LTL for Constraint-based Security Protocol Analysis

Sandro Etalle
with Ricardo Corin & Ari Saptawijaya
University of Twente

Outline

- security protocols
- constraint-based verification of security protocols
- contribution PS-LTL
  - specification of security properties
  - verification

The Context

- One of the best protocol verifiers:
  - Millen&Shmatikov (CCS’01)
    - Bounded sessions
    - Constraint solving
      - Unbounded message space
    - Perfect crypto, Dolev-Yao

Roles and Protocols

- Roles: sequences of send & recv actions
  - init(A, B, N_A, K_{st}, K_{dt})
  - resp(A, B, N_A, K_{st}, K_{dt})

Constraints

- constraint = m:K
  - a symbolic trace contains variables
  - and has an associated constraint store:

Problem

- Encoding properties in MS
  - Secrecy: add to scenario a special role
  - Authentication: construct scenario in which a role has no corresponding party

- Problems
  - indirect
  - built-in
  - inflexible
The Solution: PsLTL

- Based on LTL with past operators
  - similar to NPATRL

- Syntax
  - \( p(d_1, \ldots, d_n), \text{learn}(m) \)
  - \( \forall \phi, \exists_1 \phi_2, \phi = (\equiv \text{true } \phi) \), \( H \phi = (\equiv \neg \phi) \)
  - \( \neg \phi, \phi_1 \land \phi_2, \phi_1 \lor \phi_2, \exists v. \phi, \forall v. \phi \)

Decorating Protocols

\[
\text{init}(A, B, N, K, K_0) = \langle (A : (N_A) > B) \rangle(A : (N_A, K_0) \wedge B) \text{run}(A, B, \text{initiator}, N_A, K_0, K_0) \\
(A : (N_A) \wedge B) \\
\text{resp}(A, B, N, K, K_0) = \langle (B : (N_A, K_0) < A) \rangle(B : (N_A, K_0) \wedge A) \text{run}(B, A, \text{responder}, N_A, K_0, K_0) \\
(B : (N_A, K_0) \wedge A) \text{end}(B, A, \text{responder}, N_A, K_0, K_0) \rangle
\]

Properties

- Aliveness
- Non-injective agreement

\( \forall A, B, D_1, D_2, D_3, \text{end}(A, B, \text{responder}, D_1, D_2, D_3) \rightarrow \neg \text{run}(B, A, \text{initiator}, D_1, D_2, D_3) \)

- Freshness
- Standard secrecy
- Perfect forward secrecy (DH key agreement):
  - disclosure of long term-keys does not compromise secrecy of earlier exchanged short-term keys

Challenge

- Checking a formula \( \phi \) on a SYMBOLIC trace \( tr \)
  - given \( tr \), find \( \sigma \) s.t.
  - \( tr\sigma \) is a valid concreteization of \( tr \)
    - (the associated C5S is solvable)
    - \( \sigma \) falsifies \( \phi \)
      - We are looking for an attack
- Idea: \( \phi \) guides the concretization of \( tr \)

Sketch

- 1 symbolic trace

- \( n \) (a) concrete traces

- positive equalities: (tweaked) unification
- positive constraints: MS procedure
- negative constraints:
  - safe approximation
- negative equalities: syntactic check

how we do it
Conclusions

• Separate the properties from the spec.
• Constraint solving for protocol engineering
  – (we used it successfully in three large case studies: WSN protocol, OSA/Parlay auth protocol and DRM protocol)
• Among the simplest D-Y like approaches
• PS-LTL clarifies the difference between model and requirements