TOWARDS UNCONDITIONAL SOUNDNESS

Gergei Bana and Hubert Comon-Lundh

comon@lsv.ens-cachan.fr

06/16/2011

THE CONTEXT

THE CONTEXT

- Can we trust attacks on protocols ?

THE CONTEXT

- Can we trust attacks on protocols ?
- Can we trust security proofs ?

FIRST STEP: SYMBOLIC MODELS AND PROOFS

Consider the protocol:

$$A: \ \nu N, r, \quad \begin{array}{l} \{A, N\}^{r}_{pk(B)} \rightarrow \\ \{B, N\}^{r}_{pk(A)} \leftarrow \end{array} \qquad\qquad B: \ \nu r', \quad \begin{array}{l} \rightarrow \{x, y\}^{-}_{pk(B)} \\ \leftarrow \{B, y\}^{r'}_{pk(x)} \end{array}$$

security property: N is a shared secret between A and B (when the protocol is completed).

First step: symbolic models and proofs

Consider the protocol:

$$A: \ \nu N, r, \quad \begin{array}{l} \{A, N\}^{r}_{pk(B)} \rightarrow \\ \{B, N\}^{r}_{pk(A)} \leftarrow \end{array} \qquad B: \ \nu r', \quad \begin{array}{l} \rightarrow \{x, y\}^{-}_{pk(B)} \\ \leftarrow \{B, y\}^{r'}_{pk(x)} \end{array}$$

security property: N is a shared secret between A and B (when the protocol is completed).

True in the symbolic model

FIRST STEP: SYMBOLIC MODELS AND PROOFS

Consider the protocol:

$$A:\ \nu N, r, \quad \begin{array}{l} \{A, N\}^{r}_{pk(B)} \rightarrow \\ \{B, N\}^{r}_{pk(A)} \leftarrow \end{array} \qquad\qquad B:\ \nu r', \quad \begin{array}{l} \rightarrow \{x, y\}^{-}_{pk(B)} \\ \leftarrow \{B, y\}^{r'}_{pk(x)} \end{array}$$

security property: N is a shared secret between A and B (when the protocol is completed).

True in the symbolic model

False for some malleable encryption schemes

SECOND STEP: SOUNDNESS RESULTS

Theorem: Assuming **H** then any symbolically secure protocol is also computationally secure.

Proof: Hard and (very) long.

SECOND STEP: SOUNDNESS RESULTS

Theorem: Assuming **H** then any symbolically secure protocol is also computationally secure.

Proof: Hard and (very) long.

Typical **H**:

SECOND STEP: SOUNDNESS RESULTS

Theorem: Assuming **H** then any symbolically secure protocol is also computationally secure.

Proof: Hard and (very) long.

Typical **H**:

- the encryption scheme is IND-CCA

SECOND STEP: SOUNDNESS RESULTS

Theorem: Assuming **H** then any symbolically secure protocol is also computationally secure.

Proof: Hard and (very) long.

Typical **H**:

- the encryption scheme is IND-CCA
- bitstrings can be efficiently parsed into terms

SECOND STEP: SOUNDNESS RESULTS

Theorem: Assuming **H** then any symbolically secure protocol is also computationally secure.

Proof: Hard and (very) long.

Typical **H**:

- the encryption scheme is IND-CCA
- bitstrings can be efficiently parsed into terms
- There is no key-cycle

SECOND STEP: SOUNDNESS RESULTS

Theorem: Assuming **H** then any symbolically secure protocol is also computationally secure.

Proof: Hard and (very) long.

Typical **H**:

- the encryption scheme is IND-CCA
- bitstrings can be efficiently parsed into terms
- There is no key-cycle
- There is no dynamic corruption

SECOND STEP: SOUNDNESS RESULTS

Theorem: Assuming **H** then any symbolically secure protocol is also computationally secure.

Proof: Hard and (very) long.

Typical **H**:

- the encryption scheme is IND-CCA
- bitstrings can be efficiently parsed into terms
- There is no key-cycle
- There is no dynamic corruption
- There is no “bad key” : keys are certified

06/16/2011

SECOND STEP: SOUNDNESS RESULTS

Theorem: Assuming **H** then any symbolically secure protocol is also computationally secure.

Proof: Hard and (very) long.

Typical **H**:

- the encryption scheme is IND-CCA
- bitstrings can be efficiently parsed into terms
- There is no key-cycle
- There is no dynamic corruption
- There is no “bad key” : keys are certified
- ...

SECOND STEP: SOUNDNESS RESULTS

Theorem: Assuming **H** then any symbolically secure protocol is also computationally secure.

Proof: Hard and (very) long.

Typical **H**:

- the encryption scheme is IND-CCA — necessary ?
- bitstrings can be efficiently parsed into terms
- There is no key-cycle
- There is no dynamic corruption
- There is no “bad key” : keys are certified
- ...

SECOND STEP: SOUNDNESS RESULTS

Theorem: Assuming **H** then any symbolically secure protocol is also computationally secure.

Proof: Hard and (very) long.

Typical **H**:

- the encryption scheme is IND-CCA — necessary ?
- bitstrings can be efficiently parsed into terms — necessary ?
- There is no key-cycle
- There is no dynamic corruption
- There is no “bad key” : keys are certified
- ...

SECOND STEP: SOUNDNESS RESULTS

Theorem: Assuming **H** then any symbolically secure protocol is also computationally secure.

Proof: Hard and (very) long.

Typical **H**:

- the encryption scheme is IND-CCA — necessary ?
- bitstrings can be efficiently parsed into terms — necessary ?
- There is no key-cycle — necessary ?
- There is no dynamic corruption
- There is no "bad key" : keys are certified
- ...

Second step: Soundness results

Theorem: Assuming **H** then any symbolically secure protocol is also computationally secure.

Proof: Hard and (very) long.

Typical **H**:

- the encryption scheme is IND-CCA — necessary ?
- bitstrings can be efficiently parsed into terms — necessary ?
- There is no key-cycle — necessary ?
- There is no dynamic corruption — necessary ?
- There is no “bad key” : keys are certified
- ...

SECOND STEP: SOUNDNESS RESULTS

Theorem: Assuming **H** then any symbolically secure protocol is also computationally secure.

Proof: Hard and (very) long.

Typical **H**:

- the encryption scheme is IND-CCA — necessary ?
- bitstrings can be efficiently parsed into terms — necessary ?
- There is no key-cycle — necessary ?
- There is no dynamic corruption — necessary ?
- There is no “bad key” : keys are certified — necessary ?
- ...

SECOND STEP: SOUNDNESS RESULTS

Theorem: Assuming **H** then any symbolically secure protocol is also computationally secure.

Proof: Hard and (very) long.

Typical **H**:

- the encryption scheme is IND-CCA necessary ?
- bitstrings can be efficiently parsed into terms necessary ?
- There is no key-cycle necessary ?
- There is no dynamic corruption necessary ?
- There is no “bad key” : keys are certified necessary ?
- ...

necessary ?

ALTERNATIVE SOLUTIONS

Formal proofs in a computational model

- CRYPTOVERIF [Bruno Blanchet]
- CERTICRYPT [G. Barthe, S. Zanella et al]

Alternative solutions

Formal proofs in a computational model

- CryptoVerif [Bruno Blanchet]
- CertiCrypt [G. Barthe, S. Zanella et al]

Drawbacks:

ALTERNATIVE SOLUTIONS

Formal proofs in a computational model

- CRYPTOVERIF [Bruno Blanchet]
- CERTICRYPT [G. Barthe, S. Zanella et al]

Drawbacks:

- Takes time to develop

ALTERNATIVE SOLUTIONS

Formal proofs in a computational model

- CRYPTOVERIF [Bruno Blanchet]
- CERTICRYPT [G. Barthe, S. Zanella et al]

Drawbacks:

- Takes time to develop
- Minimal assumptions ? Small modifications, experiments,...

ALTERNATIVE SOLUTIONS

Formal proofs in a computational model

- CRYPTOVERIF [Bruno Blanchet]
- CERTICRYPT [G. Barthe, S. Zanella et al]

Drawbacks:

- Takes time to develop
- Minimal assumptions ? Small modifications, experiments,...
- Full automation ?

ALTERNATIVE SOLUTIONS

Formal proofs in a computational model

- CRYPTOVERIF [Bruno Blanchet]
- CERTICRYPT [G. Barthe, S. Zanella et al]

Drawbacks:

- Takes time to develop
- Minimal assumptions ? Small modifications, experiments,...
- Full automation ?
- What if the proof fails ?

SOUNDNESS RESULTS (CNTD)

SOUNDNESS RESULTS (CNTD)

- Can we trust Soundness theorems ?

06/16/2011

SOUNDNESS RESULTS (CNTD)

- Can we trust Soundness theorems ?
- Is the list of assumptions exhaustive ?

Soundness results (Cntd)

- Can we trust Soundness theorems ?
- Is the list of assumptions exhaustive ?

Why are the soundness proofs so complicated ?

GUILLAUME SCERRI'S EXPLANATION

The symbolic model specifies What is allowed

The computational assumptions specify What is forbidden

GUILLAUME SCERRI'S EXPLANATION

The symbolic model specifies What is allowed

The computational assumptions specify What is forbidden

Idea: design a symbolic model that specifies What is forbidden

A PERMISSIVE SYMBOLIC MODEL

Anything that is not explicitly forbidden is possible:

A transition is possible as long as the required equalities/deductions are consistent with the current assumptions

A PERMISSIVE SYMBOLIC MODEL

Anything that is not explicitly forbidden is possible:

A transition is possible as long as the required equalities/deductions are consistent with the current assumptions

Advantages:

- All assumptions are necessarily formally stated
- Any model that (also) satisfies the negation of the security assumption is a potential attack
- We may (in principle) use any first-order consistency checker
- Arbitrary primitives, modularity,....

A PERMISSIVE SYMBOLIC MODEL

Anything that is not explicitly forbidden is possible:

A transition is possible as long as the required equalities/deductions are consistent with the current assumptions

Advantages:

- All assumptions are necessarily formally stated
- Any model that (also) satisfies the negation of the security assumption is a potential attack
- We may (in principle) use any first-order consistency checker
- Arbitrary primitives, modularity,....

Difficulties/questions:

- Design (in FO) the appropriate assumptions
- What about the computational attacks ?
- Is automation so easy ?

SUMMARY

1. The (symbolic) execution model
2. The main result
3. The computational validity

1. THE EXECUTION MODEL

THE LOGIC

Atomic formulas:

- Terms over an arbitrary signature (encryption, pairs and names in the examples) including handles
- Equalities $s = t$ between terms
- Deducibility:

$$\phi, t_1, \ldots, t_n \vdash t$$

  where $t_1, \ldots, t_n$ are terms and ϕ is interpreted, in any state, as a sequence of ground terms.
- Possibly, Interpreted predicates...

Formulas:
For the transition system: only Boolean combinations of ground atomic formulas.

Interpretation:
Any FO structure.

THE EXECUTION MODEL : AN EXAMPLE

$$A: \ \nu N, r, \quad \begin{array}{l} \{A, N\}^{r}_{pk(B)} \rightarrow \\ \{B, N\}^{-}_{pk(A)} \leftarrow \end{array} \qquad\qquad B: \ \nu r', \quad \begin{array}{l} \rightarrow \{x, y\}^{-}_{pk(B)} \\ \leftarrow \{B, y\}^{r'}_{pk(x)} \end{array}$$

Initial state: $q_0, \emptyset, \top$

A successor state: $q_1, \{A, N\}^{r}_{pk(B)}, \top$

A succsucc state: $q_3, \{A, N\}^{r}_{pk(B)},$

$$\{A, N\}^{r}_{pk(B)} \vdash h \quad \wedge \quad \text{dec}(h, sk(A)) =< B, N >$$

AXIOMS: EXAMPLES

$\phi \vdash A$,
$\phi \vdash B$,
$\phi \vdash x, \phi \vdash y \;\rightarrow\; \phi \vdash f(x, y)$,...

Anything that the Dolev-Yao attacker can do

AXIOMS: EXAMPLES

$\phi \vdash A$,
$\phi \vdash B$,
$\phi \vdash x, \phi \vdash y \;\rightarrow\; \phi \vdash f(x,y)$,...

Anything that the Dolev-Yao attacker can do

Secrecy:

$$\forall x. \quad \phi, \{x\}^r_{pk(A)} \vdash x \quad \rightarrow \quad \phi \vdash x \;\vee\; \phi, \{x\}^r_{pk(A)} \vdash sk(A)$$

AXIOMS: EXAMPLES

$\phi \vdash A$,
$\phi \vdash B$,
$\phi \vdash x, \phi \vdash y \ \rightarrow \ \phi \vdash f(x, y)$,...

Anything that the Dolev-Yao attacker can do

Secrecy:

$$\forall x. \quad \phi, \{x\}^r_{pk(A)} \vdash x \quad \rightarrow \quad \phi \vdash x \ \vee \ \phi, \{x\}^r_{pk(A)} \vdash sk(A)$$

Integrity:

$$\forall y. \quad \phi \vdash y \ \wedge \ \phi, \mathsf{dec}(y, sk(K)) \vdash N \ \wedge \ y \not\sqsubseteq \phi \quad \rightarrow \quad \phi \vdash sk(K) \ \vee \ \phi \vdash N$$

THE EXECUTION MODEL : AN EXAMPLE

$$A:\ \nu N, r,\quad \begin{array}{l}\{A,N\}^{r}_{pk(B)} \rightarrow \\ \{B,N\}^{-}_{pk(A)} \leftarrow\end{array} \qquad\qquad B:\ \nu r',\quad \begin{array}{l}\rightarrow \{x,y\}^{-}_{pk(B)} \\ \leftarrow \{B,y\}^{r'}_{pk(x)}\end{array}$$

Initial state: $q_0, \emptyset, \top$

A successor state: $q_1, \{A,N\}^{r}_{pk(B)}, \top$

A succsucc state: $q_3, \{A,N\}^{r}_{pk(B)},$

$$\{A,N\}^{r}_{pk(B)} \vdash h \ \wedge\ \mathsf{dec}(h, sk(A)) = < B, N >$$

This state is now discarded because the formula is inconsistent with the axioms

The integrity axiom is necessary (otherwise the formula is consistent with the axioms).

2. THE MAIN RESULT

THE COMPUTATIONAL SOUNDNESS

Theorem: Assume that the axioms are computationally valid. If there is a computational attack, then there is a symbolic attack.

Note: this is independent of the security primitives, independent of the properties...

THE COMPUTATIONAL SOUNDNESS

Theorem: Assume that the axioms are computationally valid. If there is a computational attack, then there is a symbolic attack.

Note: this is independent of the security primitives, independent of the properties...

Computational validity of axions, for instance:

Proposition: If the encryption scheme is IND-CCA, then the secrecy and integrity axioms are computationally valid.

3. THE COMPUTATIONAL VALIDITY

THE COMPUTATIONAL INTERPRETATION

- $\mathcal{A}$ is a PPT machine and τ is a sample (mapping names to bit-strings)
- Each function symbol is interpreted as a deterministic polynomial algorithm.
- For any term t, $[\![t]\!]_\tau$ is the homomorphic extension of τ to terms
- $\mathcal{A}, \tau \models^c s = t$ iff $[\![t]\!]_\tau = [\![s]\!]_\tau$.
- $\mathcal{A}, \tau \models^c t_1, \ldots, t_n \vdash t$ iff $\mathcal{A}([\![t_1]\!]_\tau, \ldots, [\![t_n]\!]_\tau) = [\![t]\!]_\tau$.

THE COMPUTATIONAL INTERPRETATION

- $\mathcal{A}$ is a PPT machine and τ is a sample (mapping names to bit-strings)
- Each function symbol is interpreted as a deterministic polynomial algorithm.
- For any term t, $[\![t]\!]_\tau$ is the homomorphic extension of τ to terms
- $\mathcal{A}, \tau \models^c s = t$ iff $[\![t]\!]_\tau = [\![s]\!]_\tau$.
- $\mathcal{A}, \tau \models^c t_1, \ldots, t_n \vdash t$ iff $\mathcal{A}([\![t_1]\!]_\tau, \ldots, [\![t_n]\!]_\tau) = [\![t]\!]_\tau$.

We wish however to reason on families of first-order structures interpreting the formulas. Otherwise, there is always an $\mathcal{A}$ breaking

$$\forall x. \quad \phi, \{x\}^r_{pk(A)} \vdash x \quad \rightarrow \quad \phi \vdash x \;\vee\; \phi, \{x\}^r_{pk(A)} \vdash sk(A)$$

THE COMPUTATIONAL INTERPRETATION

- $\mathcal{A}$ is a PPT machine and τ is a sample (mapping names to bit-strings)
- Each function symbol is interpreted as a deterministic polynomial algorithm.
- For any term t, $[\![t]\!]_\tau$ is the homomorphic extension of τ to terms
- $\mathcal{A}, \tau \models^c s = t$ iff $[\![t]\!]_\tau = [\![s]\!]_\tau$.
- $\mathcal{A}, \tau \models^c t_1, \ldots, t_n \vdash t$ iff $\mathcal{A}([\![t_1]\!]_\tau, \ldots, [\![t_n]\!]_\tau) = [\![t]\!]_\tau$.

We wish however to reason on families of first-order structures interpreting the formulas. Otherwise, there is always an $\mathcal{A}$ breaking

$$\forall x. \quad \phi, \{x\}^r_{pk(A)} \vdash x \quad \rightarrow \quad \phi \vdash x \ \vee \ \phi, \{x\}^r_{pk(A)} \vdash sk(A)$$

For any τ, $\mathcal{A}$ returns

- $[\![n_1]\!]_\tau$ on input $[\![n_1]\!]_\tau, [\![n_2]\!]_\tau, [\![\{n_1\}^r_{pk(A)}]\!]_\tau$
- $[\![n_2]\!]_\tau$ on input $[\![n_1]\!]_\tau, [\![n_2]\!]_\tau$.

THE COMPUTATIONAL INTERPRETATION (CNTD)

S, S_1, S_2, ... are sets of samples

THE COMPUTATIONAL INTERPRETATION (CNTD)

S, S_1, S_2, ... are sets of samples

- $\mathcal{A}, \Pi, S \models^c \exists x.\theta$ if there is a PPT $\mathcal{A}_x$ such that $\mathcal{A}, \Pi, S, \mathcal{A}_x \models \theta$.
  In what follows: σ is an assigment of PPT machines to the free variables of the formula.

THE COMPUTATIONAL INTERPRETATION (CNTD)

S, S_1, S_2, ... are sets of samples

- $\mathcal{A}, \Pi, S \models^c \exists x.\theta$ if there is a PPT $\mathcal{A}_x$ such that $\mathcal{A}, \Pi, S, \mathcal{A}_x \models \theta$.
  In what follows: σ is an assigment of PPT machines to the free variables of the formula.
- $\mathcal{A}, \Pi, S, \sigma \models^c \theta_1 \wedge \theta_2$ if $\mathcal{A}, \Pi, S, \sigma \models^c \theta_1$ and $\mathcal{A}, \Pi, S, \sigma \models^c \theta_2$

The computational interpretation (cntd)

S, S_1, S_2, ... are sets of samples

- $\mathcal{A}, \Pi, S \models^c \exists x.\theta$ if there is a PPT $\mathcal{A}_x$ such that $\mathcal{A}, \Pi, S, \mathcal{A}_x \models \theta$. In what follows: σ is an assigment of PPT machines to the free variables of the formula.
- $\mathcal{A}, \Pi, S, \sigma \models^c \theta_1 \wedge \theta_2$ if $\mathcal{A}, \Pi, S, \sigma \models^c \theta_1$ and $\mathcal{A}, \Pi, S, \sigma \models^c \theta_2$
- $\mathcal{A}, \Pi, S, \sigma \models^c \theta_1 \vee \theta_2$ if $S = S_1 \cup S_2$ and $\mathcal{A}, \Pi, S_1, \sigma \models^c \theta_1$ and $\mathcal{A}, \Pi, S_2, \sigma \models^c \theta_2$

The computational interpretation (cntd)

S, S_1, S_2, ... are sets of samples

- $\mathcal{A}, \Pi, S \models^c \exists x.\theta$ if there is a PPT $\mathcal{A}_x$ such that $\mathcal{A}, \Pi, S, \mathcal{A}_x \models \theta$. In what follows: σ is an assigment of PPT machines to the free variables of the formula.
- $\mathcal{A}, \Pi, S, \sigma \models^c \theta_1 \wedge \theta_2$ if $\mathcal{A}, \Pi, S, \sigma \models^c \theta_1$ and $\mathcal{A}, \Pi, S, \sigma \models^c \theta_2$
- $\mathcal{A}, \Pi, S, \sigma \models^c \theta_1 \vee \theta_2$ if $S = S_1 \cup S_2$ and $\mathcal{A}, \Pi, S_1, \sigma \models^c \theta_1$ and $\mathcal{A}, \Pi, S_2, \sigma \models^c \theta_2$
- $\mathcal{A}, \Pi, S, \sigma \models^c \neg\theta$ if $\mathcal{A}, \Pi, S', \sigma \models \theta$ implies that S' is negligible.

THE COMPUTATIONAL INTERPRETATION (CNTD)

S, S_1, S_2, ... are sets of samples

- $\mathcal{A}, \Pi, S \models^c \exists x.\theta$ if there is a PPT $\mathcal{A}_x$ such that $\mathcal{A}, \Pi, S, \mathcal{A}_x \models \theta$.
  In what follows: σ is an assigment of PPT machines to the free variables of the formula.
- $\mathcal{A}, \Pi, S, \sigma \models^c \theta_1 \wedge \theta_2$ if $\mathcal{A}, \Pi, S, \sigma \models^c \theta_1$ and $\mathcal{A}, \Pi, S, \sigma \models^c \theta_2$
- $\mathcal{A}, \Pi, S, \sigma \models^c \theta_1 \vee \theta_2$ if $S = S_1 \cup S_2$ and $\mathcal{A}, \Pi, S_1, \sigma \models^c \theta_1$ and $\mathcal{A}, \Pi, S_2, \sigma \models^c \theta_2$
- $\mathcal{A}, \Pi, S, \sigma \models^c \neg\theta$ if $\mathcal{A}, \Pi, S', \sigma \models \theta$ implies that S' is negligible.
- $\mathcal{A}, \Pi, S, \sigma \models^c \phi, t_1, \ldots, t_n \vdash t$ if

    For every non negl. $S' \subseteq S$, there is a non-negl. $S'' \subseteq S'$ s.t.

      There is a PPT $\mathcal{A}_D$ such that, $\forall \tau \in S''$,

        The computation of Π, $\mathcal{A}$ yields a bitstring b s.t.

$$\mathcal{A}_D([\![\phi]\!]_\tau, [\![t_1]\!]_\tau^{\sigma(b)}, ..., [\![t_n]\!]_\tau^{\sigma(b)}) = [\![t]\!]_\tau^{\sigma(b)}$$

CONCLUSIONS

What remains to do ?

Conclusions

What remains to do ?

- Automation: simulating the symbolic execution requires a consistency check.
  We conjecture that, for saturated sets of axioms, this consistency check is in PTIME (ongoing work with Véronique Cortier and Guillaume Scerri).

Conclusions

What remains to do ?

- Automation: simulating the symbolic execution requires a consistency check.
  We conjecture that, for saturated sets of axioms, this consistency check is in PTIME (ongoing work with Véronique Cortier and Guillaume Scerri).
- Design (and prove the computational validity for classical cryptographic assumptions) axioms for several primitives. Note: this is modular.

CONCLUSIONS

What remains to do ?

- Automation: simulating the symbolic execution requires a consistency check.
  We conjecture that, for saturated sets of axioms, this consistency check is in PTIME (ongoing work with Véronique Cortier and Guillaume Scerri).
- Design (and prove the computational validity for classical cryptographic assumptions) axioms for several primitives. Note: this is modular.
- Try several examples of protocols.