

CT-T: Resource-guided Implementation of Secure Embedded Software

Rajeev Alur, Pavol Černý, Andre Scedrov, Steve Zdancewic

University of Pennsylvania



Model Checking of Secrecy Properties

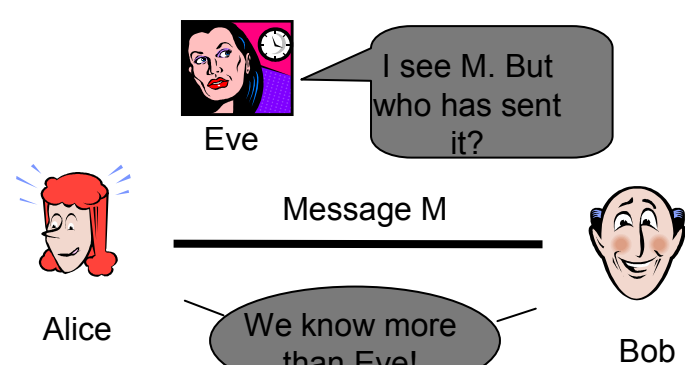
Secrecy and information-flow properties are not expressible in classical temporal logics, and therefore cannot be verified using existing model checkers. We developed two remedies: 1) richer tree models and logics that allow specification of secrecy and corresponding model-checking algorithms, 2) refinement notions for preserving secrecy.

Model: Trees with Path Equivalences

- Traditional model of systems: trees whose paths encode executions of system.
- Richer model: trees with path equivalences. Two nodes are equivalent, if the unique paths leading to them are indistinguishable to an observer.
- Secrecy: Variable x is secret at a node iff there exists an equivalent node where x has a different value.

Example: Isotropic Key Agreement Protocol

Isotropic Channels



- Isotropism: Eavesdropper can read a message, but cannot determine who sent it.
- Suppose passive eavesdropper and suppose Alice and Bob prefix each message with their names in alphabetical order. Then e.g. if Alice receives a message, she has more information than Eve – Alice knows that Bob was the sender.
- Isotropism can thus be used to exchange a bit of the secret key.

- We verified that Eve does not learn anything from the sequence of messages used to exchange a bit for the recent protocol proposed by Anand et al

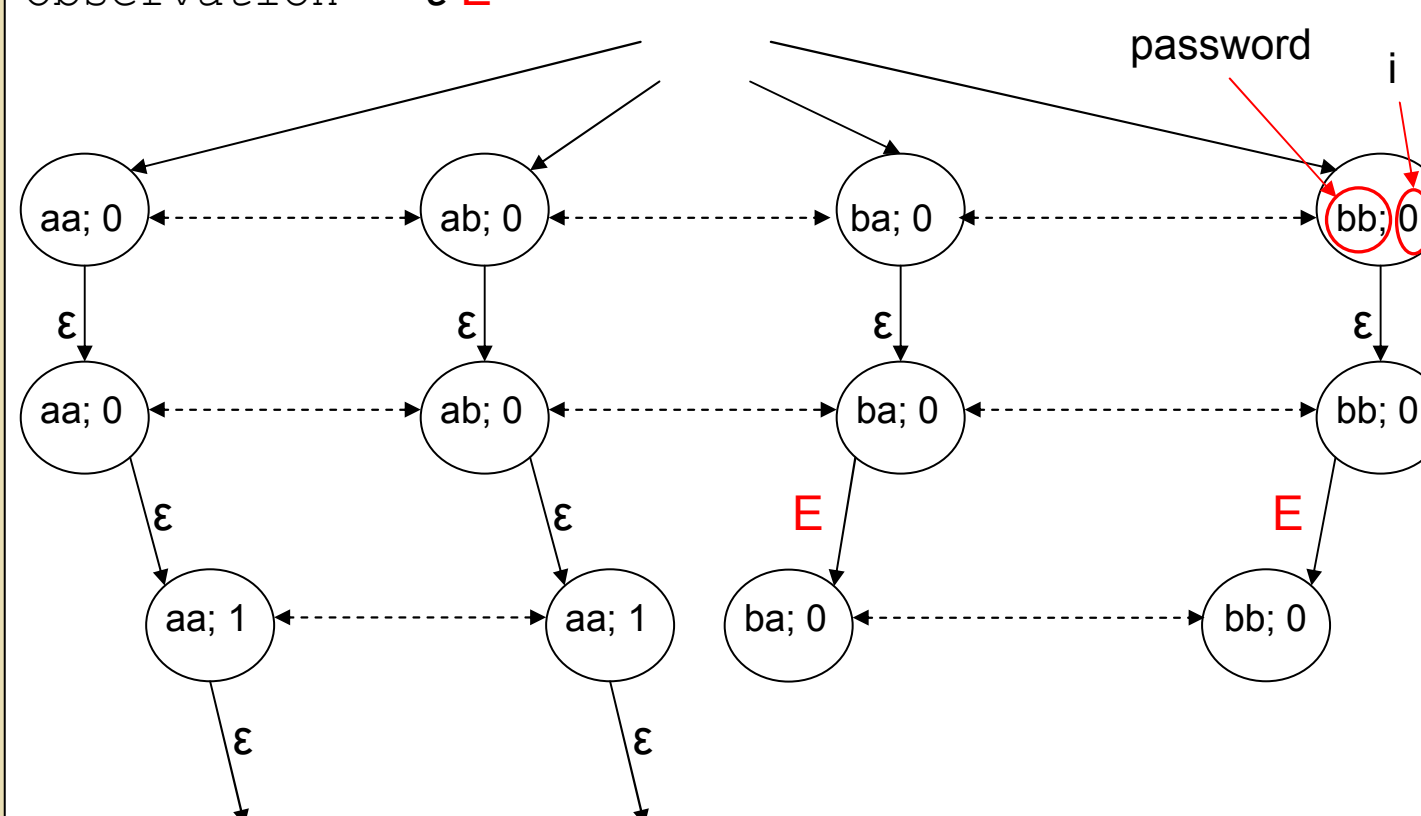
Example: Information leak through timing

TENEX OS bug

Trial = aa

Observation = ϵE

```
for (i=0; i<8; i++) {
  if (password[i] != trial[i]) break;
}
if (i < 8) return( ERROR );
```



- The code fragment above leaks password through timing.
- tree edges with observations
- equivalence edges
- If input is aa and observation is ϵE , the observer can conclude that the first letter of the password is b .

Approach and Impact

New approach: Framework for reasoning about information flow properties

- Model: Trees with path equivalences
- Logics: CTL_{\approx} , μ_{\approx} -calculus - expressive logics for safety, liveness and information flow properties
- Model-checking algorithms
- Algorithm for checking that a refinement is secrecy-preserving

Research Impact

- Algorithmic verification of secrecy and time properties for programs and protocols. Example: "Agent A does not reveal x (a secret) until agent B reveal y (a password)."
- Secrecy preserving refinement checking: e.g. for resource-guided transformations
- Software analysis: noninterference, taint analysis (future work).

▪ **Model:** For an agent a , two nodes are considered a -equivalent if the two paths (executions) leading to them if the observable behavior of a is identical along the two executions. We define an **equivalence graph**, where an a -labeled edge is added between every pair of a -equivalent nodes. Timing-insensitive (stuttering) equivalences and corresponding equivalence graph are defined similarly.

▪ **Logics:** CTL_{\approx} (μ_{\approx} -calculus) – similar to CTL (μ -calculus) with next operators EI_a for each agent a that are interpreted on a -equivalence edges.

▪ **Model checking algorithm:** Given a finite state model K of a system, the semantics of a CTL_{\approx} (μ_{\approx} -calculus) formula φ is not (and cannot be) defined on states of K , but rather on the states of a tree unfolding of K (denoted by T_K). Therefore, given K and φ in order to answer the model-checking question (Is K a model of φ ?), we need to construct a finite state structure adequate for evaluating φ . The construction is similar to (multiple levels of) the standard subset construction. For fragments of CTL_{\approx} (μ_{\approx} -calculus) that are expressive enough to capture the properties of interest, the model-checking problem is **PSPACE-complete** (EXPTIME-complete).

▪ **Secrecy preserving refinement:** is a strengthening of the classical trace-based refinement so that the implementation leaks a secret only if the specification also leaks it. We developed a **simulation based proof technique** for secrecy-preserving refinement. This result shows that existing refinement checkers can be used to check preservation of secrecy.

[ACZ06] R. Alur, P. Černý, S. Zdancewic: Preserving Secrecy Under Refinement, ICALP 2006

[ACC07] R. Alur, P. Černý, S. Chaudhuri: Model Checking on Trees with Path Equivalences, TACAS 2007 (to appear)