Abstractions Preserving Parameter Confidentiality [⋆]

Sigrid Gürgens, Peter Ochsenschläger, Carsten Rudolph

Fraunhofer – Institute for Secure Information Technology SIT, Germany
{guergens,ochsenschlaeger,rudolphc}@sit.fraunhofer.de

Abstract. Confidentiality of certain parameters is an essential security requirement for many security sensitive applications. In this paper, conditions for abstractions are formulated in terms of formal language theory to be able to prove parameter confidentiality in an abstract view of a system and then conclude that an *adequate representation* of the property is satisfied in the refined system as well. These conditions essentially depend on an agent's view as well as on an agent's initial knowledge of the system behaviour, which explicitely formalizes assumptions about the system.

1 Introduction

Typically, the well-known concepts of non-interference or information flow control address confidentiality of actions: the occurrence or non-occurrence of certain actions of an agent shall not be deducible for another agent based on what it observes. In the literature there is a variety of formalizations of this concept, Mantel [11] gives a good insight into this topic. The subtle differences between these definitions show the spectrum of this kind of confidentiality.

However, non-interference is not suitable for the specification of security requirements in distributed open systems. Here, it can be assumed that all actions concerned with communication might indeed be visible to malicious agents. Nevertheless, confidentiality is required for data transmitted using these actions. An adequate notion of confidentiality therefore has to provide the flexibility to define confidentiality for arbitrary parameters of the actions. The notion of *parameter-confidentiality* presented in [6] provides this flexibility. Parameter confidentiality formalizes the following property: An agent R that monitors a sequence of actions ω of a system S cannot determine the value of a certain parameter (a certain part of the message, the agent performing the action, etc.) of a specific action or set of actions of the sequence, even if it knows the set of possible parameter values.

It is well known that security of a system is not an add on but must be considered during the whole design process from abstract requirement specifications

[⋆] Part of the work described in this paper was developed within the project Sicari being funded by the German Ministry of Education and Research.

to a concrete realization of the system. To efficiently employ the notion of parameter confidentiality it is therefore necessary to be able to prove satisfaction of this property in an abstract view of a system and then conclude that an *adequate representation* of the property is satisfied in the refined system as well. Using our formal framework for security properties [7] in terms of formal languages and language homomorphisms, this paper gives sufficient conditions for such a top down design and explains by a characteristic example its functionality.

Only a few papers discuss confidentiality in combination with refinement. [9] considers degrees of confidentiality and shows that degrees of functionality and confidentiality are inversely related w.r.t. refinement. [12] shows how refinement can be modified to preserve flow properties. [10] gives conditions for certain refinement operators on stream functions to preserve a kind of explicit confidentiality which does not capture implicit information flow. [8] identifies confidentiality preserving refinements in a probabilistic setting.

Another aspect to be taken into consideration is that security properties can only be satisfied relative to particular sets of underlying system assumptions. Examples include assumptions on cryptographic algorithms, secure storage, trust in the correct behaviour of agents or reliable data transfer. Relatively small changes in these assumptions can result in huge differences concerning satisfaction of security properties. Every model for secure systems must address these issues. However, most existing models rely on a fixed set of underlying assumptions (see for example [3] and [13]). These assumptions are often implicitly given by particular properties of the model framework. Thus, it is very hard to verify whether a particular implementation actually satisfies all of these assumptions. Further, imprecise security assumptions might result in correct but useless security proofs and finally in insecure implementations. Therefore, a model for secure systems needs to provide the means to accurately specify underlying system assumptions in a flexible way.

In order to provide the required flexibility, we extend the system specification by two components: *agents' knowledge* about the global system behaviour and *agents' view*. The knowledge about the system consists of all traces that an agent initially considers possible, i.e. all traces that do not violate any of its assumptions about the system, and the view of an agent specifies which parts of the system behaviour the agent can actually see.

The main result of this paper shows that preserving parameter confidentiality under refinement essentially depends on agents' view as well as on agents' knowledge. The effect of agents' view is also considered in [9] and in [8] whereas agents' knowledge is ignored by all papers mentioned above.

In Section 2 the formalization background is introduced. Section 3 recapitulates the definition of parameter confidentiality from [6]. Sufficient conditions for abstractions preserving parameter confidentiality are given in Section 4 and are discussed by examples in Sections 5 and 6.

2 System behaviour and abstractions

In this section we first give a short summary of the necessary concepts of formal languages to describe system behaviour and abstractions.

The behaviour S of a discrete system can be formally described by the set of its possible sequences of actions (traces). Therefore $S \subseteq \Sigma^*$ holds where Σ is the set of all actions of the system, and Σ^* is the set of all finite sequences of elements of Σ, including the empty sequence denoted by ε. This terminology originates from the theory of formal languages [5], where Σ is called the alphabet (not necessarily finite), the elements of Σ are called letters, the elements of Σ^* are referred to as words and the subsets of Σ^* as formal languages. Words can be composed: if u and v are words, then uv is also a word. This operation is called *concatenation*; especially $\varepsilon u = u\varepsilon = u$. A word u is called a *prefix* of a word v if there is a word x such that $v = ux$. The set of all prefixes of a word u is denoted by $\mathrm{pre}(u)$; $\varepsilon \in \mathrm{pre}(u)$ holds for every word u.

Formal languages which describe system behaviour have the characteristic that $\mathrm{pre}(u) \subseteq S$ holds for every word $u \in S$. Such languages are called *prefix closed*. System behaviour is thus described by prefix closed formal languages. Labeled transition systems are uniquely determined by their sets of labeled paths, which are prefix closed languages. So our method will apply to specifications with an interleaving semantics based on labeled transition systems, as for example any kind of communicating automata.

Different formal models of the same application/system are partially ordered with respect to different levels of abstraction. Formally, abstractions are described by so called alphabetic language homomorphisms. These are mappings $h^* : \Sigma^* \longrightarrow \Sigma'^*$ with $h^*(xy) = h^*(x)h^*(y)$, $h^*(\varepsilon) = \varepsilon$ and $h^*(\Sigma) \subseteq \Sigma' \cup \{\varepsilon\}$ which implies $h^*(S) \subseteq (\Sigma')^*$. So they are uniquely defined by corresponding mappings $h : \Sigma \longrightarrow \Sigma' \cup \{\varepsilon\}$. In the following we denote both the mapping h and the homomorphism h^* by h.

The example In order to illustrate our approach using an example as simple as possible, in the following we introduce an artificial price offer–order example (which is not supposed to represent appropriate security requirements for any realistic application). In the project CASENET [2, 4] funded by the European Commission our approach was successfully applied to real life e-government and e-business applications.

The scenario for the example system consists of a set of two users $\boldsymbol{\mathcal{U}} = \{U, V\}$ and a set of two service providers $\boldsymbol{\mathcal{S}} = \{S, T\}$. The following actions can occur in the example system: a service provider sends a price offer for a certain service to a particular user, which is then received by the user. Subsequently, the user can place an order which is in turn received by the service provider. The price is assumed to be the critical parameter to remain confidential. For simplicity, we assume that only two prices are possible. Therefore, the set of prices is $\boldsymbol{M} = \{\mathit{cheap}, \mathit{exp}\}$. For user $\mathit{USER} \in \boldsymbol{\mathcal{U}}$, service provider $\mathit{SP} \in \boldsymbol{\mathcal{S}}$ and $\mathit{price} \in \boldsymbol{M}$, the actions of the system are *send-offer(SP,USER,price)*,*rec-offer(USER,SP,price)*, *send-order(USER,SP,price)* and *rec-order(SP,USER,price)*. The first parameter denotes the agent executing the particular action, the second parameter the agent the action is associated with. Note that throughout the paper we denote sets, homomorphisms etc. that belong to our example by expressions in bold face. Thus $\boldsymbol{\Sigma}$ denotes the set of all possible actions of the example, while Σ indicates an arbitrary alphabet.

We will now define an abstraction of this system. Here we consider the same sets of agents and prices, but a reduced set of possible actions.

$$\boldsymbol{\Sigma'} = \bigcup_{\substack{USER\in\boldsymbol{\mathcal{U}}\\ SP\in\boldsymbol{\mathcal{S}},price\in\boldsymbol{M}}} \{\mathit{send\text{-}offer}(SP, USER, price), \mathit{rec\text{-}offer}(USER, SP, price)\}$$

The language homomorphism $\boldsymbol{h} : \boldsymbol{\Sigma}^* \longrightarrow \boldsymbol{\Sigma'}^*$ simply removes all actions for sending and receiving orders:

Let $\boldsymbol{\Sigma'}$ and $\boldsymbol{\Sigma}$ be as defined above. Then we define a homomorphism $\boldsymbol{h}$: $\boldsymbol{\Sigma}^* \longrightarrow \boldsymbol{\Sigma'}^*$ by

$$\boldsymbol{h}(a) = \begin{cases} a \text{ if } a \in \boldsymbol{\Sigma'} \\ \varepsilon \text{ else} \end{cases}$$

Figure 1 illustrates the effect of applying $\boldsymbol{h}$ to one particular sequence in $\boldsymbol{\Sigma}^*$.

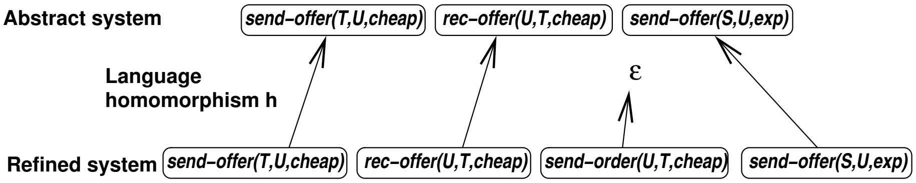

Fig. 1. Application of $\boldsymbol{h}$ to a specific sequence in $\boldsymbol{\Sigma}^*$

In the remaining sections of the paper we will formally define confidentiality of the price in the abstract system and then show that this property is satisfied in the abstract system and that $\boldsymbol{h}$ preserves this property. Thus, we are able to conclude that confidentiality of the price is satisfied in the more complex (refined) system as well.

3 Parameter confidentiality

This section gives a brief introduction to parameter confidentiality as introduced in [6].

3.1 Agents' view and knowledge about the global system behaviour

We will now explain in more detail our approach to specify agents' view and knowledge about the system and their relations. The specification of the desired system behaviour generally does not include behaviour of malicious agents which has to be taken into account in open systems. An approach which is frequently used for the security analysis of cryptographic protocols is to extend the system specification by explicit specification of malicious behaviour. However, in general malicious behaviour is not previously known and one may not be able to adequately specify all possible actions of dishonest agents. In our approach, the explicit specification of agents' knowledge about system and environment allows to discard explicit specification of malicious behaviour. Every behaviour which is not explicitly excluded by some W_P is allowed.

We consider all agents' knowledge sets to be part of the system specification. As explained in the introduction, agent P's knowledge $W_P \subseteq \Sigma^*$ about the global system behaviour contains all traces that P assumes to be possible. We may assume for example that a message that was received must have been sent before. Thus an agent's W_P will contain only those sequences of actions in which a message is first sent and then received.

Care must be taken when specifying the sets W_P for all agents P in order not to specify properties that are not guaranteed by verified system assumptions. In a setting for example where we assume one time passwords are used, if P trusts Q, W_P contains only those sequences of actions in which Q sends a certain password only once. However, if Q cannot be trusted, W_P will also contain sequences of actions in which Q sends a password more than once.

Denoting a system containing malicious behaviour by S and the correct system behaviour by S_C, we assume $S_C \subseteq S \subseteq \Sigma^*$. We further assume $S \subseteq W_P$, i.e. every agent considers the system behaviour to be possible. Security properties can now be defined relative to W_P.

The set W_P describes what P knows initially. However, in a running system P can learn from actions that have occurred. Satisfaction of security properties obviously also depends on what agents are able to learn. After a sequence of actions $\omega \in S$ has happened, every agent can use its *local view* of ω to determine the sequences of actions it considers to be possible. In order to determine what is the local view of an agent, we first assign every action to exactly one agent. Thus $\Sigma = \dot{\bigcup}_{P \in \mathbb{P}} \Sigma_{/P}$ (where $\mathbb{P}$ denotes the set of all agents, $\Sigma_{/P}$ denotes all actions performed by agent P, and $\dot{\bigcup}$ denotes the disjoint union). The homomorphism $\pi_P : \Sigma^* \to \Sigma^*{}_{/P}$ defined by $\pi_P(x) = x$ if $x \in \Sigma_{/P}$ and $\pi_P(x) = \varepsilon$ if $x \in \Sigma \setminus \Sigma_{/P}$ formalizes the assignment of actions to agents and is called the *projection* on P.

The projection π_P is the correct representation of P's view of the system if all information about an action $x \in \Sigma_{/P}$ is available for agent P and P can only see its own actions. In this case P's local view of the sequence of actions *send-offer(P,Q,price) rec-offer(Q,P,price)* for example is *send-offer(P,Q,price)*. However, P's view may be finer. For example it may additionally note other agents' actions without seeing the messages sent and received, respectively. In this case, P's local view of ω will be equal to *send-offer(P,Q,price) rec-offer(Q)*. P's local view may also be coarser than π_P. In a system the actions of which are represented by a triple (*global state, transition label, global successor state*), although seeing its own actions, P will not be able to see the other agents' state. Thus, we generally denote the local view of an agent P on Σ by λ_P.

For a sequence of actions $\omega \in S$ and agent $P \in \mathbb{P}$, $\lambda_P^{-1}(\lambda_P(\omega)) \subseteq \Sigma^*$ is the set of all sequences that look exactly the same from P's local view after ω has happened. Depending on its knowledge about the system S, underlying security mechanisms and system assumptions, P does not consider all sequences in $\lambda_P^{-1}(\lambda_P(\omega))$ possible. Thus it can use its knowledge to reduce this set: $\lambda_P^{-1}(\lambda_P(\omega)) \cap W_P$ describes all sequences of actions P considers to be possible when ω has happened. As this set is frequently used in this paper, we introduce the following abbreviation: $\Lambda_P(\omega, W_P) = \lambda_P^{-1}(\lambda_P(\omega)) \cap W_P$.

The set $\Lambda_P(\omega, W_P)$ is similar to the possible worlds semantics that have been defined for authentication logics in the context of cryptographic protocols [1, 14]. Our notion is more general because for authentication logics λ_P and W_P are fixed for all systems, whereas in our approach they can be defined differently for different systems.

Our approach to define agents' local view and system knowledge is the basis for the framework of security requirements introduced in [7].

Our example We now define local view and knowledge about the system for agents of the abstract version of the example system introduced in Section 2. We assume that each user and service provider can only see its own actions.

For every agent $P \in \boldsymbol{\mathcal{U}} \cup \boldsymbol{\mathcal{S}}$ we define a homomorphism $\boldsymbol{\lambda'_P} : \boldsymbol{\Sigma'^*} \longrightarrow \boldsymbol{\Sigma'^*}$ by

$$\boldsymbol{\lambda'_P}(a) = \begin{cases} a \text{ if } a \in \boldsymbol{\Sigma'}_{/\boldsymbol{P}} \\ \varepsilon \text{ else} \end{cases}$$

We further assume that the knowledge sets $\boldsymbol{W'_P}$ of the agents $P \in \boldsymbol{\mathcal{U}} \cup \boldsymbol{\mathcal{S}}$ in the abstract system are only restricted by an assumption about the communication: No agent considers a sequence of actions possible in which an offer is received without having been sent. Thus, we have the following knowledge sets for $P \in \boldsymbol{\mathcal{U}} \cup \boldsymbol{\mathcal{S}}$:

$$\boldsymbol{W'_P} = \boldsymbol{\Sigma'^*} \setminus \bigcup_{\substack{USER \in \boldsymbol{\mathcal{U}} \\ SP \in \boldsymbol{\mathcal{S}}, price \in \boldsymbol{M}}} (\boldsymbol{\Sigma'} \setminus \{\textit{send-offer}(SP, USER, price)\})^* \{\textit{rec-offer}(USER, SP, price)\} \boldsymbol{\Sigma'^*}$$

$\boldsymbol{W'_P}$ does not contain sequences of actions that start with a sequence without an action *send-offer*$(SP, USER, price)$, then continues with an action *rec-offer*$(USER, SP, price)$, and finally ends with any sequence of actions in $\boldsymbol{\Sigma^*}$.

Based on these knowledge sets, the system behaviour of the abstract system can now be defined as $\boldsymbol{S'} = \bigcap_{P \in \boldsymbol{\mathcal{U}} \cup \boldsymbol{\mathcal{S}}} \boldsymbol{W'_P} = \boldsymbol{W'_P}$ for all $P \in \boldsymbol{\mathcal{U}} \cup \boldsymbol{\mathcal{S}}$.

In the remainder of the paper, primed notation denotes expressions associated with the abstract system. Corresponding expressions for the refined system appear unprimed.

3.2 Formalizing parameter confidentiality

For the example system, we want to specify the property that the price offered to U is confidential from V's perspective. Various aspects are included in our definition. First, we have to consider an agent's local view and its knowledge about the system, as explained in Section 3.1. Then we need to identify the actions in which the price shall be confidential. In our example, only the actions where a price is sent to and received by U are of interest. Thus, we disregard all other actions, i.e. we map them with a suitably chosen homomorphism μ onto the empty word. From those actions not mapped onto ε, μ extracts the parameter to be confidential that occurs in the action and associates it with the "type" of the action. The type consists of the name of the action and all parameters that are not required to be confidential.

Hence $\mu(\Lambda_R(\omega, W_R))$ is a set of sequences of types of those actions that are of interest with respect to parameter confidentiality, paired with the respective parameter values being possible from R's local view.

If Σ_t denotes a set of types of parameter occurrences and M denotes a set of parameter values then homomorphism $\mu_{\Sigma_t,M} : \Sigma^* \rightarrow (\Sigma_t \times M)^*$ can be used to identify the parameters that shall be confidential. For simplicity we write μ if the related parameter set and the types are obvious.

Our example For $SP \in \boldsymbol{\mathcal{S}}$ and $\textit{price} \in \boldsymbol{M}$, the homomorphism $\boldsymbol{\mu'}$ for the abstract example system can be defined as follows:

$\boldsymbol{\mu'}(\textit{send-offer}(SP, V, \textit{price})) = \varepsilon$
$\boldsymbol{\mu'}(\textit{rec-offer}(V, SP, \textit{price})) \quad = \varepsilon$
$\boldsymbol{\mu'}(\textit{send-offer}(SP, U, \textit{price})) = (\textit{send-offer}(SP, U), \textit{price})$
$\boldsymbol{\mu'}(\textit{rec-offer}(U, SP, \textit{price})) \quad = (\textit{rec-offer}(U, SP), \textit{price})$

This homomorphism $\boldsymbol{\mu'}$ extracts the parameter *price* from all *send-offer* actions for and *rec-offer* actions by user U, respectively. Consequently, the type set for the example is given as $\boldsymbol{\Sigma_t} = \{\textit{send-offer}(SP, U), \textit{rec-offer}(U, SP)\}$.

Our aim is now to formalize that $\boldsymbol{\mu'}(\boldsymbol{\Lambda'_V}(\omega, \boldsymbol{W'_V}))$ contains *all possible parameter values*. What are the possible combinations of parameters is the last aspect that needs to be specified. In general it is obviously sufficient if all combinations of parameters are possible. However, in many cases interdependencies between different actions are publicly known and consequently, such a strong requirement would be impossible to be satisfied. In these cases it is reasonable to restrict the combinations of parameters. In our example we assume reliable transmission of messages and therefore $\textit{rec-offer}(USER, SP, \textit{price})$ can only occur if $\textit{send-offer}(SP, USER, \textit{price})$ has happened before.

The following language $\boldsymbol{K} \subseteq (\boldsymbol{\Sigma_t} \times \boldsymbol{M})^*$ expresses the different combinations of parameter values regarding offers sent to and received by U which are possible in the example system.

$$\boldsymbol{K} = (\boldsymbol{\Sigma_t} \times \boldsymbol{M})^* \setminus \bigcup_{\substack{SP \in \boldsymbol{\mathcal{S}} \\ \textit{price} \in \boldsymbol{M}}} ((\boldsymbol{\Sigma_t} \times \boldsymbol{M}) \setminus \{(\textit{send-offer}(SP, U), \textit{price})\})^* \{(\textit{rec-offer}(U, SP), \textit{price})\} (\boldsymbol{\Sigma_t} \times \boldsymbol{M})^*$$

Using the homomorphism $p_t : (\Sigma_t \times M)^* \rightarrow \Sigma_t^*$ that denotes the projection on the types, the following definition expresses parameter confidentiality with respect to a particular μ and K.

Definition 1 *[6] Let M be a parameter set, Σ a set of actions, Σ_t a set of types, $\mu : \Sigma^* \to (\Sigma_t \times M)^*$ a homomorphism, and $K \subseteq (\Sigma_t \times M)^*$ with $K \supseteq \mu(W_R)$. Then M is parameter confidential for agent $R \in \mathbb{P}$ with respect to μ and K if for each $\omega \in S$*

$$\mu(\Lambda_R(\omega, W_R)) = p_t^{-1}(p_t(\mu(\Lambda_R(\omega, W_R)))) \cap K$$

The left hand side of the above equation consists of the μ-image of the set of sequences of actions agent R considers possible after having monitored ω, while the right hand side contains all possible combinations of parameters by applying p_t and then p_t^{-1}. Again, the intersection with K removes all sequences of actions from this set R is not required to consider possible because of information on the system it is allowed to know.

In order to provide a reasonable confidentiality property, the occurrence of parameters in K cannot be too restricted. In most cases it is reasonable to require that apart from interdependencies between actions that are allowed to be known by all agents, all parameter values are possible for each action. This property of K is expressed by the so-called *(L,M)-completeness* of K as described in [6] and obviously holds for the above defined $\boldsymbol{K}$.

We can now show that in our abstract example system, agent V does not know more than it is allowed to know about occurrences of parameters in M when monitoring sequences of actions.

Proposition 1 *The set $\boldsymbol{M}$ is parameter confidential for V with respect to $\boldsymbol{\mu'}$ and $\boldsymbol{K}$.*

Proof: We show by contradiction that for all $\omega \in \boldsymbol{S'}$ holds

$\boldsymbol{\mu'}(\boldsymbol{\Lambda'_V}(\omega, \boldsymbol{W'_V}) \supseteq p_t^{-1}(p_t(\boldsymbol{\mu'}(\boldsymbol{\Lambda'_V}(\omega, \boldsymbol{W'_V})))) \cap \boldsymbol{K}$:

Assume we have $\omega \in \boldsymbol{S}$ such that there exists $x \in p_t^{-1}(p_t(\boldsymbol{\mu'}(\boldsymbol{\Lambda'_V}(\omega, \boldsymbol{W'_V})))) \cap \boldsymbol{K}$ with $x \notin \boldsymbol{\mu'}(\boldsymbol{\Lambda'_V}(\omega, \boldsymbol{W'_V})$. Then there exists $\omega_1 \in \boldsymbol{\Lambda'_V}(\omega, \boldsymbol{W'_V})$ with $p_t(\boldsymbol{\mu'}(\omega_1)) = p_t(x)$ but $\boldsymbol{\mu'}(\omega_1) \neq x$.

Further there exists $\omega_2 \notin \boldsymbol{\Lambda'_V}(\omega, \boldsymbol{W'_V})$ with $\boldsymbol{\mu'}\omega_2 \in \boldsymbol{K}$, $\boldsymbol{\mu'}(\omega_2) = x$ and $\omega_2 \in \boldsymbol{\mu'}^{-1}_{\boldsymbol{ID}}(\boldsymbol{\mu'_{ID}(\omega_1)})$, where the mapping $\boldsymbol{\mu'_{ID}} : \boldsymbol{\Sigma'^*} \longrightarrow (\boldsymbol{\Sigma'} \cup \boldsymbol{\Sigma_t})^*$ is defined by

$$\boldsymbol{\mu'_{ID}}(a) = \begin{cases} a & \text{if } \boldsymbol{\mu'}(a) = \varepsilon \\ p_1(\boldsymbol{\mu'}(a)) & \text{else} \end{cases}$$

Sequence ω_2 equals ω_1 except for those parameter values that are extracted by $\boldsymbol{\mu'}$. Such an ω_2 exists, because according to the assumption there have to be combinations of parameter values that do not occur in $\boldsymbol{\Lambda'_V}(\omega, \boldsymbol{W'_V})$.

For all $\delta \in \boldsymbol{\Sigma'^*}$ holds $\boldsymbol{\mu'}(\boldsymbol{\lambda'_V}(\delta)) = \varepsilon$, i.e. every action that can be seen by V is mapped to ε by $\boldsymbol{\mu'}$. Therefore, with $\omega_2 \in \boldsymbol{\mu'_{ID}}^{\boldsymbol{-1}}(\boldsymbol{\mu'_I D(\omega_1)})$ and $\omega_1 \in (\boldsymbol{\lambda'_V})^{\boldsymbol{-1}}(\boldsymbol{\lambda'_V}(\omega))$ follows $\omega_2 \in (\boldsymbol{\lambda'_V})^{\boldsymbol{-1}}(\boldsymbol{\lambda'_V}(\omega))$. As $\omega_2 \notin \boldsymbol{\Lambda'_V}(\omega, \boldsymbol{W'_V})$ it follows that $\omega_2 \notin \boldsymbol{W'_V}$. This means that in ω_2 there exists a *rec-offer* action by user U without a preceeding *send-offer*. This is in contradiction to $\boldsymbol{\mu'}(\omega_2) \in \boldsymbol{K}$. □

4 Preserving parameter confidentiality

This section explains how we conclude from the proof of parameter confidentiality of an abstract view of a system that an *adequate representation* of the property is satisfied in the refined system as well, and constitutes the main contribution of this paper.

For simplicity, we assume that for a system $S \subseteq \Sigma^*$ and homomorphism μ' on $\Sigma'^* \supseteq h(S)$, the property for the concrete system is defined using the homomorphism $\mu = \mu' \circ h$ (where $f \circ g$ denotes the composition of functions f and g). Then the same language K can be used on both levels of abstraction to express the combinations of action types and parameter values. Definition 2 below can be easily transferred to the more general case with different type sets and different homomorphisms. However, the extended definition is more technical while the simplified version is suitable for many realistic scenarios.

Preservation of parameter confidentiality by a homomorphism h is concerned with the parameter values considered possible on the different levels of abstraction. It therefore depends on the local views λ_R and λ'_R as well as the relation between the knowledge sets W_R and W'_R. These relations between the different levels of abstraction are shown in Figure 2.

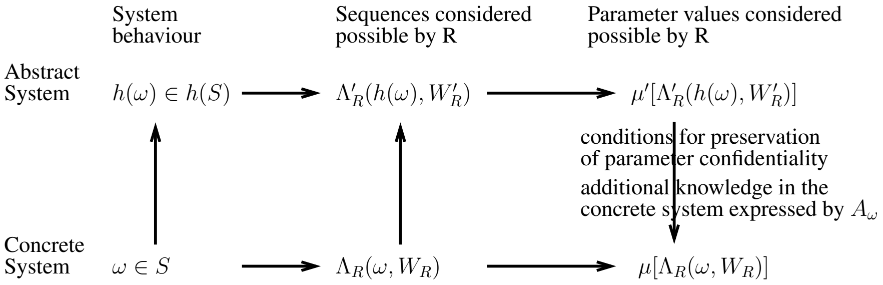
System behaviour
Sequences considered possible by R
Parameter values considered possible by R
Abstract System
$h(\omega) \in h(S)$
$\Lambda'_R(h(\omega), W'_R)$
$\mu'[\Lambda'_R(h(\omega), W'_R)]$
conditions for preservation of parameter confidentiality
additional knowledge in the concrete system expressed by A_ω
Concrete System
$\omega \in S$
$\Lambda_R(\omega, W_R)$
$\mu[\Lambda_R(\omega, W_R)]$

Fig. 2. Preserving parameter confidentiality

Definition 2 formulates a condition on language homomorphisms. We will show that homomorphisms satisfying this condition preserve parameter confidentiality, i.e. that if parameter confidentiality is satisfied in the homomorphic image of a system it is satisfied in the system as well.

Definition 2 *Let $h : \Sigma^* \longrightarrow \Sigma'^*$ and $\mu : \Sigma^* \rightarrow (\Sigma_t \times M)^*$ be language homomorphisms, $S \subseteq \Sigma^*$, $R \in \mathbb{P}$, and λ'_R the local view of agent R in Σ'^*. Then we call the homomorphism h* parameter confidential for R with respect to μ *if for all $\omega \in S$ there exists $A_\omega \subseteq \Sigma_t^*$ such that the following holds:*

$$\mu[\Lambda_R(\omega, W_R)] = \mu'[\Lambda'_R(h(\omega), h(W_R))] \cap p_t^{-1}(A_\omega)$$

So a parameter confidential homomorphism guarantees that the sequences of actions R considers possible after having monitored ω in S correspond to those R considers possible after having monitored $h(\omega)$ in S'. The definition takes into account that usually agents will be allowed to learn additional information in the refined system about the type of actions that have occurred. This additional knowledge is expressed by the set $A_\omega \subseteq \Sigma_t^*$. Applying p_t^{-1} adds all parameter values in M. Therefore, intersection with $p_t^{-1}(A_\omega)$ expresses that agents cannot gain any additional information on *parameter values*. In the case that agents do not learn any new information about action types, we simply have $A_\omega = \Sigma_t^*$ which leads to the much simpler condition $\mu[\Lambda_R(\omega, W_R)] = \mu'[\Lambda'_R(h(\omega), h(W_R))]$.

The next lemma states that if $X \subseteq (\Sigma_t \times M)^*$ is parameter confidential, then the intersection with a set in $(\Sigma_t \times M)^*$ containing all possible parameter values is also parameter confidential.

Lemma 1 *Let $X, K \subseteq (\Sigma_t \times M)^*$ and $A \subseteq \Sigma_t^*$. Then $X = p_t^{-1}(p_t(X)) \cap K$ implies $X \cap p_t^{-1}(A) = p_t^{-1}(p_t[X \cap p_t^{-1}(A)]) \cap K$.*

Proof: With Lemma A.1, $p_t^{-1}(p_t[X \cap p_t^{-1}(A)]) \cap K = p_t^{-1}(p_t(X) \cap A) \cap K$. According to the remark to Lemma A.1, this is equal to $p_t^{-1}(p_t(X)) \cap p_t^{-1}(A) \cap K$, which by assumption is equal to $X \cap p_t^{-1}(A)$. □

In Theorem 1 we now show that parameter confidential homomorphisms indeed preserve parameter confidentiality as defined in Definition 1.

Theorem 1 *Let $S' \subseteq \Sigma'^*$ be parameter confidential for agent $R \in \mathbb{P}$ with respect to some μ' and K. Let furthermore $S \subseteq \Sigma^*$ and homomorphism $h : \Sigma^* \longrightarrow \Sigma'^*$ such that $h(S) \subseteq S'$ and $h(W_R) = W'_R$. If h is parameter confidential with respect to R, then S is parameter confidential for R with respect to $\mu = \mu' \circ h$ and K.*

Proof: S' being parameter confidential with $h(S) \subseteq S'$ and $h(W_R) = W'_R$ implies that $\mu'[\Lambda_R(h(\omega), h(W_R))] = p_t^{-1}(p_t\ [\Lambda_R(h(\omega), h(W_R))]) \cap K$. From applying Lemma 1 with $X = \mu'[\Lambda_R(h(\omega), h(W_R))]$ we conclude $\mu'[\Lambda_R(h(\omega), h(W_R))] \cap p_t^{-1}(A_\omega) = p_t^{-1}(p_t\{\mu'[\Lambda'_R(h(\omega),\ h(W_R))] \cap p_t^{-1}(A_\omega)\}) \cap K$ which is equal to $p_t^{-1}(p_t(\mu[\Lambda_R(\omega, W_R)])) \cap K$, because according to the assumption h is parameter confidential. □

We now introduce a property that is sufficient for a homomorphism to be parameter confidential. In contrast to Definition 2 here we directly compare the image under homomorphism h of what R can learn from ω in the refined system S with what R can learn from the image of ω in the abstract system S' (again taking into account what R is allowed to learn). This can result in easier proofs in some cases.

Theorem 2 *Let $h : \Sigma^* \longrightarrow \Sigma'^*$ be a language homomorphism, $S \subseteq \Sigma^*$, $R \in \mathbb{P}$, and let $\psi' = p_t \circ \mu'$. h is parameter confidential for R with respect to μ if for all $\omega \in S$ there exists $A_\omega \subseteq \Sigma_t^*$ such that*

$$h[\Lambda_R(\omega, W_R)] = [\Lambda'_R(h(\omega), h(W_R)] \cap \psi'^{-1}(A_\omega).$$

Proof: Since $\mu = \mu' \circ h$, we have $\mu[\Lambda_R(\omega, W_R)] = \mu'[h(\Lambda_R(\omega, W_R))]$. By assumption, this is equal to $\mu'[\Lambda'_R(h(\omega), h(W_R)) \cap \psi'^{-1}(A_\omega)]$. This in turn is equal to $\mu'[\Lambda'_R(h(\omega), h(W_R)) \cap \mu'^{-1}(p_t^{-1}(A_\omega))]$ because $\psi' = p_t \circ \mu'$, and with Lemma A.1 follows equality with $\mu'[\Lambda'_R(h(\omega), h(W_R))] \cap p_t^{-1}(A_\omega)$. □

Note that in the case where agents do not learn additional information in the refined system, the sufficient condition of Theorem 2 for homomorphism h for being parameter confidential reduces to

$$h[\Lambda_R(\omega, W_R)] = [\Lambda'_R(h(\omega), h(W_R)],$$

i.e. A_ω is equal to Σ_t^*.

For designing a system using various steps of refinement a natural requirement is that the composition of parameter confidential homomorphisms is again parameter confidential.

Theorem 3 *Let $h : \Sigma^* \longrightarrow \Sigma'^*$ and $g : \Sigma'^* \longrightarrow \Sigma''^*$ be homomorphisms parameter confidential for $R \in \mathbb{P}$ with respect to μ and μ', respectively, and let $\mu'' = \mu' \circ g$. Then $g \circ h$ is also parameter confidential for R with respect to μ.*

Proof: Let λ_R, λ'_R, and λ''_R be the local views of agent R in Σ, Σ', and Σ'', respectively. Since h is parameter confidential, $\mu[\Lambda_R(\omega, W_R)] = \mu'[\Lambda'_R(h(\omega), h(W_R))] \cap p_t^{-1}(A_\omega)$. This is equal to $\mu''[\Lambda''_R(g(h(\omega)), g(h(W_R)))] \cap p_t^{-1}(A'_{h(\omega)}) \cap p_t^{-1}(A_\omega)$ because of parameter confidentiality of g. Equality to $\mu''[\Lambda''_R(g(h(\omega)), g(h(W_R)))] \cap p_t^{-1}(A'_{h(\omega)} \cap A_\omega)$ follows with the remark to Lemma A.1). □

We can now introduce a theorem which is the key for proving parameter confidentiality of a refined system based on parameter confidentiality of an abstraction. The theorem uses a couple of conditions that, though very technical, address quite natural conditions on compatibility of homomorphism and local views in the refined and the abstract system. The theorem refers to the case where an agent does not learn additional information about actions in the refined system (see note of Theorem 2). It essentially states that in this case the homomorphism is parameter confidential if the initial knowledge W_R of agent R in the refined system is just the inverse of what R knows in the abstract system.

Theorem 4 *Let $h : \Sigma^* \rightarrow \Sigma'^*$ be an alphabetic language homomorphism, h surjective, and for $R \in \mathbb{P}$ let λ_R and λ'_R be the homomorphisms describing the* local views *of R on Σ and Σ', respectively. Let furthermore $h'_R : \lambda_R(S) \rightarrow \lambda'_R(S')$ a mapping on S with $\lambda'_R \circ h = h'_R \circ \lambda_R$. Let additionally the following conditions hold:*

1. *For all $t' \in \Sigma'$ with $\lambda'_R(t') = \varepsilon$ there exists $t \in \Sigma$ with $\lambda_R(t) = \varepsilon$ and $h(t) = t'$.*
2. *For all $a \in \Sigma$ with $\lambda'_R(h(a)) \neq \varepsilon$ and for all $t' \in \Sigma'$ with $\lambda'_R(t') = \lambda'_R(h(a))$ there exists $t \in \Sigma$ with $\lambda_R(t) = \lambda_R(a)$ and $h(t) = t'$.*
3. *For all $a \in \Sigma$ with $h(a) \neq \varepsilon$ and $\lambda'_R(h(a)) = \varepsilon$ holds $\lambda_R(a) = \varepsilon$.*

Then $h[\Lambda_R(\omega, h^{-1}(W'_R))] = \Lambda'_R(h(\omega), W'_R)$.

For the (very technical) proof of this theorem we refer the reader to the appendix.

5 Parameter confidentiality of the refined example system

We now consider the refined system as defined in Section 2, which contains an additional set of actions for sending and receiving orders. In Section 3 we have shown that parameter confidentiality of the price offered to U is satisfied in the abstract system. We will now show that this property is preserved by homomorphism $\boldsymbol{h}$ and is therefore satisfied in the refined system as well.

The knowledge sets $\boldsymbol{W_P}$ of the agents $P \in \mathbb{P}$ in the refined system are analogous to those in the abstract system: No agent considers a sequence of actions possible in which an offer and an order, respectively, is received without having been sent before. However, the particular user V knows more about the system. It knows that user U only orders the cheap price, and that he orders only after having received an offer. This additional knowledge is formalized in $\boldsymbol{W_V^2}$ below.

The knowledge sets for agents $P \in \{U, S, T\}$ for the refined system $\boldsymbol{S}$ are defined as follows:

$\boldsymbol{W_P} = \boldsymbol{h^{-1}(W'_P)} \cap \boldsymbol{W_P^1}$

where

$$\boldsymbol{W_P^1} = \boldsymbol{\Sigma^*} \setminus \bigcup_{\substack{USER \in \boldsymbol{\mathcal{U}} \\ SP \in \boldsymbol{\mathcal{S}}, price \in \boldsymbol{M}}} (\boldsymbol{\Sigma} \setminus \{\textit{send-order}(USER, SP, \textit{price})\})^* \{\textit{rec-order}(SP, USER, \textit{price})\} \boldsymbol{\Sigma^*}$$

The knowledge set for agent V for the refined system $\boldsymbol{S}$ with its additional restrictions is defined as follows: $\boldsymbol{W_V} = \boldsymbol{h^{-1}(W'_V)} \cap \boldsymbol{W_V^1} \cap \boldsymbol{W_V^2}$

where $\boldsymbol{W_V^1} = \boldsymbol{W_P^1}$ and

$$\boldsymbol{W_V^2} = (\boldsymbol{\Sigma} \setminus \bigcup_{SP \in \boldsymbol{\mathcal{S}}} \{\textit{send-order}(U, SP, \textit{exp})\})^* \setminus \bigcup_{SP \in \boldsymbol{\mathcal{S}}} (\boldsymbol{\Sigma} \setminus \{\textit{rec-offer}(U, SP, \textit{cheap})\})^* \{\textit{send-order}(U, SP, \textit{cheap})\} \boldsymbol{\Sigma^*}$$

The refined system $\boldsymbol{S} \subseteq \boldsymbol{\Sigma}$ in our example is given as the intersection of all knowledge sets:

$$\boldsymbol{S} = \bigcap_{P \in \mathbb{P}} \boldsymbol{W_P(\Sigma)}$$

We will show that, although V knows more about the concrete system than it knows about the abstract one, with respect to the homomorphism $\boldsymbol{h}$ the two sets are identical. This allows us to make use of Theorem 4.

For specifying parameter confidentiality in $\boldsymbol{S}$, we need to define the local view of V and the type of action(s) we are interested in. The local views of the refined system are analogous to those of the abstract system, i.e. each agent sees its own actions and does not see any (part of) actions of other agents.

The local view $\boldsymbol{\lambda_V} : \boldsymbol{\Sigma} \longrightarrow \boldsymbol{\Sigma_V}$ of agent V is defined as follows:

$$\boldsymbol{\lambda_V}(a) = \begin{cases} a \text{ if } a \in \boldsymbol{\Sigma}_{/V} \\ \varepsilon \text{ else} \end{cases}$$

In the refined system we are again interested in the confidentiality of the price offered to user U, i.e. on both abstraction levels we focus on the same type of actions with respect to parameter confidentiality:

The function $\boldsymbol{\mu}$ that extracts action type and parameter is given by $\boldsymbol{\mu} = \boldsymbol{\mu'} \circ \boldsymbol{h}$, i.e.

$$\begin{aligned}
\boldsymbol{\mu}(\textit{send-offer}(SP, V, \textit{price})) &= \varepsilon \\
\boldsymbol{\mu}(\textit{send-offer}(SP, U, \textit{price})) &= (\textit{send-offer}(SP, U), \textit{price}) \\
\boldsymbol{\mu}(\textit{rec-offer}(V, SP, \textit{price})) &= \varepsilon \\
\boldsymbol{\mu}(\textit{rec-offer}(U, SP, \textit{price})) &= (\textit{rec-offer}(U, SP), \textit{price}) \\
\boldsymbol{\mu}(\textit{send-order}(USER, SP, \textit{price})) &= \varepsilon \\
\boldsymbol{\mu}(\textit{rec-order}(SP, USER, \textit{price})) &= \varepsilon
\end{aligned}$$

Proposition 2 $\boldsymbol{M}$ *is parameter confidential for* V *with respect to* $\boldsymbol{\mu}$ *and* $\boldsymbol{K}$*.*

Proof:

We will show that Proposition 2 holds by applying Theorem 1. Thus we have to show that all conditions of Theorem 1 hold, i.e. that

(i) $\boldsymbol{h}(\boldsymbol{S}) \subseteq \boldsymbol{S'}$ (i.e. $\boldsymbol{S'}$ is indeed an abstract view of $\boldsymbol{S}$),

(ii) $\boldsymbol{h}(\boldsymbol{W_V}) = \boldsymbol{W'_V}$ (i.e. knowledge sets in the refined and the abstract system are consistent),

(iii) and homomorphism $\boldsymbol{h} : \boldsymbol{\Sigma} \longrightarrow \boldsymbol{\Sigma'}^*$ is parameter confidential for V with respect to $\boldsymbol{\mu}$.

Lemma 2 $\boldsymbol{h}(\boldsymbol{S}) \subseteq \boldsymbol{S'}$*, i.e for all* $\omega \in \boldsymbol{S}$ *holds* $\boldsymbol{h}(\omega) \in \boldsymbol{S'}$*.*

Proof: We show the lemma by induction over the length of $\omega \in \boldsymbol{S}$.

Induction basis: Let $\omega = \varepsilon$. Then $\boldsymbol{h}(\omega) = \varepsilon \in \boldsymbol{S'}$.
Induction hypothesis: For $\omega_0 \in \boldsymbol{S}$ holds $\boldsymbol{h}(\omega_0) \in \boldsymbol{S'}$.
Induction step: Let us consider $\omega = \omega_0 a$.

1. $\boldsymbol{h}(a) = \varepsilon$. Then $\boldsymbol{h}(\omega) = \boldsymbol{h}(\omega_0)\boldsymbol{h}(a) = \boldsymbol{h}(\omega_0) \in \boldsymbol{S'}$.
2. $\boldsymbol{h}(a) \neq \varepsilon$,$a = \textit{send-offer}(SP, USER, \textit{price})$ for $USER \in \boldsymbol{\mathcal{U}}$, $SP \in \boldsymbol{\mathcal{S}}$, and $\textit{price} \in \boldsymbol{M}$.
   Then $\boldsymbol{h}(\omega_0) \in \boldsymbol{S'}$ implies $\boldsymbol{h}(\omega_0 a) = \boldsymbol{h}(\omega_0)a \in \boldsymbol{S'}$.
3. $\boldsymbol{h}(a) \neq \varepsilon$,$a = \textit{rec-offer}(USER, SP, \textit{price})$ for $USER \in \boldsymbol{\mathcal{U}}$, $SP \in \boldsymbol{\mathcal{S}}$, and $\textit{price} \in \boldsymbol{M}$.
   Then $\omega_0 a \in \boldsymbol{S}$ implies $\omega_0 \in \boldsymbol{\Sigma}^* \textit{send-offer}(USER, SP, \textit{price})\boldsymbol{\Sigma}^*$.
   It follows that $\boldsymbol{h}(\omega_0) \in \boldsymbol{\Sigma'}^* \textit{send-offer}(USER, SP, \textit{price})\boldsymbol{\Sigma'}^*$
   and therefore $\boldsymbol{h}(\omega_0 a) = \boldsymbol{h}(\omega_0)\boldsymbol{h}(a) = \boldsymbol{h}(\omega_0)a \in \boldsymbol{S'}$. □

For the remaining items (ii) and (iii) we first prove a preliminary consideration concerning the fact that in our particular example the additional restrictions on V's knowledge of the system $\boldsymbol{S}$ have no influence on the image under $\boldsymbol{h}$. This is formulated in the next lemma.

Lemma 3 *For all* $u \in \boldsymbol{h^{-1}(W'_V)}$ *there exists* $v \in \boldsymbol{h^{-1}(W'_V)} \cap \boldsymbol{W^1_V} \cap \boldsymbol{W^2_V}$ *such that* $\boldsymbol{h}(u) = \boldsymbol{h}(v)$.

Proof: The lemma holds trivially for $u \in \boldsymbol{h^{-1}(W'_V)} \cap \boldsymbol{W^1_V} \cap \boldsymbol{W^2_V}$. Thus let $u \in \boldsymbol{h^{-1}(W'_V)} \setminus (\boldsymbol{W^1_V} \cap \boldsymbol{W^2_V})$, i.e. $u \in \boldsymbol{h^{-1}(W'_V)} \setminus \boldsymbol{W^1_V} \cup \boldsymbol{h^{-1}(W'_V)} \setminus \boldsymbol{W^2_V}$. Then $u = u_1 \dots u_l$ contains

1. actions $u_{i_1}, \dots, u_{i_r}$ with $u_{i_x} \in \{\textit{rec-order}(SP, USER, \textit{price}) \mid SP \in \boldsymbol{\mathcal{S}}, USER \in \boldsymbol{\mathcal{U}}, \textit{price} \in \boldsymbol{M}\}$ without the respective action $\textit{send-order}(USER, SP, \textit{price})$ before, or
2. actions $u_{j_1}, \dots, u_{j_s}$ with $u_{j_y} \in \{\textit{send-order}(U, SP, \textit{exp}) \mid SP \in \boldsymbol{\mathcal{S}}\}$, or
3. actions $u_{k_1}, \dots, u_{k_t}$ with $u_{k_z} \in \{\textit{send-order}(U, SP, \textit{cheap}) \mid SP \in \boldsymbol{\mathcal{S}}\}$ without the respective action $\textit{rec-offer}(U, SP, \textit{cheap})$ before.

We define $v := f(u)$ where

$$f(u_i) = \begin{cases} \varepsilon & \text{if } u_i \in \{u_{i_1}, \dots, u_{i_r}, u_{j_1}, \dots, u_{j_s}, u_{k_1}, \dots, u_{k_t}\} \\ u_i & \text{else} \end{cases}$$

Since f maps all actions that can cause u not to be element of $\boldsymbol{W^1_V} \cap \boldsymbol{W^2_V}$ onto ε, $v = f(u) \in \boldsymbol{W^1_V} \cap \boldsymbol{W^2_V}$. Since $u \in \boldsymbol{h}^{-1}(\boldsymbol{W'_V})$ and $\boldsymbol{h}$ keeps all actions $\textit{send-offer}(SP, USER, \textit{price})$ and $\textit{rec-offer}(USER, SP, \textit{price})$, u does not contain actions $\textit{rec-offer}(USER, SP, \textit{price})$ without the respective action $\textit{send-offer}(SP, USER, \textit{price})$ before. f keeps in particular all actions *rec-offer* and *send-offer*, thus $v = f(u)$ also contains no actions *rec-offer* without the respective *send-offer* before, hence is element of $\boldsymbol{h}^{-1}(\boldsymbol{W'_V})$, and therefore element of $\boldsymbol{h}^{-1}(\boldsymbol{W'_V}) \cap \boldsymbol{W^1_V} \cap \boldsymbol{W^2_V}$. □

Lemma 4 $\boldsymbol{h(W_V) = W'_V}$

Proof: $\boldsymbol{h}(\boldsymbol{W_V}) = \boldsymbol{h}(\boldsymbol{h}^{-1}(\boldsymbol{W'_V}) \cap \boldsymbol{W^1_V} \cap \boldsymbol{W^2_V})$. Lemma 3 together with Lemma A.2 yields $\boldsymbol{h}(\boldsymbol{W_V}) = \boldsymbol{h}(\boldsymbol{h}^{-1}(\boldsymbol{W'_V}))$. Surjectivity of $\boldsymbol{h}$ implies $\boldsymbol{h}(\boldsymbol{h}^{-1}(\boldsymbol{W'_V})) = \boldsymbol{W'_V}$ and therefore, $\boldsymbol{h}(\boldsymbol{W_V}) = \boldsymbol{W'_V}$. □

Lemma 5 *For all* $u \in \boldsymbol{\lambda_V^{-1}(\lambda_V(\omega))}$ *there exists* $v \in \boldsymbol{\lambda_V^{-1}(\lambda_V(\omega))} \cap \boldsymbol{W^1_V} \cap \boldsymbol{W^2_V}$ *such that* $\boldsymbol{h}(u) = \boldsymbol{h}(v)$.

Proof: Again, the interesting case to show is $u \in \boldsymbol{\lambda_V^{-1}(\lambda_V(\omega))} \setminus \boldsymbol{W^1_V} \cap \boldsymbol{W^2_V}$. We use the same function f to define $v = f(u)$. With the same argument as above we can deduce that $v \in \boldsymbol{W^1_V} \cap \boldsymbol{W^2_V}$. It remains to show that $v \in \boldsymbol{\lambda_V^{-1}(\lambda_V(\omega))}$, which follows from $\boldsymbol{\lambda_V}(v) = \boldsymbol{\lambda_V}(\omega)$. Since $u \in \boldsymbol{\lambda_V^{-1}(\lambda_V(\omega))}$, $\boldsymbol{\lambda_V}(u) = \boldsymbol{\lambda_V}(\omega)$. Furthermore, for all actions a that are mapped by f onto ε holds $\boldsymbol{\lambda_V}(a) = \varepsilon$ (none of these actions is performed by V), hence $\boldsymbol{\lambda_V} \circ f = \boldsymbol{\lambda_V}$, thus $\boldsymbol{\lambda_V}(v) =$

$\boldsymbol{\lambda_V}(f(u)) = \boldsymbol{\lambda_V}(u) \in \boldsymbol{\lambda_V^{-1}(\lambda_V(\omega))}$. Again we can conclude $v \in \boldsymbol{\lambda_V^{-1}(\lambda_V(\omega))} \cap \boldsymbol{W_V^1} \cap \boldsymbol{W_V^2}$. □

Finally we have to show that homomorphism $\boldsymbol{h}$ is parameter confidential for V.

Lemma 6 *$\boldsymbol{h}$ is parameter confidential for V with respect to $\boldsymbol{\mu}$.*

Proof: According to Theorem 2, the assertion holds if there exists $\boldsymbol{A_\omega}$ such that for all $\omega \in \boldsymbol{S}$, $\boldsymbol{h(\Lambda_V(\omega, W_V))} = [\boldsymbol{\Lambda'_V}(\boldsymbol{h}(\omega), \boldsymbol{h}(\boldsymbol{W_V}))] \cap \boldsymbol{\psi'^{-1}(A_\omega)}$. We show that this equation holds for $\boldsymbol{A_\omega} = \boldsymbol{\Sigma_t^*}$, i.e. we show that

$$\boldsymbol{h(\Lambda_V(\omega, W_V))} = [\boldsymbol{\Lambda'_V}(\boldsymbol{h}(\omega), \boldsymbol{h}(\boldsymbol{W_V}))] \tag{1}$$

Lemma A.2 (see Appendix) states that $f(X \cap Y) = f(X)$ if for all $x \in X$ there exists $y \in X \cap Y$ such that $f(x) = f(y)$. According to Lemma 5, this holds for $X = \boldsymbol{\lambda_V^{-1}(\lambda_V(\omega))}$ and $Y = \boldsymbol{W_V^1} \cap \boldsymbol{W_V^2}$. Since $\boldsymbol{W_V} = \boldsymbol{h^{-1}}(\boldsymbol{W'_V}) \cap \boldsymbol{W_V^1} \cap \boldsymbol{W_V^2}$, this yields $\boldsymbol{h(\Lambda_V(\omega, W_V))} = \boldsymbol{h(\Lambda_V(\omega, h^{-1}(W'_V)))}$. Together with Lemma 4 we can reduce equation 1 to the case where V's knowledge about the system $\boldsymbol{S}$ is just the inverse image of what it knows from $\boldsymbol{\Sigma'}$. It remains to show

$$\boldsymbol{h(\Lambda_V(\omega, h^{-1}(W'_V))) = \Lambda'_V(h(\omega), W'_V)}$$

which holds by Theorem 4. Finally we have to show that our example satisfies the respective properties. We do not provide formal proofs here as the assertions are easy to see.

With $h = \boldsymbol{h}$ and $h_R = \boldsymbol{h}$, obviously $\boldsymbol{\lambda'_R} \circ \boldsymbol{h} = \boldsymbol{h} \circ \boldsymbol{\lambda_R}$. Furthermore, we need to exclude some pathological cases concerning inconsistency between the homomorphism and the agents' local views on both abstraction levels (see Lemma A.5). The homomorphism $\boldsymbol{h}$ just maps part of the refined system $\boldsymbol{S}$ onto ε, the other part of $\boldsymbol{S}$ forms $\boldsymbol{S'}$. Of course this homomorphism is surjective. It is also consistent with the agents' local view, as $\boldsymbol{\lambda'_R}$ is simply $\boldsymbol{\lambda_R}$ restricted to the actions in $\boldsymbol{\Sigma'}$. For the same reason, pathological cases as described by the conditions of Theorem 4 cannot occur. Hence the preconditions of Theorem A.4 are fulfilled and the above equation holds.

Thus $\boldsymbol{h}$ is parameter confidential for V with respect to $\boldsymbol{\mu}$. □

This concludes the proof that M is parameter confidential for V with respect to $\boldsymbol{\mu}$ and K.

This proof seems to be rather complex for this artificial small example. However, the general structure of this proof applies to all examples where conditions for Theorem 4 are satisfied. More complicated systems will only result in more elaborate case differentiations.

6 A different refined system not parameter confidential

In order to give an impression why the condition in Definition 2 fails to hold for a system not being parameter confidential, let us consider a system based on the same set of actions $\boldsymbol{\Sigma}$, the same set of agents, the same knowledge sets, etc., but which has a slightly different local view of agent V. V sees all actions performed by itself and additionally all types of actions performed by some other agent, i.e. it sees who sends to whom which type of message, but it cannot see the actual price used in the message. Although V never sees the price, in this example it can deduce it from the observed behaviour combined with its knowledge about the system. This example shows that often it is not sufficient to protect only the transfer of confidential data. In order to achieve confidentiality the complete system behaviour needs to be considered.

Let $\boldsymbol{\Sigma}$, $\boldsymbol{W_P}$, $\boldsymbol{S}$ etc. be as defined in the previous sections. Then $\tilde{\boldsymbol{\lambda}}_{\boldsymbol{V}} : \boldsymbol{\Sigma} \longrightarrow \boldsymbol{\Sigma_t} \cup \boldsymbol{\Sigma}_{/V}$ with

$$\tilde{\boldsymbol{\lambda}}_{\boldsymbol{V}}(a) = \begin{cases} a & \text{if } a \in \boldsymbol{\Sigma}_{/V} \\ \textit{send-offer}(SP, U) & \text{if } a = \textit{send-offer}(SP, U, \textit{price}) \\ \textit{rec-offer}(U, SP) & \text{if } a = \textit{rec-offer}(U, SP, \textit{price}) \\ \textit{send-order}(U, SP) & \text{if } a = \textit{send-order}(U, SP, \textit{price}) \\ \textit{rec-order}(SP, U) & \text{if } a = \textit{rec-order}(SP, U, \textit{price}) \end{cases}$$

is the local view of agent V in $\boldsymbol{S}$.

Proposition 3 *$\boldsymbol{M}$ is not parameter confidential for V with respect to $\boldsymbol{\mu}$ and $\boldsymbol{K}$ if the local view of $\boldsymbol{V}$ is given by $\tilde{\boldsymbol{\lambda}}_{\boldsymbol{V}}$.*

Proof sketch: We give a short proof sketch for the proposition. The complete proof can be found in the appendix in Section A.2. In order to show that the condition of Definition 2 does not hold we need to show that there exists $\omega \in \boldsymbol{S}$ containing an action *send-offer(S,U,price)* with *price*$\in \{\textit{cheap}, \textit{exp}\}$ where V knows the value of the price. Indeed, such an ω exists because V knows (through its knowledge set W_V) that U only orders the cheap price, and he sees that U receives an offer by S and then orders, thus the price offered by S must be equal to *cheap.*

7 Conclusions

In this paper we gave sufficient conditions to prove parameter confidentiality in an abstract view of a system and then conclude that an *adequate representation* of the property is satisfied in the refined system as well. The notion of parameter-confidentiality was introduced in a preceding paper [6] to specify confidentiality of certain parameters relative to an agent's knowledge about the system, especially knowledge about dependencies between parameter values in different actions.

As it was discussed in a typical example, the formulated conditions essentially depend on an agent's view as well as on an agent's initial knowledge of the systems behaviour, which explicitely formalizes assumptions about the system.

The universality of our formal definitions, based on formal languages and language homomorphisms, allows to apply them to any specification language with a semantics based on labeled transition systems.

The conditions introduced in this paper fit in our design method for security sensitive systems, where security properties are specified independently from the abstraction level. Suitable language homomorphisms map from lower to higher levels of abstraction. Our design method was successfully applied in the project CASENET funded by the European Commission (IST-2001-32446), where it was used to develop real life applications with certain security properties.

A Appendix

A.1 Several technical lemmata and proof of Theorem 4

For arbitrary sets X and Y and $A, C \subseteq X$, $B, D \subseteq Y$ and a mapping $f : X \longrightarrow Y$ we always have the equality $f^{-1}(B) \cap f^{-1}(D) = f^{-1}(B \cap D)$, but only the inclusion $f(A \cap C) \subseteq f(A) \cap f(C)$. However, for particular intersections we have equality:

Lemma A.1 *Let X, Y be arbitrary sets, $f : X \longrightarrow Y$ a mapping, and $A \subseteq X, B \subseteq Y$. Then $f(A \cap f^{-1}(B)) = f(A) \cap B$.*

Proof: $a \in f(A \cap f^{-1}(B))$ is equivalent to the existence of $b \in A$ with $f(b) \in B$ and $a = f(b)$, which in turn is equivalent to $a \in f(A)$ and $a \in B$. □

Lemma A.2 *Let X, Y and Z be arbitrary sets, and $f : X \longrightarrow Z$ a homomorphism. If for all $x \in X$ there exists $y \in X \cap Y$ such that $f(x) = f(y)$ then $f(X \cap Y) = f(X)$.*

Proof:

Let $a \in f(X)$. Then there exists $b \in X$ with $f(b) = a$. With the assumption if follows the existence of some $c \in X \cap Y$ with $f(c) = f(b) = a$. Thus $a \in f(X \cap Y)$. On the other hand, if $a \in f(X \cap Y)$, then $a \in f(X) \cap f(Y) \subseteq f(X)$, thus $a \in f(X)$. □

Lemma A.3 *Let $h : \Sigma^* \to \Sigma'^*$ be an alphabetic language homomorphism and for $R \in \mathbb{P}$ let λ_R and λ'_R be the homomorphisms describing the* local views *of R on Σ a Σ', respectively. If there exists a mapping $h'_R : \lambda_R(S) \to \lambda'_R(S')$ with $\lambda'_R \circ h = h'_R \circ \lambda_R$ on S, then $h(\Lambda_R(\omega, W_R)) \subseteq \Lambda_R(h(\omega), h(W_R))$*

Proof: $x \in h(\Lambda_R(\omega, W_R))$ implies the existence of $y \in W_R$ such that $x = h(y)$ and $\lambda_R(y) = \lambda_R(\omega)$. This in turn implies that there exists $y \in W_R$ with $x = h(y)$ and $h_R(\lambda_R(y)) = h_R(\lambda_R(\omega))$. It follows that there exists $y \in W_R$ with $x = h(y)$ and $\lambda'_R(h(y)) = \lambda'_R(h(\omega))$ which finally implies that $x \in \Lambda_R(h(\omega), h(W_R))$.

Lemma A.4 *Let h, h'_R, λ_R and λ'_R be as defined above. If h is surjective, $h(W_R) = W'_R$, and $\lambda'_R \circ h = h'_R \circ \lambda_R$, then $h[\Lambda_R(\omega, h^{-1}(W'_R))] \subseteq \Lambda'_R(h(\omega), W'_R)$.*

Proof: Replacing W_R in $h[\Lambda_R(\omega, W_R)] \subseteq \Lambda_R(h(\omega), h(W_R))$ of Lemma A.3 by $h^{-1}(W'_R)$ yields $h[\Lambda_R(\omega, h^{-1}(W'_R))] \subseteq \Lambda'_R(h(\omega), h(h^{-1}(W'_R)))$. Since the surjectivity of h implies $h(h^{-1}(W'_R)) = W'_R$, it follows the assertion.

Lemma A.5 *Let $h, \Sigma, \Sigma^*, \lambda_R$ and λ'_R be as defined above. Let furthermore the following conditions hold:*

(1) For all $t' \in \Sigma'$ with $\lambda'_R(t') = \varepsilon$ there exists $t \in \Sigma$ with $\lambda_R(t) = \varepsilon$ and $h(t) = t'$.
(2) For all $a \in \Sigma$ with $\lambda'_R(h(a)) \neq \varepsilon$ and for all $t' \in \Sigma'$ with $\lambda'_R(t') = \lambda'_R(h(a))$ there exists $t \in \Sigma$ with $\lambda_R(t) = \lambda_R(a)$ and $h(t) = t'$.
(3) For all $a \in \Sigma$ with $h(a) \neq \varepsilon$ and $\lambda'_R(h(a)) = \varepsilon$ holds $\lambda_R(a) = \varepsilon$.

Then for all $\omega \in \Sigma^*$ and $x \in \lambda'^{-1}_R(\lambda'_R(h(\omega)))$ holds $h^{-1}(x) \cap \lambda^{-1}_R(\lambda_R(\omega)) \neq \emptyset$.

Proof: We prove the lemma by induction over the length of ω.

Induction basis: Let $\omega = \varepsilon$. Then $\lambda'_R(x) = \varepsilon$ implies $x \in [\Sigma' \cap \lambda'^{-1}_R(\varepsilon)]^*$. By condition (1) it follows the existence of $y \in \Sigma^* \cap \lambda^{-1}_R(\varepsilon)$ with $h(y) = x$ and further $y \in h^{-1}(x) \cap \lambda^{-1}_R(\lambda_R(\varepsilon))$.
Induction hypothesis: The assertion holds for $\omega \in \Sigma^*$ and $x \in \lambda'^{-1}_R(\lambda'_R(h(\omega)))$.
Induction step: Let $\omega \in \Sigma^*$, $a \in \Sigma$ and $\lambda'_R(x) = \lambda'_R(h(\omega)h(a)) = \lambda'_R(h(\omega))\lambda'_R(h(a))$.

1. $\lambda'_R(h(a)) \neq \varepsilon$
   Then there exists $x = u't'v'$ with $t' \in \Sigma'$, $\lambda'_R(u') = \lambda'_R(h(\omega))$, $\lambda'_R(t') = \lambda'_R(h(a))$ and $\lambda'_R(v') = \varepsilon$. Because of the induction hypothesis there exists $u \in \Sigma^*$ with $h(u) = u'$ and $\lambda_R(u) = \lambda_R(\omega)$. Because of condition (2) there exists $t \in \Sigma$ with $h(t) = t'$ and $\lambda_R(t) = \lambda_R(a)$.
   As for the induction basis, it follows that there exists $v \in \Sigma^*$ with $\lambda_R(v) = \varepsilon$ and $h(v) = v'$. As a consequence we have $h(utv) = u't'v' = x$ and $\lambda_R(utv) = \lambda_R(\omega)\lambda_R(a) = \lambda_R(\omega a)$.
2. $\lambda'_R(h(a)) = \varepsilon$
   (a) $h(a) = \varepsilon$
   Then we have $\lambda'_R(x) = \lambda'_R(h(\omega)h(a)) = \lambda'_R(h(\omega))$. The induction hypothesis implies the existence of $y \in \Sigma^*$ with $h(y) = x$ and $\lambda_R(y) = \lambda_R(\omega)$. It follows $h(ya) = x$ and $\lambda_R(ya) = \lambda_R(\omega a)$.
   (b) $h(a) \neq \varepsilon$ and $\lambda'_R(h(a)) = \varepsilon$.
   As above there exists $y \in \Sigma^*$ with $h(y) = x$ and $\lambda_R(y) = \lambda_R(\omega)$. Condition (3) implies $\lambda_R(y) = \lambda_R(\omega a)$.

□

Lemma A.6 *If the preconditions of Lemma A.5 hold, then $\Lambda'_R(h(\omega), W'_R) \subseteq h[\Lambda_R(\omega,\, h^{-1}(W'_R))]$.*

Proof: Let $x \in \Lambda'_R(h(\omega), W'_R)$. By Lemma A.5 there exists $y \in \Sigma^*$ with $h(y) = x$ and $\lambda_R(y) = \lambda_R(\omega)$. Thus it follows $y \in \Lambda_R(\omega, h^{-1}(W'_R))$, hence $x \in h[\Lambda_R(\omega, h^{-1}(W'_R))]$. □

Now Lemma A.6 and Lemma A.4 prove Theorem 4.

A.2 Proof of Proposition 3

In order to show that the condition of Definition 2 does not hold we need to show that there exist s $\omega \in \boldsymbol{S}$ such that for all $A_\omega \subseteq \boldsymbol{\Sigma_t^*}$ holds

$$\boldsymbol{\mu}[\tilde{\boldsymbol{\Lambda}}_{\boldsymbol{V}}(\omega, \boldsymbol{W_V})] \overset{\subset}{\neq} \boldsymbol{\mu'}[\tilde{\boldsymbol{\Lambda}}'_{\boldsymbol{V}}(\boldsymbol{h}(\omega), \boldsymbol{W_V})] \cap \boldsymbol{p_t}^{-1}(A_\omega) \tag{2}$$

Consider $\boldsymbol{\omega} = \textit{send-offer}(S, U, \textit{cheap})\textit{rec-offer}(U, S, \textit{cheap})\textit{send-order}(U, S, \textit{cheap})$. V's local view of this particular sequence of actions is

$\tilde{\boldsymbol{\lambda}}_{\boldsymbol{V}}(\boldsymbol{\omega}) = \textit{send-offer}(S, U)\textit{rec-offer}(U, S)\textit{send-order}(U, S)$

and the set of sequences V considers possible contains only one sequence:

$\tilde{\boldsymbol{\Lambda}}_{\boldsymbol{V}}(\boldsymbol{\omega}, W_V) = \textit{send-offer}(S, U, \textit{cheap})\textit{rec-offer}(U, S, \textit{cheap})\textit{send-order}(U, S, \textit{cheap})$
$= \omega$

The reason for this is that V knows (through his knowledge set W_V) that U only orders the cheap price, and he sees that U receives an offer by S and then orders, thus the price offered by S must be equal to *cheap*.

Thus we have for the left hand side of Equation 2

$\boldsymbol{\mu}[\tilde{\boldsymbol{\Lambda}}_{\boldsymbol{V}}(\boldsymbol{\omega}, \boldsymbol{W_V})] = \mu(\omega) = (\textit{send-offer}(S, U), \textit{cheap})(\textit{rec-offer}(U, S), \textit{cheap})$

For the right hand side of Equation 2 we have

$$\begin{aligned} \boldsymbol{h}(\boldsymbol{\omega}) &= \textit{send-offer}(S, U, \textit{cheap})\textit{rec-offer}(U, S, \textit{cheap}) \\ & \text{thus} \\ \tilde{\boldsymbol{\lambda}}'_{\boldsymbol{V}}(\boldsymbol{h}(\boldsymbol{\omega})) &= \varepsilon \\ & \text{therefore} \\ \tilde{\boldsymbol{\lambda}}'^{-1}_{\boldsymbol{V}}(\tilde{\boldsymbol{\lambda}}'_{\boldsymbol{V}}(\boldsymbol{h}(\boldsymbol{\omega}))) \cap \boldsymbol{W}'_{\boldsymbol{V}} &= (\boldsymbol{\Sigma'} \setminus \boldsymbol{\Sigma'}_{/V})^* \cap \boldsymbol{W}'_{\boldsymbol{V}} \end{aligned}$$

Thus in order to show that $\boldsymbol{h}$ is parameter confidential we would have to show

$(\textit{send-offer}(S, U), \textit{cheap})(\textit{rec-offer}(U, S), \textit{cheap}) = \boldsymbol{\mu'}[(\boldsymbol{\Sigma'} \setminus \boldsymbol{\Sigma'}_{/\boldsymbol{V}})^* \cap \boldsymbol{W}'_{\boldsymbol{V}}] \cap p_t^{-1}(A_\omega)$

for some suitable $A_\omega \subseteq \boldsymbol{\Sigma_t}^*$.

Consequently, the sequence $(\textit{send-offer}(S, U), \textit{exp})(\textit{rec-offer}(U, S), \textit{exp})$ may not be element of $p_t^{-1}(A_\omega)$. However, we either have

$\{(\textit{send-offer}(S, U), \textit{exp})(\textit{rec-offer}(U, S), \textit{exp}),$
$(\textit{send-offer}(S, U), \textit{cheap})(\textit{rec-offer}(U, S), \textit{cheap})\} \subseteq p_t^{-1}(A_\omega) \cap \boldsymbol{\mu'}[(\boldsymbol{\Sigma'} \setminus \boldsymbol{\Sigma'}_{/\boldsymbol{V}})^*]$

that is

$\boldsymbol{\mu}[\tilde{\boldsymbol{\Lambda}}_{\boldsymbol{V}}(\boldsymbol{\omega}, \boldsymbol{W_V})] \overset{\subset}{\neq} p_t^{-1}(A_\omega) \cap \boldsymbol{\mu'}[(\boldsymbol{\Sigma'} \setminus \boldsymbol{\Sigma'}_{/\boldsymbol{V}})^*]$

or we have

$\{(\textit{send-offer}(S, U), \textit{exp})(\textit{rec-offer}(U, S), \textit{exp}),$
$(\textit{send-offer}(S, U), \textit{cheap})(\textit{rec-offer}(U, S), \textit{cheap})\} \cap p_t^{-1}(A_\omega) \cap \boldsymbol{\mu'}[(\boldsymbol{\Sigma'} \setminus \boldsymbol{\Sigma'}_{/\boldsymbol{V}})^*] = \emptyset$

that is

$\boldsymbol{\mu}[\tilde{\boldsymbol{\Lambda}}_{\boldsymbol{V}}(\boldsymbol{\omega}, \boldsymbol{W_V})] \not\subseteq p_t^{-1}(A_\omega) \cap \boldsymbol{\mu'}[(\boldsymbol{\Sigma'} \setminus \boldsymbol{\Sigma'}_{/\boldsymbol{V}})^*]$

So there is no $A_\omega \subseteq \boldsymbol{\Sigma_t}^*$ such that the condition in Definition 2 is satisfied.

References

1. M. Abadi and M.R Tuttle. A Semantics for a Logic of Authentication. In *Tenth Annual ACM Symposium on Principles of Distributed Computing, Montreal, Canada*, pages 201–216, August 1991.

2. I. Agudo, S. Gürgens, and J. Lopez. Casenet: One year later. In *18th IFIP International Information Security Conference 2003*, Athens,Greece, 2003.
3. M. Burrows, M. Abadi, and R. Needham. A Logic of Authentication. *ACM Transactions on Computer Systems*, 8, 1990.
4. CASENET. IST project 2001-32446. http://www.casenet-eu.org/.
5. S. Eilenberg. *Automata, Languages and Machines*. Academic Press, New York, 1974.
6. S. Gürgens, P. Ochsenschläger, and C. Rudolph. Parameter confidentiality. In *Informatik 2003 - Teiltagung Sicherheit*. Gesellschaft für Informatik, 2003.
7. S. Gürgens, P. Ochsenschläger, and C. Rudolph. On a formal framework for security properties. *International Computer Standards & Interface Journal (CSI), Special issue on formal methods, techniques and tools for secure and reliable applications*, 2005.
8. M. Heisel, Pfitzmann A., and Santen T. Confidentiality-preserving refinement. In *Proceedings of the 14th IEEE Computer Security Foundations Workshop*, pages 295–305. IEEE Computer Society Press, 2001.
9. J. Jacob. Basic Theorems About Security. *Journal of Computer Security*, 1(4):385–411, 1992.
10. J. Jürjens. Secrecy-preserving Refinement. In *Formal Methods Europe 2001*, LNCS. SV, 2001.
11. H. Mantel. Possibilistic definitions of security – an assembly kit. In *Proceedings of the IEEE Computer Security Foundations Workshop*, pages 185–199, 2000.
12. H. Mantel. Preserving Information Flow Properties under Refinement. In *IEEE Symposium on Security and Privacy, Oakland*, pages 78–91. IEEE Computer Science, 2001.
13. L. C. Paulson. Proving Properties of Security Protocols by Induction. Technical Report 409, Computer Laboratory, University of Cambridg, 1996.
14. G. Wedel and V. Kessler. Formal Semantics for Authentication Logics. In *Computer Security - Esorics 96*, volume 1146 of *LNCS*, pages 219–241, 1996.