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

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_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; $[K_{AB} \mapsto (M, A, B)]$

$$\begin{aligned} A \rightarrow B: &\quad M, A, B, \mathrm{scrypt}(k, (M, A, B)) \\ B \rightarrow A: &\quad M, \mathrm{scrypt}(k, (N_A, K_{AB})) \end{aligned}$$

Attack: Unifying K_{AB} and (M, A, B) enables the intruder to send the second message $\Longrightarrow$ K_{AB} 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 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

$$\begin{array}{ll} A \rightarrow B: & M, A, B, \mathrm{scrypt}(k, f_1(M, A, B)) \\ B \rightarrow A: & M, \mathrm{scrypt}(k, f_2(N_A, K_{AB})) \end{array}$$

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

$$\begin{aligned} A \to B: &\quad M, A, B, \mathrm{scrypt}(k, f_1(M, A, B)) \\ B \to A: &\quad M, \mathrm{scrypt}(k, f_2(N_A, K_{AB})) \end{aligned}$$

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

The proofs of these theorems depend on the lazy intruder

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:
  ($\exists$simple constraint ψ. $\phi \leadsto^* \psi \land \mathcal{I} \models \psi$) iff $\mathcal{I} \models \phi$,
  $\{\psi \mid \phi \leadsto^* \psi\}$ 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:
  ($\exists$simple constraint ψ. $\phi \rightsquigarrow^* \psi \wedge \mathcal{I} \models \psi$) iff $\mathcal{I} \models \phi$,
  $\{\psi \mid \phi \rightsquigarrow^* \psi\}$ is finite

Example constraint C:

$$\{pk, \mathrm{crypt}(pk, secret)\} \vdash \mathrm{crypt}(pk, X) \wedge \{pk, \mathrm{crypt}(pk, secret), h(X)\} \vdash Y$$

one solution is the following:

$$C \rightsquigarrow \{pk, \mathrm{crypt}(pk, secret), h(secret)\} \vdash Y$$

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 ϑ_1, ϑ_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 σ 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:

$$\begin{aligned} &\{pk, \mathrm{crypt}(pk, secret)\} \vdash \mathrm{crypt}(pk, X) \\ \wedge\, &\{pk, \mathrm{crypt}(pk, secret), h(X)\} \vdash \cdots \end{aligned}$$

Simplification: Constraints As Strands

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

Example:

$$
\begin{aligned}
&\{pk, \mathrm{crypt}(pk, secret)\} \vdash \mathrm{crypt}(pk, X) \\
\wedge\, &\{pk, \mathrm{crypt}(pk, secret), h(X)\} \vdash \cdots
\end{aligned}
$$

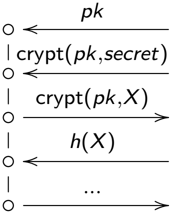
Simplification: Constraints As Strands

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

Example:

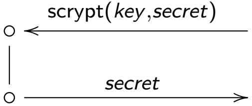
$$\begin{aligned} &\{pk, \mathrm{crypt}(pk, secret)\} \vdash \mathrm{crypt}(pk, X) \\ \wedge\, &\{pk, \mathrm{crypt}(pk, secret), h(X)\} \vdash \cdots \end{aligned}$$

An easier representation:



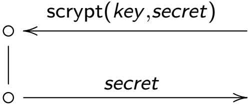
Intruder knowledges implicit, monotonically growing

Simplification: Analysis As Protocol Steps



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

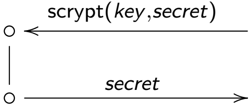
Simplification: Analysis As Protocol Steps



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

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

Simplification: Analysis As Protocol Steps

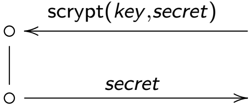


Solving requires analysis of the term scrypt(*key*, *secret*)

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 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 π is *simple* if

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

Lemma 2 *Let $\mathcal{C}$ be an unsolved constraint system, θ be a solution of $\mathcal{C}$ and $T_i \Vdash u_i$ be a minimal unsolved constraint of $\mathcal{C}$. 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 $t \in St(T_i) \setminus \mathcal{X}$ such that $t\theta = u$.*

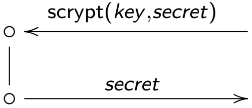
Example: Informal Reasoning

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

– If the node is an application of the (*Decompose*) rule, then consider the ground term t that is being decomposed in the derivation proof for $\mathcal{I}(t_i)$. 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, $secret$)

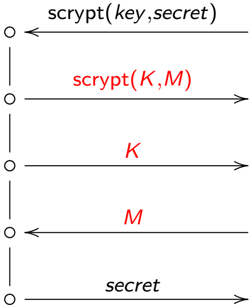
But: Proving completeness + termination is difficult when analysis steps are present

- 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

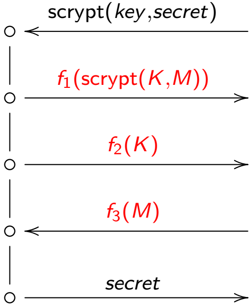
With explicit analysis



Only finitely many analyzable terms given a finite intruder knowledge
Ill-typed unification between variables possible!

Example: Analysis As Protocol Steps

With explicit analysis



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

$$trp(\{f_1(f_2(a)), f_3(b)\}) = \{f_1(f_2(a)), f_2(a), a, f_3(b), b\}$$

... but still not trivial

- Solution might contain formats in image

$$\mathcal{I}(trp(\mathcal{M})) \subseteq trp(\mathcal{I}(\mathcal{M}))$$

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

$$trp(\{f_1(f_2(a)), f_3(b)\}) = \{f_1(f_2(a)), f_2(a), a, f_3(b), b\}$$

... but still not trivial

- Solution might contain formats in image

$$\mathcal{I}(trp(\mathcal{M})) \subseteq trp(\mathcal{I}(\mathcal{M}))$$

For well-formed constraints:

Theorem

$trp(\mathcal{I}(\mathcal{M})) \vdash_c t$ if and only if $\mathcal{I}(trp(\mathcal{M})) \vdash_c t$

Conclusion

Formalization of the lazy intruder in Isabelle/HOL

- With simplifications
- Soundness, completeness, termination proved

Relative soundness typing theorem formalized

- "exists attack $\Longrightarrow$ 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]