Available online at www.sciencedirect.com
ScienceDirect
Electronic Notes in Theoretical Computer Science
ELSEVIER
Electronic Notes in Theoretical Computer Science 229 (2009) 97–115
www.elsevier.com/locate/entcs

Towards Secrecy for Rewriting in Weakly Adhesive Categories

Tobias Heindel

Abteilung für Informatik und angewandte Kognitionswissenschaft
Universität Duisburg-Essen
Duisburg, Germany

Abstract

Inspired by the scope extrusion phenomenon of name passing calculi that allow to reason about knowledge of (secret) names, we propose an abstract formulation of the concept of secret in any weakly adhesive category. The guiding idea is to mark part of a system state as visible or publicly accessible; further, in principle, something that has become public knowledge will stay accessible indefinitely.
The main technical contribution consists in providing a proof which shows that a recently proposed categorical construction, which produces a category having monomorphisms as objects and pullback squares as morphisms, preserves weak adhesivity. Finally we sketch how it is possible to verify certain secrecy properties using unfolding based verification approaches that lately have been generalized to rewriting systems in weakly adhesive categories.

Keywords: adhesive categories, interaction models, verification

1 Introduction

In every day communication, private information is usually exchanged only between communication partners that trust each other, as the consequences of public availability of private information tend to be numerous and subtle. In fact, more often than not, one would rather prefer that a certain piece of private information will never be become known to the public. Here we are not only talking about issues of embarrassment or reputation, but also about secret data like personal identification numbers for (on-line) banking accounts.

Nevertheless, as the example of on-line banking illustrates, there often occur situations in which secret data need to be transmitted via protected channels between trustworthy communication partners, and moreover the critical data must not become disclosed to a third party. The running example of this paper will be concerned with access keys to a private network, e.g. the intranet of some banking institute. Obviously, in this scenario, it is important that such access keys do not become publicly available.

doi:10.1016/j.entcs.2009.06.041

One of the first approaches to formally reason about the security of key exchange protocols using cryptographic methods, is the spi-calculus [1], which extends the π-calculus [14] by cryptographic primitives. Based on this name passing calculus, there has been carried out a large amount of work concerning the verification of concrete protocols. The actual protocol verification tools however do sometimes use techniques from other fields of computer science (see e.g. [3]). Alternatively, protocols might also be specified and verified using graph transformation systems [4,12]; the latter have the advantage that they are often easier understandable by laypersons.

Now the aim of this paper is *not* another concrete proposal of a modelling technique for protocols. Instead we strive for a better understanding of the fundamental distinction between private and public knowledge, which corresponds to the open/bound names dichotomy of name passing calculi. Moreover the scope extrusion phenomenon of the latter captures the possibility to exchange secret information and, in the extreme, to make secret information publicly available.

Taking a more abstract point of view, given an arbitrary state of a system, then part of of this state is open to public access (while at the same time other parts are still secret). In the process calculus world, the open part corresponds to the free names of a process. In graph transformations systems using the borrowed context approach [5], the open part is singled out by a sub-graph of the graph which models the whole system state.

The main question is now, when the private part and the public part (in the model) of a given system state should be considered "sufficiently" distinct such that all secret information is protected from public access. Though this question usually has an intuitive answer in concrete example cases, the question seems more difficult in the abstract setting of this paper, as we consider system states as objects of an arbitrary (weakly) adhesive category.

To help answer this question, we proceed as follows. First we introduce the protected links calculus as an toy example of a simple name passing calculus, which nevertheless is sufficiently rich to illustrate the private/public dichotomy and allows to give a precise characterization of *secrecy violations*. Then we give the graphical representation of this calculus in section 3. With these concrete examples at hand, in section 4, we set out to lift the notion of *secrecy violation* to the abstract setting of adhesive categories in such a way that the results of [2] apply.

2 The protected links calculus

The running example of this paper will be the protected links calculus (PLC), which couples the ideas of (the implementation of) the explicit fusion calculus [16] with a basic access control mechanism. Recall that the explicit fusion calculus was developed with the goal of providing an implementation of Milner's π-calculus [14]. The "machine model" was the fusion machine described in [16]; a simplified version of the latter has been proposed in [8], where also a "low-level" encoding of the π-calculus was presented.

Now the main characteristic of the protected links calculus that it shares with

the explicit fusion calculus and the fusion machine, is that it does not use any name substitution at the meta-level but instead uses a "low-level" approach similar to the one of the fusion machine. In the latter, substitution of names is implemented by a forwarding mechanisms that was explored in more detail in [8].

Indeed, the major part of the primitives of the protected links calculus are taken from the calculus of explicit fusions and its fusion machine, namely (asynchronous) input and output, parallel composition, and forwarders, which we here more often call *links*; the latter however are equipped with an access control mechanism based on the notion of *access right*, which is the new entity kind of the PLC.

Definition 2.1 (Syntax of the protected links calculus) *Let* $\mathcal{N}$ *be a collection of* names*, which is the disjoint union of* public *names* $\ddot{\mathcal{N}}$ *and* private *names* $\hat{\mathcal{N}}$*, which means* $\mathcal{N} = \ddot{\mathcal{N}} \uplus \hat{\mathcal{N}}$*. Then the set of* (raw) terms *of the* PL*-calculus is given by the following specification.*

$$\begin{array}{rll} P ::= u\langle x\rangle & u, x \in \mathcal{N} & \text{(output action)} \\ \mid u(y) & u, y \in \mathcal{N} & \text{(input action)} \\ \mid u \dashv w & u, w \in \mathcal{N} & \text{(protected link)} \\ \mid x \succcurlyeq u & x \in \mathcal{N}, u \in \hat{\mathcal{N}} & \text{(access right)} \\ \mid 0 & & \text{(inaction)} \\ \mid P \mid P & & \text{(parallel composition)} \end{array}$$

Let P *be a raw term; then the set of* free names of P*, written* $\operatorname{fn} P$*, contains all names that occur in* P *but are not private, i.e.* $\operatorname{fn} P = \{x \in \ddot{\mathcal{N}} \mid x \text{ occurs in } P\}$.

A process *of the* PL*-calculus is a raw term up to* structural congruence*, written* $\equiv$*, which is the smallest equivalence relation on raw terms satisfying the following axioms where* P, Q, R *range over* PLC *terms.*

$$P \mid 0 \equiv P \qquad P \mid Q \equiv Q \mid P \qquad P \mid (Q \mid R) \equiv (P \mid Q) \mid R$$

If a name v is public, i.e. $v \in \ddot{\mathcal{N}}$, we sometimes write $\ddot{v}$ instead of v to emphasize this fact; conversely, if $\ddot{v}$ is a name, then we implicitly assume that $\ddot{v} \in \ddot{\mathcal{N}}$; by a similar convention, if we write $\hat{v}$, then we silently presuppose that $\hat{v} \in \hat{\mathcal{N}}$.

As mentioned above, the main difference w.r.t the calculus of explicit fusions consists in the new entity, called access right. Following a common interpretation of process calculi, names are often referred to as *channels* through which input and output actions may synchronise and communicate. Further we will talk about *scopes*, which are those parts of terms that share a private name $\hat{u}$, or names that are related to $\hat{u}$ via a chain of bi-directional links, i.e. $\hat{u}$ and v are in the same scope, if there is a chain $\hat{u} \dashv w_1 \mid w_1 \dashv \hat{u} \mid \cdots v \dashv w_n \mid w_n \dashv v$. Relying on this word usage, the ideas of the protected links calculus can be described as follows.

The protection mechanism of links ensures that an output action can enter into a scope only if it has the access right for the scope in question. This is captured by the conditional forwarding mechanism of protected links, which checks access rights before output actions are relocated. In contrast, modelling the possibility of attacks

and careless users, access rights may always spread between linked channels. These phenomena are made precise in the formal definition of the reaction relation over PLC terms, and is discussed in more detail afterwards.

Definition 2.2 (Reaction in the PLC) *The* reaction relation *over* PLC *terms, written* $\twoheadrightarrow$*, is the smallest relation satisfying the axioms and rules of Figure 1.*

$$\textsc{prot}\ \frac{\hat{v} \in \hat{}\mathcal{N}}{(u\langle x\rangle \mid x \succcurlyeq \hat{v} \mid u \dashv \hat{v}) \ \twoheadrightarrow\ (\hat{v}\langle x\rangle \mid x \succcurlyeq \hat{v} \mid u \dashv \hat{v})}$$

$$\textsc{pub}\ \frac{\ddot{v} \in \ddot{}\mathcal{N}}{(u\langle x\rangle \mid u \dashv \ddot{v}) \ \twoheadrightarrow\ (\ddot{v}\langle x\rangle \mid u \dashv \ddot{v})}$$

$$\textsc{exch}\ \frac{}{(x \succcurlyeq \hat{v} \mid x \dashv y) \ \twoheadrightarrow\ (y \succcurlyeq \hat{v} \mid x \dashv y)}$$

$$\textsc{comm}\ \frac{}{(u\langle x\rangle \mid u(y)) \ \twoheadrightarrow\ (x \dashv y \mid y \dashv x)}$$

$$\textsc{struct}\ \frac{P \equiv P' \qquad P' \twoheadrightarrow Q' \qquad Q' \equiv Q}{P \twoheadrightarrow Q}$$

Fig. 1. Reaction rules of the protected links calculus

These axioms and rules can be explained as follows. The protection mechanism of links is captured by the PROT-axiom: an output action $u\langle x\rangle$ is forwarded along a link $u \dashv \hat{v}$ only if the data x come equipped with the access right $x \succcurlyeq \hat{v}$ to the channel $\hat{v}$, to which the action could be transported via the link. To simplify the modelling process we think of the data x as a representation of the actual user that is trying to send x.

However, the protection mechanism does not restrict transmissions to public channels, i.e. everyone can send on public channels. This is formalized by the PUB-rule, which says that given a link $u \dashv \ddot{v}$, which models a direct network link connecting u to $\ddot{v}$, an output action $u\langle x\rangle$ at the origin u can always travel to the target $\ddot{v}$, provided that the target channel $\ddot{v}$ is a public.

Next we come to the formal counterpart of the phenomenon that, as known from practical experience, keys are often stolen or exchanged with untrustworthy partners. Hence, assuming the worst case, the distribution of access keys is unconditional, i.e. whenever there is a direct means of communication a key may be exchanged. Precisely the EXCH-axiom says that whenever the user x has a key to enter the scope v, modelled by the access right $x \succcurlyeq \hat{v}$, in the presence of a link $x \dashv y$, which models a direct means of communication, the user y will always manage to obtain the "key" granting access to v, which then results in the access right $y \succcurlyeq \hat{v}$.

A request for a (new) channel for the transmission of (possibly confidential) data is modelled by a send action $u\langle x\rangle$ where x is a channel name which, simplifying again, is thought of as the user issuing the channel request. The receiver, corresponding to the input action $u(y)$, then will establish direct links between x and y, which is

represented by complementary forwarders $x \neg y$ and $y \neg x$; the described communication protocol is captured by the COMM-axiom. Note, that this axiom is essentially the same as the single reaction axiom of the calculus of explicit fusions [16]. Finally the STRUCT-rule says that reaction is closed under structural congruence.

Example 2.3 (Key exchange) *To show the* PL*-calculus at work, we consider key exchange. That a user $\hat{x}$ has a key granting access to $\hat{v}$ is modelled by the access right $\hat{x} \succcurlyeq \hat{v}$. Now suppose user $\hat{x}$ wants to exchange this key with user $\hat{y}$ and that $\hat{x}$ and $\hat{y}$ usually communicate via channel u, possibly a private channel as well. The* PLC *process $\hat{x} \succcurlyeq \hat{v} \mid u\langle \hat{x} \rangle \mid u(\hat{y})$ is a possible solution to achieve this, as we have the following reactions.*

$$\hat{x} \succcurlyeq \hat{v} \mid u\langle \hat{x} \rangle \mid u(\hat{y}) \rightharpoonup \hat{x} \succcurlyeq \hat{v} \mid \hat{x} \neg \hat{y} \mid \hat{y} \neg \hat{x} \quad \text{(COMM)}$$
$$\rightharpoonup \hat{y} \succcurlyeq \hat{v} \mid \hat{x} \neg \hat{y} \mid \hat{y} \neg \hat{x} \quad \text{(EXCH)}$$

Hence after these two steps, user $\hat{y}$ has the key granting access to $\hat{v}$, which is modelled by the access right $\hat{y} \succcurlyeq \hat{v}$.

Before we come to the main theme of this paper, namely secrecy, we give a short comparison of the PL-calculus on the one hand, and the calculus of explicit fusions, the fusion machine and the linear forwarder calculus on the other hand. The only properly new primitive of the PL-calculus is the access right since the fusion related calculi do not have any similar entities; however the latter calculi are not designed to reason about secrecy but only about communication via name passing.

The second theme which allows to discern the mentioned calculi concerns the mechanisms that are used to connect channels or "fuse" names. Whereas the calculus of explicit fusions addresses the issue of connection of names at the level of structural congruence, which intuitively corresponds to (irreversible) fusion of names, both the linear forwarder calculus [8] and the protected links calculus choose a "low-level" approach, and add reaction rules that "implement" the fusion of names; this "low-level" approach has the advantage, that additional reaction rules might be added to model the break down of network links, which corresponds to the removal of links between names.

Finally we would like to stress that the PL-calculus is just an example which allows to illustrate the idea of secrecy without the need to formally introduce the technical details of double pushout rewriting [6]. Hence we also omitted replication, and synchronous input and output, which only would have burdened the presentation. Moreover, for the purpose of this paper, it seemed suitable to avoid the notions of *α-equivalence* and *bound name*.

Summarizing, the PL-calculus can be seen as a simplified version of the linear forwarder calculus with a new entity called access right. The latter allows to reason about secrecy, as demonstrated in the following, central example of the paper, which illustrates how secrecy holes are modelled in the PL-calculus.

Example 2.4 (Secrecy hole) *A secrecy hole of a network, is a channel through which private access keys are made public. A process term P contains an* immediate *secrecy hole, if P contains a sub-term of the form $\ddot{v} \succcurlyeq \hat{w}$, i.e. if in the modelled system,*

there is a publicly available key granting access to some private channel $\hat{w}$.

Further a process Q *models a system with a* covert *secrecy hole, if* Q *does not contain any sub-term of the form* $\ddot{v}\!\succcurlyeq\!\hat{w}$ *but such a sub-term might arise after a number of reductions of* P. *For example the process* $\hat{u}\!\succcurlyeq\!\hat{w} \mid \hat{u}\!\neg\!\ddot{v}$ *does not contain any immediate secrecy hole; however we have the reaction* $(\hat{u}\!\succcurlyeq\!\hat{w} \mid \hat{u}\!\neg\!\ddot{v}) \rightarrowtriangle (\ddot{v}\!\succcurlyeq\!\hat{w} \mid \hat{u}\!\neg\!\ddot{v})$ *via the* EXCH*-rule, and the latter contains an immediate secrecy hole. Hence, given a term* Q *one might want to prove that it does not contain any (covert) secrecy holes.*

We would like to emphasize that we do not claim that this is the only class of secrecy holes that might be worthwhile to investigate.

How the verification of secrecy w.r.t this "definition" may be achieved using the unfolding technique of [2], is sketched in 4.6. However we first need to recall the necessary concepts concerning transformation systems and categories that allow to model systems following the double pushout approach of [6].

3 The graphical counterpart of the PLC

In this section we give a graphical representation of the PLC; more precisely, for each process there will be a corresponding graph and vice versa. Moreover each reaction rule will be a graph transformation rule based on the double pushout approach (DPO) [6]. However, omitting the details of DPO rewriting, we give an informal presentation that nevertheless should convey the main ideas of graph transformation.

The channels or names of PLC processes will correspond to nodes in the graphical representation. All entities of the PL-calculus correspond to different kinds of edges between the nodes. An output action $u\langle x\rangle$ is represented by a *send* arrow $\textcircled{x} \leftarrowtail \textcircled{u}$, an input action $u(y)$ corresponds to a *receive* edge $\textcircled{u} \rightarrowtail \textcircled{y}$, an access right $x\!\succcurlyeq\!\hat{u}$ is drawn as $\textcircled{x} \rightarrowtail \textcircled{\hat{u}}$, a link $u\!\neg\!w$ becomes a *connection* arc $\textcircled{u} \multimap \textcircled{w}$, and the inaction is the empty graph $\varnothing$. Finally parallel composition is achieved by union of graphs. To avoid clutter, a pair of complementary links $u\!\neg\!w \mid w\!\neg\!u$ is represented by $\textcircled{u} \Box \textcircled{w}$. For a private node $\textcircled{\hat{u}}$, the label $\hat{u}$ already contains the information that this is a private node, and the gray boundary only emphasizes this fact.

Moreover, not only does each process correspond to a graph, but also each reaction rule has a corresponding transformation rule. A (linear) graph transformation *rule* or *production* q is essentially a pair of graphs L, R (called left- and right-hand side, respectively) with a common sub-graph K (referred to as interface), i.e. $q = L \supseteq K \subseteq R$. Assuming that the sets of nodes and edges in rules are disjoint, and writing as if graphs were mere sets, the rewriting mechanism of such rules can be described as follows.

Suppose that the left-hand side L of a rule $q = L \supseteq K \subseteq R$ is a sub-graph of some larger graph G, i.e. $L \subseteq G$, then the rule q first removes from G all those nodes and edges that are covered by the left-hand side L but not contained in the interface K, which results in an intermediate graph $D \subseteq G$; in a second step, those nodes and edges of R that are not contained in K are adjoined to the intermediate result D, yielding a graph $H \supseteq D$. Provided that $R \cap G \subseteq K$ (and the inclusion $L \subseteq G$ satisfies the so-called *dangling condition* [4]), the result H can be described

as $H = (G \setminus (L \setminus K)) \cup R$.

Now the graphical counterpart of the rules of the PLC is illustrated in Figure 2. Note that in this encoding of the PLC rules, one directly mentions which resources

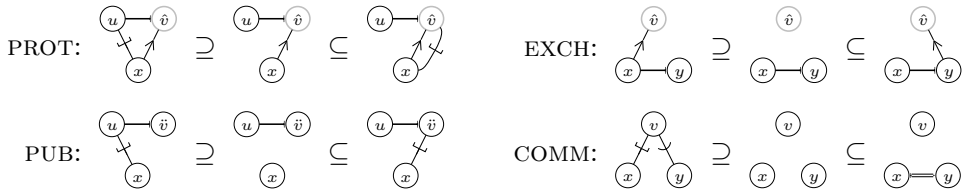


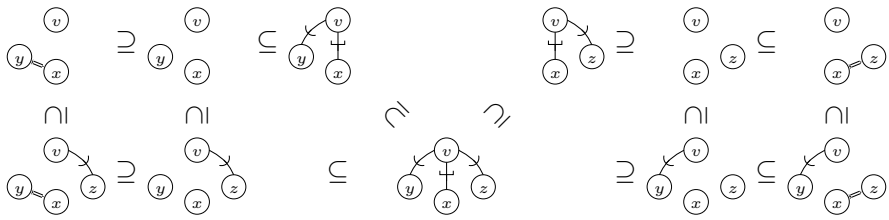
Fig. 2. The graphical representation of the PLC calculus

are only used as "catalysts", and hence remain unchanged during the reactions. To see the correspondence between the calculus and its graphical presentation, consider the following example concerning the COMM-rule.

Example 3.1 (Communication in PLC via graphs) *The protected link calculus process* $u(y) \mid u\langle x\rangle \mid u(z)$ *has two possibilities to evolve: either* $u\langle x\rangle$ *reacts with* $u(y)$ *or with* $u(z)$*, i.e.*

$$(y \dashv x \mid x \dashv y \mid u(z)) \leftarrowtail (u(y) \mid u\langle x\rangle \mid u(z)) \rightarrowtail (u(y) \mid z \dashv x \mid x \dashv z)$$

The corresponding graph transformation steps can be illustrated as follows.

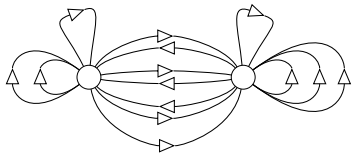


So far we have not elaborated on the concept of graph we have actually used in this encoding. As a starting point, we will work with suitably labelled graphs.To emphasize that the set of public nodes is actually a fully fledged sub-graph of the graph which represents the whole process, we will later use *marked graphs* instead; the latter are pairs of a graph an a marked sub-graph (see Definition 3.2). The use of marked graphs is motivated by the fact that the notion of *sub-graph* allows for a straightforward categorical generalization, viz. *sub-object*, whereas this is not the case for labels. In the end, marking of objects might be thought of as an abstract labelling mechanism.

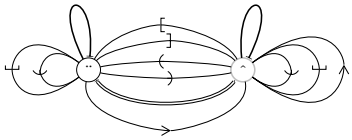
3.1 The graphical representation in a slice category

A suitable choice of a category for a precise, graphical presentation of processes of the protected links calculus is the category of graphs *typed* [1] over the *type graph*

[1] This terminolgoy is in accordance with the theory and applications of graph grammars, see e.g. http://tfs.cs.tu-berlin.de/agg/typegraph.html.

$T =$  $\in \mathbb{G}$,

i.e. processes correspond to objects of the slice category $\mathbb{G}{\downarrow}T$ where $\mathbb{G}$ is the usual category of directed multi-graphs (see Definition A.1 in Appendix A). In the graph T, the left node corresponds to public channels, the right one to private ones, the three loops on the left node stand for inputs, outputs and links between pairs of public names, respectively, the four loops on the right one represent inputs, outputs, links, and access rights between pairs of private channels, respectively, and similar explanations can be provided for the remaining edges between the two different nodes. This means that the $\mathbb{G}{\downarrow}T$-object $T{-}\mathrm{id}_T{\to}T$ would be depicted as

 .

Note that this formal rendering in the slice category $\mathbb{G}{\downarrow}T$ means that Figure 2 actually gives *schemes* for rules, since it is not specified for several nodes whether they are public or private, i.e. whether they are mapped to $\ddot{\circ}$ or $\hat{\circ}$ in T; in other words there are actually eight different instances of the COMM-rule, and four instances for each of the other rules.

In this formal setting, a secrecy violation $\ddot{v}{\rightarrowtail}\hat{w}$ is an object of $\mathbb{G}{\downarrow}T$, which can also be seen as a sub-graph of T, corresponding to the obvious graph morphism $c\colon \ddot{v}{\triangleright}\!{\rightarrowtail}\hat{w} \rightarrowtail T$ in $\mathbb{G}$. Further, observe that a (typed) graph $(A{-}a{\to}T) \in \mathbb{G}{\downarrow}T$ does not contain any secrecy violation $\ddot{v}{\rightarrowtail}\hat{w}$, i.e. there does not exist any monomorphism $\ddot{v}{\rightarrowtail}\hat{w} \rightarrowtail (A{-}a{\to}T)$ in $\mathbb{G}{\downarrow}T$, if and only if pulling back along c yields a discrete graph (in the slice category $\mathbb{G}{\downarrow}\ddot{v}{\triangleright}\!{\rightarrowtail}\hat{w}$). More general, given any graph $(B{-}b{\to}T) \in \mathbb{G}{\downarrow}T$ representing some PL-process P, pulling back along $c\colon \ddot{v}{\triangleright}\!{\rightarrowtail}\hat{w} \rightarrowtail T$ followed by post-composition with c yields the *secrecy relevant* part of $(B{-}b{\to}T)$.

This *pullback-compose* construction can be generalized to any graph morphism $\varphi\colon T' \to T$ in $\mathbb{G}$, and actually gives rise to functors $\varphi^{\circledast}\colon \mathbb{G}{\downarrow}T \to \mathbb{G}{\downarrow}T$ which act by pulling back along φ followed by post-composition with φ, i.e. $\varphi^{\circledast} = \Sigma_\varphi \circ \varphi^*$ where $\varphi^*\colon \mathbb{G}{\downarrow}T \to \mathbb{G}{\downarrow}T'$ and $\Sigma_\varphi\colon \mathbb{G}{\downarrow}T' \to \mathbb{G}{\downarrow}T$ are a choice of a pullback functor and its left adjoint, respectively. The co-unit $\varepsilon\colon \varphi^{\circledast} \to \mathrm{id}_{\mathbb{G}{\downarrow}T}$ of this adjunction $\Sigma_\varphi \dashv \varphi^*$ embeds the secrecy relevant part of an $\mathbb{G}{\downarrow}T$-object.

In the same way, the *public* part of an object $(A{-}a{\to}T) \in \mathbb{G}{\downarrow}T$ is given by $i^{\circledast}(a)$ where $i\colon \ddot{\circ} \rightarrowtail T$ is the inclusion morphism mapping the single node to the left node of T. As i is a monomorphism, also the co-unit $\varepsilon\colon i^{\circledast} \rightarrowtail \mathrm{id}_{\mathbb{G}{\downarrow}T}$ is monic. Hence ε gives for each typed graph $(A{-}a{\to}T)$ the embedding $\varepsilon_a\colon i^{\circledast}(a) \rightarrowtail a$ of its public part into the whole object. This exactly corresponds to the free names of a process if $(A{-}a{\to}T)$ arose from a PL-process. Abstracting away from the functors $i^{\circledast}$ and $c^{\circledast}$,

we thus lead to the following alternative labelling mechanism.

3.2 Marking graphs

An alternative to attribution, labelling or typing of graphs is the idea to "mark" part of a given graph. This idea has been discussed recently within the graph transformation community and is presented in detail in [10]. In the present case, one might for example choose to mark only the public nodes of a graph, i.e. the objects we are working with are pairs $\langle G, G'\rangle$ such that $G \supseteq G'$. In the next section we will replace graphs by objects of any (weakly) adhesive category. Hence the details about the graphs that we will use are deferrred to the end of this section.

The main idea of the graphical presentation of PL-calculus terms is as follows: let P be a PLC term and let $[\![P]\!]$ be the presentation of P, which is an (edge and node labelled) graph. Then the public nodes of P form the set of free names $\mathrm{fn}\, P$, which is – when considered as a discrete graph, i.e. a graph without edges – a sub-graph of $[\![P]\!]$, i.e. the pair $\langle [\![P]\!], \mathrm{fn}\, P\rangle$ satisfies $[\![P]\!] \supseteq V'$.

For the remainder of the section we write $\langle G, G'\rangle_{\supseteq}$ if G and G' are graphs, such that the inclusion $G \supseteq G'$ holds, and call the pair $\langle G, G'\rangle_{\supseteq}$ a *marked graph*. Next we will supply as suitable notion of morphism between marked graphs, such that the marked part of a graph will remain marked and moreover the marked and the unmarked part will be kept distinct. This is made formal in the next definition for the case of (unlabelled multi-)graphs.

Definition 3.2 (Marked Graph) *A* marked graph *is a pair $G_{\supseteq} = \langle G, G'\rangle_{\supseteq}$ of graphs G and G' such G' is a sub-graph of G, i.e. $G \subseteq G'$. A* marked graph mapping *between marked graphs $G_{\supseteq} = \langle G, G'\rangle_{\supseteq}$ and $H_{\supseteq} = \langle H, H'\rangle_{\supseteq}$ is a pair of graph morphisms $f_{\supseteq} = \langle f\colon G \to H, f'\colon G' \to H'\rangle$ such that $\imath_H \circ f' = f \circ \imath_G$ is satisfied where $\imath_H\colon H' \rightarrowtail H$ and $\imath_G\colon G' \rightarrowtail G$ are the inclusion morphisms and moreover*

(i) *if the image of a node v of G is in the marked part H', then the node itself is marked, i.e. for every node v of G, if $f_V(v) \in H'$ then $v \in G'$, and*

(ii) *the same holds for all edges e in G, i.e. $f_E(e) \in H'$ implies $e \in G'$.*

A marked graph mapping $f_{\supseteq} = \langle f, f'\rangle$ is an inclusion mapping *if f and f' are inclusion morphisms.*

To prepare the definition of the category of reflected monos, we give a categorical characterization of the category of marked graphs and mappings. For this we recall that, given a graph H, a sub-graph $H' \subseteq H$, and a morphism $f\colon G \to H$ in the category of graphs and graph morphisms, the (natural choice of a) pullback of the co-span $G \xrightarrow{f} H \xleftarrow{\imath} H'$ is the span $G \xleftarrow{\imath} f^{-1}(H') \xrightarrow{f\restriction_{H'}} H'$, giving rise to the pullback square $\begin{smallmatrix} f^{-1}(H') & \to & H' \\ \downarrow & & \downarrow \\ G & \to & H \end{smallmatrix}$ where $f^{-1}(|H'|) = \{h \in |G| \mid f(h) \in |H'|\}$ is the pre-image of $|H'|$, and the graph morphism $f\restriction_{H'}\colon f^{-1}(H') \to H'$ is the co-domain restriction of $f\colon G \to H$, which maps $x \in f^{-1}(|H'|)$ to $f\restriction_{H'}(x) = f(x) \in |H'|$.

Lemma 3.3 (Marked mappings as pullback squares) *Let $G_{\supseteq} = \langle G, G'\rangle_{\supseteq}$ and $H_{\supseteq} = \langle H, H'\rangle_{\supseteq}$ be marked graphs and $f_{\supseteq} = \langle f, f'\rangle\colon G_{\supseteq} \to H_{\supseteq}$ be a marked graph mapping. Then $G' = f^{-1}(H')$ and $f' = f\restriction_{H'}$, which means that $G \xleftarrow{\imath} G' \xrightarrow{f'} H'$ is*

a pullback of $G \xrightarrow{f} H \xhookleftarrow{\iota} H'$*, giving rise to the pullback square* $\begin{smallmatrix} G' & \to & H' \\ \downarrow & \lrcorner & \downarrow \\ G & \to & H \end{smallmatrix}$.

Proof. First we show that $G' = f^{-1}(H')$. To show that $G' \subseteq f^{-1}(H')$, let $x \in |G'|$; then $f(x) = f'(x) \in |H'|$, i.e. $x \in f^{-1}(|H'|)$. To prove the converse, let $x \in f^{-1}(|H'|)$, i.e. $f(x) \in |H'|$, whence also $x \in G'$ by Definition 3.2. Having shown this, the equation $f' = f\restriction_{H'}$ follows immediately. □ □

Marked graphs and their mappings satisfy a certain reflection property, namely for any given mapping $\langle f, f' \rangle \colon \langle G, G' \rangle_{\supseteq} \to \langle H, H' \rangle_{\supseteq}$, the marked part G' can be recovered from its image, i.e. the equation $|G'| = f^{-1}(f(|G'|))$ holds.

4 A categorical approach to secrecy

Having introduced the protected links calculus to motivate the notion of marked graph, we now set out to lift the latter notion to an abstract level, namely the so-called categories of reflected monos [10]. This is followed by suggestions for the description of secrecy related concepts using category theoretical language. Finally we discuss possibilities to verify secrecy properties on this abstract level.

4.1 Categories of reflected monos

Using Lemma 3.3, the idea of the category of marked graphs and their mappings can easily transferred to any category, by the reflected monos construction. This lemma at the same time illustrates the main difference between categories of reflected monos and the arrow category $\mathbb{G}^{\to}$, more precisely to the full subcategory of the arrow category $\mathbb{G}^{\to}$ having the class of all $\mathbb{G}$-monomorphisms as objects.

Definition 4.1 (Reflected Monos) *Let* $\mathbb{C}$ *be a category with pullbacks along monomorphisms, i.e. for each co-span* $A \xrightarrow{f} D \xhookleftarrow{m} M$ *with monic* m*, a pullback span* $A \xhookleftarrow{n} N \xrightarrow{g} M$ *exists, yielding a pullback square* $\begin{smallmatrix} A & \leftarrow & N \\ \downarrow & \lrcorner & \downarrow \\ D & \leftarrow & M \end{smallmatrix}$.

Then the category of reflected $\mathbb{C}$-monos*, written* $\mathrm{RMon}(\mathbb{C})$*, has* $\mathbb{C}$*-monomorphisms* $A' \rightarrowtail A$ *as objects and an* $\mathrm{RMon}(\mathbb{C})$*-morphism* $f_{\hookleftarrow} \colon (A' \rightarrowtail A) \to (B' \rightarrowtail B)$ *is a pair* $f_{\hookleftarrow} = \langle f \colon A \to B, f' \colon A' \to B' \rangle$ *of* $\mathbb{C}$*-morphisms such that* $A \xhookleftarrow{a} A' \xrightarrow{f'} B'$ *is a pullback of* $A \xrightarrow{f} B \xhookleftarrow{b} B'$*, yielding a pullback square* $\begin{smallmatrix} A' & \to & B' \\ \downarrow & \lrcorner & \downarrow \\ A & \to & B \end{smallmatrix}$.

The category of reflected monos $\mathrm{RMon}(\mathbb{C})$ comes equipped with a forgetful functor $\lbag \_ \rbag \colon \mathrm{RMon}(\mathbb{C}) \to \mathbb{C}$ which maps each monomorphism $A' \rightarrowtail A$ to A, i.e. $\lbag a \rbag = A$, and each morphism $f_{\hookleftarrow} = \langle f \colon A \to B, f' \colon A' \to B' \rangle$ to f, i.e. $\lbag \langle f, f' \rangle \rbag = f$.

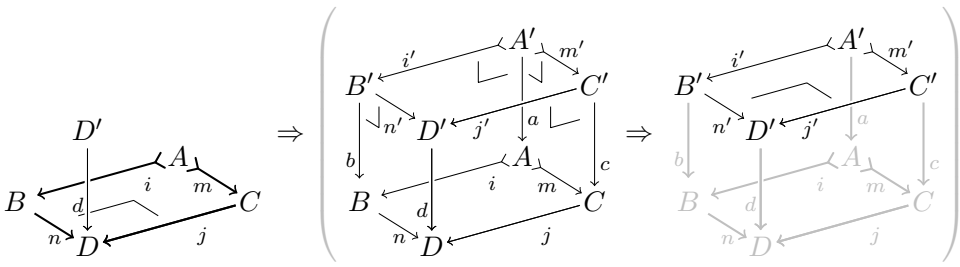
4.2 Reflected monos in weakly adhesive categories

We will now recapitulate the notion of weakly adhesive category presented in [2], which provides a framework that is suitable for double pushout rewriting, and moreover is compatible with the RMon construction, a fact which will be made precise in Proposition 4.4. Weakly adhesive categories generalize adhesive categories [13] and are closely related to weak adhesive HLR categories [7]. The advantage of this weaker notion of adhesivity lays in the fact that it captures additional examples, e.g. those presented in [7], which are of practical relevance.

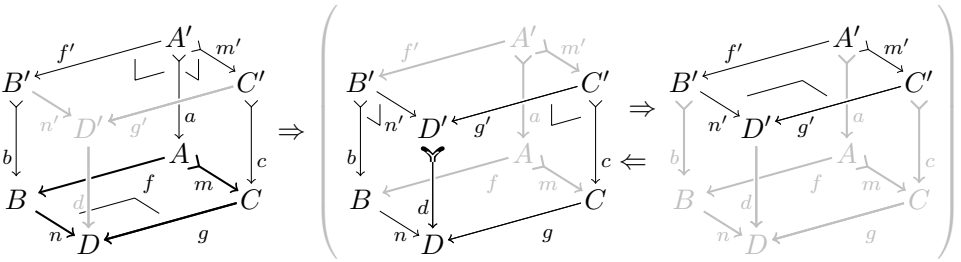
Definition 4.2 (Weakly adhesive category) *A category is* weakly adhesive *if*

(i) *pullbacks along monomorphisms exist and also pushouts along monomorphisms exist, i.e. for each span* $B \leftarrow f - A \rightarrowtail m \rightarrow C$ *with monic* m*, a pushout* $B - n \rightarrow D \leftarrow g - C$ *exists, yielding a* pushout square $\begin{smallmatrix} B & \leftarrow & A \\ \downarrow & & \downarrow \\ D & \leftarrow & C \end{smallmatrix}$*;*

(ii) *pushouts of pairs of monomorphisms are universal (or stable under pullback), i.e. in each commutative cube over a pushout square* $\begin{smallmatrix} B & \leftarrow & A \\ & \searrow & \\ & D & \leftarrow & C \end{smallmatrix}$ *having pullback squares as lateral faces as shown in the middle diagram in the display below, the top face is a pushout square;*



(iii) *pushouts along monomorphisms are* mono-universal *and* converse mono-universal: *in each commutative cube on top of a pushout square* $\begin{smallmatrix} B & \leftarrow & A \\ & \searrow & \\ & D & \leftarrow & C \end{smallmatrix}$ *as in the left diagram in the display below, with pullback squares as back faces and the "corner"-arrows* b *and* c *monic, its* top face is a pushout square *if and only if* the front faces are pullback squares and the morphism d is monic.



Note that adhesive categories [13], apart from having all pullbacks, satisfy the simpler (and stronger) version of Condition iii that does not contain any conditions on the vertical morphisms a, b, c and d. Condition iii is equivalent to the requirement that pushouts along monomorphisms are *hereditary* in the sense of [11]. Finally, the subtle difference to the weak adhesive HLR-categories of [7] is that in the definition of the latter, the vertical morphism d into the "tip" of the bottom pushout is required to be monic already in the antecedent, which implies the "top face-front faces"-equivalence. This is also the reason why the proof of Proposition 4.4 does carry over to weak adhesive HLR-categories. The exact relation among weakly adhesive, adhesive, and weak adhesive HLR categories is also discussed in [2].

An example of a category that is weakly adhesive but not adhesive and hence deserves being mentioned, is the category of undirected multi-graphs. That this

category is weakly adhesive but not adhesive can be shown by adapting the results presented in [15].

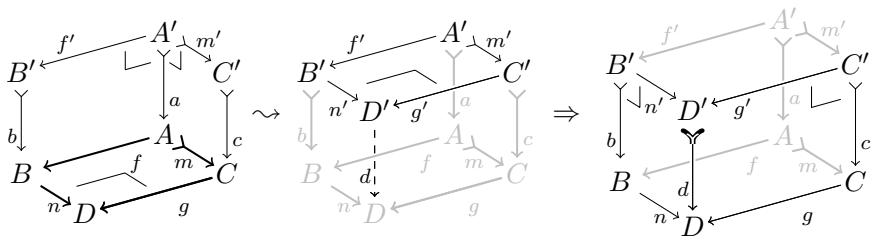
Example 4.3 (Undirected multigraphs) *An* undirected multigraph *is a triple* $M = \langle E, V, c\colon E \to V^{\oplus}\rangle$, *where* $V^{\oplus}$ *is the free commutative monoid over the set of* vertices V *and the* connection function c *assigns to each* edge $e \in E$ *a multiset* $c(e) \in V^{\oplus}$ *of* adjacent vertices. *Finally, given another multigraph* $M' = \langle E', V', c'\colon E' \to V'^{\oplus}\rangle$, *a* multigraph morphism $f\colon M \to M'$ *is a pair of functions* $(f_E\colon E \to E', f_V\colon V \to V')\colon M \to M'$ *such that* $f_V^{\oplus} \circ c = c' \circ f_E$ *where the homomorphism* $f_V^{\oplus}\colon V^{\oplus} \to E^{\oplus}$ *is the freely adjoined monoid homomorphism of the function* $f_V\colon V \to E$.

Now we come to the main technical contribution of this paper, which says that the RMon construction yields a weakly adhesive categories when applied to a weakly adhesive category.

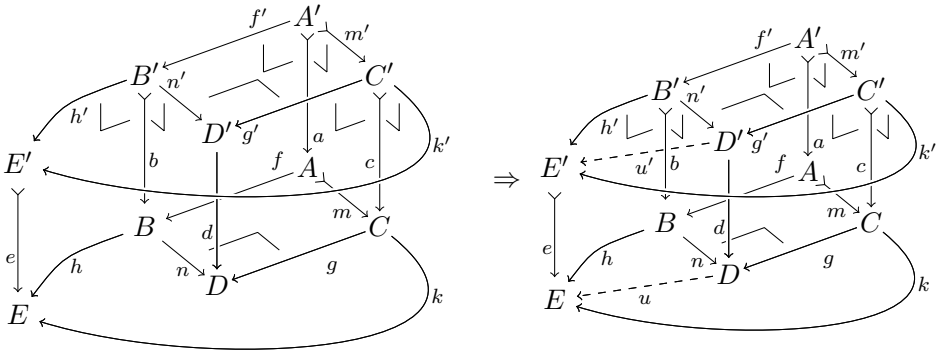
Proposition 4.4 (Weakly adhesive reflected monos) *Let* $\mathbb{C}$ *be a weakly adhesive category. Then the category of reflected monos* $\mathrm{RMon}(\mathbb{C})$ *is weakly adhesive.*

Proof. First one shows that a morphism $f_{\leftarrowtail} = \langle f, f'\rangle\colon a \to b$ in $\mathrm{RMon}(\mathbb{C})$ is monic, if and only if both f and f' are monic in $\mathbb{C}$. Moreover it is straightforward to show that pullbacks along monomorphisms are constructed component-wise.

The main task of the proof consists in showing that also pushouts along monomorphisms are constructed component-wise. Given a morphism $f_{\leftarrowtail} = \langle f, f'\rangle\colon a \to b$ and a monomorphism $m_{\leftarrowtail} = \langle m, m'\rangle\colon a \to c$, then the candidate for the pushout directly arises from the definition of weakly adhesive categories.

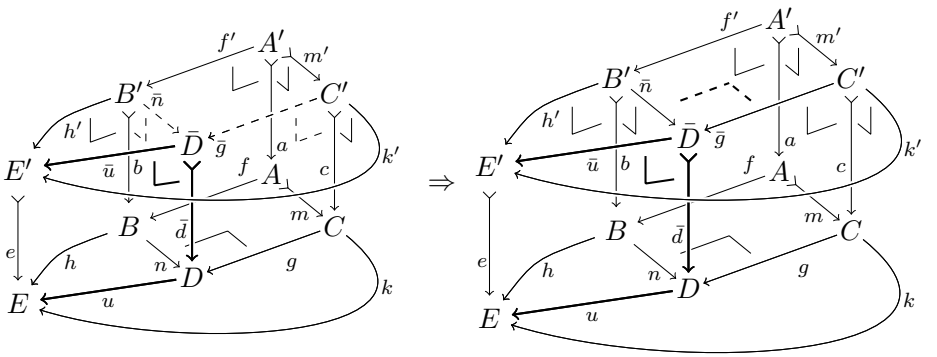


It remains to show that this square satisfies the universal property of pushouts. Hence let $\langle h, h'\rangle\colon b \to e$ and $\langle k, k'\rangle\colon c \to e$ be morphisms such that the left one of the following $\mathbb{C}$-diagrams commutes.



Now we obtain a pair of morphisms $u' \colon D' \to E'$ and $u \colon D \to E$ such that the right one of these diagrams commutes, since $\{n', g'\}$ are jointly epic. It remains to show that the pair u', u actually defines a mediating morphism as it then is easy to show that it is unique.

To derive existence, let $E' \leftarrow \bar{u} - \bar{D} \rightarrowtail \bar{d} \rightarrow D$ be a pullback of $E' \rightarrowtail e \rightarrow E \leftarrow u - D$ as shown in the left one of the following diagrams.



Now there exist unique $\mathbb{C}$-morphisms $\bar{n} \colon B' \to \bar{D}$ and $\bar{g} \colon C' \to \bar{D}$ making this $\mathbb{C}$-diagram commute, which moreover yield pullback squares as illustrated. Then, by the definition of weakly adhesive category, the co-span $B' \rightarrowtail \bar{n} \rightarrow \bar{D} \leftarrow \bar{g} - C'$ is a pushout of the span $B' \leftarrow f' - A' \rightarrowtail m' \rightarrow C'$ as indicated in the right one of the above diagrams. Finally, by straightforward calculation, we obtain an isomorphism $i \colon D' \to \bar{D}$ satisfying both $u' = \bar{u} \circ i$ and $d = \bar{d} \circ i$, i.e. $\langle u, u' \rangle \colon d \to e$ is actually a mediating morphism. □ □

Thus we have laid the foundations for discussing the notions of secret and secrecy on an abstract level. To illustrate the idea, we turn back to Example 2.4, and the discussion of secrecy holes in systems that are modelled using the PL-calculus.

4.3 Secrecy violations revisited

Having established the theoretical framework, namely reflected monos in weakly adhesive categories, we are now ready to study secrecy related phenomena at a more abstract level. In particular we address the questions of how to describe secrecy holes, of when the private/public separation might be violated, and of how persistence of public information can be ensured.

As for the first point, recall that an immediate secrecy hole in a PLC process P is witnessed by a sub-term of the form $\ddot{v}{\gg}\hat{w}$, which corresponds to the fact, that the encoding $[\![P]\!]$ as sketched in section 3 contains a sub-graph of the form $\textcircled{\ddot{v}}\!\!\rightarrowtail\!\!\textcircled{\hat{w}}$. More precisely, the graphical representation of P is actually a marked graph $\langle [\![P]\!], \operatorname{fn} P\rangle_{\supseteq}$ and also $\textcircled{\ddot{v}}\!\!\rightarrowtail\!\!\textcircled{\hat{w}}$ is a marked graph, namely $\langle \textcircled{\ddot{v}}\!\!\rightarrowtail\!\!\textcircled{\hat{w}}, \textcircled{\ddot{v}}\rangle_{\supseteq}$. Moreover P has an (immediate) secrecy hole if and only if the inclusion $\langle \textcircled{\ddot{v}}\!\!\rightarrowtail\!\!\textcircled{\hat{w}}, \textcircled{\ddot{v}}\rangle_{\supseteq} \subseteq \langle [\![P]\!], \operatorname{fn} P\rangle_{\supseteq}$ holds in the category of marked graphs, which is the case if $\textcircled{\ddot{v}}\!\!\rightarrowtail\!\!\textcircled{\hat{w}}$ is a sub-graph of $[\![P]\!]$ and $\ddot{v} \in \operatorname{fn} P$.

Now, the first observation is that the marked graph $\langle \textcircled{\ddot{v}}\!\!\rightarrowtail\!\!\textcircled{\hat{w}}, \textcircled{\ddot{v}}\rangle_{\supseteq}$ is an object of the category of marked graphs that witnesses a *secrecy violation*. Hence on the abstract level, given a category $\mathbb{D}$ and a system with states being modelled as objects of the category $\mathbb{D}$, then a state $S \in \mathbb{D}$ will have a secrecy violation $V \in \mathbb{D}$ if there is a monomorphism $m\colon V \rightarrowtail S$. This leaves us the task of motivating, for the case of $\mathbb{D} = \mathrm{RMon}(\mathbb{C})$, when an object $W' \rightarrowtail w \rightarrow W \in \mathrm{RMon}(\mathbb{C})$ should be considered as a secrecy violation.

4.4 An abstract characterization of secrecy violations

We assume that in applications it is usually clear which parts of a given system are secrecy relevant. This assumption seems harmless in the case of (on-line) banking, where personal identification numbers are clearly secrecy relevant data, and similarly it is obvious that private keys that are used in (asymmetric) cryptographic protocols should remain secret.

The process of pointing out the secrecy relevant part of a state is formalized by a certain functor $\mathcal{R}\colon \mathbb{D} \to \mathrm{RMon}(\mathbb{D})$ which marks the respective part of each object $A \in \mathbb{D}$. It is clear that $\mathcal{R}$ should satisfy the equation $\wr\_\int \circ \mathcal{R} = \mathrm{id}_{\mathbb{D}}$ (see the paragraph after Definition 4.1 for $\wr\_\int\colon \mathrm{RMon}(\mathbb{D}) \to \mathbb{D}$), which means that the application of $\mathcal{R}$ really yields a monomorphism into any given object $A \in \mathbb{D}$. For the case of $\mathbb{D} = \mathrm{RMon}(\mathbb{C})$, the value $\mathcal{R}(a)\colon m \rightarrowtail a$ at an object $A' \rightarrowtail a \rightarrow A \in \mathrm{RMon}(\mathbb{C})$, is a $\mathbb{C}$-pullback square ${}^{A'}_{A}\!\downarrow^{\leftarrow}_{\leftarrow}\!\downarrow^{M'}_{M}$; with some abuse of notation, we often will write ${}^{A'}_{A}\!\downarrow^{\leftarrow}_{\leftarrow}\!\downarrow^{\mathcal{R}A'}_{\mathcal{R}A}$ in such a situation.

The general idea of a secrecy violation in the category $\mathrm{RMon}(\mathbb{C})$ w.r.t. to a suitable functor $\mathcal{R}\colon \mathrm{RMon}(\mathbb{C}) \to \mathrm{RMon}^2(C)$, i.e. $\mathcal{R}$ satisfies $\wr\_\int \circ \mathcal{R} = \mathrm{id}_{\mathrm{RMon}(\mathbb{C})}$, is as follows. Given a state modelled by an object $S' \rightarrowtail s \rightarrow S \in \mathrm{RMon}(\mathbb{C})$ with associated secrecy relevant part $\mathcal{R}(s)\colon m \rightarrowtail s$, resulting in a pullback square ${}^{S'}_{S}\!\downarrow^{\leftarrow}_{\leftarrow}\!\downarrow^{M'}_{M}$, the (completely) private part of the secrecy relevant part M is represented by the

(pseudo-)complement[2] $\underline{m}\colon \underline{M'} \rightarrowtail M$ of $m\colon M' \rightarrowtail M$ (which exists if the subobject poset over M in $\mathbb{C}$ is finite or the category $\mathbb{C}$ is sufficiently rich). Now secrecy relevant information is kept private in s provided that M is actually a coproduct

$$\underline{M'} \rightarrowtail \underline{m} \rightarrow \underbrace{\underline{M'} + \bar{M'}}_{M} \leftarrow \bar{m} \prec \bar{M'}$$

and there exists some $\mathbb{C}$-arrow $\ell\colon M' \rightarrowtail \bar{M'}$ satisfying $m = \bar{m} \circ \ell$. This condition appears to be compatible with the intuition that the (secrecy relevant) private and public data should be located in disjoint parts of the state.

Conversely, a secrecy violation is an object $V' \rightarrowtail v \rightarrow V \in \mathrm{RMon}(\mathbb{C})$ with secrecy relevant part $\mathcal{R}(v)\colon n \rightarrowtail v$ such that either the pseudo-complement $\underline{n}\colon \underline{N'} \rightarrowtail N$ is not a co-product injection, or if $\underline{N'} \rightarrowtail \underline{n} \rightarrow N \leftarrow \bar{n} \prec \bar{N'}$ is a co-product then $\bar{n}$ does not factor through n, i.e. there is no ℓ such that $\bar{n} \circ \ell = n$.

The running example of the paper can be regained by taking the slice category $\mathbb{G}\!\downarrow\!T$ for $\mathbb{C}$. As mentioned in subsection 3.1, taking the "pullback" of a typed graph $A \rightarrowtail t \rightarrow T \in \mathbb{G}\!\downarrow\!T$ along the $\mathbb{G}$-morphism $i\colon \ddot{\bigcirc} \rightarrowtail T$ gives rise to an $\mathrm{RMon}(\mathbb{G}\!\downarrow\!T)$-object as follows: after constructing the $\mathbb{G}$-pullback $\ddot{\bigcirc} \leftarrow t' - A' \rightarrowtail a \rightarrow A$ of $\ddot{\bigcirc} \rightarrowtail i \rightarrow T \leftarrow a - A$, which gives rise to a pullback square $\begin{smallmatrix} A' \\ \ddot{\bigcirc} \end{smallmatrix} \downarrow\!\!\begin{smallmatrix} \urcorner \\ \rightarrow \end{smallmatrix}\!\!\downarrow \begin{smallmatrix} A \\ T \end{smallmatrix}$, we can define the $\mathrm{RMon}(\mathbb{G}\!\downarrow\!T)$ counterpart of $A \rightarrowtail t \rightarrow T$ as $a\colon (i \circ t') \rightarrowtail t$.

This "pullback" construction can be generalized to any monomorphism $\varphi\colon T' \rightarrowtail T$ in $\mathbb{G}$, and actually gives rise to functors $\varphi^{\circledast}\colon \mathbb{G}\!\downarrow\!T \rightarrow \mathrm{RMon}(\mathbb{G}\!\downarrow\!T)$ which act by pulling back along φ followed by post-composition with φ; functoriality is a consequence of the universal properties of pullbacks. In the same way, the "global" secrecy relevant part of an $\mathbb{G}\!\downarrow\!T$-object is obtained by applying the functor $c^{\circledast}\colon \mathbb{G}\!\downarrow\!T \rightarrow \mathrm{RMon}(\mathbb{G}\!\downarrow\!T)$ where c is the "inclusion" $\ddot{v} \triangleright \hat{w} \rightarrowtail T$ in $\mathbb{G}$.

Having the two functors $c^{\circledast}$ and $i^{\circledast}$ at our disposal, we obtain, for each $\mathbb{G}\!\downarrow\!T$-object S, an $\mathrm{RMon}(\mathbb{G}\!\downarrow\!T)$-object $i^{\circledast}(S) = S' \rightarrowtail s \rightarrow S$ with a marked public part, and a (global) secrecy relevant part $c^{\circledast}(S) = M \rightarrowtail \varrho \rightarrow S$; taking the pullback $S' \leftarrow \varrho' \prec M' \rightarrowtail m \rightarrow M$ of $S' \rightarrowtail s \rightarrow S \leftarrow \varrho \prec M$ results in the expected pullback square $\begin{smallmatrix} S' \\ S \end{smallmatrix} \downarrow\!\!\begin{smallmatrix} \ulcorner \\ \leftarrow \end{smallmatrix}\!\!\downarrow \begin{smallmatrix} M' \\ M \end{smallmatrix}$. The technical details of the definition of a suitable functor $\mathcal{R}\colon \mathrm{RMon}(\mathbb{G}\!\downarrow\!T) \rightarrow \mathrm{RMon}^2(\mathbb{G}\!\downarrow\!T)$ based on these pullback constructions are numerous but straightforward.

The fact that the typed graph $\ddot{v} \rightarrowtail \hat{w}$ is a secrecy violation can now be recovered by inspecting the pullback square

$$\begin{array}{ccc} \ddot{v} & \leftarrow & \ddot{v} \\ \downarrow & & \downarrow \\ \ddot{v} \rightarrowtail \hat{w} & \leftarrow & \ddot{v} \rightarrowtail \hat{w} \end{array},$$

since the pseudo-complement $\hat{w} \rightarrowtail \ddot{v} \rightarrowtail \hat{w}$ of the monomorphism $\ddot{v} \rightarrowtail \ddot{v} \rightarrowtail \hat{w}$ is not a co-product injection.

[2] See for example [9] for a definition.

We remark that the proposed notion of secrecy violation is intimitely related with the private/public distinction. Moreover we are "only" studying those secrecy holes, that have a "topological" or *structural* representation of the described kind; hence, though it does not appear to be very strict, this restriction needs to be kept in mind during the system modelling process.

We would also like to remark once more that the goal of this paper is not a proposal of a new "concrete" technique: for practical applications working with typed graphs, i.e. with objects of $\mathbb{G}{\downarrow}T$, is usually sufficient. However, in a manner of speaking, the category $\mathbb{G}{\downarrow}T$ has *too much* structure which is not directly relevant. The aim of this paper now consists in the exploration of ways to describe the "essentials" of secrecy related phenomena using the language of category theory.

4.5 Dynamic aspects

So far we have only spoken about properties of system states, modelled as objects of categories of reflected monos, and gave suggestions of how to determine whether private and public areas are disjoint in a given state. However we might also want to reason about the dynamic evolution of a given system.

For example one might ask whether public information will always stay available, or whether it may eventually be lost; this question concerns the transformation rules of the system. Recall that a rule $R \leftarrow\!\alpha\!\prec K \succ\!\beta\!\rightarrow R$ in a category $\mathbb{C}$ is *non-deleting* if α is an isomorphism. Now, a simple sufficient condition which ensures that a rule $l \leftarrow\!\langle\iota,\iota'\rangle\!\prec k \succ\!\langle\jmath,\jmath'\rangle\!\rightarrow r$ in the category $\mathrm{RMon}(\mathbb{C})$ leaves all public information untouched is the requirement that ι' is an isomorphism since this corresponds to the fact that every application of this rule does not delete anything in the public part of the state to which the rule is applied.

4.6 Verifying secrecy properties

We have argued that (at least in certain cases) it is possible to model secrecy violations as objects which have a structure that models disclosure of private information, the prime example being the marked graph $\langle \textcircled{\scriptsize v} \rightarrow \textcircled{\scriptsize w}, \textcircled{\scriptsize v} \rangle_{\supseteq}$ in the context of the graphical representation of the protected links calculus (see Figure 2). In analogy, suppose that the $\mathrm{RMon}^2(\mathbb{C})$-object ${}^{V'}_{V}{\downarrow}{}^{\mathcal{R}V'}_{\mathcal{R}V}$ models a secrecy violation where $\mathbb{C}$ is a weakly adhesive category and $\mathcal{R}\colon \mathrm{RMon}(\mathbb{C}) \to \mathrm{RMon}^2(\mathbb{C})$ is a "secrecy relevance" functor. Now, as the results of [2] apply (because $\mathrm{RMon}^2(\mathbb{C})$ is weakly adhesive by Proposition 4.4), one can in principle verify that in a system that is modelled by a set of rules $\{q_n = l_n \leftarrowtail k_n \rightarrowtail r_n \mid n = 1, \ldots, m\}$ and an object $S \in \mathrm{RMon}^2(\mathbb{C})$ corresponding to the start state, there is no reachable system state with a secrecy violation, i.e. the system does not have (structural) secrecy holes.

Speaking in Petri net terms, the object ${}^{V'}_{V}{\downarrow}{}^{\mathcal{R}V'}_{\mathcal{R}V}$ corresponds to a marking, the rules $\{q_n = l_n \leftarrowtail k_n \rightarrowtail r_n \mid n \in N\}$ with a start object S in the category $\mathrm{RMon}^2(\mathbb{C})$ together correspond to a marked Petri net. The fact that no reachable object contains ${}^{V'}_{V}{\downarrow}{}^{\mathcal{R}V'}_{\mathcal{R}V}$ is in analogy to the fact that a given "bad" marking is not coverable in the marked Petri net. Further, in the long rung, the results of [2] might lead to

generalizations of the methods for Petri nets and graph transformation systems (cf. [12]), that allow to automatically verify that a "bad" marking is not coverable.

5 Conclusion

Based on the recently proposed *reflected monos*-construction of [10], we have discussed possibilities for abstract, formal counterparts of secrecy related notions, that allow to reason about secret keeping in systems that are faithfully modelled by transformation systems in categories of reflected monos. To ensure that transformation systems in categories of reflected monos can be given in terms of double pushout rewriting [6], we have established that the reflected monos construction preserves weak adhesivity in the sense of [2]. Finally, we have sketched how the results of the latter work might eventually lead to automatic verification of secrecy properties, working on the abstract level of weakly adhesive categories.

Acknowledgement

I would like to thank the referees of the interactive reviewing process for their constructive criticism, and Sander Bruggink, Barbara König, and Arend Rensink for enlightening discussions about the content and the presentation of the paper.

References

[1] Abadi, M. and A. D. Gordon, *A calculus for cryptographic protocols: The spi calculus*, Information and Computation **148** (1999), pp. 1–70.

[2] Baldan, P., A. Corradini, B. König, T. Heindel and P. Sobociński, *Unfolding weakly adhesive grammars*, submitted.

[3] Blanchet, B., *An efficient cryptographic protocol verifier based on prolog rules*, in: *CSFW* (2001), pp. 82–96.

[4] Corradini, A., U. Montanari, F. Rossi, H. Ehrig, R. Heckel and M. Löwe, *Algebraic approaches to graph transformation – part I: Basic concepts and double pushout approach*, in: G. Rozenberg, editor, *Handbook of Graph Grammars* (1997), pp. 163–246.

[5] Ehrig, H. and B. König, *Deriving bisimulation congruences in the DPO approach to graph rewriting with borrowed contexts*, Mathematical Structures in Computer Science **16** (2006), pp. 1133–1163.

[6] Ehrig, H., M. Pfender and H. J. Schneider, *Graph-grammars: An algebraic approach*, in: *14th Annual Symposium on Switching and Automata Theory* (1973), pp. 167–180.

[7] Ehrig, H. and U. Prange, *Weak adhesive High-Level Replacement categories and systems: A unifying framework for graph and petri net transformations*, in: K. Futatsugi, J.-P. Jouannaud and J. Meseguer, editors, *Essays Dedicated to Joseph A. Goguen*, Lecture Notes in Computer Science **4060** (2006), pp. 235–251.

[8] Gardner, P., C. Laneve and L. Wischik, *Linear forwarders*, in: R. Amadio and D. Lugiez, editors, *CONCUR 2003*, Lecture Notes in Computer Science **2761** (2003), pp. 415–430.
URL http://www.wischik.com/lu/research/linfwd.html

[9] Grätzer, G., "Lattice theory," W.H. Freeman and Company, San Francisco, 1971.

[10] Kastenberg, H. and A. Rensink, *Graph attribution through sub-graphs*, submitted.

[11] Kennaway, R., *Graph rewriting in some categories of partial morphisms*, in: H. Ehrig, H.-J. Kreowski and G. Rozenberg, editors, *Graph-Grammars and Their Application to Computer Science*, Lecture Notes in Computer Science **532** (1990), pp. 490–504.

[12] König, B. and V. Kozioura, *Augur—a tool for the analysis of graph transformation systems*, EATCS Bulletin **87** (2005), pp. 125–137, appeared in The Formal Specification Column.

[13] Lack, S. and P. Sobociński, *Adhesive and quasiadhesive categories*, Theoretical Informatics and Applications **39** (2005), pp. 511–546.

[14] Milner, R., J. Parrow and D. Walker, *A calculus of mobile processes, i and ii*, Information and Computation **100** (1992), pp. 1–40, 41–77.

[15] Prange, U., *Algebraic High-Level Nets as Weak Adhesive HLR Categories*, Electronic Communications of the EASST **2** (2007).

[16] Wischik, L., "Explicit Fusions: Theory and Implementation," Ph.D. thesis, Computer Laboratory, University of Cambridge (2001).
URL http://www.wischik.com/lu/research/efti.html

A Basic Definitions

Definition A.1 (Graphs and morphisms, marked graphs and mappings)

An unlabelled multi-graph *is a quadruple* $G = \langle V, E, s, t\rangle$ *where* V *is the set of* nodes, E *is the set of* edges *and* $s, t\colon E \to V$ *are the* source *and* target *functions, respectively, which assign to each edge the* source *and* target *of the edge, respectively. W.l.o.g. we assume that nodes and edges are a pair of disjoint sets, i.e.* $E \cap V = \varnothing$*; then the* carrier of G *can be defined as the set* $|G| := E \cup V$.

Next, a graph morphism between two graphs $G = \langle V, E, s, t\rangle$ *and* $G' = \langle V', E', s', t'\rangle$ *is a pair of functions* $f = \langle f_V\colon V \to V', f_E\colon E \to E'\rangle$ *such that the following two diagrams commute*

$$\begin{array}{ccc} E & \xrightarrow{f_E} & E' \\ {\scriptstyle s}\downarrow & & \downarrow{\scriptstyle s'} \\ V & \xrightarrow[f_V]{} & V' \end{array} \qquad \begin{array}{ccc} E & \xrightarrow{f_E} & E' \\ {\scriptstyle t}\downarrow & & \downarrow{\scriptstyle t'} \\ V & \xrightarrow[f_V]{} & V' \end{array} \quad ,$$

i.e. the two equations $f_E \circ s' = s \circ f_V$ *and* $f_E \circ t' = t \circ f_V$ *are satisfied; such a morphism* f *corresponds to a unique* carrier function $|f|\colon |G| \to |G'|$.

Given a graph $G = \langle V, E, s, t\rangle$*, then another graph* $G' = \langle V', E', s', t'\rangle$ *is a* sub-graph *of* G *if both inclusions* $G' \subseteq G$ *and* $V' \subseteq V$ *hold, and moreover for each edge* $e \in E'$ *the two equations* $s'(e) = s(e)$ *and* $t'(e) = t(e)$ *hold. The fact that* G' *is a sub-graph of* G *is expressed by* $G' \subseteq G$*. If* G' *is a sub-graph of* G *then the obvious* inclusion morphism *is written* $\imath_G\colon G' \rightarrowtail G$*. Graphs and graph morphsims congregate into the* category of graphs $\mathbb{G}$.

We will often identify a graph with its carrier and a graph morphism with the corresponding carrier function.