Symbolic methods applied to the automation of computational proofs

Charlie Jacomme

August 29, 2018

LSV, INRIA Nancy

Introduction to security

Introduction

unnecessary !

unnecessary !

$\hookrightarrow$ Bottom line : we want proofs of security

Symbolic model

Proofs by saturation

1. Define exactly which operations can the attacker perform.
2. Define the security of our protocol/scheme.
3. Try all the possibles attacker's actions, until either we break the security, or there is nothing else to do.

Proofs by saturation

1. Define exactly which operations can the attacker perform.
2. Define the security of our protocol/scheme.
3. Try all the possibles attacker's actions, until either we break the security, or there is nothing else to do.

Realm

- Messages are abstract terms: $enc(message, sk)$
- An equationnal theory capture the attacker power:

$$dec(enc(m, sk), sk)) = M$$

- The attacker can intercept everything over the network

A symbolic method example

Deducibility

Given a set of messages, can an attacker deduce a secret ?

A symbolic method example

Deducibility

Given a set of messages, can an attacker deduce a secret ?

Example

$$x, g^{x \times y}, g^{y^2} \vdash g^{y^2+y}$$

A symbolic method example

Deducibility

Given a set of messages, can an attacker deduce a secret ?

Example

$$x, g^{x \times y}, g^{y^2} \vdash g^{y^2+y}$$

$$g^{y^2+y} = (g^{x \times y})^{-x} \times g^{y^2}$$

Computational model

Proof by reductions

1. Assume that some problem is difficult
2. Define the security of our protocol/scheme
3. Show that if one can break the security, one can break the difficult problem

Proof by reductions

1. Assume that some problem is difficult
2. Define the security of our protocol/scheme
3. Show that if one can break the security, one can break the difficult problem

Realm

- Messages are bitstrings
- Attackers are any PPT

Computational vs Symbolic

Fight !

Symbolic model

- Network controlled by the attacker
- Primitives are perfect
- Many automated proofs
- Missed attacks

Computational model

- Network controlled by the attacker
- Attacker is any PPT
- Few automated proofs
- Hand made proofs hard to check

Fight !

Symbolic model

- Network controlled by the attacker
- Primitives are perfect
- Many automated proofs
- Missed attacks

Computational model

- Network controlled by the attacker
- Attacker is any PPT
- Few automated proofs
- Hand made proofs hard to check

$\hookrightarrow$ Our focus : improving automation in the computational model

A formal framework for computational proofs

A bit of syntax

Expressions: $\times$, $\div$ and $(\_)^{(\_)}$

A bit of syntax

Expressions: $\times$, $\div$ and $(\_)^{(\_)}$

$$\text{Games:}\quad \begin{cases} a : \mathbb{F}_q \\ \text{if } \_ = \_ \text{ then } \ldots \text{ else } \ldots \\ \mathcal{A}(\ldots) \end{cases}$$

Game equivalence

Goal example:

$$a, b : \mathbb{F}_q.\mathcal{A}(g^a, g^b, g^{ab}) \simeq a, b, c : \mathbb{F}_q.\mathcal{A}(g^a, g^b, g^c)$$

Game equivalence

Goal example:

$$a, b : \mathbb{F}_q . \mathcal{A}(g^a, g^b, g^{ab}) \simeq a, b, c : \mathbb{F}_q . \mathcal{A}(g^a, g^b, g^c)$$

$\hookrightarrow$ No PPT attacker can distinguish between the two cases

$\forall \mathcal{A} \in \text{PPT}.$
$$|Pr(\mathcal{A}(g^a, g^b, g^{ab}) = 1 | a, b \leftarrow \mathbb{F}_q) - Pr(\mathcal{A}(g^a, g^b, g^c) = 1 | a, b, c \leftarrow \mathbb{F}_q) - \frac{1}{2}|$$

is negligible

A formal definition of reductions

The reduction rule

$$\mathrm{Reduc}(B)\ \frac{G \simeq G'}{G\{\mathcal{A} \mapsto B(\mathcal{A})\} \simeq G'\{\mathcal{A} \mapsto B(\mathcal{A})\}}$$

A formal definition of reductions

The reduction rule

$$\mathrm{Reduc}(B)\ \frac{G \simeq G'}{G\{\mathcal{A} \mapsto B(\mathcal{A})\} \simeq G'\{\mathcal{A} \mapsto B(\mathcal{A})\}}$$

Informally

$$\forall \mathcal{B} \in \mathrm{PPT}. ...\mathcal{A}(args_1) \simeq ...\mathcal{A}(args_2) \Rightarrow ...B(\mathcal{A}, args_1) \simeq ...B(\mathcal{A}, args_2)$$

Reduction example

The DDH assumption

$$H_1 = (a, b : \mathbb{F}_q . \mathcal{A}(g^a, g^b, g^{ab})) \simeq H_2 = (a, b, c : \mathbb{F}_q . \mathcal{A}(g^a, g^b, g^c))$$

Reduction example

The DDH assumption

$$H_1 = (a, b : \mathbb{F}_q.\mathcal{A}(g^a, g^b, g^{ab})) \simeq H_2 = (a, b, c : \mathbb{F}_q.\mathcal{A}(g^a, g^b, g^c))$$

The simulator

We replace $\mathcal{A}$ by:

$$B(e_1, e_2, e_3) := d : \mathbb{F}_q, A(e_1, e_2, g^d, e_3^d)$$

Reduction example

The DDH assumption

$$H_1 = (a, b : \mathbb{F}_q . \mathcal{A}(g^a, g^b, g^{ab})) \simeq H_2 = (a, b, c : \mathbb{F}_q . \mathcal{A}(g^a, g^b, g^c))$$

The simulator

We replace $\mathcal{A}$ by:

$$B(e_1, e_2, e_3) := d : \mathbb{F}_q, A(e_1, e_2, g^d, e_3^d)$$

The result

$$(a, b, d : \mathbb{F}_q . \mathcal{A}(g^a, g^b, g^d, g^{abd})) \simeq (a, b, c, d : \mathbb{F}_q . \mathcal{A}(g^a, g^b, g^d, g^{cd}))$$

Reduction example

The simulator

$$B(A)(e_1, e_2, e_3) := d : \mathbb{F}_q, A(e_1, e_2, g^d, e_3^d)$$

The rule application

$$\text{Reduc}(B)\ \frac{(a, b : \mathbb{F}_q.\mathcal{A}(g^a, g^b, g^{ab})) \simeq (a, b, c : \mathbb{F}_q.\mathcal{A}(g^a, g^b, g^c))}{H_1\{\mathcal{A} \mapsto B(\mathcal{A})\} \simeq H_2\{\mathcal{A} \mapsto B(\mathcal{A})\}}$$

Automated construction of simulators

The problem

Question

Given an assumption and a goal, can we find B to apply Reduc ?

The problem

Question

Given an assumption and a goal, can we find B to apply Reduc ?

Simulator

$$\text{Partial assumption: } (a, b : \mathbb{F}_q.\mathcal{A}(g^a, g^b, g^{ab}))$$

$$\text{Partial goal: } (a, b, \underline{c : \mathbb{F}_q.\mathcal{A}(g^a, g^b, g^c, g^{abc})})$$

$$\Rightarrow$$

Given (g^a, g^b, g^{ab}), can a simulator compute (g^a, g^b, g^c, g^{abc}) ?

The problem

Question

Given an assumption and a goal, can we find B to apply Reduc ?

Simulator

$$\text{Partial assumption: } (a, b : \mathbb{F}_q.\mathcal{A}(g^a, g^b, g^{ab}))$$

$$\text{Partial goal: } (a, b, \underline{c : \mathbb{F}_q.\mathcal{A}(g^a, g^b, g^c, g^{abc}}))$$

$$\Rightarrow$$

Given (g^a, g^b, g^{ab}), can a simulator compute (g^a, g^b, g^c, g^{abc}) ?

$\hookrightarrow$A deducibility problem

Deducibility

Disadvantage

Something not deducible in the symbolic word might be deducible in the actual world.

$$enc(a, sk), enc(b, sk) \not\vdash enc(a + b, sk)$$

Advantage

If something is deducible in the symbolic world, it is always deducible.

$\hookrightarrow$ We may find valid simulators using deducibility.

Main idea

Game hypothesis:

$$x_1, ..., x_n : \mathbb{F}_q.\mathcal{A}(e_1, .., e_k)$$

Goal:

$$x_1, ..., x_n, ..., x_n + k : \mathbb{F}_q.R$$

Check, if for all terms t that appears in R (a game):

$$e_1, ..., e_k \vdash t$$

Options

Option 1

Many existing efficient results for many simple theories in the symbolic models.

Options

Option 1

Many existing efficient results for many simple theories in the symbolic models.

$\hookrightarrow$ use them to provides fast automation.

- In case of success, we can construct a simulator
- If it fails, we have nothing

Options

Option 1

Many existing efficient results for many simple theories in the symbolic models.

$\hookrightarrow$ use them to provides fast automation.

- In case of success, we can construct a simulator
- If it fails, we have nothing

Option 2

Extend the symbolic technics to more complex theories, which might be complete.

Options

Option 1

Many existing efficient results for many simple theories in the symbolic models.

$\hookrightarrow$ use them to provides fast automation.

- In case of success, we can construct a simulator
- If it fails, we have nothing

Option 2

Extend the symbolic technics to more complex theories, which might be complete.

$\hookrightarrow$ use them to provides slow but complete automation.

Contributions[1]

Existing work

- Deducibility only for polynomials of degree one in the exponent, without axioms
- AutoGnP [Barthe et al, CCS15] used heuristics to construct simulators

[1] Symbolic Proofs for Lattice-Based Cryptography, CCS18
G. Barthe, X. Fan, J. Gancher, B. Gregoire, C. Jacomme, E. Shi

Contributions[1]

Existing work

- Deducibility only for polynomials of degree one in the exponent, without axioms
- AutoGnP [Barthe et al, CCS15] used heuristics to construct simulators

Contributions

- Axioms ($a \neq 0$)
- Bilinear maps
- Any polynomials in the exponent
- Matrices

[1] Symbolic Proofs for Lattice-Based Cryptography, CCS18
G. Barthe, X. Fan, J. Gancher, B. Gregoire, C. Jacomme, E. Shi

Our generalized problem

$$\Gamma \models X, g_{i_1}^{f_1}, \ldots, g_{i_k}^{f_k} \vdash_{\mathcal{E}} g_t^h$$

$$\begin{cases} \Gamma \text{ axioms} \\ X \text{ public variables} \\ g_t \text{ target group} \\ \mathcal{E} \text{ equational theory} \\ f_i \text{ polynomial} \end{cases}$$

First step

Saturation

Obtain a problem with only one group using a previous result :

$\hookrightarrow$ compute all the possible map applications and obtain a saturated set

Example

$$g_{i_1}^{f_1}, g_{i_2}^{f_2} \vdash_{\mathcal{E}} g_t^h$$

$$\Leftrightarrow$$

$$g_t^{f_1}, g_t^{f_2}, g_t^{f_1^2}, g_t^{f_2^2}, g_t^{f_1 \times f_2} \vdash_{\mathcal{E}} g_t^h$$

Second Step (our contribution)

$$\Gamma \models X, g_t^{f_1}, \ldots, g_t^{f_k} \vdash_{\mathcal{E}} g_t^h$$

With Groebner Basis

Second Step (our contribution)

$$\Gamma \models X, g_t^{f_1}, ..., g_t^{f_k} \vdash_{\mathcal{E}} g_t^h$$

With Groebner Basis

1. Characterize the attacker knowledge:

$$M = \{\sum_i e_i \times f_i | e_i \in \mathbb{K}[X]\}$$

Second Step (our contribution)

$$\Gamma \models X, g_t^{f_1}, \dots, g_t^{f_k} \vdash_{\mathcal{E}} g_t^h$$

With Groebner Basis

1. Characterize the attacker knowledge:

$$M = \{\sum_i e_i \times f_i | e_i \in \mathbb{K}[X]\}$$

2. Saturate using the axioms, if $\Gamma = \{p_k \neq 0\}$:

$$M :_{\mathbb{K}[X,Y]} (p_1 \dots p_n)^\infty = \{f \in \mathbb{K}[X, Y] | \exists n \in \mathbb{N}, f \times (p_1 \dots p_n)^n \in M\}$$

Second Step (our contribution)

$$\Gamma \models X, g_t^{f_1}, \ldots, g_t^{f_k} \vdash_{\mathcal{E}} g_t^h$$

With Groebner Basis

1. Characterize the attacker knowledge:

$$M = \{\sum_i e_i \times f_i | e_i \in \mathbb{K}[X]\}$$

2. Saturate using the axioms, if $\Gamma = \{p_k \neq 0\}$:

$$M :_{\mathbb{K}[X,Y]} (p_1 \ldots p_n)^{\infty} = \{f \in \mathbb{K}[X, Y] | \exists n \in \mathbb{N}, f \times (p_1 \ldots p_n)^n \in M\}$$

3. Test the membership.

Conclusion

Conclusions

A complete procedure

Given an hypothesis and a goal, we provide a complete procedure to decide if the rule Reduc can be applied.

Conclusions

A complete procedure

Given an hypothesis and a goal, we provide a complete procedure to decide if the rule Reduc can be applied.

WIP

Use symbolic methods (deducibility, static equivalence, unification):

- to automatize more complex crypto proofs (RND rule)
- to verify masking schemes
- to handle multistage games, oracle games, ...

Conclusions

A complete procedure

Given an hypothesis and a goal, we provide a complete procedure to decide if the rule Reduc can be applied.

WIP

Use symbolic methods (deducibility, static equivalence, unification):

- to automatize more complex crypto proofs (RND rule)
- to verify masking schemes
- to handle multistage games, oracle games, ...