Available online at www.sciencedirect.com
ScienceDirect
Electronic Notes in Theoretical Computer Science
ELSEVIER
Electronic Notes in Theoretical Computer Science 194 (2007) 61–83
www.elsevier.com/locate/entcs

A Probabilistic Scheduler for the Analysis of Cryptographic Protocols[1]

Srečko Brlek*, Sardaouna Hamadou**, John Mullins*,**

* *Lab. LaCIM, Dép. d'Informatique, Université du Québec à Montréal. CP 8888 Succursale Centre-Ville, Montreal (Quebec), Canada, H3C 3P8.*

** *Lab. CRAC, Dép. de Génie Informatique, École Polytechnique de Montréal P.O. Box 6079, Station Centre-ville, Montreal (Quebec), Canada, H3C 3A7.*

Abstract

When modelling cryto-protocols by means of process calculi which express both nondeterministic and probabilistic behavior, it is customary to view the scheduler as an intruder. It has been established that the traditional scheduler needs to be carefully calibrated in order to more accurately reflect the intruder's capabilities for controlling communication channels. We propose such a class of schedulers through a semantic variant called $\mathrm{PPC}_{\nu\sigma}$, of the Probabilistic Poly-time Calculus (PPC) of Mitchell *et al.* [11] and we illustrate the pertinence of our approach by an extensive study of the *Dining Cryptographers* (DCP) [8] protocol. Along these lines, we define a new characterization of Mitchell *et al.*'s observational equivalence [11] more suited for taking into account any observable trace instead of just a single action as required in the analysis of the DCP.

Keywords: Process algebra, observational equivalence, probabilistic scheduling, analysis of cryptographic protocols

1 Introduction

Systems that combine both probabilities and nondeterminism are very convenient for modelling probabilistic security protocols. In order to model such systems, some efforts have been taken to extend (possibilistic) models based on process algebras such as either the π-calculus or the CSP, by including probabilities. One can distinguish two classes of such models. On one hand, we have all purpose probabilistic models adding probabilities to nondeterministic models [1,6,3]. On the other hand, we have process algebraic frameworks that define probabilistic models in order to make them more suitable for applications in security protocols [11,4,10].

While it is customary to use schedulers for resolving non-determinism in probabilistic systems, scheduling processes must be carefully designed in order to reflect

[1] Research partially supported by NSERC grants (Canada)

doi:10.1016/j.entcs.2007.10.009

as accurately as possible the intruder's capabilities for controlling the communication network without controlling the internal reactions of the system. Consider the protocol $\overline{c}(a).\mathbf{0}|\overline{c}(b).\mathbf{0}$ transmitting the messages a and b over c and the intruder $c(x).\mathbf{0}$ eavesdropping on this channel. As the protocol is purely non deterministic, the probability of the intercepted message being either a or b should be the same. A scheduler that could assign an arbitrary probabilistic distribution to these two messages could also force the protocol to transmit either a or b. Such schedulers are too strong however, and should therefore not be admissible. But restricting the power of schedulers should also be carefully done, otherwise this could result in adversaries that are too weak. Forcing schedulers to give priority to internal actions, for example, makes internal actions completely invisible to attackers. An intruder is then unable to distinguish a process P from another process which could do some internal action, and then behaves like P. Consider now the following process:

$$P = \nu c'(c(x).\overline{c'}(x).\mathbf{0}|\overline{c'}(1).\mathbf{0}|c'(y).[y = 0]\overline{c}(\mathit{secret}).\mathbf{0}).$$

In this obviously unsecure protocol, an intruder could send 0 to P over c and thus allow P to publish the secret. Such a flaw will never be detected in semantics giving priority to internal actions, since in that case P will never broadcast *secret* on public channel c.

Contribution. Our contribution is threefold. Firstly, we define a semantic variant of the *Probabilistic Polynomial-time Process Calculus* PPC [11] (Section 2), called $\mathrm{PPC}_{\nu\sigma}$, to cope with the problem of characterizing the intruders's capacity. Contrary to most probabilistic models, our operational semantics does not normalize probabilities. The reason is that normalizing has the effect of removing control of its own actions from the intruder. Consider the process $P = \overline{c}(m).Q_1|\overline{c}(m).Q_2$: depending on whether P represents a protocol or an intruder, the scheduling of a component is respectively equiprobable or arbitrarily chosen by the intruder. A solution might be to discriminate semantically between a protocol and an intruder, but this rapidly becomes rather intricate since synchronization actions could be committed by both. We propose here a simpler solution to this problem. It consists of equipping the intruder with an attack strategy i.e., a selection process called *external scheduler* (Section 2.3), allowing it to choose the next action to perform at each evaluation step. This scheduling is carefully designed to reflect as accurately as possible the intruder's real capacities, i.e. to control the communication network without controlling internal reactions of the system under its stimuli.

Secondly, we reformulate (Section 3) the observational equivalence of [11] into a more amenable form to take account of all observable traces instead only single step.

Finally, to illustrate the pertinence of our approach, we conclude the paper by an extensive case study (Section 4): the analysis of the Chaum's *Dining Cryptographers* protocol [8]. We give a probabilistic version of the possibilistic specification of the *anonymity* property as per Schneider and Sidiropoulos [12], and prove that restricting the scheduler's power too much may lead to very weak models which cannot detect flawed specifications of the protocol.

Related work. The technical precursor of our framework is the process calculus of Mitchell *et al.* [11]. Though any of the models [1,6,3,11,4,10] or any similar framework could have been an interesting starting point, the framework of Mitchell *et al.* [11] appears appropriate for the following reasons. Although it is a formal model, it is still close to computational setting of modern cryptography since it works directly on the cryptographic level. Indeed it defines an extension of the CCS process algebra with finite replication and probabilistic polynomial-time terms denoting cryptographic primitives. It turns out that these probabilistic polynomial functions are useful for modelling the probabilistic behaviour of security systems. Unlike formalisms such as [6,4,10], the scheduling is probabilistic, better reflecting so the ability of the attacker to control the communication network. Finally it also appears as a natural formal framework for capturing and reasoning about a variety of fundamental cryptographic notions.

The problem of characterizing the schedulers' capacities has recently been considered in [5,7,9]. In [5] the authors treated the problem of overly powerful *schedulers* in the context of systems modelled in a *Probabilistic I/O Automata* framework. They restricted the scheduler by defining two levels of schedulers. A high-level scheduler called *adversarial scheduler* is a component of the system and controls the communication network, i.e. it schedules public channels. This component has limited knowledge of the behaviour of other components in the system: their internal choices and secret information are hidden. On the other hand, a low-level scheduler called *tasks scheduler* resolves the remaining non-determinism by a *task schedule*. These tasks are equivalence classes of actions that are independent of the high-level scheduler choices. We believe that these tasks may correspond to our "strategically equivalent action".

In Garcia *et. al.* [9] a dual problem to the one we have considered here, namely the problem that arises when traditional schedulers are overly powerful, is addressed. In the context of security protocols modelled by probabilistic automata, they define a probabilistic scheduler that assign, in the current state, a probability distribution on the possible non-deterministic next transitions. Unlike our scheduler, it is history-dependent since it defines equiprobable paths and it is not stochastic, and might therefore halt execution at any time. Roughly speaking, admissible schedulers are defined w.r.t bisimulation equivalence: any observably trace equivalent paths are equiprobably scheduled and lead to bisimilar states.

Another recent paper on the scheduling issue is presented in [7]. Unlike our scheduler and that of [9] which are both defined on the semantic level, [7] proposes a framework in which schedulers are defined and controlled on the syntactic level. They make random choices in the protocol invisible to the adversary. Note that we achieve the same goal by using the operational semantics `Eval` rule which reduces unblocked processes, as well as our strategically equivalent classes of actions. More investigation is needed for papers [5,7] to determine how the approaches may benefit from each other.

Finally an alternate approach is proposed in [4,10]. Instead of scheduling a single action (like ours), or a path (like the one of [9]), a process is scheduled. The

problems of discriminating between a protocol's actions and intruder's ones, and the privileging of internal actions, is meaningless in these models because scheduling is included implicitly in the specification. In other words, the protocol designer determines when control passes from the protocol to the attacker. Let us explain this last point. Consider the protocol $P = \nu(c)(\overline{c}(1).|c(x).\overline{c'}(0).)$ which, after an internal communication, outputs 0 on the public channel c'. In the frameworks of [4,10], it may be specified in two different manners

$$P_1 = \nu(c)(start().\overline{c}(1).\mathbf{0}|c(x).\overline{c'}(0).\mathbf{0})$$

and

$$P_2 = \nu(c)(start().\overline{c}(1).\mathbf{0}|c(x).\overline{contr2Intr}().getContr().\overline{c'}(0).\mathbf{0})$$

depending on whether or not we want to make the internal action completely invisible to the attacker. In this way the user has total freedom and can eliminate undesirable schedulers at the specification level. The drawback is that the protocol designer who has incomplete knowledge about the system, may specify his intuition of the protocol and so get some properties that might not be satisfied by the actual protocol.

2 The PPC$_{\nu\sigma}$ model

The process algebra PPC$_{\nu\sigma}$ extends semantically the *Probabilistic Polynomial-time Process Calculus PPC* [11] to better take into account the analysis of probabilistic security protocols.

2.1 Syntax of PPC$_{\nu\sigma}$

Terms. The set of terms $\mathcal{T}$ of the process algebra PPC$_{\nu\sigma}$ consists of variables $\mathcal{V}$, numbers $\mathbb{N}$, pairs, and a specific term $\mathbf{N}$ standing for the security parameter. The security parameter may be understood as the cryptographic primitives' key length and may appear in the probabilistic polynomial functions defined below. Formally we have

$$t ::= n \ (integer) \mid x \ (variable) \mid \mathbf{N} \ (secur.\ param.) \mid (t,t) \ (pair)$$

For each term t, $fv(t)$ is the set of variables in t. A *message* is a closed term (i.e. not containing variables). The set of messages is denoted $\mathcal{M}$.

Functions. The call of probabilistic as well as deterministic cryptographic primitives, such as keys and nonces generation, encryption, decryption, etc., is modelled by probabilistic polynomial functions[2] $\Lambda : \mathcal{M}^k \to \mathcal{M}$ satisfying

$$\forall (m_1, \ldots, m_k) \in \mathcal{M}^k,\ \forall m \in \mathcal{M},\ \forall \lambda \in \Lambda,\ \exists p \in [0,1] \text{ such that}$$
$$\text{Prob}[\lambda(m_1, \cdots, m_k) = m] = p.$$

We denote $\lambda(m_1, \cdots, m_k) \hookrightarrow x$ the assignment of the value $\lambda(m_1, \cdots, m_k)$ to the variable x and by $\lambda(m_1, \cdots, m_k) \overset{p}{\hookrightarrow} m$ if $\lambda(m_1, \cdots, m_k)$ evaluates to m with

[2] See Appendix A for a formal definition of a *probabilistic polynomial function*

probability p. From Definition A.1 (see Appendix A), the set

$$\mathrm{Im}(\lambda(m_1, \cdots, m_k)) = \{m \mid \exists p \in]0..1]\ \lambda(m_1, \cdots, m_k) \overset{p}{\hookrightarrow} m\}$$

of m s.t. $\lambda(m_1, \cdots, m_k)$ evaluates to m with non-zero probability, is a finite set.

For instance, RSA encryption which takes as parameters, a message m to encrypt and an encryption key formed of the pair (e, n), and returns the number $m^e \ mod\ n$, is the function $\lambda_{RSA}(m, e, n)$ returning c with probability 1 if $c = m^e \ mod\ n$, and 0 otherwise. Similarly, we can model the key guessing attack of a cryptosystem by the product $[\mathrm{rand}(1^k) \hookrightarrow key][\mathrm{dec}(c, key) \hookrightarrow x]$ where 1^k is the size of the key randomly generated by the function rand, and the decryption function dec returns m with probability 1 if c is the cryptogram of m encrypted by k and $key = k$. The success of such an attack has probability $p = \frac{1}{2^{|k|}}$. These few examples illustrate the expressive power offered by these functions. We limit ourselves to the probabilistic polynomial ones in order to model all attacks realizable (in the model) in polynomial time.

Processes. Let $\mathcal{C}$ be a countable set of public channels. We assume that each channel is equipped with a bandwidth given by the polynomial function $bw : \mathcal{C} \longrightarrow \mathbb{N}$. We say that a message m belongs to the domain of a channel c, written $m \in \mathrm{dom}(c)$, if the message length $|m|$ is less than or equal to the channel bandwidth, i.e. $m \in \mathrm{dom}(c) \iff |m| \leq bw(c)$. Note that $|(m, m')| = |m| + |m'| + r$ where r is the length of a fixed bits string, which allows us to concatenate and decompose two terms without any ambiguity.

Processes in $\mathrm{PPC}_{\nu\sigma}$ are built as follows :

$$\begin{aligned} P ::= \mathbf{0} \quad &| \quad c(x).P \quad | \quad \overline{c}(m).P \quad | \quad P|P \quad | \quad (\nu c)P \quad | \quad [t = t]P \quad | \\ &| \quad !_{q(\mathbf{N})}P \quad | \quad [\lambda(t_1, \cdots, t_n) \hookrightarrow x]P \end{aligned}$$

Given a process P, the set $fv(P)$ of *free variables*, is the set of variables x in P which are not in the scope of any prefix either input (of the form $c(x)$) or probabilistic evaluation (of the form $[\lambda(t_1, \cdots, t_n) \hookrightarrow x]$). A process without free variables is called *closed* and the set of closed processes is denoted by $\mathcal{P}roc$. Hereafter, all processes are considered closed.

The mechanisms for reading, emitting, parallel composition, restriction and matching are all standard. The finite replication $!_{q(\mathbf{N})}P$ is the $q(\mathbf{N})-$fold parallel composition of P with itself, where q is a polynomial function. The novelty is the call and return of probabilistic polynomial functions

$$[\lambda(t_1, \cdots, t_n) \hookrightarrow x]P$$

This feature allows to model (probabilistic) polynomial cryptographic primitives as well as the probabilistic character of a protocol. In fact it is the main source of probability in this model.

The following examples illustrate the way $\mathrm{PPC}_{\nu\sigma}$ can be used to model protocols.

Example 2.1 Let $Kgen$ be a (probabilistic) polynomial function that, given the security parameter $\mathbf{N}$, generates a secret encryption key and enc a (probabilistic)

encryption function that, given a secret key K and a message M, returns the cryptogram of M under K. The process P that receives a message over the public channel c generates an encryption key and returns the cryptogram of this message over the same public channel. This is modeled in $\mathrm{PPC}_{\nu\sigma}$ as follows:

$$P := c(x).[Kgen(\mathbf{N}) \hookrightarrow y][enc(x,y) \hookrightarrow z]\overline{c}(z).\mathbf{0}$$

Example 2.2 Though that the calculus does not consider the probabilistic alternative choice operator "+", it is possible to simulate it by means of the parallel composition operator and some special probabilistic functions.

Let $flips$ be a probabilistic polynomial function that given a coin return $Head$ with probability p and $Tail$ with probability $1-p$. Then the process Q that flips a coin and then behaves as the process Q' if the result of coin flipping is $Head$ and as the process Q'' otherwise, is modeled in our calculus as follows:

$$Q := [flips(coin) \hookrightarrow x]([x = Head]Q' | [x = Tail]Q'')$$

2.2 Operational semantics

The set of actions

$$\mathcal{A}ct = \{\overline{c}(m), c(m), \overline{c}(m)\cdot c(m), c(m)\cdot\overline{c}(m), \tau \mid m \in \mathcal{M} \text{ and } c \in \mathcal{C}\}$$

consists of the set of partial input and output actions, of the set of synchronization actions on public channels, and of the internal action τ:

$$\mathcal{P}artial = \{\overline{c}(m), c(m) \mid m \in \mathcal{M} \text{ and } c \in \mathcal{C}\}$$
$$\mathcal{A}ctual = \{\overline{c}(m)\cdot c(m), c(m)\cdot\overline{c}(m), \tau \mid m \in \mathcal{M} \text{ and } c \in \mathcal{C}\}$$

The set of observable actions is given by $\mathcal{V}is = \mathcal{A}ct - \{\tau\}$. The operational semantics of $\mathrm{PPC}_{\nu\sigma}$ is a probabilistic transition system $(\mathcal{E}, \mathcal{T}, E_0)$ generated by the inference rules given in Table 1 where $\mathcal{E} \subseteq \mathcal{P}roc$ is the set of states, $\mathcal{T} \subseteq \mathcal{E} \times Act \times [0,1] \times \mathcal{E}$ the set of transitions and $E_0 \in \mathcal{P}roc$ the initial state. The notation $P \xrightarrow{\alpha[p]} P'$ stands for $(P, \alpha, p, P') \in \mathcal{T}$. It is an extension of the *CCS* semantics, with a mechanism for calling probabilistic polynomial functions. We sketch it briefly here.

To make sure that internal computations of functions do not interfere with communication actions (in particular with those on public channels controlled by the intruder), all exposed functions in a process (`Eval` rule) are simultaneously evaluated by the probabilistic polynomial function *eval* defined in Table 2 below as illustrated in Example 2.3. This evaluation step allows us to get what we call a *blocked process*, i.e. a process having no more internal computations to perform. The set of blocked processes is denoted by $\mathcal{B}locked$.

Example 2.3 If processes Q' and Q'' of Example 2.2 are blocked processes then Q reduces to Q' with probability p and to Q'' with probability $1-p$. In other words, $\mathrm{Prob}[eval(Q) = Q'] = p$ and $\mathrm{Prob}[eval(Q) = Q''] = 1-p$.

The output mechanism allows a principal A to send a message on public channels (`Output` rule). Dually, the input mechanism must be ready to receive any message

$$\texttt{Eval.}\ \frac{eval(P) \xrightarrow{p} P' \quad P \notin \mathcal{B}locked}{P \xrightarrow{\tau[p]} P'}$$

$$\texttt{Output}\ \frac{m \in \mathrm{dom}(c)}{\overline{c}(m).P \xrightarrow{\overline{c}(m)[1]} P} \qquad \texttt{Input}\ \frac{m \in \mathrm{dom}(c)}{c(x).P \xrightarrow{c(m)[1]} P[m/x]}$$

$$\texttt{ParL.}\ \frac{P_1 \xrightarrow{\alpha[p]} P_1' \quad (P_1|P_2) \in \mathcal{B}locked}{P_1|P_2 \xrightarrow{\alpha[p]} P_1'|P_2} \qquad \texttt{ParR.}\ \frac{P_2 \xrightarrow{\alpha[p]} P_2' \quad (P_1|P_2) \in \mathcal{B}locked}{P_1|P_2 \xrightarrow{\alpha[p]} P_1|P_2'}$$

$$\texttt{SyncL.}\ \frac{P_1 \xrightarrow{\overline{c}(m)[p_1]} P_1' \quad P_2 \xrightarrow{c(m)[p_2]} P_2'}{P_1|P_2 \xrightarrow{\overline{c}(m).c(m)[p_1.p_2]} P_1'|P_2'} \qquad \texttt{SyncR.}\ \frac{P_1 \xrightarrow{c(m)[p_1]} P_1' \quad P_2 \xrightarrow{\overline{c}(m)[p_2]} P_2'}{P_1|P_2 \xrightarrow{c(m).\overline{c}(m)[p_1.p_2]} P_1'|P_2'}$$

$$\texttt{RestCL.}\ \frac{P \xrightarrow{\overline{c}(m).c(m)[p]} P'}{(\nu c)P \xrightarrow{\tau[p]} (\nu c)P'} \qquad \texttt{RestCR.}\ \frac{P \xrightarrow{c(m).\overline{c}(m)[p]} P'}{(\nu c)P \xrightarrow{\tau[p]} (\nu c)P'}$$

$$\texttt{Rest}\ \frac{P \xrightarrow{\alpha[p]} P' \quad \alpha \notin \{c(m), \overline{c}(m), \overline{c}(m)\cdot c(m), c(m)\cdot\overline{c}(m) : m \in \mathcal{M}\} \quad P \in \mathcal{B}locked}{(\nu c)P \xrightarrow{\alpha[p]} (\nu c)P'}$$

Table 1
Operational semantics of PPC$_{\nu\sigma}$

$$\mathrm{Prob}[eval(\mathbf{0}) = \mathbf{0}] = 1$$
$$\mathrm{Prob}[eval(c(x).P) = c(x).P] = 1 \ \text{ and } \ \mathrm{Prob}[eval(\overline{c}(m).P) = \overline{c}(m).P] = 1$$
$$\mathrm{Prob}[eval((\nu c)P) = (\nu c)Q] = \mathrm{Prob}[eval(P) = Q]$$
$$\begin{cases} \mathrm{Prob}[eval([m = m']P) = Q] = \mathrm{Prob}[eval(P) = Q] & \text{if } m = m' \\ \mathrm{Prob}[eval([m = m']P) = \mathbf{0}] = 1 & \text{else} \end{cases}$$
$$\mathrm{Prob}[eval(P|Q) = P'|Q'] = \mathrm{Prob}[eval(P) = P'] \times \mathrm{Prob}[eval(Q) = Q']$$
$$\mathrm{Prob}[eval([\lambda(m_1, \cdots, m_k) \hookrightarrow x]P) = Q] =$$
$$\textstyle\sum_{m \in Im(\lambda(m_1, \cdots, m_k))} \mathrm{Prob}[\lambda(m_1, \cdots, m_k) = m] \times \mathrm{Prob}[eval(P[m/x]) = Q]$$

Table 2
Reduction of unblocked processes

on a public channel (`Input` rule). The parallelism (`Par.` rules) operator is defined as usual. It is worth noting that the semantics keep track of information involved in an interaction (the message and the communication channel) (`Syn.` rules), contrary to most process algebra semantics where this information is lost, as the only action resulting from such a communication is usually the invisible action τ. The restriction operator ν is used to model private channels. The process $(\nu c)P$ behaves like P restricted to actions not on c unless a synchronization occurs on c (i.e. actions of the form $c(m) \cdot \overline{c}(m)$ or $\overline{c}(m) \cdot c(m)$). In this case, they are observed as an invisible action τ (`Rest.` rules).

Note that, transition systems generated by the operational semantics (Table 1) of PPC$_{\nu\sigma}$ processes, are not purely probabilistic. Consider for example process $P = \overline{c_1}(a)|\overline{c_2}(b)$. Clearly, the sum of probabilities of outgoing transitions of P, is equal to 2. This is due to the parallel composition which introduces nondeterminism. In order to resolve this nondeterminism it is mandatory to schedule, at each evaluation step of the process, all available distinct actions. However, security protocols are assumed to be executed in hostile environments, i.e. environments with external intruders having full control of the communication network, with the ability to assign any probabilistic distribution to the controlled channels (to the public actions). This

is modelled by putting public actions under control of an external scheduler. Since we do not want to define a particular attack strategy, the scheduling is not included in the semantics of processes, but rather in the intruder's definition: the intruder is then formed by the pair (Π, S) of process Π and scheduler S. One may view the hostile environment as Π interacting with the protocol, and S as its attack strategy.

2.3 External Scheduler

Given a protocol P attacked by the intruder (Π, S), evaluation of $P|\Pi$ along the strategy S is a four step process consisting of:

Reduction: evaluation of all exposed probabilistic functions in $P|\Pi$.

Localization: indexing of executable actions along $eval(P|\Pi)$ to discriminate whether or not an executable action interferes with an intruder's action.

Selection: scheduling[3] S among available actions.

Execution: the action chosen by S is executed and the process is repeated until there are no more executable actions.

Localization. Scheduling should be capable of determining whether or not the intruder is attached to an action. Since a system P attacked by the intruder Π is simply modelled by $P|\Pi$, actions are indexed by the positions of the components they belong to, e.g. if P and Π have respectively n and k parallel components, then $P|\Pi$ consists of $n+k$ components. By convention the attacker is on the right side, so that actions indexed by integers less than or equal to n belong to the protocol while those indexed by integers greater than n belong to the intruder. A partial action is indexed with an integer denoting the component to which it belongs, while a communication action is indexed by a pair of integers denoting the components to which complementary partial actions belong.

Let $\mathcal{I}ndex = \mathbb{N} \cup \mathbb{N}^2$ and the function support : $Act \setminus \{\tau\} \to \mathcal{C}$ where support(α) is the name of the channel where α occurred.

Definition 2.4 The localization function $\chi : \mathcal{B}locked \longrightarrow 2^{\mathcal{A}ct \times \mathcal{I}ndex}$ is defined recursively as follows:

$$
\begin{aligned}
\chi(\mathbf{0}) &= \emptyset \\
\chi(\alpha.P) &= \{(\alpha, 1)\} \\
\chi(P|Q) &= \chi(P) \cup \{(\alpha, \rho(P)+i) \mid (\alpha, i) \in \chi(Q)\} \\
&\quad \cup \{(\alpha \cdot \overline{\alpha}, i, \rho(P)+j) \mid (\alpha, i) \in \chi(Q) \text{ and } (\overline{\alpha}, j) \in \chi(Q)\} \\
\chi((\nu c)P) &= \{(\tau, i, j) \mid (\alpha \cdot \overline{\alpha}, i, j) \in \chi(P) \text{ and } \mathrm{support}(\alpha) = c\} \\
&\quad \cup \{(\tau, k, l) \mid (\tau, k, l) \in \chi(P)\}
\end{aligned}
$$

where $\alpha \in \mathcal{P}artial$, $\max(i,(j,k)) = \max(i,j,k)$. and

$$
\rho(P) = \begin{cases} \max\{ID \in \mathcal{I}ndex \mid (\beta, ID) \in \chi(P),\ \beta \in \mathcal{V}is\} & \text{if } \chi(P) \neq \emptyset \\ 0 & \text{otherwise} \end{cases}
$$

[3] Note that scheduling is defined only for blocked processes. In fact, the only action available to an unblocked process is the internal action corresponding to functions evaluation.

Selection. The function χ allows for action localization, but more guidelines are needed to know whether or not an indexed action interferes with an intruder's components. Actually, a partition of the set of indexed actions into *classes of strategically equiprobable actions*, i.e. classes of actions uniformly chosen in a strategy S, is needed. Intuitively, a class corresponds to actions which can not be distinguished by any scheduler. The construction of the quotient set must agree with the following principles:

(1) No strategy distinguishes between internal actions of a protocol.

(2) No strategy allows the intruder to control internal reactions of a protocol P to any external stimulus. So, if P can react in many positions to a stimulus of the intruder, then all these positions should have the same probability to react to the intruder's request.

(3) In any strategy, the intruder has complete control on its own actions.

Given a protocol P and an attacker Π, we use χ to compute two sets I_1 and I_2 s.t. indices corresponding to the protocol's components belong to I_1 and those corresponding to the intruder's ones belong to I_2. Formally:

$$I_1 = \begin{cases} \{1, 2, \cdots, \rho(\mathit{eval}(P))\} & \text{if } \chi(\mathit{eval}(P)) \neq \emptyset \\ \emptyset & \text{otherwise} \end{cases}$$

$$I_2 = \begin{cases} \{\rho(\mathit{eval}(P)) + 1, \cdots, \rho(\mathit{eval}(P)) + \rho(\mathit{eval}(\Pi))\} & \text{if } \chi(\mathit{eval}(\Pi)) \neq \emptyset \\ \emptyset & \text{otherwise} \end{cases}$$

Example 2.5 Let $P = \overline{c}(m).P_1|\overline{c}(m').P_2$ et $\Pi = c(x).\Pi'$ then $\chi(\mathit{eval}(P|\Pi))$ is the set

$$\{(\overline{c}(m), 1), (\overline{c}(m'), 2), (c(m), 3), (c(m'), 3), (\overline{c}(m) \cdot c(m), 1, 3), (\overline{c}(m') \cdot c(m'), 2, 3)\}$$

and so, $I_1 = \{1, 2\}$ et $I_2 = \{3\}$.

The quotient set of strategically equiprobable actions is summarized in Table 3 where $[(\alpha, ID)]_{I_1 \times I_2}$ denotes the equiprobable class of the indexed action (α, ID) w.r.t. sets I_1 and I_2. Let us briefly describe these classes.

Due to principle (1), *internal actions* of P are equiprobable: it is reflected in the definition of $[(\tau, i, j)]_{I_1 \times I_2}$; τ actions indexed by the components positions of P (i.e. in I_1) are equivalent. Otherwise $[(\tau, i, j)]_{I_1 \times I_2}$ is reduced to itself w.r.t (3). Due to principle (2), *partial outputs* on a given public channel are equiprobable: although the intruder can choose the public channel to spy on, it has no control of messages transmitted on it. Otherwise $[(\overline{c}(m), i)]$ is reduced to itself: being an intruder's action, it can choose both the message and the component to build its attack. *Partial input* is the dual case of partial output, with, by contrast, the intruder controlling the message (sent by itself) that is received by P. The same principles apply to *public synchronization.* The intruder can choose a listening channel c and act merely as an observer, then any communication on c takes place between two components of P. It has no control on either the messages exchanged, nor on components where communication occurs. However, if communication arises

$$[(\tau,i,j)]_{I_1\times I_2} = \begin{cases} \{(\tau,i',j') \mid i',j' \in I_1\} & \text{if } i,j \in I_1 \\ \{(\tau,i,j)\} & \text{otherwise} \end{cases}$$

$$[(\overline{c}(m),i)]_{I_1\times I_2} = \begin{cases} \{(\overline{c}(m'),i') \mid i' \in I_1, m' \in \mathrm{dom}(c)\} & \text{if } i \in I_1 \\ \{(\overline{c}(m),i)\} & \text{otherwise} \end{cases}$$

$$[(c(m),i)]_{I_1\times I_2} = \begin{cases} \{(c(m),j) \mid j \in I_1\} & \text{if } i \in I_1 \\ \{(c(m),i)\} & \text{otherwise} \end{cases}$$

$$[(\overline{c}(m)c(m),i,j)]_{I_1\times I_2} = \begin{cases} \{(\alpha\overline{\alpha},i',j') \mid i',j' \in I_1, \mathrm{support}(\alpha)=c\} & \text{if } i,j \in I_1 \\ \{(\overline{c}(m')c(m'),i',j) \mid i' \in I_1, m' \in \mathrm{dom}(c)\} & \text{if } i \in I_1, j \in I_2 \\ \{(\overline{c}(m)c(m),i,j)\} & \text{otherwise} \end{cases}$$

$$[(c(m)\overline{c}(m),i,j)]_{I_1\times I_2} = \begin{cases} \{(\alpha\overline{\alpha},i',j') \mid i',j' \in I_1, \mathrm{support}(\alpha)=c\} & \text{if } i,j \in I_1 \\ \{(c(m)\overline{c}(m),i',j) \mid i' \in I_1\} & \text{if } i \in I_1, j \in I_2 \\ \{(c(m)\overline{c}(m),i,j)\} & \text{otherwise} \end{cases}$$

Table 3
Strategically equiprobable actions

from the protocol (output) to the intruder (input), then the intruder can select the channel and its (input) component. For synchronization arising in the opposite direction, the intruder can select not only the channel but also the message and its output component as well. Finally, if communication arises between two components of the intruder, then the intruder controls everything.

Definition 2.6 [External scheduler] An *external scheduler* is a stochastic polynomial probabilistic function $S : 2^{Act\times\mathcal{I}ndex} \times 2^{\mathbb{N}} \times 2^{\mathbb{N}} \to Act \times \mathcal{I}ndex$ s.t. for any non empty set $A \subseteq Act \times \mathcal{I}ndex$ and any pair of sets $I_1, I_2 \subseteq \mathbb{N}$ (with $I_1 \neq \emptyset$ or $I_2 \neq \emptyset$) satisfying $\forall_{(i_1,i_2)\in I_1\times I_2} i_1 < i_2$, the following holds:

(i) $\sum_{(\tau,i,j)\in A, i,j\in I_1} \mathrm{Prob}[S(A,I_1,I_2) = (\tau,i,j)] \in \{0,1\}$.

(ii) $\forall_{\alpha,\beta\in A}\alpha \in [\beta]_{I_1\times I_2} \Rightarrow \mathrm{Prob}[S(A,I_1,I_2) = \alpha] = \mathrm{Prob}[S(A,I_1,I_2) = \beta]$.

The set of schedulers is denoted by $Sched$[4].

From the stochasticity condition (being itself a progress condition since it states that at each step of the process at least one of the executable actions will be scheduled), we have the following result:

Lemma 2.7 *Let P be a process s.t. $A = \chi(eval(P)) \neq \emptyset$ then the following holds:* $\forall_{S\in Sched}\exists_{\alpha\in A}\mathrm{Prob}[S(A) = \alpha] \neq 0$.

Our main result on schedulers follows.

Theorem 2.8 *The sum of probabilities of outgoing transitions in any state along any external scheduler is smaller than or equal to 1.*

Proof. Let P be a process and $\mathrm{Exec}(P)$ the set of outgoing transitions of P. Two cases follow:

[4] Given a protocol P and an intruder (Π, S), we know how to compute I_1 et I_2 induced from $A = \chi(P|\Pi)$. Hereafter, for the sake of simplicity, we write $S(A)$ for $S(A, I_1, I_2)$.

$[P \notin \mathcal{B}locked]$: in this case there is no scheduling and

$$\mathrm{Exec}(P) = \{P \xrightarrow{\tau[q]} Q \mid \mathit{eval}(\mathrm{P}) \overset{q}{\hookrightarrow} Q\}.$$

The sum of the probabilities of outgoing transitions of P is

$$\begin{aligned} p &= \sum_{Q \in Im(\mathit{eval}(\mathrm{P}))} \mathrm{Prob}[\mathit{eval}(\mathrm{P}) = Q] \\ &\leq 1 \qquad \text{by definition of } \mathit{eval}. \end{aligned}$$

$[P \in \mathcal{B}locked]$: in this case:

$$\mathrm{Exec}(P) = \{t_{ID} = P \xrightarrow{\alpha[q_{ID}]} Q \mid (\alpha, ID) \in \chi(P)\}.$$

The sum of the probabilities of outgoing transitions of P according to the scheduler S is

$$\begin{aligned} p &= \sum_{(\alpha,ID)\in\chi(P)} \mathrm{Prob}[S(\chi(P)) = (\alpha, ID)] \times q_{ID} \\ &\leq \sum_{(\alpha,ID)\in\chi(P)} \mathrm{Prob}[S(\chi(P)) = (\alpha, ID)] \qquad \text{since } \forall ID \in \mathcal{I}ndex,\ q_{ID} \leq 1 \\ &\leq 1 \qquad \text{by definition of } S. \end{aligned}$$

□

2.4 Cumulative probability distribution

Transition systems induced by the operational semantics of Table 1 may have a state P with several outgoing transitions labeled by the same action and the same probability. But to correctly compute the probability of outgoing transitions of P according to a scheduler, we must ensure that they can be uniquely identified. If P is blocked then χ enables us to uniquely index the outgoing transitions of P. If P is unblocked then there exists a finite number $n = |Im(\mathit{eval}(P))|$ of processes Q_i $(1 \leq i \leq n)$ s.t. $\exists_{q_i \neq 0}\ P \xrightarrow{\tau[q_i]} Q_i$ is an outgoing transition of P. We can order Q_i from 1 to n and use this ordering to index outgoing transitions of P s.t. $P \xrightarrow{\tau[q_i]} Q_i$ being indexed by (i, i) [5].

Let $\sigma = (\alpha_1, id_1) \ldots (\alpha_n, id_n)$ be a sequence of indexed actions. Then σ is a path from P to Q if there exist nonzero probabilities $p_1, \ldots, p_n$ s.t.

$$P_0 \xrightarrow{\alpha_1[p_1]} P_1 \xrightarrow{\alpha_2[p_2]} \cdots \xrightarrow{\alpha_n[p_n]} P_n,$$

$P = P_0$ and $Q = P_n$. Similarly, we say that P reaches Q by path σ with probability p according to S, denoted by $P \overset{\sigma[p]}{\Longrightarrow}_S Q$, if the probability that S chooses σ is $\mathrm{Prob}[S(\sigma)] = \prod_{1\leq i\leq n} q_i = p$ where

[5] by analogy to the indexing of τ actions by χ

$$q_i = \begin{cases} p_i & \text{if } P_{i-1} \notin \mathcal{B}locked \\ \text{Prob}[S(\chi(P_{i-1})) = (\alpha_i, id_i)] \times p_i & \text{otherwise} \end{cases}$$

Example 2.9 Let P be the process whose transition system in shown in Figure 1 where $P_1 \notin \mathcal{B}locked$ and $\text{Prob}[eval(P_1) = P_i] = p_i$ for $3 \leq i \leq 5$. Let S be scheduler such that

- $\text{Prob}[S(\{(\alpha_1, 1); (\alpha_2, 2)\}) = (\alpha_1, 1)] = q$
- $\text{Prob}[S(\{(\alpha_1, 1); (\alpha_2, 2)\}) = (\alpha_2, 2)] = 1 - q$ [6]
- $\text{Prob}[S(\{(\alpha_3, 1)\}) = (\alpha_3, 1)] = 1$ (according to the stochasticity condition).

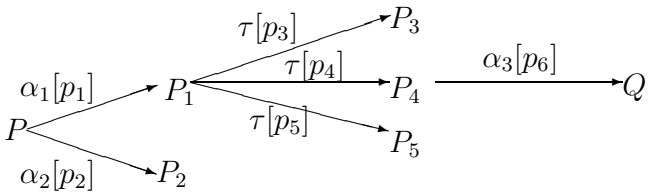

Fig. 1. Transition system of P

Then the probability that P reaches Q by the path $\sigma = (\alpha_1, 1)(\tau, (2, 2))(\alpha_3, 1)$ according to the scheduler S is

$$\text{Prob}[S(\sigma)] = (p_1 \times q) \times p_4 \times (p_6 \times 1).$$

An *α-path* is a path of type $(\tau, id_1)(\tau, id_2) \ldots (\tau, id_{n-1})(\alpha, id_n)$ $(n \geq 1)$. The notation $P \stackrel{\alpha[p]}{\Longrightarrow}_S Q$ means that there exists an α-path σ s.t. $P \stackrel{\sigma[p]}{\Longrightarrow}_S Q$. Similarly $P \stackrel{\hat{\alpha}[p]}{\Longrightarrow}_S Q$ denotes $P \stackrel{\alpha[p]}{\Longrightarrow}_S Q$ if $\alpha \neq \tau$ and $P \stackrel{\tau^*[p]}{\Longrightarrow}_S Q$ otherwise.

Let $\mathscr{E} \subseteq \mathcal{P}roc$ be a set of processes and $Q \in \mathscr{E}$. Let P be a process, and σ an α-path from P to Q. Then σ is *minimal* w.r.t $\mathscr{E}$ if no other α-path σ' exists from P to Q' s.t. σ' is a prefix [7] of σ and $Q' \in \mathscr{E}$. We denote by $Paths(P, \stackrel{\alpha}{\Longrightarrow}, \mathscr{E})$ the set of all minimal α-paths from P to an element of $\mathscr{E}$.

Definition 2.10 Let $\mathscr{E} \subseteq \mathcal{P}roc$ be a set of processes. The cumulative probability that P reaches a process in $\mathscr{E}$ by an α-path according to S is computed by the *cumulative probability function* $\mu : \mathcal{P}roc \times Act \times 2^{\mathcal{P}roc} \times Sched \to [0, 1]$

$$\mu(P, \stackrel{\hat{\alpha}}{\Longrightarrow}_S, \mathscr{E}) = \begin{cases} 1 & \text{if } P \in \mathscr{E}, \alpha = \tau \\ \sum\{\text{Prob}[S(\sigma)] : \sigma \in Paths(P, \stackrel{\alpha}{\Longrightarrow}, \mathscr{E})\} & \text{otherwise} \end{cases}$$

The next theorem follows by induction on the length of α-paths.

Theorem 2.11 *The cumulative probability function is well defined i.e.* $\forall_{P,\alpha,\mathscr{E},S}\ \mu(P, \stackrel{\hat{\alpha}}{\Longrightarrow}_S, \mathscr{E}) \leq 1$.

[6] Note that if $(\alpha_1, 1)$ and $(\alpha_2, 2)$ are equiprobable then $q = 1 - q$, i.e. $q = \frac{1}{2}$.

[7] Prefixing does not take into account any indexing, e.g. (α_1, id_3) is a prefix of $(\alpha_1, id_1)(\alpha_2, id_2)$ for all index id_1 and id_3. Note also that the minimality condition applies only to τ-paths.

3 Probabilistic behavioural equivalences

Now we plan to establish equivalences ensuring that a protocol satisfies a security property if and only if it is observationally equivalent to an abstraction of the protocol, satisfying the security property by construction. In other words we request two processes to be equivalent if and only if, when subject to same attacks, they generate "approximately" the same observations. By "approximately" we mean *asymptotically* closed w.r.t. the security parameter [8].

3.1 Asymptotic observational equivalence

We start by defining the notion of an observable and the probability that a given process P generates a particular observable. An observable is simply a pair (c, m) of a public channel and a message. The set of all observables is denoted by $\mathcal{O}bs$. The probability of observing (c, m) is defined as the sum of the probability of observing it directly, i.e. the cumulative probability of executing an $\overline{c}(m) \cdot c(m)$-path or an $c(m) \cdot \overline{c}(m)$-path to reach any state, and the probability of observing it indirectly, i.e. by observing first some different visible actions before observing it. For that purpose we extend the notion of cumulative probability to the so-called *cumulative probability up to* H.

Definition 3.1 Let $\mathscr{E} \subseteq \mathcal{P}roc$ be a set of processes, P a process, S a scheduler and $H \subset \mathcal{A}ctual \setminus \{\tau\}$ a set of visible actions. The *cumulative probability up to* H is defined inductively as follows: $\forall \alpha \in \mathcal{A}ctual \setminus H$

$$\mu(P, \stackrel{\hat{\alpha}}{\Longrightarrow}_{S/H}, \mathscr{E}) = \begin{cases} 1 \quad \text{if } P \in \mathscr{E} \text{and } \alpha = \tau, \\ \mu(P, \stackrel{\hat{\alpha}}{\Longrightarrow}_{S}, \mathscr{E}) + \\ \sum_{\beta \in H,\, Q \in \mathcal{P}roc} \mu(P, \stackrel{\hat{\beta}}{\Longrightarrow}_{S}, \{Q\}) \mu(Q, \stackrel{\hat{\alpha}}{\Longrightarrow}_{S/H}, \mathscr{E}) \text{ otherwise} \end{cases}$$

Lemma 3.2 *The cumulative probability up to* H *is well defined, i.e.* $\forall P, \alpha, \mathscr{E}$, S *and* H, $\mu(P, \stackrel{\hat{\alpha}}{\Longrightarrow}_{S/H}, \mathscr{E}) \leq 1$.

Proof. If $P \in \mathscr{E}$ and $\alpha = \tau$, then $\mu(P, \stackrel{\hat{\alpha}}{\Longrightarrow}_{S/H}, \mathscr{E}) = 1$ and the result follows. Assume that we are not in that case. Let $\sigma = \tau^* \alpha_1 \cdots \tau^* \alpha_n$ be a path of n actual visible actions separated by finite numbers of internal actions (we omit the index of the actions). We say that action α_i $(1 \leq i \leq n)$ is at distance i from process P on path σ according to the scheduler S, denoted $d_\sigma(P, \alpha_i, S) = i$, if there exist both a nonzero probability p, and a process Q such that $P \stackrel{\sigma[p]}{\Longrightarrow}_S Q$ and $\forall j < i \ \alpha_j \neq \alpha_i$. In other words, $d_\sigma(P, \alpha, S)$ is the first position of the visible action α on path σ. Similarly, let $\alpha \notin H$ be an actual visible action and

$$\Sigma_H(\alpha) = \{\sigma = \tau^* \alpha_1 \tau^* \alpha_2 \cdots \tau^* \alpha_n \tau^* \alpha \mid n \in \mathbb{N} \text{ and } \forall i \leq n,\ \alpha_i \in H\}.$$

Then α is at maximal distance k w.r.t H from the process P according to the scheduler S, denoted by $d_H(P, \alpha, S) = k$, if $\sup_{\sigma \in \Sigma_H(\alpha)} d_\sigma(P, \alpha, S) = k$. To prove

[8] For verification purpose, all along this section, we consider only actual actions (i.e. actions in $\mathcal{A}ctual$) keeping in mind that partial actions are never executable. So far partial actions have been considered purely for the sake of semantic soundness and completeness.

the claim, we proceed by induction on $d_H(P, \alpha, S)$. Let $\alpha \in \mathcal{A}ctual \setminus (H \cup \{\tau\})$. We have:

[Basis] if $d_H(P, \alpha, S) = 0$ then

$$\mu(P, \stackrel{\hat{\alpha}}{\Longrightarrow}_{S/H}, \mathscr{E}) = \mu(P, \stackrel{\hat{\alpha}}{\Longrightarrow}_{S}, \mathscr{E}) \leq 1.$$

[Inductive step] Assume that the claim holds for all Q s.t. $d_H(Q, \alpha, S) < n$.

Since

$$\mu(P, \stackrel{\hat{\alpha}}{\Longrightarrow}_{S/H}, \mathscr{E}) = \mu(P, \stackrel{\hat{\alpha}}{\Longrightarrow}_{S}, \mathscr{E}) + \sum_{\beta \in H,\, Q \in \mathcal{P}roc} \mu(P, \stackrel{\hat{\beta}}{\Longrightarrow}_{S}, \{Q\}) \mu(Q, \stackrel{\hat{\alpha}}{\Longrightarrow}_{S/H}, \mathscr{E})$$

and by induction hypothesis, $\mu(Q, \stackrel{\hat{\alpha}}{\Longrightarrow}_{S/H}, \mathscr{E}) \leq 1$, it remains to show that

$$\mu(P, \stackrel{\hat{\alpha}}{\Longrightarrow}_{S}, \mathscr{E}) + \sum_{\beta \in H,\, Q \in \mathcal{P}roc} \mu(P, \stackrel{\hat{\beta}}{\Longrightarrow}_{S}, \{Q\}) \leq 1,$$

But we have

$$\mu(P, \stackrel{\hat{\alpha}}{\Longrightarrow}_{S}, \mathscr{E}) + \sum_{\beta \in H,\, Q \in \mathcal{P}roc} \mu(P, \stackrel{\hat{\beta}}{\Longrightarrow}_{S}, \{Q\}) \leq \sum_{\beta \in H \cup \{\alpha\}} \mu(P, \stackrel{\hat{\beta}}{\Longrightarrow}_{S}, \mathcal{P}roc)$$

But that sum is smaller than the sum of the probabilities of outgoing transitions from the state P according to scheduler S, which is less than or equal to 1 according to the Theorem 2.8. □

Definition 3.3 Let $o = (c, m)$ be an observable and $L_o = \{\overline{c}(m) \cdot c(m),\ c(m) \cdot \overline{c}(m)\}$. The cumulative probability that P generates o according to S is

$$\text{Prob}[P \rightsquigarrow_S o] = \sum_{\alpha \in L_o} \mu(P, \stackrel{\hat{\alpha}}{\Longrightarrow}_{S/(\mathcal{A}ctual \setminus (L_o \cup \{\tau\}))}, \mathcal{P}roc).$$

Lemma 3.4 *The cumulative probability of an observable is well defined, i.e.* $\forall P, o,$ *and* S, $\text{Prob}[P \rightsquigarrow_S o] \leq 1$.

Proof. The proof is given in Appendix B. □

We define our asymptotic observational equivalence relation as stating that two processes are equivalent if they generate the same observables with approximately the same probabilities when they are attacked by the same enemy.

Definition 3.5 Let $\mathcal{P}oly : \mathbb{N} \to \mathbb{R}^+$ be the set of positive polynomials and $\mathcal{E} = \mathcal{P}roc \times \mathcal{S}ched$ the set of attackers. Two processes P and Q are observationally equivalent, written $P \simeq Q$, iff $\forall_{q \in \mathcal{P}oly}$, $\forall_{o \in \mathcal{O}bs}$, $\forall_{(\Pi, S) \in \mathcal{E}}$, $\exists_{i_0}$ s.t. $\forall_{\mathbf{N} \geq i_0}$

$$|\text{Prob}[P|\Pi \rightsquigarrow_S o] - \text{Prob}[Q|\Pi \rightsquigarrow_S o]| \leq \frac{1}{q(\mathbf{N})}$$

Theorem 3.6 *The observational relation $\simeq$ is an equivalence relation.*

Proof. Reflexivity and symmetry are obvious. For transitivity, let P, Q and R be processes such that $P \simeq Q$ and $Q \simeq R$. Let q be a polynomial, o an observation and (Π, S) an attacker. Then $P \simeq Q \Rightarrow \exists i_0$ such that $\forall \mathbf{N} \geq i_0$

$$|\text{Prob}[P|\Pi \rightsquigarrow_S o] - \text{Prob}[Q|\Pi \rightsquigarrow_S o]| \leq \frac{1}{2q(\mathbf{N})},$$

and $Q \simeq R \Rightarrow \exists j_0$ such that $\forall \mathbf{N} \geq j_0$

$$|\text{Prob}[Q|\Pi \rightsquigarrow_S o] - \text{Prob}[R|\Pi \rightsquigarrow_S o]| \leq \frac{1}{2q(\mathbf{N})}.$$

It follows that $\forall \mathbf{N} \geq k_0 = \max(i_0, j_0)$ we have
$|\text{Prob}[P|\Pi \rightsquigarrow_S o] - \text{Prob}[R|\Pi \rightsquigarrow_S o]|$

$$\begin{aligned}
&= |\text{Prob}[P|\Pi \rightsquigarrow_S o] - \text{Prob}[Q|\Pi \rightsquigarrow_S o] \\
&\qquad + \text{Prob}[Q|\Pi \rightsquigarrow_S o] - \text{Prob}[R|\Pi \rightsquigarrow_S o]| \\
&\leq |\text{Prob}[P|\Pi \rightsquigarrow_S o] - \text{Prob}[Q|\Pi \rightsquigarrow_S o]| \\
&\qquad + |\text{Prob}[Q|\Pi \rightsquigarrow_S o] - \text{Prob}[R|\Pi \rightsquigarrow_S o]| \\
&\leq \frac{1}{2q(\mathbf{N})} + \frac{1}{2q(\mathbf{N})} = \frac{1}{q(\mathbf{N})}.
\end{aligned}$$

□

In order to develop methods for reasoning about security properties of cryptoprotocols based on observable traces, we reformulate the observational equivalence to take into account any observable trace.

3.2 Trace equivalence

We start by defining the cumulative probability that a process P generates a sequence of observables $o_1 o_2 \cdots o_n$ recursively as the probability that the process directly generates o_1 and reaches a state Q, times the cumulative probability that the process Q generates the remaining sequence.

Definition 3.7 Let $o_1 o_2 \cdots o_n$ be a sequence of observables s.t. $\forall i \leq n, o_i = (c_i, m_i)$ and P be a process. Let $\alpha_i = \overline{c_i}(m_i) \cdot c_i(m_i)$ and $\beta_i = c_i(m_i) \cdot \overline{c_i}(m_i)$ $\forall 1 \leq i \leq n$. The cumulative probability that P generates the sequence $o_1 o_2 \cdots o_n$ according to scheduler S is

$$\begin{aligned}
&\text{Prob}[\, P \rightsquigarrow_S^{tr} o_1 o_2 \cdots o_n] \\
&\qquad = \sum_{Q \in \mathcal{P}roc} (\mu(P, \stackrel{\hat{\alpha_1}}{\Longrightarrow}_S, \{Q\}) + \mu(P, \stackrel{\hat{\beta_1}}{\Longrightarrow}_S, \{Q\}))\text{Prob}[Q \rightsquigarrow_S^{tr} o_2 \cdots o_n].
\end{aligned}$$

Lemma 3.8 *The cumulative probability of observing a sequence of observables is well defined, i.e. $\forall P, o_1, o_2, \cdots, o_n$ and S,* $\text{Prob}[P \rightsquigarrow_S^{tr} o_1 o_2 \cdots o_n] \leq 1$.

Proof. It follows easily by induction on the length of sequences and from Lemma 3.2 and Theorem 2.8. □

Definition 3.9 Two processes P and Q are trace equivalent, as denoted by $P \simeq^{tr} Q$, iff $\forall_{q\in\mathcal{P}oly}$, $\forall_{o_1,o_2,\cdots,o_n\in\mathcal{O}bs}$, $\forall_{(\Pi,S)\in\mathcal{E}}$, $\exists_{i_0}$ s.t. $\forall_{\mathbf{N}\geq i_0}$

$$|\mathrm{Prob}[P|\Pi \rightsquigarrow_S^{tr} o_1o_2\cdots o_n] - \mathrm{Prob}[Q|\Pi \rightsquigarrow_S^{tr} o_1o_2\cdots o_n]| \leq \frac{1}{q(\mathbf{N})}$$

Theorem 3.10 *$\simeq^{tr}$ is an equivalence relation.*

Proof. Similar to the proof of Theorem 3.6. □

To conclude this section we have the following important result that relates the two equivalence relations.

Theorem 3.11 *Trace and observational equivalences are equivalent, i.e.*

$$\forall_{P,Q\in\mathcal{P}roc} P \simeq^{tr} Q \Leftrightarrow P \simeq Q.$$

Proof. The proof is given in Appendix B. □

4 Case study: the Dining Cryptographers protocol

The *Dining Cryptographers* [8] protocol is a paradigmatic example of a protocol which ensures anonymity. Its author defines it as follows:

> *Three cryptographers are sitting down to dinner at their favorite restaurant. Their waiter informs them that arrangements have been made with the* maitre d'hotel *for the bill to be paid anonymously. One of the cryptographers might be paying for the dinner, or it might have been NSA (U.S. National Security Agency). The three cryptographers respect each other's right to make an anonymous payment, but they wonder if NSA is paying.*

To fairly resolve their uncertainty, they carry out the following protocol: each cryptographer flips a coin between himself and the cryptographer on his right, so that only the two of them can see the outcome. Each one of them then states whether or not the two coins he can see fell on the same side. The payer (if any!) states the opposite of what he sees. The idea is that if the coins are unbiased and the protocol is carried out faithfully then an odd number of "different" indicates that one of them is paying and neither of the other two learns anything about his identity; otherwise NSA is paying.

4.1 A flawed specification of the protocol

In the following specification [9] we suppose that the NSA makes his choice according to a probabilistic distribution (known only to himself) defined by the function λ_{NSA} and informs each cryptographer over a secure channel if whether or not he is the payer. To ensure fairness between cryptographers, i.e. no one having advantage over another, each flip of a coin is made by an "outside trusted third party" by

[9] where $\oplus$ is the addition modulus 2.

means of the function $flips$ and the result is made available to both concerned cryptographers.

$$
\begin{array}{ll}
NSA ::= & [\lambda_{NSA}(3) \hookrightarrow x](\prod_{0\leq i\leq 3}[x=i]Payer_i) \\
Payer_3 ::= & \overline{c}_0(nopay).\overline{c}_1(nopay).\overline{c}_2(nopay).\mathbf{0} \\
Payer_i ::= & \overline{c}_i(pay).\overline{c}_{i\oplus 1}(nopay).\overline{c}_{i\oplus 2}(nopay).\mathbf{0} \text{ if } 0\leq i\leq 2. \\
Crypts ::= & [flips(coin_0)\hookrightarrow y_0][flips(coin_1)\hookrightarrow y_1][flips(coin_2)\hookrightarrow y_2]\prod_{0\leq i\leq 2} Crypt_i \\
Crypt_i ::= & c_i(z_i).([z_i=pay]P_i|[z_i=nopay]Q_i) \\
P_i ::= & [y_i=y_{i\oplus 1}]\overline{pub}_i(desagree).\mathbf{0}|[y_i\neq y_{i\oplus 1}]\overline{pub}_i(agree).\mathbf{0} \\
Q_i ::= & [y_i=y_{i\oplus 1}]\overline{pub}_i(agree).\mathbf{0}|[y_i\neq y_{i\oplus 1}]\overline{pub}_i(desagree).\mathbf{0}
\end{array}
$$

The protocol is then specified as follows: $DCP^1 ::= \nu c_0c_1c_2(NSA|Crypts)$

4.2 Specification of anonymity

We give a probabilistic version of the anonymity specification of [12] in the possibilistic model CSP. The idea is that given a set A and a set O, of anonymous and observable events respectively, a protocol P ensures the anonymity of events A to any observer who can see only events in O if P does not allow the observer to determine any causal dependency between the probabilistic distributions of A and O.

Let A and O be the sets of anonymous and observable events respectively. Perm(A) denotes the set of permutations of the elements of A and $\pi(P)$ the process obtained by replacing any occurrence of the event a in the process P by the event $\pi(a)$. We defined formally the anonymity property as follows.

Definition 4.1 [Anonymity property] A protocol P ensures anonymity of events A if and only if it is trace equivalent to any of its permutations (w.r.t anonymous events), i.e.

$$\forall_{\pi\in Perm(A)} P \simeq^{tr} \pi(P)$$

In the above specification the anonymous events of the DCP protocol are $A_{DCP} = \{(c_i, m) \mid 0\leq i\leq 2 \text{ and } m = pay,\ nopay\}$ and the observable events are any communication over a public channel. Let

$$Sched_\tau = \{S\in Sched \mid \sum_{(\tau,i,j)\in A, i,j\in I_1} \text{Prob}[S(A,I_1,I_2)=(\tau,i,j)] = 1\}$$

denote the subset of schedulers that give priority to internal actions of the protocol and $\simeq^{tr}_\tau$ the observational trace equivalence induced by $Sched_\tau$. Then we have the following results which show that $Sched$ can detect the flaw in DCP^1 while $Sched_\tau$ cannot.

Theorem 4.2 *With the notation above, the following conditions hold:*

- $\forall_{\pi\in Perm(A_{DCP})}$, *if* $\forall_{i=0,1,2}$ Prob$[flips(coin_i) = Head] = \frac{1}{2}$ *then*

$$DCP^1 \simeq^{tr}_\tau \pi(DCP^1.$$

- *Whatever the probabilistic distributions of the coins are, if π is not the identity permutation then*

$$DCP^1 \not\approx^{tr} \pi(DCP^1.$$

The flaw in DCP^1 results from the fact that the real payer (if any) has an advantage over the others since, because of the definition of the process $Payer_i$, he always receives the message from the NSA before them. A scheduler that gives priority to observable actions (i.e. an intruder who attacks the protocol as soon as possible), will only generate observable traces beginning with the public channel of the real payer i.e. with an observable (c, m) where c denotes the public channel of the real payer. Under such schedulers, DCP^1 and $\pi(DCP^1)$ will generate different observable traces and are hence not equivalent. A correct specification of the protocol (DCP^2) follows

$$\begin{array}{ll}
NSA ::= & [\lambda_{NSA}(3) \hookrightarrow x](\prod_{0\leq i\leq 3} [x = i]Payer_i) \\
Payer_3 ::= & \overline{c}_0(nopay).\overline{c}_1(nopay).\overline{c}_2(nopay).\mathbf{0} \\
Payer_i ::= & \overline{c}_i(pay).\overline{c}_{i\oplus 1}(nopay).\overline{c}_{i\oplus 2}(nopay).\mathbf{0} \text{ if } 0 \leq i \leq 2. \\
Crypts ::= & c_0(z_0).c_1(z_1).c_2(z_2).Flip \\
Flip ::= & [flips(coin_0) \hookrightarrow y_0][flips(coin_1) \hookrightarrow y_1][flips(coin_2) \hookrightarrow y_2](\prod_{0\leq i\leq 2} Crypt_i) \\
Crypt_i ::= & [z_i = pay]P_i|[z_i = nopay]Q_i \\
P_i ::= & [y_i = y_{i\oplus 1}]\overline{pub}_i(different).\mathbf{0}|[y_i \neq y_{i\oplus 1}]\overline{pub}_i(same).\mathbf{0} \\
Q_i ::= & [y_i = y_{i\oplus 1}]\overline{pub}_i(same).\mathbf{0}|[y_i \neq y_{i\oplus 1}]\overline{pub}_i(different).\mathbf{0}
\end{array}$$

The protocol is then specified as follows: $DCP^2 ::= \nu c_0 c_1 c_2(NSA|Crypts)$. It blocks the coins flipping until all cryptographers receive their message from the NSA.

Acknowledgement

The authors are grateful to the reviewers for their careful reading and helpful comments that improved the paper's readability.

References

[1] A. Aldini, M. Bravetti, and R. Gorrieri. A process algebra approach for the analysis of probabilistic non-interference. Technical report, 2002.

[2] M.J. Atallah. *Algorithms and Theory of Computation Handbook*. CRC Press LLC, 1999.

[3] J. Bengt, K.G Larson, and W. Yi. Probabilistic extension of process algebra. In *Handbook of process algebra*, page 565, 2002.

[4] Bruno Blanchet. A computationally sound mechanized prover for security protocols. In *S&P*, pages 140–154. IEEE Computer Society, 2006.

[5] Ran Canetti, Ling Cheung, Dilsun Kirli Kaynar, Moses Liskov, Nancy A. Lynch, Olivier Pereira, and Roberto Segala. Time-bounded task-pioas: A framework for analyzing security protocols. In Shlomi Dolev, editor, *DISC*, volume 4167 of *Lecture Notes in Computer Science*, pages 238–253. Springer, 2006.

[6] K. Chatzikokolakis and C. Palamidessi. A Framework for Analysing Probabilistic Protocols and its Applications to the Partial Secrets Exchange. In *Proc. of the Sym. on Trust. Glob. Comp. (STGC'05)*, LNCS. Spr.-Ver., 2005.

[7] K. Chatzikokolakis and C. Palamidessi. Making random choices invisible to the scheduler. In *Proc. of CONCUR'07). To appear.*, 2007.

[8] David Chaum. The dining cryptographers problem: Unconditional sender and recipient untraceability. *J. Cryptology*, 1(1):65–75, 1988.

[9] Flavio D. Garcia, Peter van Rossum, and Ana Sokolova. Probabilistic anonymity and admissible schedulers. *http://arxiv.org/abs/0706.1019*, 2007.

[10] Peeter Laud and Varmo Vene. A type system for computationally secure information flow. In Maciej Liskiewicz and Rüdiger Reischuk, editors, *FCT*, volume 3623 of *Lecture Notes in Comp. Sc.*, pages 365–377. Springer, 2005.

[11] J.C. Mitchell, A. Ramanathan, A. Scedrov, and V. Teague. A probabilistic polynomial-time process calculus for the analysis of cryptographic protocols. *Theoretical Computer Science*, 353:118–164, 2006.

[12] S. Schneider and A. Sidiropoulos. Csp and anonymity. In *Proc. Comp. Security - ESORICS 96*, volume 1146 of *LNCS*, pages 198–218. Springer-Vale, 1996.

A Probabilistic polynomial functions

The following definitions are standard: see for instance [2] (chapter 24, pp. 19-28).

Definition A.1 A probabilistic function F from X to Y is a function $X \times Y \rightarrow [0,1]$ that satisfies the following conditions.

- $\forall x \in X : \sum_{y \in Y} F(x,y) \leq 1$
- $\forall x \in X$, the set $\{y | y \in Y, F(x,y) > 0\}$ is finite.

For $x \in X$ and $y \in Y$, we say $F(x)$ evaluates to y with probability p, written $\text{Prob}[F(x) = y] = p$, if $F(x,y) = p$.

Definition A.2 The composition $F = F_1 \circ F_2 : X \times Z \rightarrow [0,1]$ of two probabilistic functions $F_1 : X \times Y \rightarrow [0,1]$ and $F_2 : Y \times Z \rightarrow [0,1]$ is the probabilistic function:

$$\forall x \in X, \forall z \in Z : F(x,z) = \sum_{y \in Y} F_1(x,y) \cdot F_2(y,z).$$

Definition A.3 An oracle Turing machine is a Turing machine with an extra oracle tape and three extra states q_{query}, q_{yes} and q_{no}. When the machine reaches the state q_{query}, control is passed either to the state q_{yes} if the contents of the oracle tape belongs to the oracle set, or to the state q_{no} otherwise.

Given an oracle Turing machine M, $M_\sigma(\overrightarrow{a})$ stands for the result of the application of M to $\overrightarrow{a}$ by using the oracle σ.

Definition A.4 An oracle Turing machine executes in polynomial time if there exists a polynomial $q(\overrightarrow{x})$ such that for all σ, $M_\sigma(\overrightarrow{a})$ halts in time $q(|\overrightarrow{a}|)$, where $\overrightarrow{a} = (a_1, \cdots, a_k)$ and $|\overrightarrow{a}| = |a_1| + \cdots + |a_k|$.

Let M be an oracle Turing machine with execution time bounded by the polynomial $q(\overrightarrow{a})$. Since $M(\overrightarrow{a})$ may call an oracle with at most $q(\overrightarrow{a})$ bits, we have a finite set $\mathcal{Q}$ of oracles for which M executes in time that is bounded by $q(\overrightarrow{a})$.

Definition A.5 We say that an oracle Turing machine is *probabilistic polynomial* and write $\text{Prob}[M(\overrightarrow{a}) = b] = p$ the probability that M applied to $\overrightarrow{a}$ returns b is p, if and only if, by choosing uniformly an oracle σ in the finite set $\mathcal{Q}$, the probability that $M_\sigma(\overrightarrow{a}) = b$ is p.

Definition A.6 A probabilistic function F is said to be *polynomial* if it is computable by a probabilistic polynomial Turing machine, that is, for all input $\overrightarrow{a}$ and all output b, $\text{Prob}[F(\overrightarrow{a}) = b] = \text{Prob}[M(\overrightarrow{a}) = b]$.

B Proofs

Proof. [Lemma 3.4] Let $o = (c,m)$ be an observable, $L_o = \{\overline{c}(m) \cdot c(m), c(m) \cdot \overline{c}(m)\}$, S a scheduler and P a process. Let H be the set $H = \mathcal{A}ctual \setminus (L_o \cup \{\tau\})$. To prove the lemma, we proceed by induction on $d_H(P, \alpha, S)$ defined in the above proof of Lemma 3.2.

- *[Basis:]* If $\forall_{\alpha\in L_o} d_H(P,\alpha,S)=0$ then

$$\begin{aligned}\mathrm{Prob}[P\rightsquigarrow_S o] &= \sum_{\alpha\in L_o}\mu(P,\stackrel{\hat{\alpha}}{\Longrightarrow}_{S/H},\mathcal{P}roc) = \sum_{\alpha\in L_o}\mu(P,\stackrel{\hat{\alpha}}{\Longrightarrow}_{S},\mathcal{P}roc)\\ &\le \sum_{\beta\in Act}\mu(P,\stackrel{\hat{\beta}}{\Longrightarrow}_{S},\mathcal{P}roc)\\ &\le 1 \qquad \text{(according to Theorem 2.8.)}\end{aligned}$$

- *[Induction step:]* Suppose that for any process Q such that $\forall_{\alpha\in L_o} d_H(Q,\alpha,S)<n$, $\mathrm{Prob}[Q\rightsquigarrow_S o]\le 1$. Then we have

$$\begin{aligned}\mathrm{Prob}[P\rightsquigarrow_S o] &= \sum_{\alpha\in L_o}\mu(P,\stackrel{\hat{\alpha}}{\Longrightarrow}_{S/H},\mathcal{P}roc)\\ &= \sum_{\alpha\in L_o}\mu(P,\stackrel{\hat{\alpha}}{\Longrightarrow}_{S},\mathcal{P}roc) +\\ &\quad \sum_{\alpha\in L_o}\sum_{\beta\in H,\ Q\in\mathcal{P}roc}\mu(P,\stackrel{\hat{\beta}}{\Longrightarrow}_{S},\{Q\})\mu(Q,\stackrel{\hat{\alpha}}{\Longrightarrow}_{S/H},\mathscr{E})\\ &= \sum_{\alpha\in L_o}\mu(P,\stackrel{\hat{\alpha}}{\Longrightarrow}_{S},\mathcal{P}roc) +\\ &\quad \Big(\sum_{\beta\in H,\ Q\in\mathcal{P}roc}\mu(P,\stackrel{\hat{\beta}}{\Longrightarrow}_{S},\{Q\})\Big)\mathrm{Prob}[Q\rightsquigarrow_S o]\\ &\le \sum_{\alpha\in L_o}\mu(P,\stackrel{\hat{\alpha}}{\Longrightarrow}_{S},\mathcal{P}roc) +\\ &\quad \sum_{\beta\in H,\ Q\in\mathcal{P}roc}\mu(P,\stackrel{\hat{\beta}}{\Longrightarrow}_{S},\{Q\}) \quad \text{(induction hypotheses)}\\ &\le \sum_{\beta\in Act}\mu(P,\stackrel{\hat{\beta}}{\Longrightarrow}_{S},\mathcal{P}roc)\\ &\le 1 \qquad \text{(according to Theorem 2.8.)}\end{aligned}$$

□

Proof. [Theorem 3.11] ($\Rightarrow$). Let $H\subset \mathcal{A}ctual\setminus\{\tau\}$ be a set of visible actions, $o=(c,m)$ an observable s.t. $\{\overline{c}(m)\cdot c(m), c(m)\cdot\overline{c}(m)\}\cap H=\emptyset$, S a scheduler and R a process. Then $d_H(R,o,S)$ denotes

$$d_H(R,o,S)=\max(d_H(R,\overline{c}(m)\cdot c(m),S), d_H(R,c(m)\cdot\overline{c}(m),S)).$$

Now let O be a set of observables s.t. $o\notin O$, and $Tr_k^O(R,o,S)$ the set

$$Tr_k^O(P,o,S)=\{s=o_1o_2\cdots o_ko \mid \forall i\le k, o_i\in O \text{ and } \mathrm{Prob}[R\rightsquigarrow_S^{tr} s]\neq 0\}.$$

Then for any observable o, we have

$$\mathrm{Prob}[R\rightsquigarrow_S o]=\sum_{k\ge 0}\sum_{s\in Tr_k^{\mathcal{O}bs\setminus\{o\}}(R,o,S)}\mathrm{Prob}[R\rightsquigarrow_S^{tr} s].$$

Let P and Q be two processes s.t. $P \simeq^{tr} Q$, $o = (c, m) \in \mathcal{O}bs$, $(\Pi, S) \in \mathcal{E}$, $q \in Poly$ and

$$H = \{\alpha \mid \exists (c', m') \in \mathcal{O}bs \setminus \{o\} \text{ and } \alpha \in \{\overline{c'}(m') \cdot c'(m'), c'(m') \cdot \overline{c'}(m')\}\}.$$

To prove the theorem, we proceed by induction on

$$\max(d_H(P|\Pi, o, S), d_H(Q|\Pi, o, S)).$$

[Basis:] If $d_H(P|\Pi, o, S) = d_H(Q|\Pi, o, S) = 0$, then we have:

$$\text{Prob}[P|\Pi \rightsquigarrow_S o] = \mu(P|\Pi, \stackrel{\hat{\alpha}}{\Longrightarrow}_S, \mathcal{P}roc) + \mu(P|\Pi, \stackrel{\hat{\beta}}{\Longrightarrow}_S, \mathcal{P}roc)$$

where $\alpha = \overline{c}(m) \cdot c(m)$ and $\beta = c(m) \cdot \overline{c}(m)$. Similarly

$$\text{Prob}[Q|\Pi \rightsquigarrow_S o] = \mu(Q|\Pi, \stackrel{\hat{\alpha}}{\Longrightarrow}_S, \mathcal{P}roc) + \mu(Q|\Pi, \stackrel{\hat{\beta}}{\Longrightarrow}_S, \mathcal{P}roc).$$

$P \simeq^{tr} Q \Rightarrow$

$$\begin{aligned}|&\mu(P|\Pi, \stackrel{\hat{\alpha}}{\Longrightarrow}_S, \mathcal{P}roc) + \mu(P|\Pi, \stackrel{\hat{\beta}}{\Longrightarrow}_S, \mathcal{P}roc) \\ &\qquad - \mu(Q|\Pi, \stackrel{\hat{\alpha}}{\Longrightarrow}_S, \mathcal{P}roc) + \mu(Q|\Pi, \stackrel{\hat{\beta}}{\Longrightarrow}_S, \mathcal{P}roc)| \leq \frac{1}{q(\mathbf{N})}\end{aligned}$$

Hence

$$|\text{Prob}[P|\Pi \rightsquigarrow_S o] - \text{Prob}[Q|\Pi \rightsquigarrow_S o]| \leq \frac{1}{q(\mathbf{N})}$$

[Induction step:] assume that $\max(d_H(P|\Pi, o, S), d_H(Q|\Pi, o, S) = n$, and $\forall_{q \in Poly}$ we have

$$\sum_{k<n} \sum_{s \in Tr_k^{\mathcal{O}bs \setminus \{o\}}(R,o,S)} |\text{Prob}[P|\Pi \rightsquigarrow_S^{tr} s] - \text{Prob}[Q|\Pi \rightsquigarrow_S^{tr} s]| \leq \frac{1}{2q(\mathbf{N})}.$$

Now set $U_k = Tr_k^H(P|\Pi, o, S) \cup Tr_k^H(Q|\Pi, o, S)$, then we have

$$\begin{aligned}|&\text{Prob}[P|\Pi \rightsquigarrow_S o] - \text{Prob}[Q|\Pi \rightsquigarrow_S o]| \\ &= |\sum_{k \leq n} \sum_{s \in U_k} (\text{Prob}[P|\Pi \rightsquigarrow_S^{tr} s] - \text{Prob}[Q|\Pi \rightsquigarrow_S^{tr} s])| \\ &\leq |\sum_{k<n} \sum_{s \in U_k} \text{Prob}[P|\Pi \rightsquigarrow_S^{tr} s] - \text{Prob}[Q|\Pi \rightsquigarrow_S^{tr} s]| \\ &\quad + \sum_{s \in U_n} |\text{Prob}[P|\Pi \rightsquigarrow_S^{tr} s] - \text{Prob}[Q|\Pi \rightsquigarrow_S^{tr} s]|\end{aligned}$$

By induction hypothesis we have

$$|\sum_{k<n} \sum_{s \in U_k} \text{Prob}[P|\Pi \rightsquigarrow_S^{tr} s] - \text{Prob}[Q|\Pi \rightsquigarrow_S^{tr} s]| \leq \frac{1}{2q(\mathbf{N})}.$$

Set $r_n = |Tr_n^H(P|\Pi, o, S)| + |Tr_n^H(Q|\Pi, o, S)|$. $P \simeq^{tr} Q \Rightarrow \forall s \in U_n$

$$|\text{Prob}[P|\Pi \rightsquigarrow_S^{tr} s] - \text{Prob}[Q|\Pi \rightsquigarrow_S^{tr} s]| \leq \frac{1}{r_n \cdot 2q(\mathbf{N})},$$

then

$$\sum_{s \in U_n} |\text{Prob}[P|\Pi \rightsquigarrow_S^{tr} s] - \text{Prob}[Q|\Pi \rightsquigarrow_S^{tr} s]| \leq \frac{r_n}{r_n \cdot 2q(\mathbf{N})} = \frac{1}{2q(\mathbf{N})}.$$

Hence we have

$$\begin{aligned} |\text{Prob}[P|\Pi \rightsquigarrow_S o] - \text{Prob}[P|\Pi \rightsquigarrow_S o]| &\leq \frac{1}{2q(\mathbf{N})} + \frac{1}{2q(\mathbf{N})} \\ &\leq \frac{1}{q(\mathbf{N})}. \end{aligned}$$

($\Leftarrow$). Suppose now that $P \simeq Q$ but $P \not\simeq^{tr} Q$. Then we will show that we reach a contradiction. Indeed, if $P \not\simeq^{tr} Q$, then there exists an attacker (Π, S), an observable trace $o_1 o_2 \cdots o_n$ and a polynomial q s.t.

$$\forall_{\mathbf{N}} \; |\text{Prob}[P|\Pi \rightsquigarrow_S^{tr} o_1 o_2 \cdots o_n] - \text{Prob}[Q|\Pi \rightsquigarrow_S^{tr} o_1 o_2 \cdots o_n]| > \frac{1}{q(\mathbf{N})}.$$

Let $o' = (c', m)$ s.t. c' is neither a channel of P or Q nor a channel of Π and let (Π', S') be the attacker who behaves exactly like (Π, S) exept that anytime he observes the trace $o_1 o_2 \cdots o_n$ he makes the observable o' with probability 1. Then we have

$$\begin{aligned} &|\text{Prob}[\, P|\Pi' \rightsquigarrow'_S o'] - \text{Prob}[Q|\Pi' \rightsquigarrow'_S o']| \\ &\qquad = \; |\text{Prob}[P|\Pi \rightsquigarrow_S^{tr} o_1 o_2 \cdots o_n o'] - \text{Prob}[Q|\Pi \rightsquigarrow_S^{tr} o_1 o_2 \cdots o_n o']| \end{aligned}$$

Hence $P \not\simeq Q$ which contradicts our hypothesis. □