On Bana-Comon Logic

Seminar-Formal Methods in IT-Security

Zixuan Fan
Lehrstuhl für Sicherheit in der Informatik

Wintersemester 2022/23
8. Februar 2023

Outline

1. Backgrounds
2. The complexity analysis
3. A Case study of Squirrel
4. Summary

An overview of 2 kinds of models for communication protocols:

Model	Dolev-Yao	Computation
Primitives	Blackbox functions	Maps from bitstrings to bitstrings
Adversary	Restricted behaviour	Probablistic Turing Machines
Range	Small	Large
Formalisation and Verification	Easy	Hard

How can we take adavantage of the both models?

What is an interactive theorem prover(ITP)?

What is an interactive theorem prover(ITP)?

A software tool that assists with the development of formal proofs by human-machine collaboration.

How are they important in terms of protocol security?

How are they important in terms of protocol security?

Automation, easying modelling, universal.

How do our study take advantage of them?

How do our study take advantage of them?

Our study is based on the 2 ITPs, Isabelle and Squirrel.

Recall the question of 2 kinds of protocols: How to take advantage of them both?

Recall the question of 2 kinds of protocols: How to take advantage of them both?
Answer: **BC-logic**

Recall the question of 2 kinds of protocols: How to take advantage of them both?
Answer: **BC-logic**
Some new aspects:

- Efficiency $\Longrightarrow$ Complexity analysis
- Usefulness $\Longrightarrow$ Squirrel

Recall the question of 2 kinds of protocols: How to take advantage of them both?
Answer: **BC-logic**
Some new aspects:

- Efficiency $\Longrightarrow$ Complexity analysis
- Usefulness $\Longrightarrow$ Squirrel

To summarize: Automation of the proof of protocols under the computational model

Outline

1. Backgrounds
2. The complexity analysis
3. A Case study of Squirrel
4. Summary

The Complexity Analysis of BC-logic

- The evaluation of BC-logic terms
- The folding process of the protocols
- Examples in Isabelle

The Complexity Analysis of BC-logic

- The evaluation of BC-logic terms
- The folding process of the protocols
- Examples in Isabelle

How should we show that a variable is in certain complexity classes?

The Complexity Analysis of BC-logic

- The evaluation of BC-logic terms
- The folding process of the protocols
- Examples in Isabelle

How should we show that a variable is in certain complexity classes?

$$\exists a\ b. f(n) = a \cdot n + b \Longrightarrow f \in \mathcal{O}(n)$$

Our definitions

$$
\begin{aligned}
\mathit{msg} &:= \textbf{EMPTY} \mid \textbf{ITE}\ \mathit{bl}\ \mathit{msg}\ \mathit{msg} \\
&\quad \mid \textbf{CONS}\ \mathit{msg}\ \mathit{msg} \mid \textbf{ATOM}\ \mathit{string} \\
\mathit{bl} &:= \textbf{EQ}\ \mathit{msg}\ \mathit{msg} \mid \textbf{EQL}\ \mathit{msg}\ \mathit{msg} \mid \textbf{T} \mid \textbf{F} \\
\mathit{functions} &: \textbf{T\_eval T\_bval msg\_len bl\_len}
\end{aligned}
$$

Our definitions

$$\begin{aligned} msg &:= \textbf{EMPTY} \mid \textbf{ITE}\ bl\ msg\ msg \\ &\quad \mid \textbf{CONS}\ msg\ msg \mid \textbf{ATOM}\ string \\ bl &:= \textbf{EQ}\ msg\ msg \mid \textbf{EQL}\ msg\ msg \mid \textbf{T} \mid \textbf{F} \\ functions &: \textbf{T\_eval T\_bval msg\_len bl\_len} \end{aligned}$$

Our goals

$$\exists a\ b.\textbf{T\_eval}\ msg \leq a \cdot (\textbf{msg\_len}\ msg) + b$$
$$\exists a\ b.\textbf{T\_bval}\ bl \leq a \cdot (\textbf{bl\_len}\ bl) + b$$

Time for Examples...

Evaluation of our modelling

- Parallel channels

Evaluation of our modelling

- Parallel channels
- Logical operators, e.g. **AND XOR**

Evaluation of our modelling

- Parallel channels
- Logical operators, e.g. **AND XOR**
- Length of messages, bit-streams or characters?

How should we model a series of communication in a protocol?

How should we model a series of communication in a protocol?

As a graph, as a set of transition rules, or ...?

How should we model a series of communication in a protocol?

As a graph, as a set of transition rules, or ...?
Our choice: as a tree!

How should we model a series of communication in a protocol?

As a graph, as a set of transition rules, or ...?
Our choice: as a tree!

Consider the following transition rules:

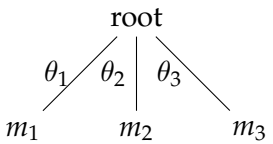
$$root, a \xrightarrow{\theta_1} m_1, b \qquad root, a \xrightarrow{\theta_2} m_2, b \qquad root, a \xrightarrow{\theta_3} m_3, b$$

$$m_2, b \xrightarrow{\eta_1} n_1, c \qquad m_3, b \xrightarrow{\eta_2} n_2, c \qquad m_3, b \xrightarrow{\eta_3} n_3, c$$

Consider the following transition rules:

$$root, a \xrightarrow{\theta_1} m_1, b \qquad root, a \xrightarrow{\theta_2} m_2, b \qquad root, a \xrightarrow{\theta_3} m_3, b$$

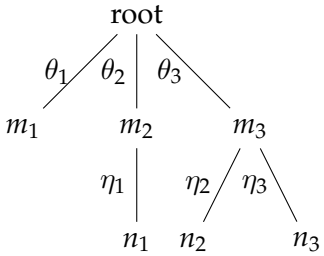
$$m_2, b \xrightarrow{\eta_1} n_1, c \qquad m_3, b \xrightarrow{\eta_2} n_2, c \qquad m_3, b \xrightarrow{\eta_3} n_3, c$$



Consider the following transition rules:

$$root, a \xrightarrow{\theta_1} m_1, b \qquad root, a \xrightarrow{\theta_2} m_2, b \qquad root, a \xrightarrow{\theta_3} m_3, b$$

$$m_2, b \xrightarrow{\eta_1} n_1, c \qquad m_3, b \xrightarrow{\eta_2} n_2, c \qquad m_3, b \xrightarrow{\eta_3} n_3, c$$



How do we profit from this tree structure?

How do we profit from this tree structure?

Folding!

root $\overset{\text{true}}{\dashrightarrow}$ if θ_1 then m_1
else if θ_2 then (m_2;
if η_1 then n_1
else empty)
else (m_3;
if η_2 then n_2
else n_3)

Our definitions:

$$\begin{aligned} \mathit{proc\_trie} := {} & \mathbf{Rt}\ (\mathit{bl} \times \mathit{proc\_trie})\ \mathit{list} \\ & \mid \mathbf{Nd}\ \mathit{msg}\ (\mathit{bl} \times \mathit{proc\_trie})\ \mathit{list} \\ \mathbf{fun\ valid} :: {} & \mathit{proc\_trie} \Rightarrow \mathit{bool} \end{aligned}$$

Our goals:

$$\mathbf{eval}\ (\mathbf{get\_msg}\ \mathit{proc}) = \mathbf{eval}\ (\mathbf{fold}\ \mathit{proc})$$

$$\exists a\ b.\mathbf{valid}\ \mathit{proc} \Longrightarrow \mathbf{T\_fold}\ \mathit{proc} \leq a \cdot (\mathbf{sz}\ \mathit{proc}) + b$$

Have you found any problem with regards to the transition rules?

Have you found any problem with regards to the transition rules?

Evaluation:

- Protocol subjects are never repeated.

Have you found any problem with regards to the transition rules?

Evaluation:

- Protocol subjects are never repeated.
- Configuration of our tries/prefix trees.

Outline

1. Backgrounds
2. The complexity analysis
3. A Case study of Squirrel

A case study of Squirrel

Squirrel in an overview:

- 3 basic types of object: message, timestamp and index.
- Various proof tactics available. Among them, **prf(Pseudorandom Function)** and **euf(Existential Unforgeability)** are Important.

A case study of Squirrel

Technische Universität München

Our experiments:

1. The tutorial
2. The simple examples
3. Exploration on the Kerberos protocol

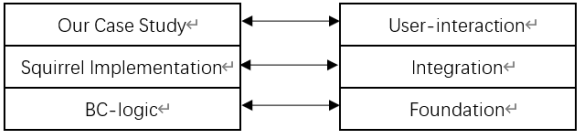
Conclusion: Not easy for starters, incomplete as of beta-version, good future to expect

Expectations:

- Post-quantum cryptographic schemes
- A larger library of examples
- Better automated proof tactics

Summary

Relation of our 2 parts of work



We have answered the 2 questions in the paper of BC-logic:

1. The evaluation and folding of BC-logic is efficient within polynomial time.
2. The BC-logic has its practical usage.

Questions?

"Rules of logic are to mathematics what those of structure to architecture."

— Bertrand Russell

Thanks!