Engineering Secure Software Systems

February 2, 2021: Information Flow: Unwindings, Algorithms, Beyond P-Security

Henning Schnoor

Institut für Informatik, Christian-Albrechts-Universität zu Kiel

Admin: Evaluation, Exam

Your feedback

- 6 of you answered, thanks!
- let's look at the feedback

Lecture Evaluation

my comments

- “It seems like there is too much to do and not enough time. It feels like one shouldn’t ask any questions during the lecture since we already skip many review questions and topics.”
  (also in our review session: students seem to like the review questions, but hardly participate)
  - questions too easy / too difficult?
  - no time for preparation?
- “If there is something like iLearn . Where we can find what you taught us so far in synchronous order , will be best.”
  - you can find these in `slides/esss_lecture_slides_xx.pdf`
- “Since there is no skript, it is also realy hard to repeat the topics from the slides.”
  - There are the lecture notes, `lecture-notes/esss_notes.pdf` that contain a lot of additional information (but no complete script).

Exam via BigBlueButton

Technicalities

- We use BigBlueButton, either
    - standalone,
    - as part of ESSS-Mattermost channel, or
    - as part of OLAT.
- You need a working camara/microphone, and your (student) id **with photo** readable through the camera
- Use a computer so you can draw/type in the shared working area
- Test your setup before the exam.
- TEST YOUR SETUP BEFORE THE EXAM!

Technical Issues?

- things can go wrong: internet connection, device crashes, camera, microphone, you name it!
- then: exam counts as “not taken,” not as failed
- new date probably only possible in second examination period (starting March 29)

Exam: Corona Info

from Vice President for Studies & Teaching

- A **free attempt** will be granted for all examinations taken and failed during the examination periods of the winter semester 2020/21
- Please take into account that the lecturers have a significantly higher workload than in regular semesters. **We therefore ask you to only register for examinations for which you have prepared and which you wish to take.** If you will not be taking an exam at short notice, please inform the respective examiners, especially in the case of oral exams.

Exam: My Expectations

I expect you to …

- 4,0 know central definitions, results (formally correct) and can apply them to simple examples

  basic reproduction

- 3,0 explain relationships between and motivations for central definitions

  basic understanding

- 2,0 explain the ideas behind the central proofs

  advanced understanding

- 1,0 reason about alternative defintions, applications, …

  application of knowledge to new situations

caveats

- this is **not** a “guaranteed performance $\to$ grade mapping”
- this is a **theory lecture**, you need to be **formally precise** when required.

Exam: Your Preparation

material

- slides
- exercises (with solutions)
- videos (of some central proofs)
- notes (contain all proofs)

preparation: are you ready?

- Do you know the central definitions, results, **precisely**?
- Can you answer the review questions? (Answers not provided on purpose)
- Do you have ideas for most of the exercise tasks? (Most exercises have solutions, except the ones that are meant to lead to discussions)
- Can you explain the relationship between different but related concepts in the lecture?
- Can you explain the proofs of the main formal results of the lecture?

Part II: Information Flow

Overview

Part II: Information Flow

Examples

Introduction and Motivation

P-Security

Automatic Verification

IP-Security

Motivation and Definition

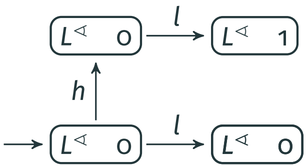
conditions

OC $s \sim_u t$, then $\mathtt{obs}_u(s) = \mathtt{obs}_u(t)$

SC $s \sim_u t$, then $s \cdot a \sim_u t \cdot a$

LR $\mathtt{dom}(a) \not\rightarrow u$, then $s \sim_u s \cdot a$

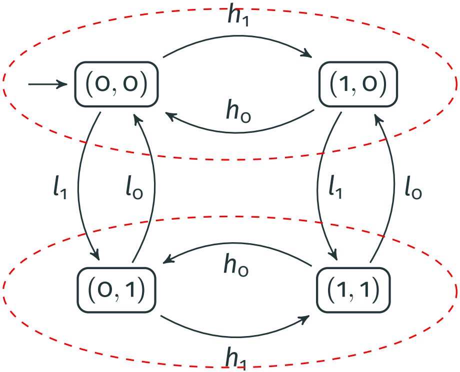
system 1



insecure

- $\alpha_1 = l$
- $\alpha_2 = hl$

system 2



Exercise

Task (uniqueness of unwindings)

Show that P-unwindings are not unique, but that mininal P-unwindings are, that is:

1. give an example for a system M and a policy $\rightarrowtail$ such that there are (at least) two different P-unwindings for M and $\rightarrowtail$,
2. show that if M is P-secure with respect to a policy $\rightarrowtail$, then there is a P-unwinding for M and $\rightarrowtail$ that is contained (via set inclusion) in all P-unwindings for M and $\rightarrowtail$.

Algorithm for P-Security

seen

P-security is characterized by unwindings

algorithmic approach

check whether unwinding exists, accept if unwinding found.

issues?

- what are “candidates” for unwindings?
- how many equivalence relations on a set with $|S|$ elements?
- candidate given by proof:

$$s \sim_u t \text{ iff } \forall \alpha_1, \alpha_2 \text{ with } \mathtt{purge}(\alpha_1) = \mathtt{purge}(\alpha_2) : \mathtt{obs}_u(s \cdot \alpha_1) = \mathtt{obs}_u(t \cdot \alpha_2)$$

- difficult to construct algorithmically!

Required: "Easier" Unwinding

lemma

If M is P-secure, then this algorithm constructs unwinding:

Input: $(S, A, \mathtt{step}, D, \mathtt{dom})$
  for each $u \in D$ **do**
    $\sim_u := \{(s, s) \mid s \in S\}$
    while elements added to $\sim_u$ **do**
      close $\sim_u$ under transitivity
      close $\sim_u$ under symmetry
      close $\sim_u$ under left respect
      close $\sim_u$ under step consistency
    end while
  end for

corollary

P-Security can be verified in polynomial time.

proof

Algorithm:

- construct $(\sim_u)_{u \in U}$ as in algorithm
- accept iff each relation satisfies output consistency

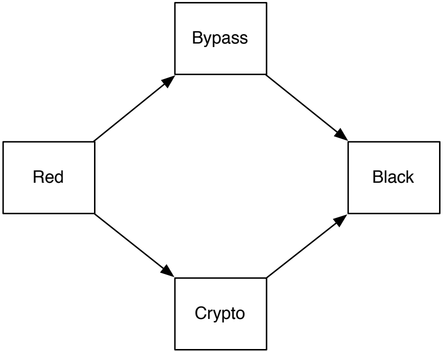
Overview

Overview

Intransitive Noninterference

P-Security

- reasonable definition of security
- assumes that policies are transitive
- intransitive policies occur in more complex scenarios



Intransitive Noninterference

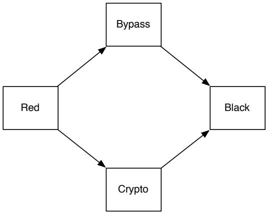
issue

- information may flow from `Red` to `Black`, but must pass `Crypto` or `Bypass`
- all-or-nothing approach of P-security does not suffice

goals for definition

- `Red`'s actions may have impact on `Black`'s view
- but `Black` may **only** learn of these actions "via `Bypass` or `Crypto`"
- question whether `Black` may learn of action depends on what happens *after* action

intransitive policy



Intransitive Noninterference

downgrading

- indirect interference, introduced in [HY87]
- standard example: trusted "downgrader" D: declassifier, encryption device, ... *small enough to be formally verified*
- intransitive policies:

$$H \rightarrowtail D \rightarrowtail L$$

- H's actions "transmitted" to L by actions of D
- L must not learn about H's actions directly

intransitive noninterference

meaningful semantics for intransitive policies

Intransitive Noninterference

question

- action sequence: $a\alpha$
- may L “learn” that a was performed?

$$
\begin{tikzcd}
& B \arrow[r, tail] & D \arrow[dr, tail] & \\
A \arrow[ur, tail] \arrow[dr, tail] & & & L \\
& C \arrow[r, tail] & E \arrow[ur, tail] &
\end{tikzcd}
$$

downgrading

transmission of actions by sequence of actions

With each action

Agent performing action “transmits” knowledge about previous events

step-by-step downgrading

- sequence $abece$: who may “know” that a occured?
- knowledge “spreads” in each step: $a\ b\ e\ c\ e$

Intransitive Noninterference: IP-Security

overview

- adaptation of P-security to intransitive case, defined in [HY87]
- replaces `purge` with `ipurge`: keeping track of "allowed interferences"

definition (`sources`)

- $\texttt{sources}(\alpha, u)$: agents who may interfere with u in sequence α
- $\texttt{sources}\colon A^* \times D \to \mathcal{P}(D)$
    - $\texttt{sources}(\epsilon, u) = \{u\}$
    - $\texttt{sources}(a\alpha, u)$ for $a \in A$, $\alpha \in A^*$: two cases
        1. there is $v \in \texttt{sources}(\alpha, u)$ with $\texttt{dom}(a) \rightarrowtail v$, then

$$\texttt{sources}(a\alpha, u) = \texttt{sources}(\alpha, u) \cup \{\texttt{dom}(a)\}\,.$$

        2. otherwise: $\texttt{sources}(a\alpha, u) = \texttt{sources}(\alpha, u)$.

definition (ipurge)

$\mathtt{ipurge}\colon A^* \times D \to A^*$ (also: $\mathtt{ipurge}_u$) defined inductively

- $\mathtt{ipurge}(\epsilon, u) = \epsilon$
- for $a \in A, \alpha \in A^*$:

$$\mathtt{ipurge}(a\alpha, u) = \begin{cases} a\mathtt{ipurge}(\alpha, u), & \text{if } \mathtt{dom}(a) \in \mathtt{sources}(a\alpha, u), \\ \mathtt{ipurge}(\alpha, u), & \text{otherwise} \end{cases}$$

definition (IP-security)

System $(S, s_0, A, \mathtt{step}, D, O, \mathtt{obs}, \mathtt{dom})$ is IP-secure with respect to a policy $\rightarrowtail$, if for all $u \in D$, $s \in S$, $\alpha_1, \alpha_2 \in A^*$:

$$\text{If } \mathtt{ipurge}_u(\alpha_1) = \mathtt{ipurge}_u(\alpha_2), \text{ then } \mathtt{obs}_u(s \cdot \alpha_1) = \mathtt{obs}_u(s \cdot \alpha_2).$$

Exercise

Task (IP-Security examples)

Which of the following systems are IP-secure? Assume that as usual, the state names indicate the observations made by L, that lowercase letters denote actions performed by agents with the corresponding higher-case letter name, and the policy $H \rightarrowtail D \rightarrowtail L$. Additionally, assume that H and D make the same observation in each state of the system.

$$
\begin{tikzcd}
\boxed{0} \arrow[r, "h"] \arrow[d, "l"] & \boxed{0} \arrow[r, "d"] & \boxed{1} \\
\boxed{0} \arrow[r, "h"] & \boxed{0} \arrow[r, "d"] & \boxed{0}
\end{tikzcd}
\qquad
\begin{tikzcd}
\boxed{0} \arrow[r, "h"] \arrow[d, "l"] & \boxed{0} \arrow[r, "d"] & \boxed{1} \\
\boxed{0} \arrow[r, "d"] & \boxed{0} \arrow[r, "h"] & \boxed{1}
\end{tikzcd}
$$