A Higher-Order Indistinguishability Logic for Cryptographic Reasoning

David Baelde Adrien Koutsos Joseph Lallemand

Univ Rennes, CNRS, IRISA & Inria Paris

LICS 2023, Boston
Reasoning about cryptographic protocols

Computational model

Probabilistic polynomial time machines (PPTM):

- Secrets = long-enough random bitstrings, size $\eta$.
- PTIME prevents brute force attacks.

Reasoning up to negligible probability of failure:
$P(\eta)$ asymptotically smaller than any $\eta^{-k}$. 
CCSA logic: Computationally Complete Symbolic Attacker
[Bana & Comon, CCS’14]

First-order terms interpreted as PPTMs, explicit random tape $\rho \in \{0, 1\}^\infty$

- $\left[t\right]_{M} (\eta, \rho) \in \{0, 1\}^*$
- Name constants $n, m, k \ldots$ extract from $\rho$ dedicated sections of length $\eta$.

Example

$\text{att}_2(m, h(\text{att}_1(m), k))$ : attacker computes 2\textsuperscript{nd} message from $m$ and hash of first message.
CCSA logic: Computationally Complete Symbolic Attacker
[Bana & Comon, CCS’14]

First-order terms interpreted as PPTMs, explicit random tape $\rho \in \{0, 1\}^\infty$

- $\lbrack t \rbrack_M(\eta, \rho) \in \{0, 1\}^*$
- Name constants $n, m, k \ldots$ extract from $\rho$ dedicated sections of length $\eta$.

Predicates

- $t \sim t'$: “$t$ and $t'$ are computationally indistinguishable”.
- $[\varphi]$ where $\varphi$ is a boolean term: “$\varphi$ false with negligible probability”.

Example

$[n \neq \text{att}_2(m, h(\text{att}_1(m), k))]$ is valid.
Example (Local meta-logic formulas)

\[ \text{output@A(i, j)} = h(\text{input@A(i, j)}, k(i)) \]

\[ \forall k. \text{cond@B(k)} \Rightarrow \exists i, j. \text{A(i, j)} < B(k) \land \text{input@B(k)} = \text{output@A(i, j)} \]

Reasoning over all trace models \( T \) for protocol \( P \), and all computational models \( M \).

Meta-logic term \( t \) \( \stackrel{T}{\rightarrow} \) base logic term (\( t \))

Local meta-logic formula \( \phi \) \( \stackrel{T}{\rightarrow} \) base logic term (\( \phi \))

Global meta-logic formulas are first-order formulas over \( [\phi] \) and \( t \sim t' \) atoms.

Quantifiers in local meta-logic formulas

- Only allowed over index and timestamp, which are interpreted in finite domains.
- Quantifications translate to finite boolean combinations.
Example (Local meta-logic formulas)

- $\text{output@A}(i, j) = h(\text{input@A}(i, j), k(i))$
Example (Local meta-logic formulas)

- \( \text{output@A}(i, j) = h(\text{input@A}(i, j), k(i)) \)
- \( \forall k. \ \text{cond@B}(k) \Rightarrow \exists i, j. \ \text{A}(i, j) < \text{B}(k) \land \text{input@B}(k) = \text{output@A}(i, j) \)
CCSA meta-logic: reasoning about protocols

Foundation of the Squirrel proof-assistant [BDJKM, SP’21 & BDKM, CSF’22]

Example (Local meta-logic formulas)

- $\text{output}@A(i,j) = h(\text{input}@A(i,j), k(i))$
- $\forall k. \text{cond}@B(k) \Rightarrow \exists i, j. A(i,j) < B(k) \land \text{input}@B(k) = \text{output}@A(i,j)$

Reasoning over all trace models $\mathbb{T}$ for protocol $\mathcal{P}$, and all computational models $\mathcal{M}$.

Meta-logic term $t \xrightarrow{\mathbb{T}}$ base logic term $(t)_{\mathbb{T}} \xrightarrow{\mathcal{M}}$ PPTM returning bitstring

Local meta-logic formula $\varphi \xrightarrow{\mathbb{T}}$ base logic term $(\varphi)_{\mathbb{T}} \xrightarrow{\mathcal{M}}$ PPTM returning boolean

Global meta-logic formulas are first-order formulas over $[\varphi]$ and $t \sim t'$ atoms.
CCSA meta-logic: reasoning about protocols

Foundation of the Squirrel proof-assistant [BDJKM, SP’21 & BDKM, CSF’22]

Example (Local meta-logic formulas)

- \( \text{output@A}(i, j) = h(\text{input@A}(i, j), k(i)) \)
- \( \forall k. \text{cond@B}(k) \Rightarrow \exists i, j. A(i, j) < B(k) \land \text{input@B}(k) = \text{output@A}(i, j) \)

Reasoning over all trace models \( \mathbb{T} \) for protocol \( \mathcal{P} \), and all computational models \( \mathcal{M} \).

Meta-logic term \( t \xrightarrow{\mathbb{T}} \) base logic term \( (t)\xrightarrow{\mathcal{M}} \) PPTM returning bitstring

Local meta-logic formula \( \phi \xrightarrow{\mathbb{T}} \) base logic term \( (\phi)\xrightarrow{\mathcal{M}} \) PPTM returning boolean

Global meta-logic formulas are first-order formulas over \([\phi]\) and \( t \sim t' \) atoms.

Quantifiers in local meta-logic formulas

- Only allowed over index and timestamp, which are interpreted in finite domains.
- Quantifications translate to finite boolean combinations.
Higher-order CCSA logic

Letting go (at first) of PTIME, computability, bitstrings, protocols...

Terms of the old base logic:
probability polynomial-time machines.

Terms of the new logic:
$\eta$-indexed families of random variables.

$\lceil t \rceil_M(\eta, \rho) \in \{0, 1\}^*$

$\lceil t \rceil_M^\eta \in [\tau]^\eta_M$
Higher-order CCSA logic

Letting go (at first) of PTIME, computability, bitstrings, protocols...

Terms of the old base logic: probabilistic polynomial-time machines.

\[
[t]_M(\eta, \rho) \in \{0, 1\}^*
\]

Terms of the new logic: \(\eta\)-indexed families of random variables.

\[
[t]_M^{\eta; \rho} \in \left[\tau\right]_M^{\eta}
\]

Benefits

- Quantifiers at all types in local formulas, e.g. \(\forall \tau : (\tau \rightarrow \text{bool}) \rightarrow \text{bool}:

\[
\left[\forall \tau (\lambda x : \tau. \varphi)\right]_M^{\eta; \rho} = \text{true} \quad \text{when} \quad \left[\lambda x : \tau. \varphi\right]_M^{\eta; \rho} = a \in \left[\tau\right]_M^{\eta} \rightarrow \text{true}
\]
Higher-order CCSA logic

Letting go (at first) of PTIME, computability, bitstrings, protocols...

Terms of the old base logic:
probabilistic polynomial-time machines.
\[ [t]_M(\eta, \rho) \in \{0, 1\}^* \]

Terms of the new logic:
\( \eta \)-indexed families of random variables.
\[ [t]_M^{\eta; \rho} \in \llbracket \tau \rrbracket_\eta \]

Benefits

- Quantifiers at all types in local formulas, e.g. \( \forall \tau : (\tau \rightarrow \text{bool}) \rightarrow \text{bool} \):
  \[ \llbracket \forall \tau (\lambda x : \tau. \varphi) \rrbracket_\eta^{\eta; \rho} = \text{true} \quad \text{when} \quad \llbracket \lambda x : \tau. \varphi \rrbracket_\eta^{\eta; \rho} = a \in \llbracket \tau \rrbracket_\eta \rightarrow \text{true} \]

- Ability to talk about useful non-PTIME functions, e.g. discrete logarithm.
- Express abstract reasoning schemes using HOL, e.g. hybrid argument.
Recovering the meta-logic

Restricting types and terms:

- Types index and timestamp fixed and finite: $[\tau]^\eta_M$ is the same finite set for all $\eta$.
- One can restrict some terms to be constant, deterministic, PTIME, adversarial, etc.

$\mathcal{M} \models \text{const}(t)$ when $[t]^\eta_\rho_M$ independent of $\eta, \rho$
Recovering the meta-logic

Restricting types and terms:
- Types index and timestamp fixed and finite: $\llbracket \tau \rrbracket^\eta_M$ is the same finite set for all $\eta$.
- One can restrict some terms to be constant, deterministic, PTIME, adversarial, etc.

$$M \models \text{const}(t) \quad \text{when} \quad \llbracket t \rrbracket^\eta_\rho_M \text{ independent of } \eta, \rho$$

Recursive definitions:
- We allow recursive definitions with a semantical well-foundedness criterion.
- Macros of the meta-logic can be recovered, e.g. $\text{input}_P, \text{output}_P : \text{timestamp} \to \text{message}$. 
Proof system

$\mathcal{E}; \Theta \vdash \Phi$ reads as $\forall \mathcal{E}. \wedge \Theta \Rightarrow \Phi$

$\mathcal{E}; \Theta; \Gamma \vdash \varphi$ reads as $\forall \mathcal{E}. \wedge \Theta \Rightarrow [\wedge \Gamma \Rightarrow \varphi]$

$\Theta \colon$ global formulas (FO formulas)

$\Gamma, \varphi \colon$ local formulas (boolean terms)

$\mathcal{E}; \Theta; \Gamma, \varphi_1 \vdash \psi \quad \mathcal{E}; \Theta; \Gamma, \varphi_2 \vdash \psi$

\[ \frac{}{\mathcal{E}; \Theta; \Gamma, \varphi_1 \lor \varphi_2 \vdash \psi} \]

$\mathcal{E}; \Theta, [\varphi_1]; \Gamma \vdash \psi \quad \mathcal{E}; \Theta, [\varphi_2]; \Gamma \vdash \psi \quad \mathcal{E}; \Theta \vdash \text{const}(\varphi_1) \lor \text{const}(\varphi_2)$

\[ \frac{}{\mathcal{E}; \Theta, [\varphi_1 \lor \varphi_2]; \Gamma \vdash \psi} \]
### Proof system

\[ \mathcal{E}; \Theta \vdash \Phi \] reads as \( \forall \mathcal{E}. \land \Theta \Rightarrow \Phi \)

\[ \mathcal{E}; \Theta; \Gamma \vdash \varphi \] reads as \( \forall \mathcal{E}. \land \Theta \Rightarrow [\land \Gamma \Rightarrow \varphi] \)

\[ \mathcal{E}; x; \Theta \vdash \Phi \] \( x \notin \mathcal{E} \)

\[ \mathcal{E}; \Theta \vdash \forall x. \Phi \] \( x \notin \mathcal{E} \)

\[ \mathcal{E}, x; \Theta \vdash \varphi \] \( x \notin \mathcal{E} \)

\[ \mathcal{E}; \Theta; \Gamma \vdash \forall x. \varphi \] \( x \notin \mathcal{E} \)

---

### Theorem (Equivalence between local and global quantifiers)

\[ \mathcal{M} \models [\forall (x: \tau). [\varphi]] \quad \text{iff} \quad \mathcal{M} \models [\forall (x: \tau). \varphi] \]

"for any random variable \( x \) over \([\tau]\), \( \varphi \) holds almost surely"

"almost surely, \( \varphi \) holds for any value \( x \in [\tau] \)"
Proof system

\[ \mathcal{E} : \text{environment} \]
\[ \Theta : \text{global formulas (FO formulas)} \]
\[ \Gamma, \varphi : \text{local formulas (boolean terms)} \]

\[ \mathcal{E}; \Theta \vdash \Phi \quad \text{reads as} \quad \forall \mathcal{E}. \land \Theta \Rightarrow \Phi \]
\[ \mathcal{E}; \Theta; \Gamma \vdash \varphi \quad \text{reads as} \quad \forall \mathcal{E}. \land \Theta \Rightarrow [\land \Gamma \Rightarrow \varphi] \]

\[ \mathcal{E}, x; \Theta \vdash \Phi \quad x \notin \mathcal{E} \]
\[ \mathcal{E}; \Theta \vdash \forall x. \Phi \]
\[ \mathcal{E}, x; \Theta; \Gamma \vdash \varphi \quad x \notin \mathcal{E} \]
\[ \mathcal{E}; \Theta; \Gamma \vdash \forall x. \varphi \]

Theorem (Equivalence between local and global quantifiers)

\[ M \models \forall (x : \tau).[\varphi] \quad \text{iff} \quad M \models [\forall (x : \tau). \varphi] \]

“for any random variable \( x \) over \( [\tau] \), \( \varphi \) holds almost surely”

“almost surely, \( \varphi \) holds for any value \( x \in [\tau] \)”

If \( \tau \) is assumed fixed and finite, this is also equivalent to \( M \models \forall (x : \tau). \text{const}(x) \Rightarrow [\varphi] \).
Advanced axioms

The base logic comes with axioms for reasoning about names and crypto primitives. E.g. \([n \neq t]\) valid when \(t\) is ground and does not contain \(n\).

We lifted these axioms to the meta-logic:

- Occurrence conditions: account for potential macros unrollings.
- Take into account constant local formulas conditioning occurrences.

For our higher-order logic, we justify improved axioms from first principles:

- Occurrence conditions: account for unrollings of recursive definitions.
- Take into account arbitrary local formulas as conditions.
- Resulting axioms systematically handle bad occurrences, i.e. corruption.
  Case study on forward secrecy for a DH key exchange.
Conclusion

Higher-order CCSA logic

• Strictly generalizes former CCSA meta-logic and proof system.
• Decouples core logic from protocol-specific declarations and recursive definitions.
• Fragment corresponding to former meta-logic implemented in Squirrel proof assistant. All past proof developments have been ported; new ones added.

Future work

• More complex proofs of protocols.
• Proof-theoretical investigations, automated reasoning.
• Modelling other classes of protocols and attacker models.
• Relationship with other works in higher-order crypto.