Modelling Dynamic Opacity using Petri Nets with silent actions

Jeremy W. Bryans, Maciej Koutny and Peter Y.A. Ryan

School of Computing Science, University of Newcastle,
Newcastle upon Tyne, NE1 7RU, U.K.

Abstract. In a previous work, [1], we presented a Petri Net based framework in which various confidentiality properties may be expressed in terms of predicates over system state and abstraction mappings from the reachable states and transitions of the underlying Petri Net. Here we extend that work by generalising these mappings by allowing them to be state dependent. This provides a natural framework in which to model various situations of importance in security, for example key compromise and refresh, downgrading of secrecy labels and conditional anonymity. We also show how global changes in the abstraction mappings can be used to model how some secrecy requirements depend on the status of the observer. We illustrate this by modelling the various flavours of anonymity that arise in the dining cryptographers example.
A further development on the earlier work is to provide a more complete treatment of silent actions. We also discuss the expressiveness of the resulting framework and the decidability of the associated verification problems.
Keywords: opacity, non-deducibility, anonymity, Petri nets, observable behaviour, silent actions.

1 Introduction

The notion of opacity with respect to a given system predicate, see for example [9], formalises the idea that an observer of a system may never be able to establish the truth of that predicate. As such it appears to be very general and flexible and to allow a wide class of well-established notions of secrecy to be captured.

In a previous paper, [1], we presented a Petri Net based framework in which various opacity properties may be expressed in terms of predicates over system state and abstraction mappings from the reachable states and transitions of an underlying Petri Net. in this paper we extend this earlier work in a number of respects:

- Rather than just considering a static abstraction mapping we allow the transition labels to depend on the the markings, i.e., the system state.
- We give a more satisfactory treatment of silent actions.
- We introduce a further Petri Net based notion of opacity, namely *total opacity*.

- We give a more extensive discussion of the notion of anonymity in this framework. In particular we illustrate, using the full dining cryptographers example, how flavours of anonymity may change with the observer viewpoint, i.e., with the abstraction mapping.
- The decidability results of [1] are extended to this richer model.

Our earlier framework allowed us, via the notion of opacity, to capture a number of situations of importance in security but that sit awkwardly with the more familiar information flow concepts such as non-interference. These include, for example, anonymity and encrypted channels in which there is inevitably some partial information flow. The extended framework presented here allows us to go further and capture situations in which the information flow may vary with the state of the system. Note that we can include the state of an adversary in our system model. Thus we can now model key compromise or refresh as well as classification downgrades.

The new form of opacity, *total opacity*, that we introduce here further allows us to capture the notion of *non-inference*.

Using the framework of Petri nets gives us access to a raft of existing results and tools that have been developed in the Petri net community.

2 Petri nets

In this section, we introduce Petri nets with weighted arcs [11], and give their operational semantics in terms of step sequences[1].

A (weighted) *net* is a triple $N = (P, T, W)$ such that P and T are disjoint finite sets, and $W : (T \times P) \cup (P \times T) \to \mathbb{N}$. The elements of P and T are respectively the *places* and *transitions*, and W is the *weight function* of N. In diagrams, places are drawn as circles, and transitions as rectangles. If $W(x, y) \geq 1$ for some $(x, y) \in (T \times P) \cup (P \times T)$, then (x, y) is an *arc* leading from x to y. As usual, arcs are annotated with their weight if this is 2 or more. We assume that, for every $t \in T$, there is a place p such that $W(p, t) \geq 1$.

The *pre-* and *post-multiset* of a transition $t \in T$ are multisets of places, $\mathrm{PRE}_N(t)$ and $\mathrm{POST}_N(t)$, respectively given by

$$\mathrm{PRE}_N(t)(p) = W(p, t) \quad \text{and} \quad \mathrm{POST}_N(t)(p) = W(t, p),$$

for all $p \in P$. Both notations extend to finite multisets of transitions U:

$$\mathrm{PRE}_N(U) = \sum_{t \in U} U(t) \cdot \mathrm{PRE}_N(t) \qquad \text{and} \qquad \mathrm{POST}_N(U) = \sum_{t \in U} U(t) \cdot \mathrm{POST}_N(t)\ .$$

A *marking* of a net N is a multiset of places. Following the standard terminology, given a marking M of N and a place $p \in P$, we say that p is marked if

[1] We have already demonstrated in [1] that step and interleaving semantics lead to different notions of opacity (even without the τ labels we introduce later).

$M(p) \geq 1$ and that $M(p)$ is the number of tokens in p. In diagrams, M will be represented by drawing in each place p exactly $M(p)$ tokens (black dots).

Transitions represent actions which may occur at a given marking and then lead to a new marking. Here we define this dynamics in terms of multisets of (simultaneously occurring) transitions.

A *step* is a non-empty finite multiset of transitions, $U : T \to \mathbb{N}$. It is *enabled* at a marking M if $M \geq \mathrm{PRE}_N(U)$. Thus, in order for U to be enabled at M, for each place p, the number of tokens in p under M should at least be equal to the total number of tokens that are needed as an input to U, respecting the weights of the input arcs.

If U is enabled at M, then it can be *executed* leading to the marking $M' = M - \mathrm{PRE}_N(U) + \mathrm{POST}_N(U)$. This means that the execution of U 'consumes' from each place p exactly $W(p,t)$ tokens for each occurrence of a transition $t \in U$ that has p as an input place, and 'produces' in each place p exactly $W(t,p)$ tokens for each occurrence of a transition $t \in U$ with p as an output place. If the execution of U leads from M to M' we write $M[U\rangle M'$.

An *execution* from a marking M to a marking M' is a sequence

$$\mu = MU_1M_1 \ldots M_{n-1}U_nM'$$

such that

$$M\,[U_1\rangle\, M_1\, \cdots\, M_{n-1}\,[U_n\rangle\, M' \ .$$

We also say that M' is *reachable* from M.

3 Observing Petri net behaviour

In this section, we introduce a specific device aimed at modelling various observation capabilities based on the executed behaviours of a Petri net. Our framework is deliberately general to allow one to deal with a wider range of observation scenarios. We also extend the previous scheme, by allowing even greater discriminating power on the observer's side.

We start by making a small (but important from the point of view of applications) adjustment of the standard notion of a marked net, by assuming that the system specification we are given at the outset is a pair $\Sigma = (N, \mathcal{M}_0)$, where N is a net as defined in the previous section and $\mathcal{M}_0$ is a non-empty finite set of initial markings. This allows us to easily model situations where only partial information of the initial state of the system is available to an observer.

We will denote by $[\mathcal{M}_0\rangle$ the set of all markings reachable from any of the markings in $\mathcal{M}_0$, and by $RG(\Sigma)$ the reachability graph of Σ defined as the labelled directed graph whose nodes are the markings in $[\mathcal{M}_0\rangle$, and the labelled arcs represent all steps executed at these markings according to the rules from the previous section We will denote by $RG_{steps}(\Sigma)$ the set of all the steps labelling the arcs of $RG(\Sigma)$.

3.1 Visibility of reachable markings and executed steps

In our approach, we assume that there is a mapping *obs* which for each reachable marking in $[\mathcal{M}_0\rangle$ returns some label $obs(M)$ which is meant to capture the observable or visible aspects of system's global states. We further assume that the mapping is defined for steps of executed transitions; more precisely, for each reachable marking M and a step of transitions U enabled by M, $obs(M,U)$ issome label which is meant to capture the observable or visible aspects of executing step U at the global state M.

We do not place any restrictions on the nature of the *obs* mapping at this point; indeed, it is left under-specified deliberately to accommodate a wide range of observation scenarios. We only assume that markings and steps are visible through different sets of labels (i.e., $obs(M) \neq obs(M',U)$, for all $M, M' \in [\mathcal{M}_0\rangle$ and U enabled at M').

We employ a special label τ which is returned as the value of $obs(M,U)$ in cases when U is a step invisible to the observer in the system state M.

Notice that, unlike in [1], we do not define the mapping *obs* for steps, i.e., $obs(U)$ is not assumed to be given (clearly, if $obs(M,U)$ returns the same label ℓ for all markings M enabling U, then we can define $obs(U) = \ell$ and we have exactly the setup from [1]). The motivation for this is that we envisage application when the observability of executed transitions would depend on the current state of the system (for example, after breaking one of the cryptographic keys used by a system under attack, the adversary would typically be able to deduce more from the observed message exchange).

Suitable choices of *obs* mapping can be used to encode the various levels of visibility of system behaviour that we attribute to the environment or adversary. Thus transitions visible only to a secret user might be mapped to a τ label. Such events would be completely invisible to the environment, i.e., the environment would not be aware that any transition had occurred (and, if all the transitions in an executed step are invisible, then the whole step is mapped to τ). Transitions corresponding to the transmission of encrypted values could be mapped to a single label. Transitions deemed visible to the adversary may be left unchanged by the *obs* mapping.

Note that, in particular, *obs* allows us to 'detect' properties like deadlock-freeness or acceptance sets. The theory is rich enough to incorporate and reason about them. It is another matter, of course, how deadlocks would be detected or observed in the real life system, but these issues are beyond the scope of the current paper.

Having defined the observable aspects of individual markings and steps of transitions, we can define the effect of the observation mapping on the executions of the marked net Σ.

Let $\mu = M_0 U_1 M_1 \ldots M_{n-1} U_n M_n$ be an execution from a marking $M_0 \in \mathcal{M}_0$. We first introduce two auxiliary notations:

- $obs'(\mu) = \ell_0 \ell'_1 \ell_1 \ldots \ell'_n \ell_n$ is the sequence obtained from μ by replacing each M_i by $\ell_i = obs(M_i)$ and each U_i by $\ell'_i = obs(M_{i-1}, U_i)$.

– $obs''(\mu)$ is obtained from $obs'(\mu)$ by replacing each maximal subsequence $\ell_i \ell'_{i+1} \ell_{i+1} \ldots \ell'_j \ell_j$ such that

$$i \leq j \quad \text{and} \quad \ell_i = \cdots = \ell_j \quad \text{and} \quad \ell'_{i+1} = \cdots = \ell'_j = \tau\,,$$

by ℓ_i^S, where $S = \{M_i, M_{i+1}, \ldots, M_j\}$.

obs'' collapses sequences of the same (observable) states interspersed with τ's into a single state, since this is what will be observable to the user.

Suppose now that $obs''(\mu) = \hat{\ell}_1^{S_1} \hat{\ell}'_1 \hat{\ell}_2^{S_2} \ldots \hat{\ell}_m^{S_m}$. Then the *observation* of μ is given by $obs(\mu) = \hat{\ell}_1 \hat{\ell}'_1 \hat{\ell}_2 \ldots \hat{\ell}_m$. Moreover, for each $i \leq m$, $obs^i(\mu) = S_i$ and

$$\begin{aligned} obs^{init}(\mu) &= S_1 \\ obs^{fin}(\mu) &= S_m \\ obs^{all}(\mu) &= S_1 \cup \ldots \cup S_m\,. \end{aligned}$$

For example, if $\mu = M_0 U_1 M_1 U_2 M_2 U_3 M_3$ is such that $obs'(\mu) = z\tau z b w \tau y$ then $obs(\mu) = zbw\tau y$, $obs^{init}(\mu) = \{M_0, M_1\}$, $obs^{fin}(\mu) = \{M_3\}$ and $obs^{all}(\mu) = \{M_0, M_1, M_2, M_3\}$.

Note that one could have deleted from $obs(\mu)$ all the remaining τ's without changing any of the subsequent results, but this would have led to a more complicated definitions and constructions in proofs.

Examples The two basic forms of defining the *obs* mapping are *transition labelling* and *marking projection*. In the first case, we assume that each transition t has its own (not necessarily unique) label $\ell(t)$ and then the visibility of a step $U = \{t_1, \ldots, t_k\}$ is defined as the multiset

$$\ell(U) = \begin{cases} \tau & \text{if } \ell(t_1) = \cdots = \ell(t_k) = \tau \\ \{\ell(t_i) \mid \ell(t_i) \neq \tau\} & \text{otherwise}\,. \end{cases}$$

In the case of marking projection, we assume that $Vis \subseteq P$ is a set of places on which we can always see the tokens, and all places in $P \setminus Vis$ are hidden from us (in the extreme case, $Vis = \varnothing$ which effectively means that no information about the tokens is available). Then, for every marking M, we define $M|_{Vis}$ as a multiset over Vis such that $M|_{Vis}(p) = M(p)$ for every place $p \in Vis$.

Dynamic observation functions can be encoded using these projections. We simply include information that the observer has as part of the net. For example, in Figure 1, the net on the left represents the transmitting of messages encrypted using a key k. The net on the right represents the observers state of knowledge of the encryption key. Before the inverse key (k^{-1}) is known is represented by state S_2 and after the inverse key is known by state S_3. The key may be refreshed (modelled by the *ref* transition); this moves the observer back to the initial state. We include the possibilty that the *ref* transition may also occur before the key is compromised.

The (relevant part of the) *obs* mapping can then be defined as

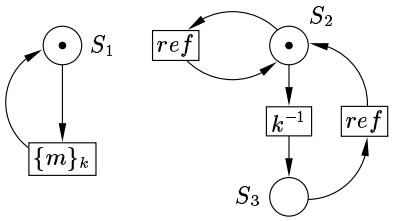

Fig. 1. An encoding of a dynamic observation mapping.

$$\begin{aligned} obs(M) &= M \\ obs(\{S_1, S_2\}, \{m\}_k) &= \{m\}_k \\ obs(\{S_1, S_3\}, \{m\}_k) &= m \end{aligned}$$

and so when the observer has the inverse key he can read any messages which pass on this channel.

Of course, once the observer has the inverse key he can go back to any previously read messages which he has saved, and decode them offline. It is possible that the observation function could also take this *post*-hoc analytic ability into consideration, but we do not pursue that idea further in this paper.

The same construction also seems suitable to model the notion of a *downgrader*: Once the appropriate downgrade action takes place an uncleared user may observe the messages or read the files which have been downgraded.

Figure 2 is an example of using the τ transitions to model information flow. Consider two system users, *high* and *low*. *high* is able to execute either of two large processes, initiating them with the action exe_1 or exe_2 as appropriate. *low* is using the same system, but is a lower-priority user. His *work* request will be disallowed if *high* is executing one or other of his processes.

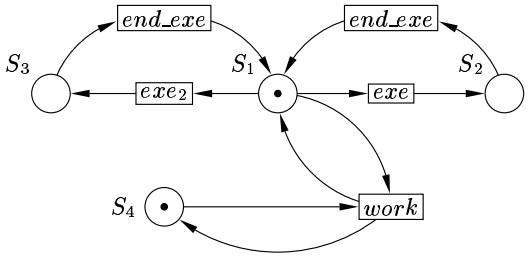

Fig. 2. An encoding of an information flow.

We can consider the exe_1 and exe_2 transitions as silent or τ transitions with respect to the low user. If the *high* user is in state S_2 or S_3 then when *low* attempts to *work* he will be unable to perform the transition. This will allow him to deduce that either S_2 or S_3 is occupied, but not which one.

The (relevant part of) the *obs* mapping would then be defined as

$$\begin{aligned} obs(M) &= M|_{S_4} \\ obs(\{S_1, S_4\}, \{work\}) &= work \\ obs(\{S_2, S_4\}, \{exe\}) &= \tau \\ obs(\{S_3, S_4\}, \{exe\}) &= \tau \\ obs(\{S_2, S_4\}, \{end\_exe\}) &= \tau \\ obs(\{S_3, S_4\}, \{end\_exe\}) &= \tau \end{aligned}$$

Although *low* cannot see the state of *high* directly, some information can be deduced by the willingness of the system to respond to his own *work* requests. Either *high* is in state S_1 and the work request will be granted, or *high* is in one of states S_2 and S_3, and the *work* request will not be granted.

In fact, this approach appears somewhat linked to the CSP refusals semantic model [8]. In that, the semantics of the system is taken to be the observed traces of events, together with the sets of events that may be *refused* to an observer. Here, the semantics of the systems (as observed by *low*) is taken as a trace of states and transitions (events) but some refusal information is captured by our treatment of silent actions.

Exploring and formalising such links to other semantic models is not pursued here, but will be considered in a forthcoming paper.

3.2 Opacity

In the present framework, we are interested in whether an observer can establish a property $\mathcal{P}$ at some specific state(s) of the execution of the system solely on the basis of its visible version. We consider here any state property, i.e., one which can be evaluated at any reachable marking in $[\mathcal{M}_0\rangle$. Clearly, any such property can simply be represented as the set of those reachable markings where it holds, and so we will take $\mathcal{P}$ to be any subset of $[\mathcal{M}_0\rangle$.

Now, given an observed execution of the system, we will be interested in finding out whether the fact that an underlying marking belongs to $\mathcal{P}$ can be deduced by the observer. Note, however, that we are not interested in establishing whether the underlying marking does not belong to $\mathcal{P}$. To do this, we would rather consider the property $\overline{\mathcal{P}} = [\mathcal{M}_0\rangle \setminus \mathcal{P}$.

What it means to deduce a property can mean different things depending on what is relevant or important from the point of view of real application. Below, we formalise four possible ways of defining variants of opacity. The first two

properties can be used to capture the fact that we are only interested in the holding of our property in the observed initial or final state, respectively.

- $\mathcal{P}$ is *initial-opaque* if for every execution μ from any marking in $\mathcal{M}_0$, if $obs^{init}(\mu) \cap \mathcal{P} \neq \varnothing$, then there exists an execution μ' from a marking in $\mathcal{M}_0$ such that $obs(\mu) = obs(\mu')$ and $obs^{init}(\mu') \cap \mathcal{P} = \varnothing$.
- $\mathcal{P}$ is *final-opaque* if for every execution μ from any marking in $\mathcal{M}_0$, if $obs^{fin}(\mu) \cap \mathcal{P} \neq \varnothing$, then there exists an execution μ' from a marking in $\mathcal{M}_0$ such that $obs(\mu) = obs(\mu')$ and $obs^{fin}(\mu') \cap \mathcal{P} = \varnothing$.

The next property reflects a view that we are interested in the holding of our property at all the specific observed states of the execution.

- $\mathcal{P}$ is *always-opaque* if for every execution μ from any marking in $\mathcal{M}_0$, if $obs^{i}(\mu) \cap \mathcal{P} \neq \varnothing$ for some i, then there exists an execution μ' from a marking in $\mathcal{M}_0$ such that $obs(\mu) = obs(\mu')$ and $obs^{i}(\mu') \cap \mathcal{P} = \varnothing$.

The last property capture the situation that the holding of our property can never be established for sure.

- $\mathcal{P}$ is *total-opaque* if for every execution μ from any marking in $\mathcal{M}_0$ such that $obs^{all}(\mu) \cap \mathcal{P} \neq \varnothing$ there exists an execution μ' from a marking in $\mathcal{M}_0$ such that $obs(\mu) = obs(\mu')$ and $obs^{all}(\mu') \cap \mathcal{P} = \varnothing$.

initial-opacity is illustrated by the dining cryptographers example. It would appear that *initial-opacity* is suited to modelling situations in which initialisation information such as crypto keys, etc., needs to be kept secret. More generally, situations in which confidential information can be modelled in terms of initially resolved non-determinism can be captured in this way. *always-opacity* would seem more appropriate to capture situation in which secret information is input at run time, for example due to high level interactions.

The distinction between *total-opacity* and *always-opacity* can be illustrated in the two diagrams below.

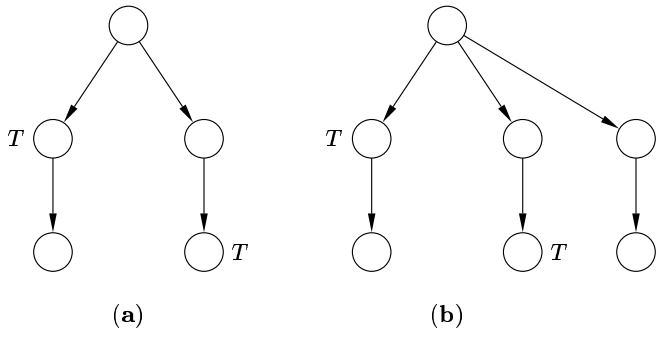

Fig. 3. The difference between total-opacity and final-opacity.

Assume that each of the paths through the nets in figure 3 gives rise to the same observation. In figure 3(a), a property which holds at each of the states marked T will be *always-opaque*, because at each of the states where it holds there is a corresponding execution for which the property does not hold *in the corresponding state*. But it is not *total-opaque*, because by the end of whichever execution we observe, we can say that the property has held at some point. For *total-opacity* we would require the extra execution of figure 3(b). Now, at the end of the execution we do not know if the property was ever true.

Informally, the notion of *non-inference*, [10], captures the idea that an observer should never be able to eliminate the possibility that the High user did nothing. More formally:

$$\forall t \in traces(S) \;\; \exists t' \in traces(S) \bullet \\ t \downarrow Low = t' \downarrow Low \quad \wedge \quad t' \downarrow High = \langle\rangle$$

We will be exploring the application of these formal properties in a future paper.

Proposition 1. *If $\mathcal{P}$ is total-opaque then it is initial-opaque, final-opaque and always-opaque. Moreover, if $\mathcal{P}$ is always-opaque then it is initial-opaque and final-opaque. No other implication of this kind in general holds.*

Proof. The first two parts follows from the definitions, and in [1] we have shown that neither initial-opacity nor final-opacity implies always-opacity. Hence, all we need is that in general always-opacity does not imply total-opacity, which can easily be shown using a suitable counterexample.

Proposition 2. *For $x \in \{initial, final, always, total\}$, it is the case that $\mathcal{P} = \varnothing$ is x-opaque, $\mathcal{P} = [\mathcal{M}_0\rangle$ is not x-opaque, and if $\mathcal{P} \subseteq \mathcal{P}'$ and $\mathcal{P}'$ is x-opaque then $\mathcal{P}$ is x-opaque.*

Proof. Follows from definitions. □

What now follows are crucial results stating that the four notions of opacity are decidable provided that the system has finitely many states. In all the results that follow, it is assumed that $[\mathcal{M}_0\rangle$ is finite.[2]

Theorem 1. *It is decidable whether $\mathcal{P}$ is initial-opaque.*

Proof. The proof of this and the next results are based on a language-theoretic argument centered around a directed graph G obtained from $RG(\Sigma)$ in the following way:

- a fresh (initial) node v_0 is added to $[\mathcal{M}_0\rangle$ and connected by an $obs(M)$-arc to every node $M \in \mathcal{M}_0$ together with any node M' for which there is a directed path

$$M_1 \; U_1 \; M_2 \; \ldots \; U_{m-1} \; M_m \tag{1}$$

[2] Note that the finiteness of $[\mathcal{M}_0\rangle$ is decidable, and can be checked using the standard coverability tree construction [11].

in $RG(\Sigma)$ such that $M_1 = M$, $M_m = M'$,

$$\begin{aligned} obs(M_1) &= \cdots = obs(M_m) \\ obs(M_1, U_1) &= \cdots = obs(M_{m-1}, U_{m-1}) = \tau\,; \end{aligned}$$

– for every directed path

$$M_1\ U_1\ M_2\ \ldots\ U_{m-1}\ M_m\ U\ M'_1\ U'_1\ M'_2\ \ldots\ U'_{n-1}\ M'_n \tag{2}$$

in $RG(\Sigma)$ with $m, n \geq 1$, we add a $obs(M_m, U)obs(M'_1)$-labelled arc[3] from M_1 to M'_n provided that that:

$$\begin{aligned} obs(M_1) &= \cdots = obs(M_m) \\ obs(M'_1) &= \cdots = obs(M'_n) \\ obs(M_1, U_1) &= \cdots = obs(M_{m-1}, U_{m-1}) = \tau \\ obs(M'_1, U'_1) &= \cdots = obs(M'_{n-1}, U'_{n-1}) = \tau\,, \end{aligned}$$

and we have that $obs(M_m, U) \neq \tau$ or $obs(M_m) \neq obs(M'_1)$.

Note that G is finite since $[\mathcal{M}_0\rangle$ is finite and so $RG(\Sigma)$ is a finite reachability graph.

To decide initial-opacity, we proceed as follows. First, we construct a finite state machine F_1 by taking G with all the states being treated as final. After that we construct a finite state machine F_2, by taking F_1 and deleting all arcs from v_0 to M' such that there is no directed path as in (1) above satisfying additionally the following condition $\{M_1, M_2, \ldots, M_m\} \cap \mathcal{P} = \varnothing$.

It is easy to see that $\mathcal{P}$ is initial-opaque iff $L(F_1) \subseteq L(F_2)$. □

Theorem 2. *It is decidable whether $\mathcal{P}$ is final-opaque.*

Proof. First, we construct a finite state machine F_1 by taking G with all the states being treated as final. After that we construct a finite state machine F_2 by taking G and making a copy M'_{copy} of each node $M' \in [\mathcal{M}_0\rangle \setminus \mathcal{P}$. The copied nodes are the final states, and there are no arcs outgoing from them. There are two cases when there are arcs incoming to M'_{copy}.

Case 1: $M' \in \mathcal{M}_0$. Then we add an $obs(M')$-arc from v_0 to M'_{copy}.

Case 2: using a directed path as in (2) above, we added an $\ell\ell'$-arc from M to M'. Then we add an $\ell\ell'$-arc from M to M'_{copy} provided that $n = 1$.

It is easy to see that $\mathcal{P}$ is final-opaque iff $L(F_1) \subseteq L(F_2)$. □

Theorem 3. *It is decidable whether $\mathcal{P}$ is total-opaque.*

Proof. First, we construct a finite state machine F_1 by taking G with all the states being treated as final. After that we construct a finite state machine F_2 by taking $RG(\Sigma)$ with all the nodes in $\mathcal{P}$ together with the adjacent arcs deleted and $\mathcal{M}'_0 = \mathcal{M}_0 \setminus \mathcal{P}$. From the resulting 'reachability graph' we construct, exactly as we have done so for the original $RG(\Sigma)$, the corresponding graph G' and F_2 is obtained from G' by assuming that all states are final.

It is easy to see that $\mathcal{P}$ is total-opaque iff $L(F_1) \subseteq L(F_2)$. □

[3] Note that the label is itself a sequence consisting of two labels returned by the *obs* mapping.

4 Dining cryptographers

To illustrate our approach, we use the example of the dining cryptographers, first presented in [3]. A simplified version (involving only two cryptographers) was presented in [1].

This example involves three diners and admits some further anonymity properties, e.g., a paying cryptographer can remain anonymous w.r.t. his or her companions.

The three cryptographers, Anne, Bob and Charlie, enjoy a meal in a restaurant. When they call for the bill, the waiter tells them that it has already been paid. Each cryptographer wishes to know whether the bill was paid by the NSA, or if it was one of them. However, if one of them paid, they do not want an eavesdropper, Yves, on the neighbouring table to know which of them paid. The protocol they choose to solve this problem is as follows:

They each toss a coin, and reveal the result only to the cryptographer sitting one their left. Yves, of course, cannot see any of the coins. If Anne paid, she lies about the parity of the two coins (she calls 'agree' if she sees a head and a tail, and 'disagree' otherwise). If Anne did not pay, she tells the truth about the parity of the coins. Similarly for Bob and Charlie. Now each cryptographer knows if the NSA paid for the meal, or if the bill was settled by one of them. If the number of "disagree" calls is even, then the NSA is paying. If the number is odd, then one of the cryptographers is paying. In this case, the other two cryptographers do not know which of their dining companions has paid for the meal. Yves cannot distinguish the paying cryptographer from the others.

Figure 4 presents a possible encoding of the protocol. The two places at the top of the diagram represent Anne's initial state (having paid is represented by placing a single token in place AP, and having not paid is represented by placing a single token in place $A\neg P$). The two places at the right represent Bob's initial state (BP and $B\neg P$) and the two places on the left Charlie's initial state (CP and $C\neg P$). The possible initial markings for these places are

$$\{AP, B\neg P, C\neg P\}, \{A\neg P, BP, C\neg P\}, \{A\neg P, B\neg P, CP\}, \{A\neg P, B\neg P, C\neg P\}.$$

The three sets of two places in the centre of the diagram represent the three coins (heads is represented by placing tokens in place $c_i h$, and tails is represented by placing tokens in place $c_i t$, for $i = 1, 2, 3$). For each pair, the marked place must contain two tokens. This is because two cryptographers must see each coin. The possible initial markings for the coins are therefore

$$\{\{1h, 1h\}, \{1t, 1t\}\} \times \{\{2h, 2h\}, \{2t, 2t\}\} \times \{\{3h, 3h\}, \{3t, 3t\}\}$$

The set of possible initial markings, $\mathcal{M}_0$, is the cross product of the cryptographer markings and the coin markings.

The eight transitions at the top represent the eight possible scenarios for Anne, given by two possibilities for each coin multiplied by the two possibilities for her own initial state. Each transition is labelled with '$A0$' (if Anne says the coins 'disagree') or '$A1$' (if Anne says the coins 'agree'). Similarly for Bob on

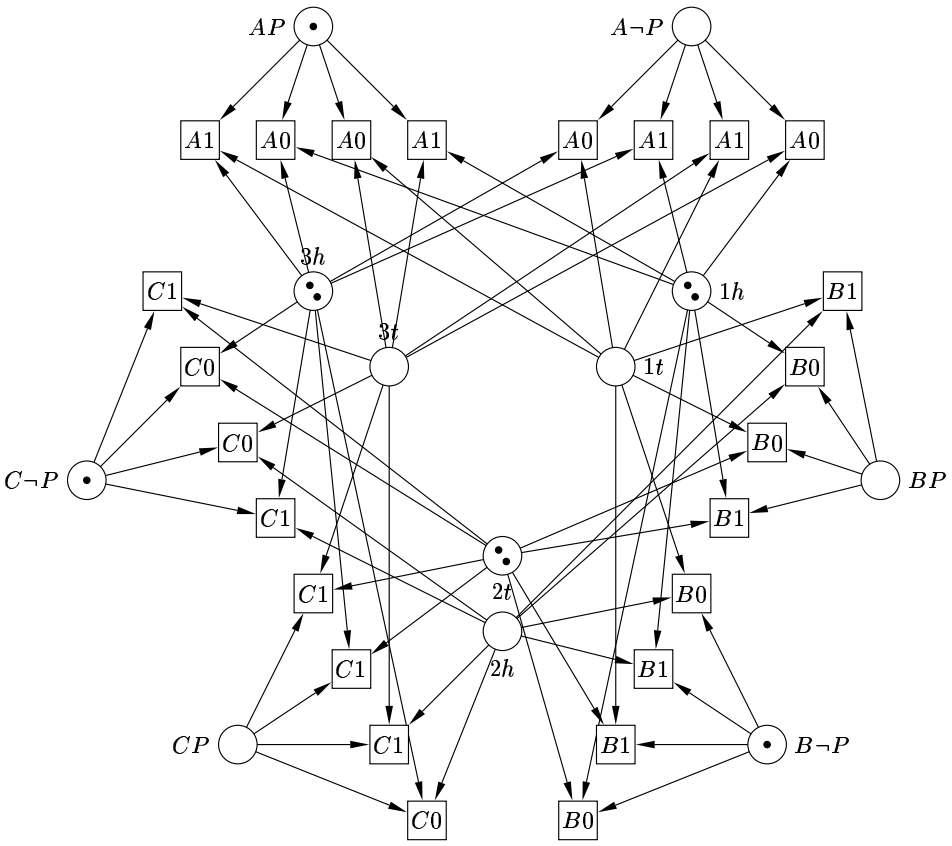

Fig. 4. Net for the 3-way dining cryptographers example with one of the n initial markings.

the right and Charlie on the left. This gives the transition labelling ℓ which will be used for defining the visibility of steps.

Yves' observation function is simple. He can see none of the places (he does not know the initial state of the cryptographers, nor the state of the coins), but he can see all of the labels of the executed transitions (he can hear all that they say). In other words, for every reachable marking M and executed step U, we have the following (see section 3.1):

$$obs_Y(M) = M|_{\varnothing} = \varnothing$$
$$obs_Y(U) = \ell(U).$$

Note that Yves also knows the structure of the original net, i.e., the protocol.

We wish to demonstrate that after observing the execution of transitions, although Yves may be able to determine whether the meal was paid for by one

of the cryptographers, he can never know which one. The three properties we wish to be initial-opaque are therefore $\mathcal{P}_A = \{M \in \mathcal{M}_0 \mid M(AP) = 1\}$ and $\mathcal{P}_B = \{M \in \mathcal{M}_0 \mid M(BP) = 1\}$. $\mathcal{P}_C = \{M \in \mathcal{M}_0 \mid M(CP) = 1\}$.

Yves cannot determine the satisfaction of any of the above predicates. Note, however, that Yves can in either case determine the satisfaction of the property $\mathcal{P} = \{M \in \mathcal{M}_0 \mid M(AP) + M(BP) + M(CP) = 1\}$, i.e., he knows when one of the cryptographers paid the bill. In terms of our framework, both $\mathcal{P}_1$ and $\mathcal{P}_2$ are initial-opaque, but $\mathcal{P}$ is not.

These properties are similar to the ones which hold for the limited version of the case study, presented in [1]. The difference is that in the situation where one of the cryptographers paid for the meal, the other two do not know who it was. The paying cryptographer remains anonymous with respect to their dining companions. To see this, we model the point of view of one of the cryptographers, by change the *obs* function to model the increased level of knowledge. For example, the observation function of Anne is such that, for every reachable marking M and executed step U,

$$obs_{Anne}(M) = M|_{\{AP, A\neg P, 1h, 1t, 3h, 3t\}}$$
$$obs_{Anne}(U) = \ell(U).$$

Anne knows her own initial state, and can see the state of her coin and the coin on her left. Given this observation function, she learns exactly what she wants to know — did the NSA pay, or was it one of her friends?

5 Conclusions and future work

We have presented an extension of our earlier Petri net framework in order be able to capture a richer class of information flow requirements. In this richer model we can encompass conditional information flow policies (e.g., downgraders) as well as scenarios that include key compromise and key refreshment.

In this paper we model the full dining cryptographers example which allows us to illustrate how various flavours of anonymity can be captured, corresponding to the various observer viewpoints.

A further advance presented here is a full treatment of invisible events.

The decidability results of the earlier paper have been extended to the richer model presented here and for the new *total opacity* property.

In future work we intend to explore the relationship of the approach presented here to process algebraic formulations of generalised non-interference [13] and anonymity [14].

A major challenge in such work is the choice of appropriate abstractions to encode the adversary's observational capabilities. This is particularly delicate where cryptographic mechanisms are involved. Adversary deductions and algebraic manipulations complicate the modelling. We intend to investigate using the dynamic *obs* mappings presented here to address such issues.

A further line of research is to explore analogues in this framework of the notion of *non-deducibility on strategies*, due to Johnson and Wittbold [16]. This

seeks to capture the possibility of a secret user and an uncleared user colluding and using adaptive strategies to cause information flows in violation of the policy. This is likely to require more precise modelling of various flavours of non-determinism within the Petri net framework.

We will also investigate the problem of preservation of opacity properties under refinement and composition of Petri nets.

Acknowledgement

This research was supported by the EPSRC GOLD project and DSTL.

References

1. J.W.Bryans, M.Koutny and P.Y.A.Ryan: Modelling Opacity using Petri Nets. Proceedings of WISP 2004 (2004)
2. N.Busi and R.Gorrieri: Structural Non-interference with Petri Nets. Proceedings of WITS 2004 (2004)
3. D. Chaum: The dining cryptographers problem: unconditional sender and receipient untraceability. Journal of Cryptology, 1, 1988
4. E.Cohen: Information Transmission in Computational Systems. Proceedings of 6th ACM Symposium on Operating System Principles (1997)
5. R.J.Feiertag: A technique for Proving Specifications are Multi-level Secure Technical. Report CSL109, CSL, SRI International (1980)
6. J.Goguen and J.Meseguer: Security Policies and Security Models. Proceedings of IEEE Symposium on Security and Privacy (1982)
7. J.Goguen and J.Meseguer: Inference Control and Unwinding. Proceedings of IEEE Symposium on Research in Security and Privacy (1984)
8. C. A. R. Hoare: Communicating Sequential Processes (1985)
9. L. Mazare: Using Unification for Opacity Properties. Proceedings of WITS 2004 (2004)
10. C. O'Halloran: A Calculus of Information Flow. Proceedings of ESORICS (1990)
11. W.Reisig and G.Rozenberg (Eds.): Lectures on Petri Nets. LNCS 1491 & 1492 (1998)
12. J.Rushby: Noninterference, Transitivity and Channel-Control Security Policies. SRI Techical Report (1992)
13. P.Y.A.Ryan: Mathematical Models of Computer Security. Proceedings of Foundations of Security Analysis and Design, LNCS 2171 (2000)
14. S.A.Schneider and A.Sidiropoulos: CSP and Anonymity. Proceedings of ESORICS (2000)
15. D.Sutherland: A Model of Information, Proceedings of 9th National Computer Security Conference (1986)
16. J.T.Wittbold and D.M.Johnson: Information Flow in Nondeterministic Systems. Proceedings of the Symposium on Research on Security and Privacy (1990)