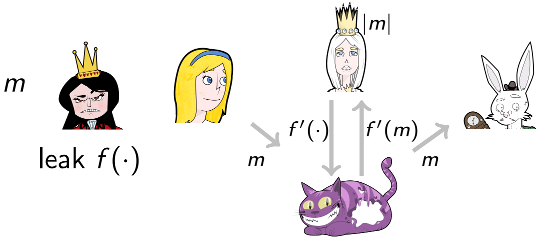
ideal world



Modeling leakage: simulation-based approach

- In the UC framework could have a very hard life
  - e.g., semantic security impossible for a single bit of leakage on m
- Natural fix: Allow to leak on the ideal state

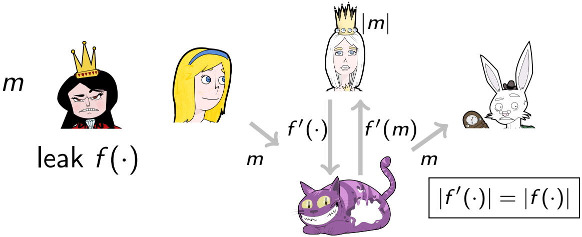
ideal world



Modeling leakage: simulation-based approach

- In the UC framework  could have a very hard life
  - e.g., semantic security impossible for a single bit of leakage on m
- Natural fix: Allow to leak on the ideal state

ideal world



Results from BCH12

- At TCC 2012 Bitanski, Canetti and Halevi defined simulation-based leakage tolerance in the UC framework

Results from BCH12

- At TCC 2012 Bitanski, Canetti and Halevi defined simulation-based leakage tolerance in the UC framework
- Leakage is modeled as a partial form of passive corruption

Results from BCH12

- At TCC 2012 Bitanski, Canetti and Halevi defined simulation-based leakage tolerance in the UC framework
- Leakage is modeled as a partial form of passive corruption
  - Instead of seeing the entire state,  can leak part of it

Results from BCH12

- At TCC 2012 Bitanski, Canetti and Halevi defined simulation-based leakage tolerance in the UC framework
- Leakage is modeled as a partial form of passive corruption
  - Instead of seeing the entire state,  can leak part of it
- Their main result is that passive adaptive security implies leakage tolerance for a large class of functionalities

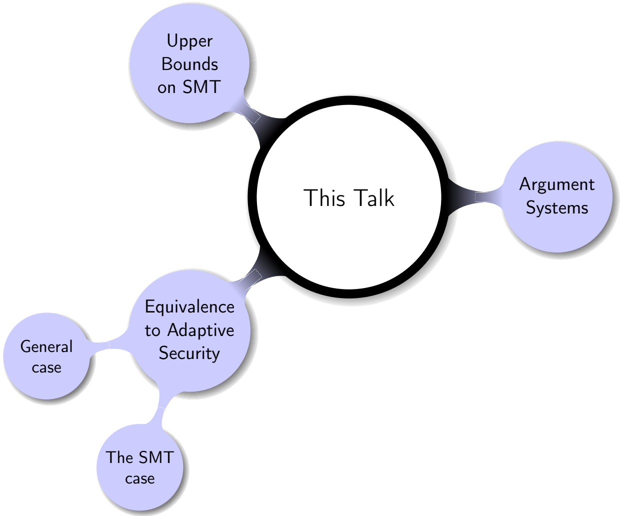
Results from BCH12

- At TCC 2012 Bitanski, Canetti and Halevi defined simulation-based leakage tolerance in the UC framework
- Leakage is modeled as a partial form of passive corruption
  - Instead of seeing the entire state,  can leak part of it
- Their main result is that passive adaptive security implies leakage tolerance for a large class of functionalities
  - Intuition: If  can fake the entire real state, it can also simulate any leakage from it!

Results from BCH12

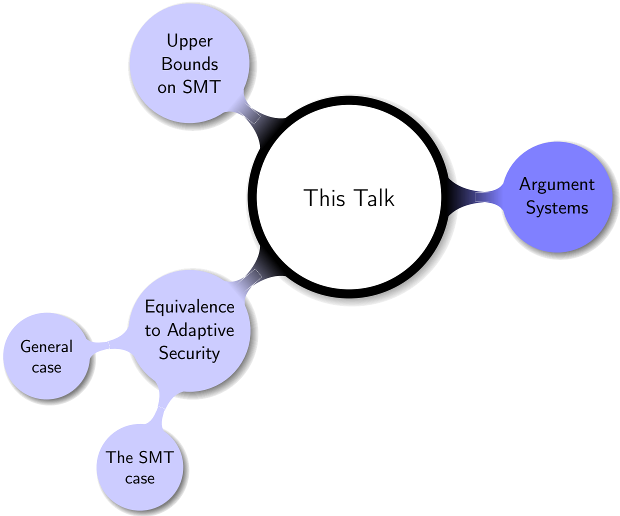
- At TCC 2012 Bitanski, Canetti and Halevi defined simulation-based leakage tolerance in the UC framework
- Leakage is modeled as a partial form of passive corruption
  - Instead of seeing the entire state,  can leak part of it
- Their main result is that passive adaptive security implies leakage tolerance for a large class of functionalities
  - Intuition: If  can fake the entire real state, it can also simulate any leakage from it!
- <u>This work:</u> We look at the other direction

Roadmap



Upper Bounds on SMT
This Talk
Argument Systems
Equivalence to Adaptive Security
General case
The SMT case

Roadmap


Upper Bounds on SMT
This Talk
Argument Systems
Equivalence to Adaptive Security
General case
The SMT case

Arguments of knowledge

- An argument system is an interactive protocol in which  convinces  that some x is in $\mathcal{L} \subset \mathbf{NP}$

Arguments of knowledge

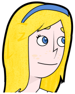
- An argument system is an interactive protocol in which  convinces  that some x is in $\mathcal{L} \subset \mathbf{NP}$

$$\mathcal{L} = \{x : \exists w \text{ s.t. } (x, w) \in \mathcal{R}_{\mathcal{L}}\}$$

Arguments of knowledge

- An argument system is an interactive protocol in which  convinces  that some x is in $\mathcal{L} \subset \mathbf{NP}$

$$\mathcal{L} = \{x : \exists w \text{ s.t. } (x, w) \in \mathcal{R}_{\mathcal{L}}\}$$



(x, w)

x

Arguments of knowledge

- An argument system is an interactive protocol in which  convinces  that some x is in $\mathcal{L} \subset \mathbf{NP}$

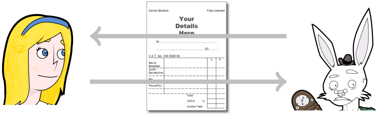
$$\mathcal{L} = \{x : \exists w \text{ s.t. } (x, w) \in \mathcal{R}_{\mathcal{L}}\}$$

(x, w) $\quad$ x

Arguments of knowledge

- An argument system is an interactive protocol in which  convinces  that some x is in $\mathcal{L} \subset \mathbf{NP}$

$$\mathcal{L} = \{x : \exists w \text{ s.t. } (x, w) \in \mathcal{R}_{\mathcal{L}}\}$$



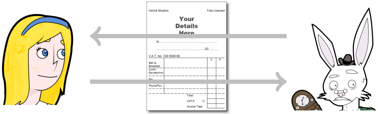
(x, w) x

Output J(x, )

Arguments of knowledge

- An argument system is an interactive protocol in which  convinces  that some x is in $\mathcal{L} \subset \mathbf{NP}$

$$\mathcal{L} = \{x : \exists w \text{ s.t. } (x, w) \in \mathcal{R}_{\mathcal{L}}\}$$



(x, w) x

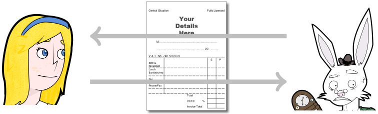
Output J(x, )

- Completeness: If $x \in \mathcal{L}$ the proof always succeeds

Arguments of knowledge

- An argument system is an interactive protocol in which  convinces  that some x is in $\mathcal{L} \subset \mathbf{NP}$

$$\mathcal{L} = \{x : \exists w \text{ s.t. } (x, w) \in \mathcal{R}_{\mathcal{L}}\}$$



(x, w) $\qquad$ x

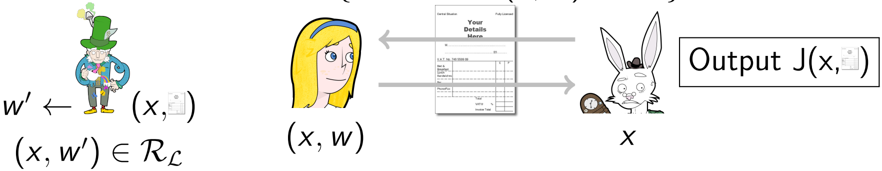
Output J(x, ·)

- Completeness: If $x \in \mathcal{L}$ the proof always succeeds
- Computational soundness: A computationally bounded  can cheat only with small probability

Arguments of knowledge

- An argument system is an interactive protocol in which  convinces  that some x is in $\mathcal{L} \subset \mathbf{NP}$

$$\mathcal{L} = \{x : \exists w \text{ s.t. } (x, w) \in \mathcal{R}_{\mathcal{L}}\}$$



- Completeness: If $x \in \mathcal{L}$ the proof always succeeds
- Computational soundness: A computationally bounded  can cheat only with small probability
- Argument of knowledge: We can extract a valid w in polynomial time from an accepting proof

Kilian's protocol

- Let **AM**(ρ, λ) be the class of argument systems with ρ messages and total communication complexity λ

Kilian's protocol

- Let $\mathbf{AM}(\rho, \lambda)$ be the class of argument systems with ρ messages and total communication complexity λ
- At STOC '92 Kilian showed that every language in $\mathbf{NP}$ has an argument of knowledge in $\mathbf{AM}(4, \mathit{poly}(\log k))$

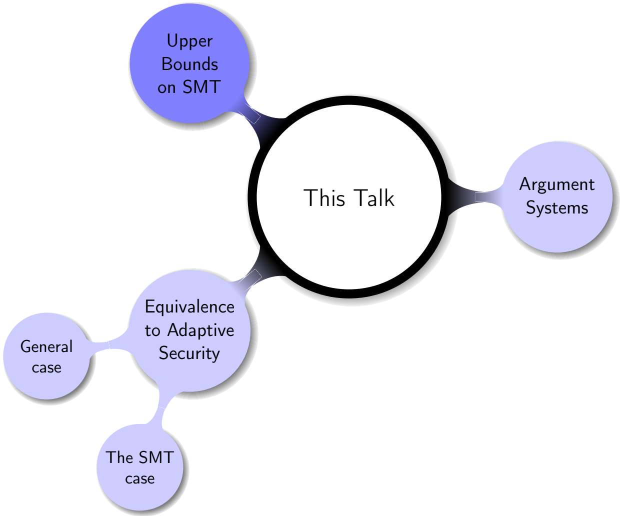
Kilian's protocol

- Let $\mathbf{AM}(\rho, \lambda)$ be the class of argument systems with ρ messages and total communication complexity λ
- At STOC '92 Kilian showed that every language in **NP** has an argument of knowledge in $\mathbf{AM}(4, \mathit{poly}(\log k))$
  - Such arguments are often called succinct

Kilian's protocol

- Let $\mathbf{AM}(\rho, \lambda)$ be the class of argument systems with ρ messages and total communication complexity λ
- At STOC '92 Kilian showed that every language in **NP** has an argument of knowledge in $\mathbf{AM}(4, \mathit{poly}(\log k))$
  - Such arguments are often called succinct
  - Main ingredients: PCP theorem + Merkle trees

Roadmap



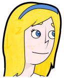
Upper Bounds on SMT
This Talk
Argument Systems
Equivalence to Adaptive Security
General case
The SMT case

Leakage-tolerant SMT requires large keys

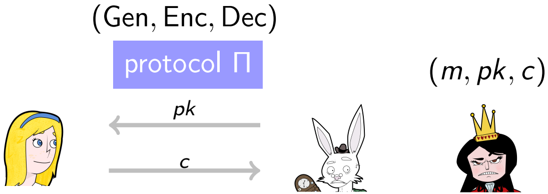
Theorem: Assume collision resistant function ensembles exist. Let Π be a leakage tolerant protocol for SMT tolerating poly-logarithmic leakage. Then,

$$|\mathcal{SK}| \geq (1-\epsilon)|\mathcal{M}| \text{ for negligible } \epsilon.$$

Sketch of the proof



Sketch of the proof



$$\sigma_B = (sk, r_G, m)$$

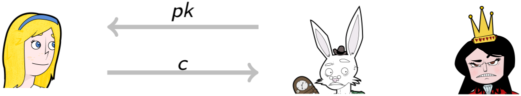
Sketch of the proof

$$\mathcal{L} = \{x := (m, pk, c) : \exists w = (sk, r_G) \text{ "explaining" } x\}$$

(Gen, Enc, Dec)

protocol Π

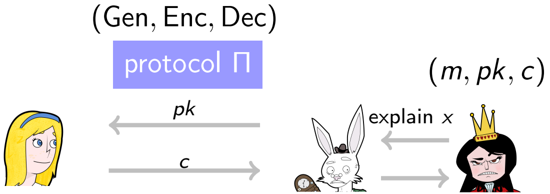
(m, pk, c)



$$\sigma_B = (sk, r_G, m)$$

Sketch of the proof

$\mathcal{L} = \{x := (m, pk, c) : \exists w = (sk, r_G) \text{ “explaining” } x\}$



$\sigma_B = (sk, r_G, m)$

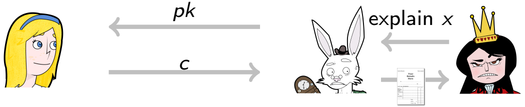
Sketch of the proof

$\mathcal{L} = \{x := (m, pk, c) : \exists w = (sk, r_G)$ "explaining" $x\}$

(Gen, Enc, Dec)

protocol Π

(m, pk, c)

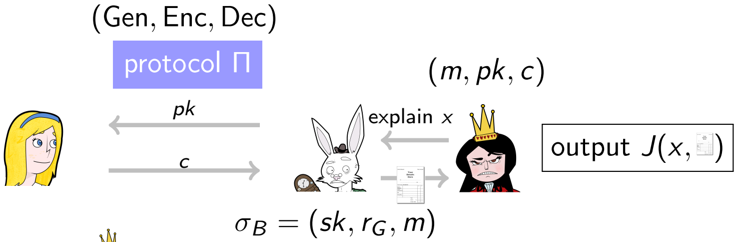


output $J(x, \cdot)$

$\sigma_B = (sk, r_G, m)$

Sketch of the proof

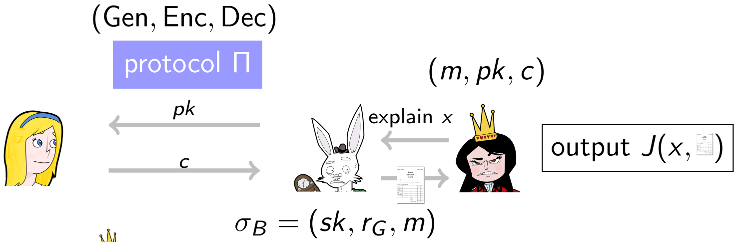
$\mathcal{L} = \{x := (m, pk, c) : \exists w = (sk, r_G) \text{ "explaining" } x\}$



- In the attack  plays the verifier in the proof system and simulates the interaction with the prover via leakage queries

Sketch of the proof

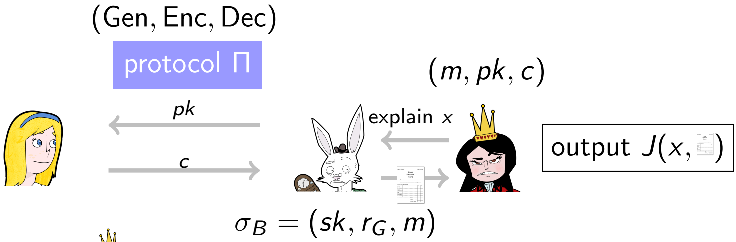
$\mathcal{L} = \{x := (m, pk, c) : \exists w = (sk, r_G) \text{ “explaining” } x\}$



- In the attack  plays the verifier in the proof system and simulates the interaction with the prover via leakage queries
  - The above can be done because  knows a witness and the communication complexity is at most poly-logarithmic

Sketch of the proof

$\mathcal{L} = \{x := (m, pk, c) : \exists w = (sk, r_G) \text{ "explaining" } x\}$



- In the attack  plays the verifier in the proof system and simulates the interaction with the prover via leakage queries
  - The above can be done because  knows a witness and the communication complexity is at most poly-logarithmic
  -  can compute the verifier's next message and "hard-wire" the result in the next leakage query

Sketch of the proof

- It follows by leakage tolerance that ∃  simulating 's view

Sketch of the proof

- It follows by leakage tolerance that $\exists$  simulating 's view
  - In particular note that  "translates" leakage queries on the real state to leakage queries on the ideal state

Sketch of the proof

- It follows by leakage tolerance that $\exists$  simulating 's view
  - In particular note that  "translates" leakage queries on the real state to leakage queries on the ideal state
- Since the proof is for sure valid in the real world, so it must be in the ideal world, i.e. $x = (m, pk, c) \in \mathcal{L}$ by soundness

Sketch of the proof

- It follows by leakage tolerance that $\exists$  simulating 's view
  - In particular note that  "translates" leakage queries on the real state to leakage queries on the ideal state
- Since the proof is for sure valid in the real world, so it must be in the ideal world, i.e. $x = (m, pk, c) \in \mathcal{L}$ by soundness

$$M_{pk,c} := \{m' \in \mathcal{M} : \exists sk' \text{ explaining } c\}$$

Sketch of the proof

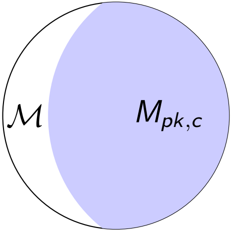
- It follows by leakage tolerance that $\exists$  simulating 's view
  - In particular note that  "translates" leakage queries on the real state to leakage queries on the ideal state
- Since the proof is for sure valid in the real world, so it must be in the ideal world, i.e. $x = (m, pk, c) \in \mathcal{L}$ by soundness

$$M_{pk,c} := \{m' \in \mathcal{M} : \exists sk' \text{ explaining } c\}$$

Claim: $\mathbb{P}[m \in M_{pk,c}] \geq 1 - \epsilon$

Sketch of the proof

- It follows by leakage tolerance that ∃  simulating 's view
  - In particular note that  "translates" leakage queries on the real state to leakage queries on the ideal state
- Since the proof is for sure valid in the real world, so it must be in the ideal world, i.e. $x = (m, pk, c) \in \mathcal{L}$ by soundness

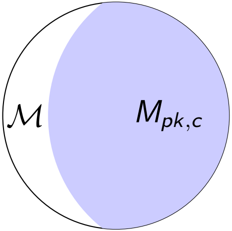


$$M_{pk,c} := \{m' \in \mathcal{M} : \exists sk' \text{ explaining } c\}$$

Claim: $\mathbb{P}[m \in M_{pk,c}] \geq 1 - \epsilon \Rightarrow |M_{pk,c}| \geq (1-\epsilon)|\mathcal{M}|$

Sketch of the proof

- It follows by leakage tolerance that $\exists$  simulating 's view
  - In particular note that  "translates" leakage queries on the real state to leakage queries on the ideal state
- Since the proof is for sure valid in the real world, so it must be in the ideal world, i.e. $x = (m, pk, c) \in \mathcal{L}$ by soundness

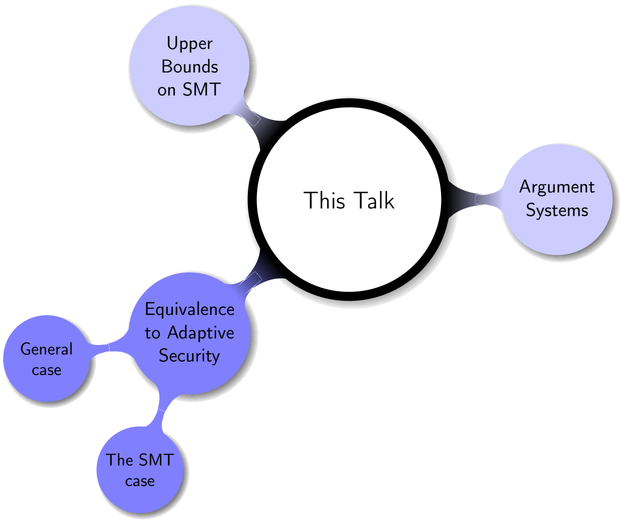


$$M_{pk,c} := \{m' \in \mathcal{M} : \exists sk' \text{ explaining } c\}$$

Claim: $\mathbb{P}[m \in M_{pk,c}] \geq 1 - \epsilon \Rightarrow |M_{pk,c}| \geq (1 - \epsilon)|\mathcal{M}|$

$$\forall m_0 \neq m_1 \in M_{pk,c} \text{ it is } sk_0 \neq sk_1 \Rightarrow |\mathcal{SK}| \geq |M_{pk,c}|$$

Roadmap



Upper Bounds on SMT
This Talk
Argument Systems
Equivalence to Adaptive Security
General case
The SMT case

Equivalence in case of SMT

Theorem: Let Π be a leakage tolerant protocol for SMT tolerating poly-logarithmic leakage at the receiver side at the end of the protocol execution. Then Π is passively secure against an adaptive corruption of the receiver at the end of the protocol execution.

Equivalence in case of SMT

Theorem: Let Π be a leakage tolerant protocol for SMT tolerating poly-logarithmic leakage at the receiver side at the end of the protocol execution. Then Π is passively secure against an adaptive corruption of the receiver at the end of the protocol execution.

- Consider the same  as before in the real world of the leakage game, but now ask to leak an argument of knowledge for $x \in \mathcal{L}$

Equivalence in case of SMT

Theorem: Let Π be a leakage tolerant protocol for SMT tolerating poly-logarithmic leakage at the receiver side at the end of the protocol execution. Then Π is passively secure against an adaptive corruption of the receiver at the end of the protocol execution.

- Consider the same  as before in the real world of the leakage game, but now ask to leak an argument of knowledge for $x \in \mathcal{L}$
- From this we get a valid simulator  in the ideal world

Equivalence in case of SMT

Theorem: Let Π be a leakage tolerant protocol for SMT tolerating poly-logarithmic leakage at the receiver side at the end of the protocol execution. Then Π is passively secure against an adaptive corruption of the receiver at the end of the protocol execution.

- Consider the same  as before in the real world of the leakage game, but now ask to leak an argument of knowledge for $x \in \mathcal{L}$
- From this we get a valid simulator  in the ideal world
- Since the proof will accept with overwhelming probability, a simulator  for the adaptive security game can run  and extract from it a consistent state $w = (sk, r_G)$

A problem and a solution

- Unfortunately, such state may not look indistinguishable from the real state! The difficulty will be to "enforce" indistinguishability

A problem and a solution

- Unfortunately, such state may not look indistinguishable from the real state! The difficulty will be to "enforce" indistinguishability
- To argue indistinguishability consider the following () in the leakage game

A problem and a solution

- Unfortunately, such state may not look indistinguishable from the real state! The difficulty will be to "enforce" indistinguishability
- To argue indistinguishability consider the following  in the leakage game
  - Let the protocol flow leading to $x = (m, pk, c)$ and $w = (sk, r_G)$

A problem and a solution

- Unfortunately, such state may not look indistinguishable from the real state! The difficulty will be to "enforce" indistinguishability
- To argue indistinguishability consider the following () in the leakage game
  - Let the protocol flow leading to $x = (m, pk, c)$ and $w = (sk, r_G)$
  - Leak (1) $h = H(w)$; (2) a proof that x and h are consistent; (3) a guess $b =$ (m, w); (4) a proof that b and h are consistent

A problem and a solution

- Unfortunately, such state may not look indistinguishable from the real state! The difficulty will be to "enforce" indistinguishability
- To argue indistinguishability consider the following () in the leakage game
  - Let the protocol flow leading to $x = (m, pk, c)$ and $w = (sk, r_G)$
  - Leak (1) $h = H(w)$; (2) a proof that x and h are consistent; (3) a guess $b =$ (m, w); (4) a proof that b and h are consistent
  - Output b

A problem and a solution

- Unfortunately, such state may not look indistinguishable from the real state! The difficulty will be to "enforce" indistinguishability
- To argue indistinguishability consider the following () in the leakage game
  - Let the protocol flow leading to $x = (m, pk, c)$ and $w = (sk, r_G)$
  - Leak (1) $h = H(w)$; (2) a proof that x and h are consistent; (3) a guess $b =$ (m, w); (4) a proof that b and h are consistent
  - Output b
- We use  to build  in the adaptive security game

A problem and a solution

- Unfortunately, such state may not look indistinguishable from the real state! The difficulty will be to "enforce" indistinguishability
- To argue indistinguishability consider the following () in the leakage game
  - Let the protocol flow leading to $x = (m, pk, c)$ and $w = (sk, r_G)$
  - Leak (1) $h = H(w)$; (2) a proof that x and h are consistent; (3) a guess $b =$ (m, w); (4) a proof that b and h are consistent
  - Output b
- We use  to build  in the adaptive security game
  - If we now extract w from the proof in (2) above, we get the right distribution (unless collision resistance is broken)

General case

- The previous statement can be generalized to an arbitrary n-party protocol where a single party gets corrupted at the end of the protocol execution

General case

- The previous statement can be generalized to an arbitrary n-party protocol where a single party gets corrupted at the end of the protocol execution
- However, the proof breaks down when $t \geq 2$ parties can be corrupted

General case

- The previous statement can be generalized to an arbitrary *n*-party protocol where a single party gets corrupted at the end of the protocol execution
- However, the proof breaks down when $t \geq 2$ parties can be corrupted
    - The reason is that we cannot send anymore  into the parties and leak its guess, since now we should get $(w_1, \ldots, w_t)$ and it is not clear how to leak this value from small leakages $f(w_1), \ldots, f(w_t)$

General case

- The previous statement can be generalized to an arbitrary *n*-party protocol where a single party gets corrupted at the end of the protocol execution
- However, the proof breaks down when $t \geq 2$ parties can be corrupted
    - The reason is that we cannot send anymore  into the parties and leak its guess, since now we should get $(w_1, \ldots, w_t)$ and it is not clear how to leak this value from small leakages $f(w_1), \ldots, f(w_t)$
- We obtain in this case a weaker form of adaptive security, i.e. we can still extract a consistent internal state but this may not be indistinguishable from a real state

Conclusions

- We have studied the connection between adaptive security and simulation-based leakage tolerance

Conclusions

- We have studied the connection between adaptive security and simulation-based leakage tolerance
- Bitansky *et al.* [BCH12] proved that adaptive security implies leakage tolerance for a large class of functionalities

Conclusions

- We have studied the connection between adaptive security and simulation-based leakage tolerance
- Bitansky *et al.* [BCH12] proved that adaptive security implies leakage tolerance for a large class of functionalities
- We have shown that for some corruption case and for poly-logarithmic leakage

Conclusions

- We have studied the connection between adaptive security and simulation-based leakage tolerance
- Bitansky *et al.* [BCH12] proved that adaptive security implies leakage tolerance for a large class of functionalities
- We have shown that for some corruption case and for poly-logarithmic leakage
  - SMT requires a key as long as the message being encrypted

Conclusions

- We have studied the connection between adaptive security and simulation-based leakage tolerance
- Bitansky *et al.* [BCH12] proved that adaptive security implies leakage tolerance for a large class of functionalities
- We have shown that for some corruption case and for poly-logarithmic leakage
    - SMT requires a key as long as the message being encrypted
    - Leakage tolerance implies adaptive security

Thank You!

entirely written in LaTeX2ε using Beamer and Tik Z, drawings by Andrea Chronopoulos