Guaranteeing Timed Opacity using Parametric Timed Model Checking

Dylan Marinho

Université de Lorraine, CNRS, Inria, LORIA, Nancy, France

Context: timing attacks

- Principle: deduce **private information** from timing data (**execution time**)
- Issues:
  - May depend on the **implementation** (**introduced by the compiler**)
  - A relatively trivial solution: make the program last always its maximum execution time
    Drawback: **loss of efficiency**
- Informal problems
  - Question: can we exhibit **secure execution times**?
  - Further question: can we also tune internal timing constants to make the system resisting to timing attacks?

Objective. Given a system modeled by a timed automaton, can we exhibit secure **execution times**, i. e., for which an attacker having only access to the global execution time cannot deduce whether some private location was visited?

A simple example of a timing attack

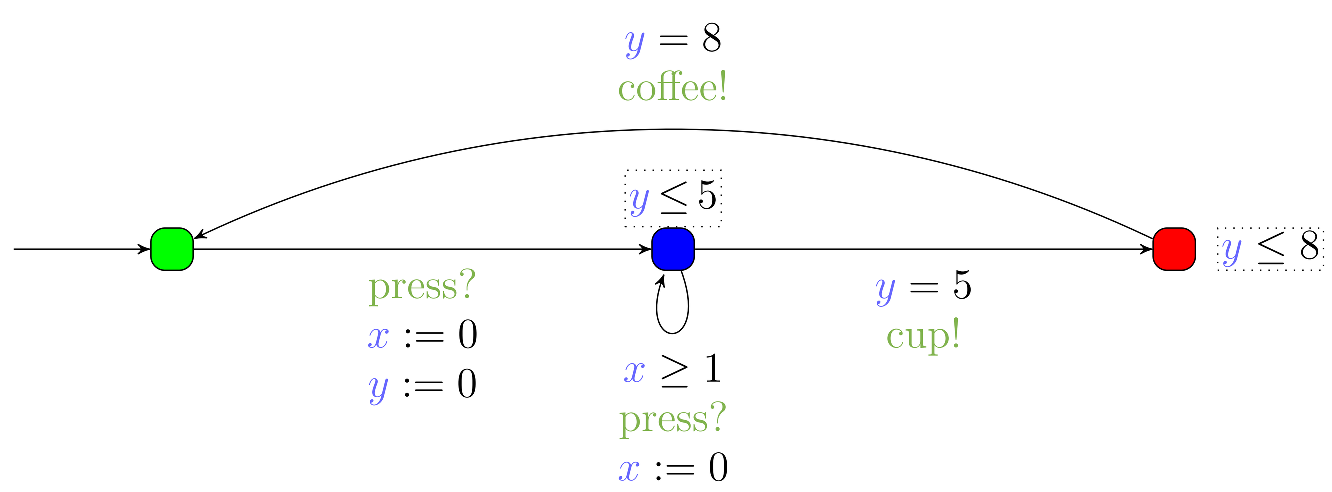
```
# input pwd     : Real password
# input attempt: Tentative password
for i = 0 to min(len(pwd, len(attempt)) − 1 do
 if pwd[i] =/= attempt[i] then
 return false
done
return true
```

Listing 1: Code describing the verification of a tentative password input by the user

`pwd`	`c`	`h`	`i`	`c`	`k`	`e`	`n`
`attempt`	`c`	`h`	`e`	`e`	`s`	`e`	
Execution time	ϵ	ϵ	ϵ				

- Problem: The execution time is proportional to the number of consecutive correct characters from the beginning of `attempt`

Timed Automaton (TA) [AD94]

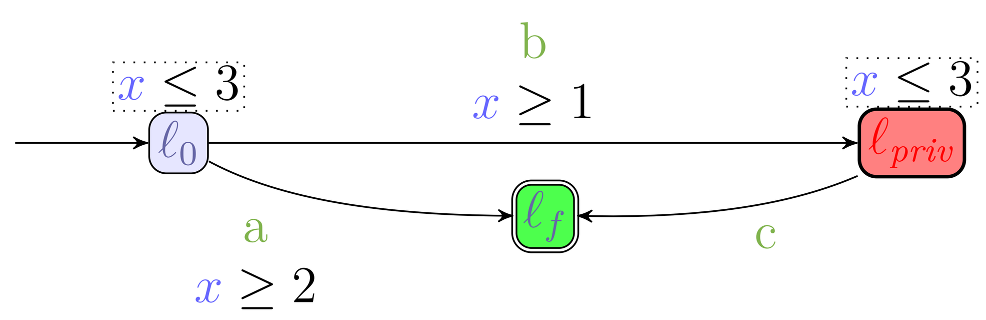


- Finite state automaton (sets of locations and actions) augmented with a set X of clocks
  - Real-valued variables evolving linearly at the same rate
  - Can be compared to integer constants in invariants and guards
- Features
  - Location invariant: property to be verified to stay at a location
  - Transition guard: property to be verified to enable a transition
  - Clock reset: some of the clocks can be set to 0 along transitions

Timed-opacity definition [TOSEM22]

Attacker model The attacker only has **access to the global execution time** from the initial location to some final location (no action is visible)

Secret Has the system visited some private location ℓ_{priv}?

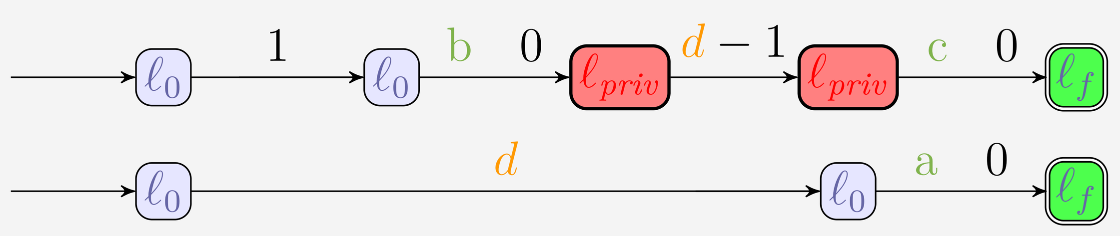


Definition (timed opacity) The system is **opaque w.r.t. ℓ_{priv} on the way to ℓ_f** for a duration d if there exist two runs from ℓ_0 to ℓ_f of duration d

1. one passing by ℓ_{priv}
2. one *not* passing by ℓ_{priv}

Example

- There exist two runs of duration d for all durations $d \in [2, 3]$:



  The system is **opaque w.r.t. ℓ_{priv} on the way to ℓ_f** for all durations in $[2, 3]$
- But it is not possible to reach ℓ_f with a path of duration 1.5 not passing by ℓ_{priv}
  The system is ***not* fully opaque w.r.t. ℓ_{priv} on the way to ℓ_f**

Theorem The durations d such that the system is opaque can be effectively computed and defined

Corollary Asking whether a TA is opaque for all its execution times (*"full timed-opacity"*) is decidable

Parametric Timed Automaton (PTA) [AHV93]

- Timed automaton (sets of locations, actions and clocks) augmented with a set P of parameters (**Unknown constants** compared to a clock in guards and invariants)
- High interest of **timing parameters: underspecified systems, or partially known systems**

Overview of our theoretical results [TOSEM22]

- General case: The mere existence of a parameter valuation for which there exists a duration for which timed-opacity is achieved **is undecidable**
- Study of a subclass known for being "at the frontier" of decidability (L/U-PTA) [Hun+02]
- Practical contribution: We adopt a "best-effort" approach for the general case of PTAs: this approach is not guaranteed to terminate

Experiments [TOSEM22]

- Verification engine: IMITATOR [And21]
- Common PTA benchmarks [TAP21]
- Library of Java programs [STA], manually translated to PTAs
  - user-input variables translated to (non-timing) parameters (supported by IMITATOR)

Perspectives

Theoretical side

- Some restricted problems remain open e. g., PTAs with one clock

Practical side

- Automatic translation of programs to PTAs
- Repairing a non-opaque system

References

[AD94] Rajeev Alur and David L. Dill. "A theory of timed automata". In: *Theoretical Computer Science* 126.2 (Apr. 1994), pp. 183–235. ISSN: 0304-3975. DOI: 10.1016/0304-3975(94)90010-8.

[AHV93] Rajeev Alur, Thomas A. Henzinger, and Moshe Y. Vardi. "Parametric real-time reasoning". In: *STOC*. San Diego, California, United States: ACM, 1993, pp. 592–601. DOI: 10.1145/167088.167242.

[And21] Étienne André. "IMITATOR 3: Synthesis of Timing Parameters Beyond Decidability". In: *CAV 2021*. 2021. DOI: 10.1007/978-3-030-81685-8_26.

[Hun+02] Thomas Hune, Judi Romijn, Mariëlle Stoelinga, and Frits W. Vaandrager. "Linear parametric model checking of timed automata". In: *JLAP* 52-53 (2002). DOI: 10.1016/S1567-8326(02)00037-1.

[STA] STAC. URL: https://github.com/Apogee-Research/STAC/.

[TAP21] Étienne André, Dylan Marinho, and Jaco van de Pol. "A Benchmarks Library for Extended Parametric Timed Automata". In: *TAP 2021*. 2021. DOI: 10.1007/978-3-030-79379-1_3.

[TOSEM22] Étienne André, Didier Lime, Dylan Marinho, and Jun Sun. "Guaranteeing Timed Opacity using Parametric Timed Model Checking". In: *TOSEM* (2022).

This work was partially supported by the ANR-NRF French-Singaporean research program ProMiS (ANR-19-CE25-0015)