Formalizing the Lazy Intruder in Isabelle:
Towards Formalized Protocol Compositionality Results

Andreas V. Hess    Sebastian Mödersheim

DTU Compute, Danmarks Tekniske Universitet, Denmark

August, 2016

Part of a Sapere Aude research project
Relative Soundness

Examples from [Almousa et al., 2015]:

**Theorem (Typing result)**

> If $P$ is a type-flaw resistant protocol and has an attack, then $P$ has a well-typed attack

---

<table>
<thead>
<tr>
<th>If $P$ is a type-flaw resistant protocol and has an attack, then $P$ has a well-typed attack.</th>
<th>Example: type-flaw attack</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A \rightarrow B$: $M, A, B, \text{scrypt}(k, f_1(M, A, B))$</td>
<td>$B \rightarrow A$: $M, \text{scrypt}(k, f_2(N_A, K_{AB}))$</td>
</tr>
</tbody>
</table>

Wrapping in different formats/tags (part of type-flaw resistance) makes such attacks unnecessary.

The proofs of these theorems depend on the lazy intruder.
Relative Soundness

Examples from [Almousa et al., 2015]:

**Theorem (Typing result)**

*If P is a type-flaw resistant protocol and has an attack, then P has a well-typed attack*

**Theorem (Parallel compositionality)**

*If $P_1$ and $P_2$ are parallel-composable and $P_1 \parallel P_2$ has an attack then either $P_1$ or $P_2$ has an attack in isolation*
Relative Soundness

Examples from [Almousa et al., 2015]:

**Theorem (Typing result)**

*If* *P* *is a type-flaw resistant protocol and has an attack, then* *P* *has a well-typed attack*

**Theorem (Parallel compositionality)**

*If* *P*<sub>1</sub> *and* *P*<sub>2</sub> *are parallel-composable and* *P*<sub>1</sub> || *P*<sub>2</sub> *has an attack then either* *P*<sub>1</sub> *or* *P*<sub>2</sub> *has an attack in isolation*

**Example:** type-flaw attack; *K<sub>AB</sub> → (M, A, B)*

\[
A \rightarrow B: \quad M, A, B, \text{scrypt}(k, (M, A, B))
\]

\[
B \rightarrow A: \quad M, \text{scrypt}(k, (N_A, K_{AB}))
\]

**Attack:** Unifying *K<sub>AB</sub>* and *(M, A, B)* enables the intruder to send the second message ⇒ *K<sub>AB</sub>* becomes known
Relative Soundness

Examples from [Almousa et al., 2015]:

**Theorem (Typing result)**

_If P is a type-flaw resistant protocol and has an attack, then P has a well-typed attack_

**Theorem (Parallel compositionality)**

_If P₁ and P₂ are parallel-composable and P₁ \parallel P₂ has an attack then either P₁ or P₂ has an attack in isolation_

**Example:** type-flaw attack

\[
A \rightarrow B : \quad M, A, B, \text{scrypt}(k, f₁(M, A, B))
\]
\[
B \rightarrow A : \quad M, \text{scrypt}(k, f₂(N_A, K_{AB}))
\]

Wrapping in different formats/tags (part of type-flaw resistance) makes such attacks unnecessary
Relative Soundness
Examples from [Almousa et al., 2015]:

**Theorem (Typing result)**

If \( P \) is a type-flaw resistant protocol and has an attack, then \( P \) has a well-typed attack.

**Theorem (Parallel compositionality)**

If \( P_1 \) and \( P_2 \) are parallel-composable and \( P_1 \parallel P_2 \) has an attack then either \( P_1 \) or \( P_2 \) has an attack in isolation.

**Example:** type-flaw attack

\[
A \rightarrow B : \ M, A, B, \text{scrypt}(k, f_1(M, A, B)) \\
B \rightarrow A : \ M, \text{scrypt}(k, f_2(N_A, K_{AB}))
\]

Wrapping in different formats/tags (part of type-flaw resistance) makes such attacks unnecessary.

The proofs of these theorems depend on the lazy intruder A. Hess, S. Mödersheim

Lazy Intruder in Isabelle August, 2016 2 / 15
The Lazy Intruder
What is the lazy intruder?
The Lazy Intruder

What is the lazy intruder?

- Set of constraint reduction rules (Unify, Compose/Synthesis, Decomposition/Analysis...)
- Constraints on Dolev-Yao style intruder deduction
- Is sound, complete, and terminating:
  \[ \exists \text{simple constraint } \psi. \phi \leadsto^* \psi \land I \models \psi \text{ iff } I \models \phi, \]
  \[ \{ \psi \mid \phi \leadsto^* \psi \} \text{ is finite} \]
The Lazy Intruder

What is the lazy intruder?

- Set of constraint reduction rules (Unify, Compose/Synthesis, Decomposition/Analysis...)
- Constraints on Dolev-Yao style intruder deduction
- Is sound, complete, and terminating:
  \[
  (\exists \text{simple constraint } \psi. \phi \leadsto^* \psi \land I \models \psi) \iff I \models \phi, \\
  \{\psi | \phi \leadsto^* \psi\} \text{ is finite}
  \]

Example constraint $C$:

\[
\{pk, \text{crypt}(pk, secret)\} \vdash \text{crypt}(pk, X) \land \{pk, \text{crypt}(pk, secret), h(X)\} \vdash Y
\]

One solution is the following:

\[
C \leadsto \{pk, \text{crypt}(pk, secret), h(secret)\} \vdash Y
\]

Constraint reduced to a simple (always satisfiable) constraint
The Lazy Intruder

Normally used for efficiency/completeness in model-checking
The Lazy Intruder

Normally used for efficiency/completeness in model-checking

But also used as a proof technique to show relative soundness theorems
- for a certain class of protocols,
- if there is an attack,
- then there is an attack with a certain property
The Lazy Intruder

Normally used for efficiency/completeness in model-checking

But also used as a proof technique to show relative soundness theorems

- for a certain class of protocols,
- if there is an attack,
- then there is an attack with a certain property

This is done as follows

- Any attack can be seen as a solution to a constraint
- Since the lazy intruder is complete, it will find a solution
- Show that all reduction steps preserve some invariant
  - e.g. no ill-typed instantiations of variables
- Show that the preservation implies the original property
- **Thus**: if there is an attack, then there is one where the solution has the property
Motivation: Unclear Argumentation

Why formalization in proof assistants (like Isabelle/HOL)?
Motivation: Unclear Argumentation

Why formalization in proof assistants (like Isabelle/HOL)?

- Pen and paper proofs of compositionality results often involve subtle details
Motivation: Unclear Argumentation

Why formalization in proof assistants (like Isabelle/HOL)?

- Pen and paper proofs of compositionality results often involve subtle details
- This can lead to unclear arguments and "mistakes"
Motivation: Unclear Argumentation

Why formalization in proof assistants (like Isabelle/HOL)?

- Pen and paper proofs of compositionality results often involve subtle details
- This can lead to unclear arguments and "mistakes"

Example: Part of a typing result proof [Almousa et al., 2015]:

(Equation). For the (Unify) rule, we proceed by cases of \( s \) and \( t \):

- If both \( s \) and \( t \) are atomic, then \( s \) and \( t \) cannot be variables, so the above property is preserved trivially, simply because they must be the same constant.
- If both are composed, then \( \sigma(s) = \sigma(t) \) and there exist \( u, v \in SMP \) and \( \vartheta_1, \vartheta_2 \) such that \( \vartheta_1(u) = s \) and \( \vartheta_2(v) = t \). Then, \( \sigma(\vartheta_1(u)) = \sigma(\vartheta_2(v)) \) and \( \Gamma(u) = \Gamma(v) = \Gamma(s) = \Gamma(t) \) as the protocol is type-flaw-resistant, and so \( \sigma \) is well-typed.
- If \( t \) is variable, then it is simple and we proved earlier that if it has an ill-typed solution, then it also has a well-typed one.
Motivation: Unclear Argumentation

Why formalization in proof assistants (like Isabelle/HOL)?

- Pen and paper proofs of compositionality results often involve subtle details
- This can lead to unclear arguments and "mistakes"

Proof assistants provide a very high guarantee of correctness
Motivation: Unclear Argumentation

Why formalization in proof assistants (like Isabelle/HOL)?

- Pen and paper proofs of compositionality results often involve subtle details
- This can lead to unclear arguments and "mistakes"

Proof assistants provide a very high guarantee of correctness

- Only requires trust in the proof assistant’s core
Motivation: Unclear Argumentation

Why formalization in proof assistants (like Isabelle/HOL)?

- Pen and paper proofs of compositionality results often involve subtle details
- This can lead to unclear arguments and "mistakes"

Proof assistants provide a very high guarantee of correctness

- Only requires trust in the proof assistant’s core
- ... but requires a huge time investment for proof development
  - Informal arguments not accepted
- Simpler definitions leading to simpler proofs can be useful
Motivation: Unification of Results

Compositionality results in the literature make slightly different assumptions in their models

- May not be compatible with each other
Motivation: Unification of Results

Compositionality results in the literature make slightly different assumptions in their models

- May not be compatible with each other

Constraint systems (e.g. lazy intruder) are used as proof techniques for relative soundness results

- Proof assistant formalization can aid in unifying such results
Motivation: Unification of Results

Compositionality results in the literature make slightly different assumptions in their models

- May not be compatible with each other

Constraint systems (e.g. lazy intruder) are used as proof techniques for relative soundness results

- Proof assistant formalization can aid in unifying such results

Contributions (finished, modulo some details):

- Formalization of a lazy intruder in Isabelle/HOL
- Formalization of a typing result based on the lazy intruder
- Work towards formalization of a parallel compositionality result based on the typing result
Simplification: Constraints As Strands

The constraints must have monotonically growing intruder knowledges and the variables must originate from the intruder.
Simplification: Constraints As Strands

The constraints must have monotonically growing intruder knowledges and the variables must originate from the intruder

Example:

\[
\{pk, \text{crypt}(pk, secret)\} \vdash \text{crypt}(pk, X)
\]
\[
\wedge \{pk, \text{crypt}(pk, secret), h(X)\} \vdash \ldots
\]
Simplification: Constraints As Strands

The constraints must have monotonically growing intruder knowledges and the variables must originate from the intruder

Example:

\[ \{pk, crypt(pk, secret)\} \vdash crypt(pk, X) \]
\[ \land \{pk, crypt(pk, secret), h(X)\} \vdash \ldots \]
Simplification: Constraints As Strands

The constraints must have monotonically growing intruder knowledges and the variables must originate from the intruder.

Example:

\{pk, \text{crypt}(pk, secret)\} \vdash \text{crypt}(pk, X) \\
\wedge \{pk, \text{crypt}(pk, secret), h(X)\} \vdash \ldots

An easier representation:

\[
\begin{align*}
&\vdash pk \\
&\vdash \text{crypt}(pk, secret) \\
&\vdash \text{crypt}(pk, X) \\
&\vdash h(X) \\
&\vdash \ldots
\end{align*}
\]

Intruder knowledges implicit, monotonically growing.
Simplification: Analysis As Protocol Steps

Solving requires **analysis** of the term `scrypt(key, secret)`
Simplification: Analysis As Protocol Steps

Solving requires analysis of the term \( \text{scrypt}(key, secret) \)

But: Proving completeness + termination is difficult when analysis steps are present
Simplification: Analysis As Protocol Steps

Solving requires analysis of the term $\text{scrypt}(key, secret)$

But: Proving completeness + termination is difficult when analysis steps are present

- Termination measure needs to keep track of analyzed terms
Simplification: Analysis As Protocol Steps

Solving requires **analysis** of the term \( \text{scrypt}(key, secret) \)

**But:** Proving completeness + termination is difficult when analysis steps are present

- Termination measure needs to keep track of analyzed terms
- Completeness proof based on traversing or restricting derivation trees
Example: Informal Reasoning

Part of proving completeness of a lazy intruder constraint system [Cortier et al., 2007]

Definition 16 (simple) We say that a proof $\pi$ is simple if

1. any subproof of $\pi$ is left-minimal,
2. a composition rule of the form $\frac{u_1 \quad u_2}{u}$ is not followed by a decomposition rule leading to $u_1$ or $u_2$,

Lemma 2 Let $C$ be an unsolved constraint system, $\theta$ be a solution of $C$ and $T_i \vdash u_i$ be a minimal unsolved constraint of $C$. Let $u$ be a term. If there is a simple proof of $T_i\theta \vdash u$ having the last rule an axiom or a decomposition then there is $t \in St(T_i) \setminus X$ such that $t\theta = u$. 
Example: Informal Reasoning

Part of proving completeness of a lazy intruder constraint system [Almousa et al., 2015]

- If the node is an application of the \((\text{Decompose})\) rule, then consider the ground term \(t\) that is being decomposed in the derivation proof for \(I(t_i)\). We first consider different cases depending on how \(t\) is derived:
  - If \(t\) is obtained by a decomposition step itself, then we regress to the respective term being decomposed, and we do so until we hit a term that is not obtained by decomposition. By the previous cases, this cannot
Simplification: Analysis As Protocol Steps

Solving requires analysis of the term `scrypt(key, secret)`

But: Proving completeness + termination is difficult when analysis steps are present

- Termination measure needs to keep track of analyzed terms
- Completeness proof based on traversing or restricting derivation trees

Idea: Analysis as protocol steps
Example: Analysis As Protocol Steps

*With explicit analysis*

Only finitely many analyzable terms given a finite intruder knowledge II-l-typed unification between variables possible!
Example: Analysis As Protocol Steps

*With explicit analysis*

\[
\text{scrypt}(\text{key}, \text{secret})
\]
\[
\leftarrow
\]
\[
\rightarrow
\]
\[
\text{f}_1(\text{scrypt}(K,M))
\]
\[
\rightarrow
\]
\[
\leftarrow
\]
\[
\text{f}_2(K)
\]
\[
\rightarrow
\]
\[
\leftarrow
\]
\[
\text{f}_3(M)
\]
\[
\rightarrow
\]
\[
\leftarrow
\]
\[
\text{secret}
\]

Messages wrapped in **formats** to prevent ill-typed unification between variables (needed for typing result)
Simplification: Analysis As Protocol Steps

Analysis/decomposition of formats much easier

- Does not require additional constraints
- Only needs to happen once before any other derivation
Simplification: Analysis As Protocol Steps

Analysis/decomposition of formats much easier

- Does not require additional constraints
- Only needs to happen once before any other derivation

\[ trp(\{f_1(f_2(a)), f_3(b)\}) = \{f_1(f_2(a)), f_2(a), a, f_3(b), b\} \]
Simplification: Analysis As Protocol Steps

Analysis/decomposition of formats much easier

- Does not require additional constraints
- Only needs to happen once before any other derivation

\[ \text{trp}([f_1(f_2(a)), f_3(b)]) = \{f_1(f_2(a)), f_2(a), a, f_3(b), b\} \]

... but still not trivial

- Solution might contain formats in image

\[ \mathcal{I}(\text{trp}(\mathcal{M})) \subseteq \text{trp}(\mathcal{I}(\mathcal{M})) \]
Simplification: Analysis As Protocol Steps

Analysis/decomposition of formats much easier

- Does not require additional constraints
- Only needs to happen once before any other derivation

\[ \text{trp}(\{f_1(f_2(a)), f_3(b)\}) = \{f_1(f_2(a)), f_2(a), a, f_3(b), b\} \]

... but still not trivial

- Solution might contain formats in image

\[ I(\text{trp}(M)) \subseteq \text{trp}(I(M)) \]

For well-formed constraints:

**Theorem**

\[ \text{trp}(I(M)) \vdash c t \text{ if and only if } I(\text{trp}(M)) \vdash c t \]
Conclusion

Formalization of the lazy intruder in Isabelle/HOL
- With simplifications
- Soundness, completeness, termination proved

Relative soundness typing theorem formalized
- "exists attack $\Rightarrow$ exists well-typed attack"

Future work (compositionality!)
- Formalize parallel compositionality theorem of [Almousa et al., 2015]
- Formalize and unify other results based on the lazy intruder, e.g. [Cortier et al., 2007]