股市日记

2015 年 10 月 26 日

An Overview of Our Approach

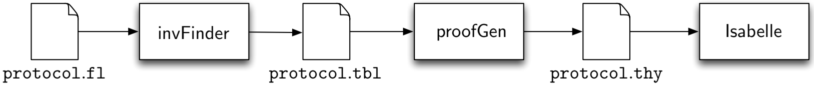

Figure: The workflow of paraVerifier.

指标是基础–2015-10-8

- 筹码指标与均线指标是两个基础指标
- 唯指标而动
- 指标应该有清晰的表示形式
- 即使是消息，也必须与技术指标结合，诺奖信息，医药股意义不大；全都开盘拉高出货；反而新铸股份满足指标爆发

State of Arts

- CMP: parameter abstraction and parameter abstraction
- Proposed, by McMillan, elaborated by Chou, Mannava, and Park (CMP) , and formalized by Krstic
- construction of an abstract instance which can simulate any protocol instance
- human provides auxiliary invariants (non-non-interference lemmas)

State of Arts

- invisible invariants,
- auxiliary invariants are computed from reachable state set in a finite protocol instance $P(c)$
- raw formula translated from BDD
- the reachable state set cann't be enumerated, e.g., the FLASH protocol

Two central and diffcult problems

- searching auxilairy invariants is not automatical
- soundness problem: the theoretical foundation is not mechanized, and a gap between existing approaches and a formal proof

Our Motivation

- automatically searching auxilairy invariants
- Formally proving all the things: both the theoretical foundation and case studies
- A formal proof script as a formal verification product

An Overview of Our Approach

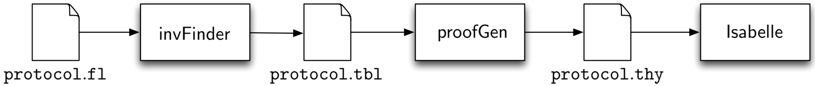

Figure: The workflow of paraVerifier.

Some Explanations

- paraVerifier=invFinder + proofGen
- protoocl.fl: a small cache coherence protocol instance
- invFinder searches auxiliary invariants automatically
- Some advanced theories: Kepler guess, Four-coloured problems

The Kepler Conjecture

“The Kepler Conjecture says that the ?cannonball packing? (see picture) is a densest packing of 3-dimensional balls of the same size. This was stated as a fact by Kepler in 1611 but only proved by Thomas Hales in 1998. His proof relies on a Java program for generating all (3000) possible counterexamples (all of which are then shown not to be counterexamples). With the help of Isabelle we were able to prove the correctness of a functional implementation of his Java program. Listen to Thomas Hales speaking about the proof (ABC Radio National Science Show, March 11th 2006). A formal proof of the Kepler conjecture was completed in 2014.”

Essentials of formal verification (by a theorem prover)

- Formally model the system (by a formula)
- Formalize the specification (by a formula)
- Prove that the model satisfies the spec (logical deduction)

Anonymity protocols

$$\begin{tikzcd}
A \arrow[dr, "ma"] & & D \\
& M \arrow[ur, "mb'"] \arrow[dr, "ma'"] & \\
B \arrow[ur, "mb"] & & C
\end{tikzcd}$$

(1)

$$\begin{tikzcd}
A \arrow[dr, "mb"] & & D \\
& M \arrow[ur, "mb'"] \arrow[dr, "ma'"] & \\
B \arrow[ur, "ma"] & & C
\end{tikzcd}$$

(2)

$$ma = \{\!|\text{Nonce } na, \text{Agent } C, \{\!|\text{Nonce } na'|\!\}_{\text{pubK } C}|\!\}_{\text{pubK } M},\ ma' = \{\!|\text{Nonce } na'|\!\}_{\text{pubK } C}$$

$$mb = \{\!|\text{Nonce } nb, \text{Agent } D, \{\!|\text{Nonce } nb'|\!\}_{\text{pubK } D}|\!\}_{\text{pubK } M},\ mb' = \{\!|\text{Nonce } nb'|\!\}_{\text{pubK } D}$$

A case study—Formal verification of anonymity protocols (by a theorem prover)

```
constdefs box::"agent⇒trace⇒trace set⇒  assertOfTrace⇒bool"
 "box A r rs Assert≡  ∀r'.r'∈rs⟶obsEquiv A r r' ⟶(Assert r')"


constdefs diamond::"agent⇒trace⇒trace set⇒  assertOfTrace⇒bool"
 "diamond A r rs Assert≡  ∃r'.r'∈rs ∧obsEquiv A r r'  ∧(Assert r')"
```

Formalization of anonymity properties

```
constdefs senderAnomity::"agent set⇒agent⇒msg⇒
 trace⇒trace set⇒bool"
 "senderAnomity AS B m r rs≡ (∀X. X∈AS⟶  r ⊨ ◇B rs (originates X m))"
constdefs unlinkability::"agent set⇒agent⇒msg⇒
 trace⇒trace set⇒bool"
 "unlinkability AS A m r rs≡
  (let P= λX m' r.  sends X m' r in  (¬ (r ⊨ □ Spy rs (P A m)) ∧
(∀X.X∈AS ⟶r ⊨ ◇Spy rs (P A m)))
```

Modelling Onion Routing Protocols

```
--- Formal inductive definition inductive_set oneOnionSession::"nat⇒agent
⇒trace set" for k::"nat" and M::"agent" where
 onionNil:  "[]∈ (oneOnionSession k M) "
 | onionCons1:  "[|tr∈(oneOnionSession k M);X≠M;Y≠M;
 Nonce n0∉(used tr);Nonce n∉(used tr); length tr<k|]
⟹ Says X M (Crypt (pubK M) { Nonce n0,Agent Y,Crypt (pubK Y) (Nonce n)}
#tr∈ oneOnionSession k M"
 | onionCons2:  "[|tr∈(oneOnionSession k M);X≠M;
 Nonce n∉(used tr);length tr<k|]⟹
 Says X M (Crypt (pubK M) (Nonce n))#tr ∈ oneOnionSession k M"
 | onionCons3:  "[|tr∈(oneOnionSession k M);
 length tr≥k; Says M Y (Crypt (pubK Y) (Nonce n))∉(set tr)|]
⟹Says M Y (Crypt (pubK Y) (Nonce n)) #tr ∈ oneOnionSession k M"
```

Proving

1. $[|(m_1, m_2) \in \text{set (zip (map msgPart } tr\text{)}$
   $\text{(map msgPart (swap } ma\ mb\ tr\text{)))}|]$
   $\Longrightarrow m_1 = m_2 \vee (m_1, m_2) = (ma, mb) \vee (m_1, m_2) = (mb, ma)$
2. $\text{sendRecvMatchL } tr \text{ (swap } ma\ mb\ tr\text{)}$
3. $\text{length (swap } ma\ mb\ tr\text{)} = \text{length } tr$
4. $\text{swap } ma\ mb\ tr = \text{swap } mb\ ma\ tr$
5. $[|(\text{Says } X\ \ M\ \ ma \in \text{set } tr)|]$
   $\Longrightarrow \text{Says } X\ \ M\ \ mb \in \text{set (swap } ma\ mb\ tr\text{))}$
6. $[|(\text{Says } X\ \ M\ \ mb \in \text{set } tr)|]$
   $\Longrightarrow \text{Says } X\ \ M\ \ ma \in \text{set (swap } ma\ mb\ tr\text{))}$
7. $[|m \neq ma; m \neq mb; (\text{Says } X\ \ M\ \ m) \in \text{set } tr|]$
   $\Longrightarrow (\text{Says } X\ \ M\ \ m \in \text{set (swap } ma\ mb\ tr\text{))}$

Conclusion

Lemma

$[|tr \in \text{oneOnionSession}\ k\ M; ma' = \{\text{Nonce}\ n\}_{\text{pubK}\ Y};$

$\text{Says}\ M\ B\ ma' \in \text{set}\ tr; \text{regularOrig}\ ma'\ tr;$

$M \notin \text{bad}; \text{cond}\ tr\ M|] \Longrightarrow$

$\text{senderAnomity}\ (\text{senders}\ tr\ M - \text{bad})$

$\text{Spy}\ ma'\ tr\ (\text{oneOnionSession}\ k\ M)$

Conclusion

Lemma

$[\![tr \in \text{oneOnionSession } k\ M;\ ma' = \{\text{Nonce } n\}_{\text{pubK } Y};$

$\text{Says } M\ B\ ma' \in \text{set } tr; \text{regularOrig } ma'\ tr;$

$\text{Says } A\ M\ m' \in \text{set } tr; A \notin \text{bad}; M \notin \text{bad};$

$\exists X, mx.\text{Says } X\ M\ mx \in \text{set } tr \wedge X \neq A \wedge X \notin \text{bad}; \text{cond } tr\ M\ n]\!]$

$\Longrightarrow \text{let } AS = \text{senders } tr\ M - \text{bad } \mathit{in}$

$\text{unlinkability } AS\ A\ m\ (\text{oneOnionSession } k\ M)$