CHRISTIAN-ALBRECHTS-UNIVERSITÄT ZU KIEL
Institut für Informatik · AG Software Engineering
Priv.-Doz. Henning Schnoor

Engineering Secure Software Systems Winter 2020/21
Exercise Sheet 10

issued: January 19, 2021 **due:** January 28, 2021

Exercise 10.1, indistinguishability (10 Points)

For the following pairs of terms, determine whether they are I-distinguishable, where $I = \{k_A, k_C, \hat{k}_C, \mathsf{yes}, \mathsf{no}\}$ contains the initial adversary knowledge.

$\mathbf{t_1}$	$\mathbf{t_2}$
$[N_A, \mathsf{enc}^{\mathsf{s}}_{N_A}(N_B)]$	$[N_B, \mathsf{enc}^{\mathsf{s}}_{N_B}(N_A)]$
$[N_B, \mathsf{enc}^{\mathsf{s}}_{N_A}(N_B)]$	$[N_A, \mathsf{enc}^{\mathsf{s}}_{N_B}(N_A)]$
$[N_A, \mathsf{enc}^{\mathsf{s}}_{N_A}(N_B)]$	$[N_A, \mathsf{enc}^{\mathsf{s}}_{N_B}(N_B)]$
$\mathsf{enc}^{\mathsf{a}}_{k_A}(N_A, \mathsf{yes})$	$\mathsf{enc}^{\mathsf{a}}_{k_A}(N_B, \mathsf{yes})$
$\mathsf{enc}^{\mathsf{a}}_{k_A}(N_A, \mathsf{yes})$	$\mathsf{enc}^{\mathsf{a}}_{k_A}(N_A, \mathsf{no})$
$[N_A, \mathsf{enc}^{\mathsf{a}}_{k_A}(\mathsf{hash}(N_A), \mathsf{yes})$	$[N_B, \mathsf{enc}^{\mathsf{a}}_{k_A}(\mathsf{hash}(N_B), \mathsf{yes})$
$[N_A, \mathsf{enc}^{\mathsf{a}}_{k_A}(\mathsf{hash}(N_A), \mathsf{yes})$	$[N_B, \mathsf{enc}^{\mathsf{a}}_{k_A}(\mathsf{hash}(N_A), \mathsf{yes})$

Exercise 10.2, strong secrecy and derivation-based secrecy (10 Points)

For an equational theory E, a term t is E-derivable from a set of terms I, if there is a term M built from E-constructors (e.g., encryption functions), E-deconstructors (e.g., decryption functions) and elements from I with $M \equiv_E t$.

Example: Let E model symmetric encryption and pairing, let $I = \{k_{AC}, \underbrace{\mathsf{enc}^{\mathsf{s}}_{k_{AC}}(\mathsf{yes}, N_A)}_{=:u}\}$. Then $t = N_A$ is E-derivable from I via $M = \mathsf{proj}_2(\mathsf{dec}^{\mathsf{s}}_{k_{AC}}(u))$.

Now, the *(nonce) derivation problem* for E is to determine, given a set I of terms and a term (a nonce) t, whether t is E-derivable from I.

Show that if static equivalence for E is decidable, then the nonce derivation problem for E is also decidable.

Note: It suffices to state the (simple) algorithm deciding nonce derivation problem, which may apply the decision algorithm for static equivalence.

Exercise 10.3, secrecy properties and events (10 Points)

In the lecture, two different kinds of (trace) properties were discussed:

- secrecy properties, modeled with derivability of the constant FAIL and in ProVerif using the statement

  query attacker(FAIL),

- event properties, modeled in ProVerif using the specification **event** and queries like

  query x:key; **event(termServer(x))** ⇒ **event(acceptsClient(x))**.

Is one of these concepts more powerful than the other? In other words, can you "translate" any secrecy query into an event quary and/or vice versa? Which, if any, extensions would our theoretical model require to be able to handle event properties?

Note: The point of this exercise is not for you to actually specify a (rather cumbersome) translation, but to conceptually consider the relationships and differences between these two types of properties.

Exercise 10.4, ProVerif modeling of Needham Schroeder (10 Points)

Study the modeling of the Needham Schroeder protocol given in the ProVerif distribution (various models of the protocol can be found in `examples/pitype/secr-auth/`). Which additional properties were modeled compared to our models from the lecture and exercise class?