

# Criteria for Static Equivalence in Applied $\Pi$ -calculus

Eike Ritter

University of Birmingham

Joint work with Aybek Mukhamedov and Mark Ryan



## *Background: Formal Protocol Verification*

---

Security protocols are message exchanges to share information  
Cryptography used to keep content private

Example: Key elements of voting protocols

Let  $k$  be a shared key between voters and election officials, not known to intruder

**Alice** sends Official message  $m_1 = \text{encrypt}(\text{vote}_A, k)$

**Bob** sends Official message  $m_2 = \text{encrypt}(\text{vote}_B, k)$

Election official outputs result on public channel by sending message  
“1 vote for **A**, 1 vote for **B**”.



Consider now different variant:

**Alice** sends Official message  $m_1 = \text{encrypt}(\text{vote}_B, k)$

**Bob** sends Official message  $m_2 = \text{encrypt}(\text{vote}_A, k)$

Election official outputs result on public channel by sending message  
“1 vote for **A**, 1 vote for **B**”.

Question: Can intruder (the environment) detect the difference?

More precisely: Is there program  $P$  with hole which intruder may apply which

- outputs 1 if first protocol substituted for hole
- outputs 2 if second protocol substituted for hole

Informal answer: No - doesn't have the encryption keys, hence encrypted messages are just meaningless bitstrings

How to formalise this?

Key assumption: Cryptography is perfect

## *Formalisation of protocols*

---

Here: Use process calculus extended with terms (Applied  $\Pi$ -calculus) to model such protocols

Each participant is modelled by one process

Communication modelled by channels

have both public and private channels (viewable by intruder or not)

So far,  $\Pi$ -calculus. Extensions for applied  $\Pi$ -calculus:

- adds capability of sending terms over channels
- terms are subject to equations
- have private names (modelling eg nonces and encryption keys)
- have separate syntactic construction modelling intruder knowledge obtained by observing output on public channels

## *Syntax of processes*

---

Have grammar for processes

$$P, Q ::= 0 \mid P|Q \mid !P \mid \nu n.P \mid \text{if } M = N \text{ then } P \text{ else } Q \\ \mid u(x).P \mid \bar{u}\langle N \rangle.P$$

where  $\nu n.P$  creates private channel or name  
and extended processes (which model also intruder knowledge)

$$A, B ::= P \mid A|B \mid \nu n.A \mid \nu x.A \mid \{M/x\}$$

where

- $\nu x.A$  models creation of private variables, used for communication between private channels;
- $\{M/x\}$  makes term  $M$  available to intruder via variable  $x$ ;

Have in addition equational theory for terms

## *Example processes*

---

**Alice** :=  $\bar{c}\langle \text{encrypt}(v_A, k) \rangle$

**Bob** :=  $\bar{c}\langle \text{encrypt}(v_B, k) \rangle$

**E0** :=  $c(v_1).\text{if } \text{encrypt}(\text{decrypt}(v_1, k), k) = v_1 \text{ then}$   
 $c(v_2).\text{if } \text{encrypt}(\text{decrypt}(v_2, k), k) = v_2 \text{ then } \bar{c}\langle \text{"OK"} \rangle$

$P_1$  :=  $\nu k.\text{Alice}|\text{Bob}|E0$

and similarly for  $P_2$  (exchanged  $v_A$  and  $v_B$ ).



## Possible traces

---

For  $P_1$ , internal communication between Alice and EO and Bob and EO lead to

$$EO := \bar{c}\langle \text{"OK"} \rangle$$

$$P'_1 := \nu k. \{ \text{encrypt}(v_A, k) / v_1, \text{encrypt}(v_B, k) / v_2 \} | EO$$

For  $P_2$ , internal communication between Alice and EO and Bob and EO lead to

$$EO := \bar{c}\langle \text{"OK"} \rangle$$

$$P'_2 := \nu k. \{ \text{encrypt}(v_B, k) / v_1, \text{encrypt}(v_A, k) / v_2 \} | EO$$



## *How to formalise desirable properties*

---

Standard notion in process calculi: Observational equivalence

Idea: Two processes  $P_1$  and  $P_2$  are observationally equivalent iff behaviour is the same in any context

Formally: for any context  $C[-]$ ,  $C[P_1]$  produces some output iff  $C[P_2]$  produces some output

Example:  $P_1$  and  $P_2$  are observationally equivalent

Key point: Encryption means outside environment (ie intruder) cannot distinguish votes

## *How to verify observational equivalence*

---

Key problem: Quantification over *all* evaluation contexts

Verification usually works in two steps:

- Show that dynamic behaviour of two processes is the same:  
For any action that  $P_1$  can do,  $P_2$  can do the same action and vice versa (dynamic part)
- Show that the intruder can perform the same computations based on the observations on public channels learnt from the execution of  $P_1$  and  $P_2$  (static equivalence)

Can use standard techniques (eg bisimulation) for  $\Pi$ -calculus for first step

In rest of talk, consider second step

## Proving static equivalences

---

Static equivalence between two so-called *frames*

$\phi = \nu \vec{n}. \{M_1/x_1, \dots, M_n/x_n\}$  and

$\psi = \nu \vec{n}. \{N_1/x_1, \dots, N_n/x_n\}$  defined as:

**Definition 1.**  $\phi$  and  $\psi$  are statically equivalent iff

$\forall$  terms  $M, N$

$$M[\vec{M}/\vec{x}] = N[\vec{M}/\vec{x}]$$

iff

$$M[\vec{N}/\vec{x}] = N[\vec{N}/\vec{x}]$$

Problem: involves quantification over all  $M, N$ !

In general static equivalence is undecidable (Abadi, Cortier)

Question: Can we examine only small subset of terms  $M, N$ ?

In this talk: YES.

Cannot use always automated tools (eg Proverif) because proofs of protocols use often symbolic manipulation

(eg if two frames are statically equivalent, then also two frames with a particular extension are also statically equivalent)



## *Algorithm for deciding static equivalence*

---

Assumption: Equality given by confluent term rewriting system.

Algorithm presented in two steps.

First, ignore reduction, and consider only substitution.

Show: it is enough to consider only the the substitutions occurring in the frame.

**Proposition 2** (Occurrence check). *Assume  $\phi$  and  $\psi$  are statically equivalent.*

*Then for all  $i$  and  $M$  s.t.  $FN(N) \cap \vec{n} = \emptyset$ ,*

*if  $M_i \equiv N[\vec{M}/\vec{x}]$ , then  $N_i = N[\vec{N}/\vec{x}]$*

*and vice versa.*

Special case of definition with  $M = x_i$ .

Not sufficient in general. However, it is in one important case:

Reduction does not allow intruder to construct new terms

Formally:

Call frame  $\phi = \nu \vec{n}. \{M_1/x_1, \dots, M_n/x_n\}$  be compatible with reduction if

for all left-hand side of reduction rules  $L$  there is no subterm  $K$  of  $L$  such that  $L$  is a subterm of  $K[\vec{M}/\vec{x}]$ .

Consequence: For all terms  $M$ , if  $M[\vec{M}/\vec{x}] \rightsquigarrow N$ ,  $\exists M'$  such that  $N \equiv M'[\vec{M}/\vec{x}]$  and  $M \rightsquigarrow M'$ .

## *Example revisited*

---

For running example, both frames

$$\nu k. \{ \text{encrypt}(v_A, k) / v_1, \text{encrypt}(v_B) / v_2 \} \text{ and}$$
$$\nu k. \{ \text{encrypt}(v_B, k) / v_1, \text{encrypt}(v_A) / v_2 \}$$

are compatible with reduction.

Key point: encryption key is private name

Occurrence condition obviously satisfied  $\Rightarrow$  above frames are statically equivalent.

## *The general case*

---

Now assume frame  $\phi = \{M_1/x_1, \dots, M_n/x_n\}$  is not compatible with reduction.

Idea: Extend frames to add all terms learnt by the intruder by applying reductions:

For each left-hand side  $L$  of reduction rule and  $K$  such that

- $K$  subterm of  $L$ ,  $FN(K) \cap \vec{n} = \emptyset$
- $L$  subterm of  $K[\vec{M}/\vec{x}]$

extend frame with  $N/x$ , where  $N$  is normal form of  $K[\vec{M}/\vec{x}]$ .

Also, define *recipe* (for intruder) of obtaining  $N$  by  $\{K/x\}$

Such  $K$  called *incompatibility witness*

Do this extension until no further terms added

Example: products

Reduction rules:

$$\text{Fst}\langle u, v \rangle \rightsquigarrow u$$

$$\text{Snd}\langle u, v \rangle \rightsquigarrow v$$

Example frame:

$$\nu k_1, k_2. \{ \langle k_1, k_2 \rangle / x \}$$

First reduction rule produces extension  $\{k_1/x_1\}$  with recipe

$$\{\text{Fst}x/x_1\}$$

Second reduction rule produces extension  $\{k_2/x_2\}$  with recipe

$$\{\text{Snd}x/x_2\}$$



## Criterion for static equivalence

---

**Theorem 3.** • Given frames  $\phi \stackrel{\text{def}}{=} \nu\vec{n}.\sigma$  and  $\psi \stackrel{\text{def}}{=} \nu\vec{n}.\tau$ .

- Let  $\phi_e = \nu\vec{n}.\sigma_e$  and  $\psi_e = \nu\vec{n}.\tau_e$  be the extensions of the frame.
- Let  $\rho$  and  $\omega$  be the recipes for  $\phi_e$  and  $\psi$ .

$\phi$  is statically equivalent to  $\psi$  iff

(i) for all  $i$  and all  $N$  such that  $FN(N) \cap \vec{n} = \emptyset$ ,  $N$  subterm of  $x_i\sigma_e$   
if  $x_i\rho\sigma_e = N\rho\sigma_e$ , then

$$x_i\rho\tau_e = N\rho\tau_e$$

(ii) for all LHS  $L$  of reduction rules, incompatibility witnesses  $K_1$   
and  $K_2$  of  $L$ , we have  $K_1\rho\sigma_e = K_2\rho\sigma_e$  implies

$$K_1\rho\tau_e = K_2\rho\tau_e$$

and symmetric conditions hold.

## Example

---

Consider frame  $\phi = \nu k_1, k_2, k. \{ \langle k_1, k_2 \rangle / x \}$

and

$\psi = \nu k_1, k_2, k. \{ k / x \}$ .

Have

$\phi_e = \nu k_1, k_2, k. \{ \langle k_1, k_2 \rangle / x, k_1 / x_1, k_2 / x_2 \}$  and  $\psi_e = \psi$ .

$\phi$  and  $\psi$  not statically equivalent because equation

$$x \rho \sigma_e = \langle x_1, x_2 \rangle \rho \sigma_e$$

holds, but not

$$x \rho \tau_e = \langle x_1, x_2 \rangle \rho \tau_e$$

Proverif shows some concrete instances of these theorems

Abadi and Cortier give general criterion for rewrite systems

Assumption: firstly, rewriting does not increase term size too much  
secondly, terms obtained by re-writing frames don't expose additional  
bound names

$\Rightarrow$  for each frame  $\phi$ , obtain finite set of equations which characterise  
static equivalence

In general, this set is a superset of the equations considered in the  
theorem.