Dynamic Noninterference: Consistent Policies, Characterizations and Verification

Sebastian Eggert, Henning Schnoor, and Thomas Wilke

Institut für Informatik, Christian-Albrechts-Universität zu Kiel, 24098 Kiel, Germany
{eggert|schnoor|wilke}@ti.informatik.uni-kiel.de

Abstract. We study non-interference based security in a dynamic setting, where the security policy may depend on the state of the system. More specifically, we 1. provide new definitions of dynamic non-interference security which conform to the intuitive notion of non-interference and give efficient algorithms to decide whether a given system is secure, and 2. obtain a characterization of secure systems using unwinding relations. Our new definitions are motivated by the fact that previous definitions show counter-intuitive behaviour, as we point out by exhibiting a system that is clearly insecure even though meeting the previous definition. We capture this deficiency by a precise study of consistency in dynamic policies.

1 Introduction

Non-interference [GMS2,GMS4] is a well-studied technique to formalize security of systems that contain trusted and untrusted components. A wide range of different definitions to capture different aspects of non-interference has been proposed [BDRS08], see [vdMZ10] for a comparison of some of these notions. At the core of these definitions lies the idea that interference, or information flow, between different components should be restricted by a security policy, which fixes the pairs of agents that may interfere with each other. This leads to a notion of states that should be “indistinguishable” for an agent operating in the system; a system is secure if in these states, the agent indeed makes the same observations.

The most basic noninterference policy consists of two agents (often also called security domains), $H$ (high) and $L$ (low), and allows interference only from $L$ to $H$, written as $L \rightarrow H$, but not from $H$ to $L$, written $H \not\rightarrow L$. Intuitively, this expresses that $H$ may “see” the actions of $L$, but $L$ must not “see” actions performed by $H$. This property can both be seen as a secrecy property (in the case that $H$ operates on sensitive data that $L$ must not obtain any information about) and as an intrusion prevention property (if $H$ is an untrusted component that should not have any influence on the remainder of the system).

Much of the noninterference literature considers policies as the above, or more generally, policies that are both transitive [Den76] (if an agent $H$ may influence an agent $D$ who in turn may influence another agent $L$, then direct information...
flow between \( H \) and \( L \) is permitted) as well as static—the policy cannot change during the runtime of a system.

For many real-world examples, both of these properties are too restrictive. As a consequence, several generalizations have been studied: Intransitive noninterference [HY87], has been proposed, which gives new semantics to intransitive policies: Their formalism, also called IP-security, can model that, in an extension of the above example, \( H \) may be allowed to influence \( L \), but only if this influence “passes through” another agent \( D \). Hence \( D \) can be used as a safeguard to control and limit the influence that \( H \) has on other parts of the system. Intransitive policies are used to model declassification [MS04]. There is a rich literature on different notions of intransitive noninterference [HY87,Rus92,vdM07].

Similarly, it is widely accepted that to model realistic systems, dynamic noninterference is crucial [BS09,ZM04,Les06]: Here the policy may change during the runtime of a system. This may happen by the addition or removal of users from a system, or a change of user’s privileges. While much of the intransitive noninterference literature concerns state-observed systems, the only paper that we are aware of that studies dynamic noninterference in this setting is [Les06].

![Fig. 1](image1.png)

**Fig. 1.** A distributed program for access control.

![Fig. 2](image2.png)

**Fig. 2.** The transition system of the program in Figure 1.

An example. The system in Figure 1 captures a simple distributed program with procedures for reading and writing on a binary variable \( x \) and an admin who can restrict the write-access on that variable. The writer can flip the value of this variable, given that the policy allows him to interfere with the reader; the reader can always read the value of the variable. This distributed program is presented as a transition system in Figure 2.

In our graphical notation, the transition function is indicated with labelled edges between states, where an action \( w \) stands for a flip action and is performed by the writer \( W \), action \( a \) stands for the deny write-access action and is performed by the admin \( A \). The read action of the reader is omitted since the observation of the reader is always the value of the variable. Ac-
tions not indicated in the graphical representation loop in the current state. The policies are indicated in each state in the obvious way.

The system is secure with respect to the definition in [Les06] However, the system is clearly insecure: The reader observes when a flip action modifies the value of the variable and can conclude whether the admin has performed the deny write-access action. However, the admin is never allowed to interfere with anyone, hence the traces (sequences of actions) $aw$ and $w$ should be indistinguishable for the reader. Since this is not the case, this system must be considered as insecure.

An important point to observe in this example is that dynamic policies can be inconsistent in the following sense: The above policy explicitly allows information flow from $W$ to $R$ in the initial state. However, using this liberty by allowing $R$ to observe the action $w$ performed by $W$ in the initial state by letting this action change $R$’s observation from 0 to 1 contradicts the policy, namely the requirement that $A$ must not interfere with any other agent in the system.

Surprisingly, if we would allow the writer to interfere with the reader in all four states, the system turns out to be insecure with respect to Leslie’s definition, in contrast to the intuition that weakening the security requirements should not turn a secure system into an insecure system. Hence the example shows that the study of dynamic policies exhibits new and subtle issues, which we explore in this paper.

Our results. In this paper, we develop a theory of dynamic noninterference which takes the above-mentioned issues into account. Our contributions are as follows:

1. We provide new and natural definitions for dynamic noninterference, both for the transitive and for the intransitive setting.
2. We give characterizations of policies that are consistent in the above sense.
3. We provide characterizations of our definitions based on unwindings, which are a popular proof technique for interference-based security definitions.
4. We study the complexity of determining whether a given system is secure with respect to a given policy. In the transitive case, this can be answered in nondeterministic logarithmic space (NL). For intransitive noninterference, the problem is fixed-parameter tractable with respect to the number of agents, and solvable in polynomial time of the number of agents is logarithmic in the state space. The general problem is NP-complete.

Our results show significant differences between the transitive and the intransitive case. For once, in the transitive case, the above-mentioned class of consistent policies coincides with uniform policies that have the natural property that each agent always knows (in a precise epistemic sense) the set of agents that may interfere with him. Hence in the transitive case, each policy is equivalent to one that is consistent and uniform. Moreover, security in the transitive case can be characterized with an unwinding relation, which immediately yields the above-mentioned complexity result. In the intransitive case, the situation is

\footnote{We will formally state and discuss the security definition from [Les06] in Section 5.6 after introducing the necessary background.}
more complicated: Again, every policy is equivalent to one which is consistent, but here, not necessarily to a uniform policy. We again obtain an unwinding-based security characterization, but this one requires computing exponentially many relations. However, if we restrict ourselves to uniform policies, we obtain simple unwindings which again lead to very efficient algorithms. Moreover, the class of uniform policies itself also has an unwinding-based characterization and hence can be verified efficiently.

From our unwinding results in the dynamic setting, we obtain as a corollary an unwinding characterization of (static) IP-security. Prior to our results, only an unwinding that was sound, but not complete for IP-security was known [Rus92].

Our new unwinding immediately implies that (static) IP-security can be verified in nondeterministic logarithmic space, improving on the polynomial-time result obtained in [vdMSW11].

Our definitions of dynamic security are inspired by, and generalizations of, static IP-security. While there are valid arguments against IP-security in the static case [vdM07], these issues are orthogonal to the issues stemming from the dynamic setting. We therefore study the effects of dynamic policies in the framework of IP-security, which is technically simpler than e.g., TA-security defined in [vdM07].

The proofs for all our results can be found in the appendix.

2 Preliminaries and Notation

We work with the standard state-observed system model: A system is a finite automaton where each action belongs to a dedicated agent, and each agent has an observation in each state.

**Definition 2.1.** A system is a 6-tuple $M = (S, s_0, A, \text{step}, \text{obs}, \text{dom})$, where $S$ is a finite set of states, $s_0 \in S$ is the initial state, $\text{step}: S \times A \rightarrow S$ is a transition function, $\text{obs}: S \times D \rightarrow O$ is an observation function, where $O$ is an arbitrary set of observations, and $\text{dom}: A \rightarrow D$ associates with each action an agent, where $D$ is an arbitrary finite set of agents (or security domains).

For a state $s$ and an agent $u$, we usually write $\text{obs}_u(s)$ instead of $\text{obs}(s, u)$. For a sequence $\alpha \in A^*$ of actions and a state $s \in S$, we denote with $s \cdot \alpha$ the state obtained when performing the sequence $\alpha$ starting in $s$, i.e., $s \cdot \epsilon = s$, and $s \cdot \alpha \alpha = \text{step}(s, \alpha, \alpha)$. A local policy is a reflexive relation $\to_s \subseteq D \times D$. A dynamic policy is a family of local policies $(\to_s)_{s \in S}$, one for each state of the system. We also more intuitively talk about “edges” in the policy, where an edge is a single entry $u \to_s v$. A policy is static if the local policies $\to_s$ are the same in all states, i.e., if the question whether $u \to_s v$ does not depend on $s$. In this case, we only write $u \to v$. We define the set $u_s^{-}$ as the set of agents that may interfere with $u$ in $s$, i.e., the set $\{v \mid v \to_s u\}$.

In our examples, we often identify states with the action sequences used to reach them from the initial state. For example, in the system shown in Figure 2, we denote the initial state with $\epsilon$, the bottom left state with $a$ the upper right
state with \( w \), and the bottom right state with \( wa \). In each state of the system, we write the policy in that state as a graph. For example, in the system from Figure 2, we have \( W \rightarrow R \), but \( W \not
rightarrow R \). We only specify the observations of the agents insofar as they are relevant for the example, which in Figure 2 is only the observation of the agent \( R \).

2.1 Static noninterference

A formulation of security for systems with a static policy was introduced by Goguen and Meseguer [GM82,GM84]. The intuition is that an agent \( u \) may only observe actions of agents that are allowed to interfere with \( u \). In other words, if an agent \( v \) performs some action \( a \) and this agent \( v \) is not allowed to interfere with some agent \( u \), i.e. \( v \not
\rightarrow u \), then whatever happened after this action \( a \), the agent \( u \) should never be able to observe whether the action \( a \) has happened or not. For formalizing this intuition, an operator purge is defined, which removes all actions from a run, which are not permitted to observe by a specific agent.

**Definition 2.2.** For any agent \( u \in D \), \( a \in A \) and \( \alpha \in A^* \), we define

\[
purge(\epsilon, u) = \epsilon, \quad \text{purge}(a\alpha, u) = \begin{cases} a \text{ purge}(\alpha, u) & \text{if } \text{dom}(a) \rightarrow u \\ \text{purge}(\alpha, u) & \text{otherwise} \end{cases}
\]

A system is considered as secure if no agent can distinguish from its observation, if a specific run \( \alpha \) or the “purged” run is performed.

**Definition 2.3 ([GM82,GM84]).** A system is P-secure if and only if for every \( u \in D \), \( \alpha \in A^* \), we have \( \text{obs}_u(s_0 \cdot \alpha) = \text{obs}_u(s_0 \cdot \text{purge}(\alpha, u)) \).

P-security can also be characterized using unwindings. These are equivalence relations that characterize states that agents should not be able to distinguish. An unwinding consists of an equivalence relation on the states of the system for every agent. For P-security the following conditions were defined by Haigh and Young [HY87]. For every \( s, t \in S \), \( a \in A \):

1. **OC:** (output consistency) If \( s \sim_u t \), then \( \text{obs}_u(s) = \text{obs}_u(t) \).
2. **SC:** (step consistency) If \( s \sim_u t \), then \( s \cdot a \sim_u t \cdot a \).
3. **LR:** (left respect) If \( \text{dom}(a) \not
\rightarrow u \), then \( s \sim_u s \cdot a \).

These conditions characterize secure systems:

**Theorem 2.4 ([HY87], [Rus92]).** A system is P-security if and only if for every \( u \in D \), there exists a relation \( \sim_u \) that satisfies OC, SC and LR.

An attractive feature of unwinding-based characterizations of security notions is that they directly lead to an efficient verification procedure [EvdMSW11].
2.2 Static Intransitive Noninterference

The notion of P-security is very restrictive on systems with a non-transitive policy. A relaxed notion of noninterference was proposed by Haigh and Young [HY87]. The intuition is that actions may transmit—or downgrade—information about actions performed earlier. If an agent $H$ is permitted to interfere with $D$, $(H \rightarrow D)$ and if $D$ is permitted to interfere with $L$, $(D \rightarrow L)$, but $H$ must not directly interfere with $L$ ($H \not\rightarrow L$), then $L$ is only allowed to observe that $H$ has performed an action, if $D$ performs an action after $H$’s action. Hence $D$’s action transmits $H$’s action, and thus makes it permitted for $L$ to observe it. It is not allowed that $L$ observes $H$’s actions directly, without any action of $D$.

As an example, consider the System in Figure 3. As usual, the action $h$ belongs to the agent $H$, action $d$ to agent $D$. It is obvious that agent $L$ obtains some information about $H$’s actions: Whenever agent $L$ observes the value 2, it is clear that the action $h$ was performed. However, this event only occurs when the $h$ action has been transmitted to agent $L$ by the action $d$ performed by the downgrader $D$. Hence the system is secure with respect to intransitive noninterference. For the formal definition, we follow the presentation of Rushby [Rus92].

He considers the set of agents $\text{sources}(\alpha, u)$ whose actions will be transmitted to agent $u$ when the action sequence $\alpha$ is performed. It is inductively defined by $\text{sources}(\epsilon, u) = \{u\}$ and, for $a \in A$, $\alpha \in A^*$ if it exists $v \in \text{sources}(\alpha, u)$ with $\text{dom}(a) \rightarrow v$ then

$$\text{sources}(aa, u) = \{\text{dom}(a)\} \cup \text{sources}(\alpha, u) ,$$

and else

$$\text{sources}(aa, u) = \text{sources}(\alpha, u) .$$

This leads to the definition of an intransitive purge function: Analogously to P-security, an action $a$ is purged from the sequence $aa$ for agent $u$ if $\text{dom}(a)$ is not among the agents whose actions will be transmitted to $u$ when $aa$ is performed. This is inductively defined by $\text{ipurge}(\epsilon, u) = \epsilon$ and for $a \in A$ and $\alpha \in A^*$ by

$$\text{ipurge}(aa, u) = \begin{cases} a \text{ ipurge}(\alpha, u) & \text{if } \text{dom}(a) \in \text{sources}(aa, u) \\ \text{ipurge}(\alpha, u) & \text{otherwise} . \end{cases}$$

The corresponding security definition is analogous to the one of P-security.

**Definition 2.5** ([HY87],[Rus92]). A system is IP-secure iff for all $u \in D$, $\alpha \in A^*$, we have $\text{obs}_u(s_0 \cdot \alpha) = \text{obs}_u(s_0 \cdot \text{ipurge}(\alpha, u))$.
3 Dynamic Policies: Security for Transitive Policies

3.1 Definition of dynamic information flow security

We first give a new security definition which generalizes P-security to the dynamic case: We require that an action which is not observable for an agent when it occurs does not have any influence on the agent’s observation in the future.

**Definition 3.1.** A system is dP-secure if and only if for all \( u \in D \), \( s \in S \), \( a \in A \) and \( \alpha \in A^* \) the following implication holds:

\[
\text{If } \text{dom}(a) \not\rightarrow_s u, \text{ then } \text{obs}_u(s \cdot \alpha) = \text{obs}_u(s \cdot a \alpha).
\]

Figure 4 shows a dP-secure system. In contrast, the system in Figure 2 is not dP-secure, since \( A \not\rightarrow \epsilon \), but \( \text{obs}_R(aw) \neq \text{obs}_R(w) \).

Our security definition takes a different approach than the definitions stated in Section 2: We assume a local point of view by only checking whether a single event is visible for an agent. Hence our definition might seem too weak. However, it is easy to give a purge-based characterization of dP-security. For this, we define a dynamic purge function as follows—note that clearly, the function depends on the state in which the “purging” starts, since the policies depend on the state as well.

**Definition 3.2.** For any \( u \in D \), \( s \in S \), \( a \in A \) and \( \alpha \in A^* \), we define \( \text{dpurge}(\epsilon, u, s) = \epsilon \) and,

\[
\text{dpurge}(a \alpha, u, s) = \begin{cases} 
    a \text{dpurge}(\alpha, u, s) & \text{if } \text{dom}(a) \not\rightarrow_s u \\
    \text{dpurge}(\alpha, u, s) & \text{otherwise}.
\end{cases}
\]

Similarly to P-security (see Theorem 2.4), we can provide unwinding relations. We adapt the requirements for P-security to the dynamic setting as follows: For an agent \( u \in D \), all states \( s, t \in S \) and all \( a \in A \).

- **OC\text{dp}:** If \( s \sim_u t \), then \( \text{obs}_u(s) = \text{obs}_u(t) \).
- **SC\text{dp}:** If \( s \sim_u t \), then \( s \cdot a \sim_u t \cdot a \).
- **LR\text{dp}:** If \( \text{dom}(a) \not\rightarrow_s u \), then \( s \sim_u s \cdot a \).

3.2 Characterizations of dynamic information flow

Both, the \( \text{dpurge} \)-function and the unwinding relations can be used to characterize dP-security, which highlights the natural similarities to the static case. We note that in contrast to the static case, it is not sufficient to consider only action sequences that start in the initial state.
Theorem 3.3. Let $M$ be a system with a dynamic policy $(\rightarrow_s)_{s \in S}$. Then the following are equivalent:

1. The system $M$ is $dP$-secure.
2. For all $u \in D$, $s \in S$ and all $\alpha, \beta \in A^*$ with $\text{dpurge}(\alpha, u, s) = \text{dpurge}(\beta, u, s)$, we have that $\text{obs}_u(s \cdot \alpha) = \text{obs}_u(s \cdot \beta)$.
3. For every agent $u \in D$ there exists an equivalence relation $\sim_u \subseteq S \times S$ that satisfies the conditions $\text{OC}^{dp}$, $\text{SC}^{dp}$ and $\text{LR}^{dp}$.

The benefit of unwinding relations is, that they lead directly to an efficient verification procedure. For verifying security, it is sufficient to compute for every $u \in D$ the smallest equivalence relation that satisfies $\text{LR}^{dp}$ and $\text{SC}^{dp}$ and check that the observation function $\text{obs}_u$ is constant on every equivalence class. This can be done with nearly the same algorithm as in the static case, described in [EvdMSW11]. The above theorem directly implies that it can be decided in nondeterministic logarithmic space whether a system is $dP$-secure with respect to a given policy.

3.3 Inconsistencies and Uniform Policies

Dynamic policies can be inconsistent: An allowed interference $u \rightarrow_s v$ may contradict a “forbidden” interference $u' \not\rightarrow_{s'} v'$ in another state $s'$. Hence an edge in a policy does not explicitly allow that information may flow, but this permission is subject to consistency with other aspects of the policy. Thus an edge $u \rightarrow_s v$ in the policy should be interpreted as “it is not explicitly forbidden that $\text{dom}(a)$ interferes with $u$.” For an example, again consider the system in Figure 2. Here, the policy in the initial state allows information flow from $W$ to $R$. However, if $R$ is allowed to observe $W$’s action in the initial state, then $R$ would know that the system is in the initial state, and would also know that $W$ has not performed an action. This is an information flow from $A$ to $R$, which is prohibited by the policy. We now discuss a class of policies where this problem does not appear: A very simple structure of the policy is one where in all states which have to be indistinguishable for an agent the incoming edges of the local policy are the same. If a system has such a policy, an agent always knows what are its incoming edges, even the agent does not know the actual state of the system. Such policies will be denoted as uniform policies.

Definition 3.4. Let $M$ be a system with a dynamic policy $(\rightarrow_s)_{s \in S}$. Then $(\rightarrow_s)_{s \in S}$ is uniform if for every $u \in D$, $s \in S$, $a \in A$ and $\alpha \in A^*$ with $\text{dom}(a) \neq \rightarrow_s u$, we have $u_{s \cdot \alpha} = u_{s \cdot a \alpha}$.

Surprisingly, it can be shown that uniform policies are sufficient in the following sense: Every policy is equivalent to the largest uniform policy it contains. Intuitively, the reason for this is that if $v \rightarrow_s u$, but $u$ is not supposed to “know” that this interference is allowed, then any actual information flow using this edge lets $u$ notice that the edge was present—and hence allows $u$ to distinguish the state $s$ from a state $s'$ with $v \not\rightarrow_{s'} u$. 

8
We show that for any system with an arbitrary policy, it is possible to give a uniform policy, such that the same system is dP-secure with the original policy if and only if the system is dP-secure with the uniform policy.

Theorem 3.5. Let $M$ be a system, and let $(\rightarrow_s)s \in S$ be a dynamic policy. Let $(\rightarrow'_s)s \in S$ be the largest uniform policy that is included in $(\rightarrow_s)s \in S$. Then $M$ is dP-secure with respect to $(\rightarrow_s)s \in S$ if and only if $M$ is dP-secure with respect to $(\rightarrow'_s)s \in S$.

We note that the policy $(\rightarrow'_s)s \in S$ in Theorem 3.5 can be obtained from $(\rightarrow_s)s \in S$ by repeatedly removing edges $v \rightarrow_s u$ from the policy whenever there is a state $s'$, such that for $u$, the states $s$ and $s'$ should be indistinguishable due to the definition of dP-security.

Uniform policies have the following property: Every edge occurring in some state represents an information flow that is actually allowed; no edge contradicts the global policy. Another way to interpret this is that any information flow that is forbidden by the policy is directly forbidden via the absence of the corresponding edge. In that sense, such a policy is closed under logical deduction. Uniform policies have several additional natural properties, for example the dpurge-function behaves very similarly to the static case: It suffices to verify action sequences that start in the initial state of the system and dpurge satisfies a natural associativity condition on uniform policies. It also can be shown that for uniform policies, our security definition and the definition from [Les06] coincide.

4 Transitive with downgrading over the time

Before considering the general intransitive case, we consider an interesting intermediate scenario. Consider a situation where in some state $s$, the agent $H$ is not allowed to interfere with $L$, i.e., we have $H \not\rightarrow_s L$, but in some later state $s \cdot \alpha$, this interference is allowed, i.e., $H \rightarrow_{s \cdot \alpha} L$. In the definition of dP-security, this only allows $H$ to transmit to $L$ information about actions that $H$ performs in the state $s \cdot \alpha$. A slight relaxation of dP-security allows agent $H$ to additionally transmit in the state $s \cdot \alpha$ information about actions that $H$ itself performed previously to the state $s \cdot \alpha$, e.g., in the state $s$. Hence this interpretation of dynamic non-interference allows an agent to transmit actions that it has performed earlier. However, it is not allowed to transmit actions performed by other agents.

---

2 i.e., if $v \rightarrow_s u$ according to $(\rightarrow'_s)s \in S$, then also $v \rightarrow_s u$ according to $(\rightarrow_s)s \in S$
The system in Figure 5 should be considered as secure, since the second $h$ action transmits the first $h$ action. This system is not dP-secure.

This is captured by the following definition:

**Definition 4.1.** A system is dotP-secure, if for all $u \in D$, $a \in A$ and $\alpha \in A^*$: If $\text{dom}(a) \not\rightarrow_s u$ and it does not exist $b \in A$, $\beta, \beta' \in A^*$ with $\alpha = \beta b \beta'$ and $\text{dom}(a) = \text{dom}(b)$ and $\text{dom}(a) \not\rightarrow_{s-a} b$, then $\text{obs}_u(s \cdot a \alpha) = \text{obs}_u(s \cdot \alpha)$.

Note that the situation here is asymmetric in the following sense: In the example given in Figure 5, the initial state and the state $h$ need to be considered “equivalent” by any unwinding-like characterization. However, performing the action $h$ in the initial state must not leave the equivalence class, while performing it in the state $H$ may. Therefore, it is not surprising that instead of an equivalence relation (which is in particular symmetric), our characterization of dotP-secure systems uses a directed relation.

For every agent $u$ and $v$, the properties that such a relation should satisfy are for every $s, t \in S$ and $a \in A$:

- **OC$_{\text{dotp}}$:** If $s \lessdot^v u$ t, then $\text{obs}_u(s) = \text{obs}_u(t)$.
- **SC$_{\text{dotp}}$:** If $s \lessdot^v u$ t and $\text{dom}(a) \neq v$ or $\text{dom}(a) = v$ and $v \not\rightarrow_t u$, then $s \cdot a \lessdot^v t \cdot a$.
- **LR$_{\text{dotp}}$:** If $\text{dom}(a) = v$ and $v \not\rightarrow_s u$, then $s \lessdot^v s \cdot a$.

It is also possible to characterize dotP-security by a sources-based definition closer to case of intransitive noninterference: This highlights that downgrading over time already features some aspects of the intransitive case, which we will study in detail in Section 5.

**Definition 4.2.** For $u \in D$, $s \in S$, $a \in A$ and $\alpha \in A^*$, we define $\text{dotsources}(\epsilon, u, s) = \{u\}$ and,

$$\text{dotsources}(a\alpha, u, s) = \begin{cases} \{\text{dom}(a)\} \cup \text{dotsources}(\alpha, u, s \cdot a) & \text{if } \text{dom}(a) \not\rightarrow_s u \\ \text{dotsources}(\alpha, u, s \cdot a) & \text{otherwise} \end{cases}$$

Both, the unwinding relations and the sources-based definition characterizes dotP-security:

**Theorem 4.3.** Let $M$ be a system with a policy $(\not\rightarrow_s)_{s \in S}$. Then the following properties are equivalent:

1. $M$ is dotP-secure.
2. For every two agents $u, v \in D$ there exists a relation $\lessdot^v_u$ that satisfies OC$_{\text{dotp}}$, SC$_{\text{dotp}}$ and LR$_{\text{dotp}}$.
3. For all $u \in D$, $a \in A$, $s \in S$, $\alpha \in A^*$: If $\text{dom}(a) \notin \text{dotsources}(a\alpha, u, s)$, then $\text{obs}_u(s \cdot a\alpha) = \text{obs}_u(s \cdot \alpha)$.

The characterization with an unwinding implies that deciding whether a system is dotP-secure with respect to a given policy can be done in nondeterministic logarithmic space.

A purge-based characterization of dotP-security can be given analogously.
5 Intransitive Case

5.1 Definition of dIP-security

We now consider the fully intransitive case, where whenever an agent performs an action, he transmits information about the actions he has performed himself as well as information about actions by other agents that was previously transmitted to him. The definition follows a similar pattern as that of IP-security: If the performance of an action sequence $a\alpha$ starting in a state $s$ does not transmit the action $a$ (possibly via several intermediate steps) to the agent $u$, then $u$ should not be able to decide from his observations whether $a$ was performed. To formalize this, we use the straightforward adaptation of Rushby’s sources-definition to the dynamic case (see also [Les06]):

**Definition 5.1.** Let $M$ be a system with policy $(\rightarrow s)_{s \in S}$. For a state $s \in S$, a sequence $\alpha \in A^*$, and an agent $u$, we define $\text{disources}(\epsilon, u, s) = \{u\}$, and

$$\text{disources}(a\alpha, u, s) = \begin{cases} \{\text{dom}(a)\} \cup \text{disources}(\alpha, u, s \cdot a), & \text{if } \exists v \in \text{disources}(\alpha, u, s \cdot a) \text{ with } \text{dom}(a) \rightarrow s v, \\ \text{disources}(\alpha, u, s \cdot a), & \text{otherwise.} \end{cases}$$

An action $a$ is transmitted to an agent $u$ in the action sequence $a\alpha$ starting in the state $s$ if $\text{dom}(a) \in \text{disources}(a\alpha, u, s)$. An alternate view of the definition is to consider the sets of agents that “know” whether the action $a$ has been performed in state $s$: Initially, this is only the set of agents $v$ with $\text{dom}(a) \rightarrow s v$. The knowledge is spread by every action performed by an agent who is “in the know”: If an action $b$ is performed in a later state $t$, and $\text{dom}(b)$ already knows that the action $a$ was performed, then all agents $v$ with $\text{dom}(b) \rightarrow t v$ obtain this information when $b$ is performed. Following the discussion above, we obtain the following natural definition of security:

**Definition 5.2.** Let $M$ be a system with a policy $(\rightarrow s)_{s \in S}$. We say that $M$ is dIP-secure, if for all states $s$, all actions $a \in A$, and all sequences $\alpha \in A^*$, with $\text{dom}(a) \notin \text{disources}(a\alpha, u, s)$, we have $\text{obs}_u(s \cdot a\alpha) = \text{obs}_u(s \cdot \alpha)$.

The definition formalizes the above arguments: If, on the path $a\alpha$, the action $a$ is not transmitted to $u$, then the question whether $a$ was performed or not should not change $u$’s observation: The runs $a\alpha$ and $\alpha$, starting in the state $s$, should be indistinguishable for $u$.

Consider the earlier example in Figure 2. As argued before, the system should be regarded as insecure. This remains true for our intransitive notion of security: Since $A$ must not interfere with any other agent in any state of the system, we have $\text{dom}(a) \notin \text{disources}(aw, R, s_0)$. Since we have $\text{obs}_R(s_0 \cdot aw) \neq \text{obs}_R(s_0 \cdot w)$, the system is insecure.
We will now consider two alternative definitions of intransitive noninterference in the dynamic case, the first one based on a purge-like function and the second one using a natural unwinding condition. We will then see that all of these approaches lead to the same definition of security, which strengthens our belief that dIP-security is indeed a natural definition of dynamic intransitive noninterference.

5.2 A purge-based definition of dIP-security

Similarly to our definition for transitive systems, dIP-security can be characterized by a purge-like function as well. There is, however, a subtle but important point that has to be considered, which we discuss after giving the formal definition.

**Definition 5.3.** Let $M$ be a system with a policy $(\rightarrow s)_{s \in S}$. For a state $s$, a sequence $\alpha \in A^*$, and an agent $u$, we define $\operatorname{dipurge}(\epsilon, u, s) = \epsilon$, and

$$
\operatorname{dipurge}(a\alpha, u, s) = \begin{cases} 
\operatorname{dipurge}(\alpha, u, s \cdot a), & \text{if } \operatorname{dom}(a) \in \operatorname{disources}(a\alpha, u, s), \\
\operatorname{dipurge}(\alpha, u, s), & \text{otherwise}.
\end{cases}
$$

The crucial point is that in the case where $a$ is not visible for the agent $u$, we define $\operatorname{dipurge}(a\alpha, u, s)$ as $\operatorname{dipurge}(\alpha, u, s)$ instead of the more intuitive choice $\operatorname{dipurge}(\alpha, u, s \cdot a)$, on which the security definition in [Les06] is based.

We briefly explain the reasoning behind this choice. Assume that we work with the more intuitive choice of the purge-function as outlined above, and consider the action sequence $aw$, performed from the initial state in the system given in Figure 2. Since the action $a$ is clearly not transmitted to the agent $R$, it is removed from the trace, and the result of the purge function would be the same as purging the sequence $w$ starting in the lower left state. However, in this state, the action $w$ is invisible for $R$, hence the purge-function would remove it, and thus purging the sequence $aw$ for agent $R$ results in the empty sequence. On the other hand, if we consider the sequence $w$, again starting in the initial state, then the $w$ is not removed, since here $w$ is directly visible for $R$. It hence follows that $aw$ and $w$ do not lead to the same purged trace. Hence a security definition based on such a purge-function does not require $aw$ and $w$ to lead to observationally equivalent states. This is the reason why, as mentioned earlier, the system in Figure 2 is considered secure in the security definition from [Les06].

However, a natural definition of security needs to require $aw$ and $w$ to lead to the same observation for agent $R$, as the event $a$ may not be transmitted to $R$ under any circumstances.

In Section 5.4 we will show that $\operatorname{dipurge}$ yields a natural equivalent characterization of dIP-security, which further supports the above arguments for our definition of the $\operatorname{dipurge}$-function.
5.3 Unwindings for dIP-security

We now present an intransitive dynamic security definition based on unwinding relations. As mentioned in our discussion of downgrading over time, dynamic intransitive security cannot easily be characterized with symmetric unwindings. Hence our characterization again uses a directed relation. In fact, we need a relation $\lesssim_{D'}$ for every set of agents $D' \subseteq D$. We require the following conditions:

**Definition 5.4.** Let $M$ be a system with agents $D$ and state set $S$, let $(\rightarrow_s)_{s \in S}$ be a policy for $M$. A dynamic intransitive unwinding for $M$ with respect to $(\rightarrow_s)_{s \in S}$ consists of a family of relations $(\lesssim_{D'})_{D' \subseteq D}$, where $\lesssim_{D'} \subseteq S \times S$ for all $D' \subseteq D$ such that the following conditions are satisfied:

- **dOC** (output consistency): If $s \lesssim_{D'} t$ and $u \in D'$, then $\text{obs}_u(s) = \text{obs}_u(t)$,
- **dSC** (step consistency): If $s \lesssim_{D''} t$, then $s \cdot b \lesssim_{D''} t \cdot b$, where $D'' = D'$ if $\text{dom}(b) \in D'$, and $D'' = D' \cap \{u \mid \text{dom}(b) \not\rightarrow_s u\}$ otherwise,
- **dLR** (local respect): $s \lesssim_{\{u \in D \mid \text{dom}(a) \not\rightarrow_s u\}} s \cdot \alpha$.

Intuitively, $s \lesssim_{D'} t$ expresses that there is a common reason for all agents in $D'$ to have the same observations in $s$ as in $t$, i.e., if there is a state $\tilde{s}$, an action $a$, and a sequence $\alpha$, such that $s = \tilde{s} \cdot \alpha$, $t = \tilde{s} \cdot \alpha$, and $\text{dom}(a) \notin \text{disources}(\alpha, u, \tilde{s})$ for all agents $u \in D'$.

5.4 Characterizations of dIP-security

As already informally stated above, we now show that the three characterizations of dynamic intransitive noninterference suggested above are equivalent: Our security definition from Section 5.1 can be equivalently phrased using a purge-based definition as well as using our unwinding characterization.

**Theorem 5.5.** Let $M$ be a system and let $(\rightarrow_s)_{s \in S}$ be a policy. Then the following are equivalent:

1. $M$ is dIP-secure with respect to $(\rightarrow_s)_{s \in S}$,
2. for all agents $u$, all states $s$, and all action sequences $\alpha$ and $\beta$ with $\text{dipurge}(\alpha, u, s) = \text{dipurge}(\beta, u, s)$, we have that $\text{obs}_u(s \cdot \alpha) = \text{obs}_u(s \cdot \beta)$,
3. there is a dynamic intransitive unwinding for $M$ with respect to $(\rightarrow_s)_{s \in S}$.

5.5 Complexity of verifying dIP-security

In contrast to unwindings for the transitive case, the unwinding characterization of dIP-security does not lead to a polynomial-time algorithm to verify security of a system: The number of relations needed to consider is exponential in the number of agents in the system. It turns out that (unless P = NP), we cannot do significantly better: The verification problem is in fact NP-complete.

**Theorem 5.6.** Determining whether a given system is dIP-secure with respect to a given policy is NP-complete.
In particular, unless P = NP, dIP-security cannot be characterized with a polynomial number of unwinding relations, each of which can be computed in polynomial time. However, the above unwinding characterization shows that the complexity stems from the number of agents in the system, as opposed to the number of states. Since the number of agents is often significantly smaller than the number of states, efficient verification of security for many realistic systems is still possible. Formally, applying a standard dynamic programming approach to the unwinding conditions given in Theorem 5.5 yields the following result:

**Corollary 5.7.** – Determining whether a given system is dIP-secure with respect to a given policy is fixed-parameter-tractable with the number of agents as parameter.

– For systems where the number of agents is logarithmic in the number of states, deciding whether a system satisfies dIP-security can be performed in polynomial time.

The concept of fixed-parameter tractability was introduced to study situations as above, where the worst-case complexity of an NP-complete problem depends on a parameter that will be small for practically relevant instances. For background on fixed-parameter tractability, we refer the reader to [DF99].

### 5.6 Relationship to other notions of security

As mentioned earlier, the only definition of dynamic non-interference in a state-based setting that we are aware of is given in [Les06]. With the notation introduced in Section 5.1 we can now formally state their security definition: Consider the function \( \text{dipurge}' \) which is defined in the same way as \( \text{dipurge} \), except that in the case that \( \text{dom}(a) \notin \text{disources}(ao, u, s) \), \( \text{dipurge}' \) is defined as \( \text{dipurge}'(ao, u, s) = \text{dipurge}'(\alpha, u, s\cdot a) \). Now a system is secure according to the definition from [Les06] if for all agents \( u \) and all action sequences \( \alpha \) and \( \beta \) with \( \text{dipurge}'(\alpha, u, s_0) = \text{dipurge}'(\beta, u, s_0) \), we have that \( \text{obs}_u(s_0 \cdot \alpha) = \text{obs}_u(s_0 \cdot \beta) \).

One can show that for uniform policies, our definition and the one given in [Les06] also coincide in the intransitive case. For irredundant policies, this is not generally true.

For static systems, our intransitive purge function as defined above and the function used to define IP-security in [HY87] (cp. Section 2.2) are identical. Therefore the above Theorem 5.5 and the fact that \( \text{dipurge} \) is idempotent immediately imply that IP-security and dIP-security are equivalent in the static case.

**Proposition 5.8.** Let \( M \) be a system, and let \( (-s)_{s \in S} \) be a static policy. Then \( M \) is IP-secure if and only if \( M \) is dIP-secure with respect to \( (-s)_{s \in S} \).

As expected, the situation is considerably more complex for dynamic policies than in the static scenario. For example, it is well-known and easy to see that for static, transitive policies, IP-security and P-security coincide. The analogous
result does not hold in the dynamic case. Consider the example given in Figure 5. Here every local policy is transitive, but there are still intransitive effects resulting from the dynamic change of the policy. The system is not dP-secure, but it dIP-secure. The class of systems in which such effects do not occur are those where, informally, “every edge in the policy that we could possible use in the future is already there in the present state.”

**Proposition 5.9.** Let $M$ be a system with policy $(\rightarrow_s)_{s \in S}$ such that for all states $s$, all actions $a \in A$, all sequences $\alpha \in A^*$, and all agents $u$ with $\text{dom}(a) \in \text{disources}(aa, u, s)$, we have $\text{dom}(a) \rightarrow_s u$. Then $M$ is dIP-secure with respect to $(\rightarrow_s)_{s \in S}$ if and only if $M$ is dP-secure with respect to $(\rightarrow_s)_{s \in S}$.

### 5.7 Consistent Policies and Redundant Edges

In our discussion of dP-security, we observed that a policy may contain edges that can never be used. Clearly, this issue also occurs in the intransitive case; however the solution is not as simple: In the transitive case, it was sufficient to “remove any incoming edge for $u$ that $u$ does not know about,” which is the informal statement of Theorem 3.5. In the intransitive case, this is clearly not true, as is evident from the example in Figure 6. When the system is in state $h_1$, then agent $L$ does not know that an edge $D \rightarrow L$ is present, since for $L$ states $\epsilon$ and $h_1$ are indistinguishable—but clearly, the edge cannot be removed from the policy without affecting security. However, unusable edges still exist in the intransitive case. We call such edges *redundant*.

To formalize this notion, it is helpful to consider a different view of our security definition. Note that our security definition essentially consists in establishing, for each agent $u$ an equivalence relation (which we will call $\sim_u$) on states, where the security requirement is that $\sim_u$-equivalent states have identical observations (for the agent $u$). It is useful to formalize this in the obvious way:

**Definition 5.10.** For a system $M$, policy $(\rightarrow_s)_{s \in S}$, and agent $u$, let $\sim_u$ be the smallest equivalence relation on the states of $M$ such that for all $s, a, \alpha$, if $\text{dom}(a) \notin \text{disources}(aa, u, s)$, then $s \cdot a \alpha \sim_u s \cdot \alpha$.

Clearly, a system is dIP-secure if and only if for all $s_1$ and $s_2$ with $s_1 \sim_u s_2$, we have that $\text{obs}_u(s_1) = \text{obs}_u(s_2)$. This equivalence relation allows us to easily formalize when an edge in the policy is redundant, i.e., can be removed without affecting security:
Definition 5.11. Let $M$ be a system with policy $(\rightarrow_s)_{s \in S}$. Let $e$ be an edge in $(\rightarrow_s)_{s \in S}$, and let $(\rightarrow'_s)_{s \in S}$ be the policy obtained from $(\rightarrow_s)_{s \in S}$ by removing $e$. Then $e$ is redundant, if for all $s, u, a, \alpha$ such that $\text{dom}(a) \in \text{disources}(a, u, s)$ when using the policy $(\rightarrow_s)_{s \in S}$, and $\text{dom}(a) \notin \text{disources}(a, u, s)$ when using policy $(\rightarrow'_s)_{s \in S}$, we have $s \cdot a \alpha \sim_u s \cdot \alpha$ (with respect to the original policy $(\rightarrow_s)_{s \in S}$). A policy is consistent if it does not contain any redundant edge.

We show below that this definition exactly captures the edges that are intuitively redundant in a policy. We stress that the question whether an edge is redundant does not depend on the observation function of the system, but only on the policy and the system’s transition function, whereas a definition of security is concerned with comparing observations in different states.

Consistency means that there is no edge in the policy which is contradicted by other aspects of the policy; we can again interpret this property in the dual way: The set of information flows forbidden by a consistent policy is closed under logical deduction—every edge that can be shown to represent a forbidden information flow is absent in the policy. This is also a property of uniform policies in the transitive case, in fact, in the transitive case, uniform and consistent policies coincide (except for degenerated cases). We will later see that this is not true in the transitive case.

As an example for a secure system with a policy that is inconsistent, consider the system given in Figure 6. It can be shown that the system is secure (the agent $L$ knows whether in the initial state, $h_1$ or $h_2$ was performed, as soon as this information is transmitted by agent $D$). We now show that the edge $H \rightarrow_{h_1} L$ is redundant. To see this, consider what happens if we remove this edge from the policy. The only states that $L$ is allowed to distinguish with the edge, but not without, are combinations of the states $\{h_1, h_1h_1, h_1h_2\}$. However, we observe that

- $h_2h_1 \sim_L h_1$, since $\text{dom}(h_2) \notin \text{disources}(h_2h_1, L, \epsilon)$,
- $h_2h_1h_1 \sim_L h_2h_1$, since $\text{dom}(h_1) \notin \text{disources}(h_1, L, h_2h_1)$,
- $h_2h_1h_1 \sim_L h_1h_1$, since $\text{dom}(h_2) \notin \text{disources}(h_2h_1, L, \epsilon)$,
- $h_2h_1h_2 \sim_L h_2h_1$, since $\text{dom}(h_2) \notin \text{disources}(h_2, L, h_2h_1)$,
- $h_2h_1h_2 \sim_L h_1h_2$, since $\text{dom}(h_2) \notin \text{disources}(h_2h_1h_2, L, \epsilon)$.

Symmetry and transitivity of $\sim_L$ imply $h_1 \sim_L h_2h_1h_1 \sim_L h_1h_1 \sim_L h_2h_1h_2 \sim_L h_1h_2$, hence all three $\{h_1, h_1h_1, h_1h_2\}$ are $\sim_L$-equivalent even with respect to the original policy. Hence the edge connective $H \rightarrow_{h_1} L$ is indeed redundant (and it follows from this discussion that the system would indeed be insecure if $h_1$, $h_1h_1$, and $h_1h_2$ would not all have the same observations).

We now show that redundant edges are exactly those that can be removed from a policy without affecting security—and thus our definition of irredudant edges captures the intuitive notion of an edge being redundant. It follows that each dynamic policy is equivalent to a consistent policy, which can be computed by removing every redundant edge.

Theorem 5.12. Let $M$ be a system with a policy $(\rightarrow_s)_{s \in S}$.
1. Let \((\rightarrow^\prime_s)_{s \in S}\) be obtained from \((\rightarrow_s)_{s \in S}\) by removing a set of edges which are redundant. Then \(M\) is dIP-secure with respect to \((\rightarrow_s)_{s \in S}\) if and only if \(M\) is dIP-secure with respect to \((\rightarrow^\prime_s)_{s \in S}\).

2. Let \((\rightarrow^\prime_s)_{s \in S}\) be obtained from \((\rightarrow_s)_{s \in S}\) by removing an edge that is not redundant. Then there exists a system \(N\) such that \(N\) differs from \(M\) only in the observation function and that is dIP-secure with respect to \((\rightarrow^\prime_s)_{s \in S}\), but not dIP-secure with respect to \((\rightarrow^\prime_s)_{s \in S}\).

3. Let \((\rightarrow^\prime_s)_{s \in S}\) be obtained from \((\rightarrow_s)_{s \in S}\) by removing the set of all edges that are redundant. Then \((\rightarrow^\prime_s)_{s \in S}\) is consistent.

Theorem 5.12 states that for every policy \((\rightarrow_s)_{s \in S}\), a consistent policy \((\rightarrow^\prime_s)_{s \in S}\) that is equivalent to \((\rightarrow_s)_{s \in S}\) can be obtained from \((\rightarrow_s)_{s \in S}\) by removing all redundant edges. The proof also implies that the order in which redundant edges are removed is irrelevant in the sense that any sequential, exhaustive removal of redundant edges will always result in the same policy. We mention that, in contrast to the transitive case, a consistent policy is not necessarily uniform (as an example, consider the system in Figure 5, which can also be studied with respect to our intransitive security definition).

### 5.8 Sound Unwinding Conditions and Uniform Intransitive Policies

In the above Section 5.3, we obtained an unwinding characterization of dIP-security, which however does not yield a polynomial-time algorithm for security verification if the number of agents is unbounded. Since the problem is NP-complete, such an algorithm—and hence a “small” unwinding—is unlikely to exist. However, we can define unwinding conditions that are sound for dIP-security, and are sound and complete for a natural subclass of systems with associated policies, namely the subclass of systems in which every agent knows the set of agents who may currently interfere with him. Formally, we define this property as follows—note that this definition is very similar to the uniformity condition for the transitive case, the only difference is the notion of states that must be indistinguishable.

**Definition 5.13.** A policy for a system is intransitively uniform, if for all agents \(u\), all actions \(a\), all states \(s\) and all sequences \(\alpha\) with \(\text{dom}(a) \notin \text{disources}(ao, u, s)\), we have \(u^\rightarrow_{s,aa} = u^\rightarrow_{s,\alpha}\).

The definition captures the above intuition: If an agent \(u\) must not distinguish two states by the security definition, then the set of agents that may interfere with \(u\) must be identical in these two states. We note that while in the transitive case, uniform and consistent policies coincide except for degenerated cases, this is not true for intransitive noninterference (in fact, neither implication holds).

Besides being a natural requirement that is often met in a concrete system, the class of intransitively uniform policies has two attractive features: First, if we have a uniform policy, then checking whether the system satisfies dIP-security can be performed in polynomial time. Second, checking whether a policy for a given system is intransitively uniform can be done in polynomial time. Both
of these results follow from characterizations of the respective conditions with unwinding relations, which in fact are very similar. In the uniform case, many of the subtle issues with dynamic policies do not occur anymore; as an example, dIP-security and the security definition from [Les06] coincide for uniform policies. However, requiring policies to be uniform is a severe restriction (note that the system shown in Figure 5 requires a non-uniform policy).

We now define the properties of the unwindings that we will be interested in:

**Definition 5.14.** A uniform dynamic intransitive unwinding for a system $M$ with respect to a policy $(\rightarrow_s)_{s \in S}$ is a family of equivalence relations $\sim_{s,v,u}$ for each choice of states $s$ and agents $v$ and $u$. We consider the following unwinding properties:

- **uOC$^{dIP}$ (output consistency):** If $s \sim_{s,v,u} t$, then $\text{obs}_u(s) = \text{obs}_u(t)$
- **uPC$^{dIP}$ (policy consistency):** If $s \sim_{s,v,u} t$, then $u_s = u_t^{-1}$
- **uSC$^{dIP}$ (step consistency):** If $s \sim_{s,v,u} t$ and $a \in A$ with $v \not\leadsto_s \text{dom}(a)$, then $s \cdot a \sim_{s,v,u} t \cdot a$
- **uLR$^{dIP}$ (local respect):** If $\text{dom}(a) \not\leadsto_s u$, then $\tilde{s} \sim_{u}^{\tilde{s},\text{dom}(a)} \tilde{s} \cdot a$

Both intransitive uniformity and dIP-security (in the case of a uniform policy) are characterized with almost exactly the same unwinding—the only difference is that for uniformity, we require the condition $uPC^{dIP}$, since here we are concerned with having the same policies in certain states, while for security, we require the condition $uOC^{dIP}$, as here we are naturally interested in observations.

**Theorem 5.15.** Let $M$ be a system with a policy $(\rightarrow_s)_{s \in S}$.

1. The policy $(\rightarrow_s)_{s \in S}$ is intransitively uniform if and only if there is a uniform dynamic intransitive unwinding for $M$ and $(\rightarrow_s)_{s \in S}$ that satisfies $uPC^{dIP}$, $uSC^{dIP}$, and $uLR^{dIP}$.
2. (a) If $M$ is dIP-secure with respect to $(\rightarrow_s)_{s \in S}$, then there is a uniform dynamic intransitive unwinding for $M$ and $(\rightarrow_s)_{s \in S}$ that satisfies $uOC^{dIP}$, $uSC^{dIP}$, and $uLR^{dIP}$.
   (b) If $(\rightarrow_s)_{s \in S}$ is intransitively uniform and there is a uniform dynamic intransitive unwinding for $M$ and $(\rightarrow_s)_{s \in S}$ that satisfies $uOC^{dIP}$, $uSC^{dIP}$, and $uLR^{dIP}$, then $M$ is dIP-secure with respect to $(\rightarrow_s)_{s \in S}$.

Due to Theorem 5.6, we cannot hope that the above conditions completely characterize secure systems, and indeed the system in Figures 5 and Figure 6 are examples for system that are dIP-secure but not intransitively uniform, and thus its security cannot be shown with a uniform dynamic intransitive unwinding. Theorem 5.15 immediately yields polynomial-time algorithms to verify the respective conditions via a standard dynamic programming approach:

**Corollary 5.16.** – Verifying whether a policy is intransitively uniform can be performed in nondeterministic logarithmic space.

– For systems with intransitively uniform policies, verifying whether a system is dIP-secure can be performed in nondeterministic logarithmic space.
In particular, it can be tested in polynomial time whether a system satisfies intransitive uniformity of the policy and dIP-security simultaneously. The above shows that the complexity of dynamic noninterference comes from the combination of dynamic policies that do not allow agents to “see” their allowed sources of information with an intransitive security definition. We note that in the transitive case, this interplay does not arise, since there a system necessarily has to allow principals to “see” their incoming edges—at least those edges that will be used for information flow (see Theorem 3.5).

5.9 An application to the static case

Recall that in the static case, our notion of noninterference security is equivalent to IP-security as defined in [HY87]. For IP-security, Rushby gave unwinding conditions that are sufficient, but not necessary. This left open the question whether there is an unwinding condition that exactly characterizes IP-security. We can answer this question positively: Clearly, a static policy is intransitively uniform. Hence our results immediately yield a characterization of IP-security with the above unwinding conditions, and from these, an algorithm verifying IP-security in nondeterministic logarithmic space can be obtained in the straightforward manner.

Corollary 5.17.

1. A system with a static intransitive policy is IP-secure if and only if it has a dynamic unwinding satisfying $u_{OC}^{dIP}$, $u_{SC}^{dIP}$, and $u_{LR}^{dIP}$.
2. Static IP-security can be verified in nondeterministic logarithmic space.

6 Conclusion

We have shown that dynamic noninterference is considerably different than static noninterference: An allowed interference in one state may contradict a forbidden interference in another state. Our new definitions of transitive and intransitive dynamic noninterference address and correct these issues. Our purge- and unwinding-based characterizations show that our definitions are natural, and directly lead to our complexity results. In this paper, we studied generalizations of IP-security to the dynamic setting. An interesting open question is to study a TA-security in a dynamic setting. Preliminary results indicate that such a generalization needs to use a very different approach from the one used in the current paper.

A Additional Results

In this Section we present and prove additional results which were informally mentioned in the main paper.
A.1 Initial-State verification suffices for uniform policies

One noteworthy difference to the static case is that it is necessary to evaluate the $\text{dpurge}$-function in every state, and not only in the initial state: The example in Figure 7 is secure if we only consider traces starting in the initial state, but can easily be seen to not be dP-secure.

However, in the case of uniform policies, it suffices to consider traces starting in the initial state, as we now show.

**Theorem A.1.** Let $M$ be a system with uniform policies. Then the system $M$ is dP-secure iff for all $u \in D$ and all $\alpha \in A^*$: $\text{obs}_u(s_0 \cdot \alpha) = \text{obs}_u(s_0 \cdot \text{dpurge}(\alpha, u, s_0))$.

**Proof.** Assume that $M$ is a secure system. Then from $s_0 \cdot \alpha \sim_u s_0 \cdot \text{dpurge}(\alpha, u, s_0)$ follows from the output consistency that $\text{obs}_u(s_0 \cdot \alpha) = \text{obs}_u(s_0 \cdot \text{dpurge}(\alpha, u, s_0))$.

For the other direction of the proof, we consider $\alpha, \beta \in A^*$ with $\text{dpurge}(\alpha, u, s) = \text{dpurge}(\beta, u, s)$. Then it exists $\gamma \in A^*$ with $s = s_0 \cdot \gamma$. It follows that $s_0 \cdot \gamma \sim_u \text{dpurge}(\gamma, u, s_0)$. This gives

\[
\text{obs}_u(s \cdot \alpha) = \text{obs}_u(s_0 \cdot \gamma \alpha) \\
= \text{obs}_u(s_0 \cdot \text{dpurge}(\gamma \alpha, u, s_0)) \\
= \text{obs}_u(s_0 \cdot \text{dpurge}(\gamma, u, s_0) \text{dpurge}(\alpha, u, s_0 \cdot \gamma)) \\
= \text{obs}_u(s_0 \cdot \text{dpurge}(\gamma, u, s_0) \text{dpurge}(\beta, u, s_0 \cdot \gamma)) \\
= \text{obs}_u(s_0 \cdot \beta).
\]
A.2 Some properties of the purge function

Here we show that our purge function in the transitive case behaves very naturally in the case of uniform policies.

Lemma A.2. Let $M$ be a system. For every $u \in D$, $s, t \in S$ and $\alpha, \beta \in A^*$, we have

1. $\text{dpurge}(\text{dpurge}(\alpha, u, s), u, s) = \text{dpurge}(\alpha, u, s),$
2. $\text{dpurge}(\alpha \beta, u, s) = \text{dpurge}(\alpha, u, s) \cdot \text{dpurge}(\beta, u, s \cdot \text{dpurge}(\alpha, u, s)),$
3. if $M$ has a uniform policy and if $\sim_u$ is an unwinding that satisfies $LR^{dp}$ and $SC^{dp}$ and if $s \sim_u t$, then $s \cdot \alpha \sim_u t \cdot \text{dpurge}(\alpha, u, t)$ and $\text{dpurge}(\alpha, u, s) = \text{dpurge}(\alpha, u, t).

Proof. 1. We show this by an induction on the length of $\alpha$. Since the base case is obvious, we proceed with the inductive step. We consider $a\alpha$ with $a \in A$ and $\alpha \in A^*$ and assume that the claim holds for $\alpha$. In the following two cases, we get

(a) If $\text{dom}(a) \rightarrow_s u$, we have

$$\text{dpurge}(\text{dpurge}(a\alpha, u, s), u, s) = \text{dpurge}(\text{adpurge}(a\alpha, u, s), us) = \text{adpurge}(\text{dpurge}(\alpha, u, s \cdot a), u, s \cdot a) \quad \text{I.H.}$$

$$= \text{adpurge}(\alpha, u, s \cdot a)$$

$$= \text{dpurge}(a\alpha, u, s).$$

(b) If $\text{dom}(a) \not\rightarrow_s u$, we have

$$\text{dpurge}(\text{dpurge}(a\alpha, u, s), u, s) = \text{dpurge}(\text{dpurge}(\alpha, u, s), u, s) \quad \text{I.H.}$$

$$= \text{dpurge}(a\alpha, u, s).$$

2. We show this claim by an induction on the length of $\alpha$. We get the following two cases

(a) If $\text{dom}(a) \rightarrow_s u$, we have

$$\text{dpurge}(a\alpha, u, s) = \text{adpurge}(a\alpha, u, s \cdot a)$$

$$\quad \text{I.H.}$$

$$= \text{adpurge}(\alpha, u, s \cdot a) \cdot \text{dpurge}(\beta, u, s \cdot \text{dpurge}(\alpha, u, s • a))$$

$$= \text{dpurge}(a\alpha, u, s) \cdot \text{dpurge}(\beta, u, s \cdot \text{dpurge}(a\alpha, u, s)).$$

(b) If $\text{dom}(a) \not\rightarrow_s u$, we have

$$\text{dpurge}(a\alpha, u, s) = \text{dpurge}(\alpha, u, s)$$

$$\quad \text{I.H.}$$

$$= \text{dpurge}(\alpha, u, s) \cdot \text{dpurge}(\beta, u, s \cdot \text{dpurge}(\alpha, u, s))$$

$$= \text{dpurge}(a\alpha, u, s) \cdot \text{dpurge}(\beta, u, s \cdot \text{dpurge}(a\alpha, u, s)).$$

3. This can be shown by an induction on the length of $\alpha$. 

$\square$
A.3 Equivalence of intransitive Security Definitions for Uniform Policies

We now show that in case of an intransitively uniform policy, a system is secure with respect to the definition of \[\text{Les06}\] if and only if it is dIP-secure.

We first show the following Lemma, which intuitively says that if the first action of $aa$ is not transmitted to $u$ on the path $aa$, then the same actions on the remaining path $\alpha$ are transmitted to $u$ when evaluating $\alpha$ from the state $s$ or from the state $s \cdot a$ in the case of uniform policies. This is the key reason why, for uniform policies, the difference between Leslie’s function $\text{dipurge}'$ and our $\text{dipurge}$ is irrelevant.

**Lemma A.3.** Let $M$ be a system with an intransitively uniform policy $(\rightarrow_s)_{s \in S}$. Let $\text{dom}(a) \notin \text{disources}(aa, u, s)$, where $a = \beta \beta'$. Then

$$\text{dom}(b) \in \text{disources}(b \beta', u, s \cdot \beta) \iff \text{dom}(b) \in \text{disources}(b \beta', u, s \cdot a \beta).$$

**Proof.** Assume this is not the case, and let $b \beta'$ be a minimal counter-example. First assume that $\text{dom}(b) \in \text{disources}(b \beta', u, s \cdot a \beta)$ and $\text{dom}(b) \notin \text{disources}(b \beta', u, s \cdot \beta)$. Then there is some $\text{dom}(c) \in \text{disources}(\beta', u, s \cdot a \beta \cdot b)$ with $\text{dom}(b) \rightarrow_s a \beta \text{dom}(c)$, and due to minimality of $b \beta'$ it follows that $\text{dom}(c) \in \text{disources}(\beta', u, s \cdot \beta b)$. Since $\text{dom}(b) \notin \text{disources}(b \beta', u, s \cdot \beta)$, it thus follows that $\text{dom}(b) \not\rightarrow_s \beta \text{dom}(c)$. This is a contradiction to the intransitive uniformity of $(\rightarrow_s)_{s \in S}$, since $\text{dom}(a) \notin \text{disources}(a \beta, \text{dom}(c), s)$, and hence $s \cdot a \beta \sim_{\text{dom}(c)} s \cdot \beta$.

The second case is essentially identical: Assume that $\text{dom}(b) \in \text{disources}(b \beta', u, s \cdot \beta)$ and $\text{dom}(b) \notin \text{disources}(b \beta', u, s \cdot a \beta)$. Then there is some $\text{dom}(c) \in \text{disources}(\beta', u, s \cdot \beta b)$ with $\text{dom}(b) \rightarrow_s a \beta \text{dom}(c)$. Due to the minimality of $b \beta'$, it follows that $\text{dom}(c) \in \text{disources}(\beta', u, s \cdot a \beta b)$, hence $\text{dom}(b) \not\rightarrow_s a \beta \text{dom}(c)$. Since $s \cdot a \beta \sim_{\text{dom}(c)} s \cdot \beta$ due to the above, we have a contradiction to the uniformity of $(\rightarrow_s)_{s \in S}$. 

From the above Lemma, we can now easily show that for uniform policies, dIP-security and security in the sense of \[\text{Les06}\] coincide:

**Theorem A.4.** Let $M$ be a system with an intransitively uniform dynamic policy $(\rightarrow_s)_{s \in S}$. Then $M$ is dIP-secure if and only if $M$ is secure with respect to the definition in \[\text{Les06}\].

**Proof.** Due to Theorem 5.5 it suffices to show that in the case of a uniform policy, the functions $\text{dipurge}$ and $\text{dipurge}'$ coincide. Assume indirectly that this is not the case, and let $\alpha$ be a minimal sequence such that there exists a state $s$ and an agent $u$ with $\text{dipurge}(\alpha, u, s) \neq \text{dipurge}'(\alpha, u, s)$. Clearly $\alpha \neq \epsilon$, hence assume that $\alpha = aa'$. Then

$$\text{dipurge}(aa', u, s) = a \text{dipurge}(a', u, s \cdot a) = \text{dipurge}'(a', u, s \cdot a) = \text{dipurge}(aa', u, s),$$

22
which is a contradiction to the choice of $\alpha$.

Hence assume that $\text{dom}(a) \notin \text{disources}(aa', u, s)$. By definition, it follows that $\text{dipurge}(aa', u, s)$ and $\text{dipurge}'(aa', u, s) = \text{dipurge}'(\alpha', u, s \cdot a) = \text{dipurge}(\alpha', u, s \cdot a)$ (the final equality is due to the minimality of $\alpha$).

It hence suffices to show that $\text{dipurge}(\alpha', u, s) = \text{dipurge}(\alpha', u, s \cdot a)$. This easily follows by induction on Lemma A.3: The same actions of $\alpha'$ are transmitted to $u$ when evaluating $\alpha'$ starting in the state $s$ and in $s \cdot a$. □

A.4 Example why symmetric unwindings do not work in the intransitive case

To see why a symmetric unwinding will not work, consider the system in Figure 8. For ease of notation, we will identify states with the action sequences that lead to them starting from the initial state, e.g., with $hdd$, we denote the right-most state in the upper branch of the system.

First note that the system is insecure: The states $hdd$ and $dd$ have different observation functions, however on the path $hdd$, the action $h$ remains invisible for $L$ (formally: $\text{dom}(h) \notin \text{disources}(hdd, L, \epsilon)$). To characterize this with an unwinding, one could proceed as follows: By the same argument as above, the states $hd$ and $d$ must be “equivalent,” as $\text{dom}(h) \notin \text{disources}(hd, u, \epsilon)$. However, if we now perform an additional $d$-action (to allow establishing equivalence between $hdd$ and $dd$), we have the following problem: In the state $d$, there is an allowed interference $D \rightarrow L$ (and in both branches of the system, $D$ “knows” whether $H$ was performed in the initial state). However, the presence of this edge in the policy cannot “transmit” the action $h$ on the path $dd$, since the $h$-action simply was not performed on that path. However, if the interference was allowed in the state $hd$, then clearly, the system would be secure. Therefore, even though we would like to call the states $hd$ and $d$ “equivalent,” this notion is not symmetric, as the presence of an interference $D \rightarrow L$ in one state has different consequences than the presence of such an edge in the other. This leads us to the following notion of an “unwinding”—here, whenever $s \preceq_D t$, the state $s$ should be seen as the state on the branch $s_0 \cdot aa$, whereas $t$ is a state on the branch $s_0 \cdot a$.

B Proofs

In this section we give proofs for the results claimed in the paper.
B.1 Proof of Theorem 3.3

Proof. First, we will show that 1 implies 3. Let $M$ be a dP-secure system. Let $u \in D$. Define for every $s,t \in S$:

$$s \sim_u t \text{ iff for all } \alpha \in A^* : \text{obs}_u(s \cdot \alpha) = \text{obs}_u(t \cdot \alpha).$$

The condition OC\textsuperscript{dp} is satisfied if $\alpha = \epsilon$. For the condition SC\textsuperscript{dp}, we consider $s,t \in S$ with $s \sim_u t$ and let $a \in A$. Then for all $\alpha \in A^*$, we have $s \cdot \alpha \sim_u t \cdot \alpha$ and also $s : a \alpha \sim_u t \cdot a \alpha$. Therefore, $s \cdot a \sim_u t \cdot a$. For the condition LR\textsuperscript{dp}, we consider $a \in A$ and $s \in S$ with $\text{dom}(a) \not\rightarrow_s u$. Since $s$ is a reachable state, it exists $\alpha \in A^*$ with $s = s_0 \cdot \alpha$. The definition of dP-security states, that for every $\beta \in A^*$ the equality of $\text{obs}_u(s \cdot a \beta)$ and $\text{obs}_u(s \cdot \beta)$ holds. Therefore, $s \sim_u a \cdot s$.

We assume that 3 holds and will proof 2. Let $M$ be a dP-secure system. Let $u \in D$. Then it exists an unwinding $\sim_u$ that satisfies LR\textsuperscript{dp}, SC\textsuperscript{dp} and OC\textsuperscript{dp}. We will show by an induction on the combined length of $\alpha$ and $\beta$, that for every state $s \in S$: $\text{dpurge}(\alpha, u, s) = \text{dpurge}(\beta, u, s)$ implies $s \sim_u s \cdot \beta$. The base case with $\alpha = \beta = \epsilon$ is clear. For the inductive step consider $\alpha$ and $\beta$ with $\text{dpurge}(\alpha, u, s) = \text{dpurge}(\beta, u, s)$ for some state $s$. We have to consider two cases:

Case 1: $\alpha = a \alpha'$ for some $a \in A$, $\alpha' \in A^*$ and $\text{dom}(a) \not\rightarrow_s u$. Then we have $\text{dpurge}(a \alpha', u, s) = \text{dpurge}(\alpha', u, s)$. From the property LR\textsuperscript{dp} follows that $s \sim_u s \cdot a$ and from LR\textsuperscript{dp} follows $s \cdot \alpha' \sim_u s \cdot a \alpha'$. Applying the induction hypothesis gives $s \cdot \alpha' \sim_u s \cdot \beta$ which can be combined to $s \cdot \alpha \sim_u s \cdot \beta$.

Case 2: $\alpha = a \alpha'$ and $\beta = b \beta'$ with $\text{dom}(a) \rightarrow_s u$ and $\text{dom}(b) \rightarrow_s u$. From

$$a \text{ dpurge}(\alpha', u, s \cdot a) = \text{dpurge}(a \alpha', u, s)$$

$$= \text{dpurge}(\alpha', u, s)$$

$$= \text{dpurge}(\beta', u, s \cdot b)$$

follows that $a = b$ and $\text{dpurge}(\alpha', u, s \cdot a) = \text{dpurge}(\beta', u, s \cdot a)$. Applying the induction hypothesis gives $s \cdot a \alpha' \sim_u s \cdot b \beta'$. In both cases follows from OC\textsuperscript{dp} that $\text{obs}_u(s \cdot \alpha) = \text{obs}_u(s \cdot \beta)$.

For proofing the missing implication we assume that 1 does not hold. Therefore, it exists $u \in D$, $s \in S$, $a \in A$ and $\alpha \in A^*$ with $\text{dom}(a) \not\rightarrow_s u$ and $\text{obs}_u(s \cdot a \alpha) \neq \text{obs}_u(s \cdot \alpha)$. Therefore, $\text{dpurge}_u(a \alpha, u, s) = \text{dpurge}_u(\alpha, u, s)$ and $\text{obs}_u(s \cdot a \alpha) \neq \text{obs}_u(s \cdot \alpha)$. \hfill \Box

B.2 Proof of Theorem 3.5

Proof. Let $M$ be a dP-secure system with respect to the policy $(\rightarrow_s)_{s \in S}$ and let $u \in D$. Then there is an unwinding $\sim_u$ that satisfies OC\textsuperscript{dp}, SC\textsuperscript{dp} and LR\textsuperscript{dp} (with respect to the policy $(\rightarrow_s)_{s \in S}$). Let be $\sim'_u$ be the a smallest equivalence
relation that satisfies \(\text{SC}_{\text{dp}}\) and \(\text{LR}_{\text{dp}}\) with respect to the policy \((\sim_s)_{s \in S}\). We will show that \(\sim_u' \subseteq \sim_u\). Let \(s, t \in S\) with \(s \sim_u t\) and \(t = s \cdot a\) form some \(a \in A\) with \(\text{dom}(a) \neq \sim_s u\). Therefore it exists \(s' \in S\) with \(s' \sim_u s\) and \(\text{dom}(a) \neq \sim_{s'} u\). From \(s' \sim_u s' \cdot a\) and \(s' \cdot a \sim_u s \cdot a\) follows \(s \sim_u t\).

The other direction of the proof follows directly from the fact, that the policy \((\sim_s)_{s \in S}\) is at least as restrictive as the policy \((\sim_s')_{s \in S}\).

## B.3 Proof of Theorem 4.3

**Proof.** First, we proof the implication from (1) to (2). Let \(M\) be a \text{dotP}-secure system. Let be \(u, v \in D\). We say a string \(\alpha \in A^*\) has the property (1) in a state \(t \in S\) iff

\[
\text{it exists no } b \in A, \beta, \beta' \in A^* \text{ with } \alpha = \beta b \beta' \text{, } v = \text{dom}(b) \text{ and } v \rightarrow_{t, \beta} u. \tag{\text{(*)}}
\]

We define a relation \(\preceq_u^v\) for every \(s, t \in S\) as

\[
(s, t) \in \preceq_u^v \text{ if it holds } \text{obs}_u(s \cdot \alpha) = \text{obs}_u(t \cdot \alpha)
\]

for all \(\alpha \in A^*\) satisfying the property (1) in \(t\).

We show that this relation \(\preceq_u^v\) satisfies the five properties \(\text{OC}_{\text{dotp}}\), \(\text{SC}_{\text{dotp}}\), and \(\text{LR}_{\text{dotp}}\). Let \(s \in S\) and \(a \in A\) with \(\text{dom}(a) \neq v \) and \(v \neq \sim_s u\). Let \(\alpha \in A^*\) that satisfies the property (1) in \(s \cdot a\). Since the system is \text{dotP}-secure, we have \(\text{obs}_u(s \cdot \alpha) = \text{obs}_u(s \cdot \alpha')\). Therefore, \(\preceq_u^v\) satisfies \(\text{LR}_{\text{dotp}}\). Let \((s, t) \in \preceq_u^v\) and assume that \((s \cdot a, t \cdot a) \notin \preceq_u^v\) for some \(a \in A\) with \(\text{dom}(a) \neq v\) or \(\text{dom}(a) = v\) and \(\text{dom}(a) \neq \sim_t u\). Therefore, it exists \(\alpha \in A^*\) that satisfies the property (1) in \(t \cdot a\) and \(\text{obs}_u(s \cdot a) \neq \text{obs}_u(t \cdot a)\). But also \(aa\) satisfies (1) in \(t\). This contradicts that \((s, t) \in \preceq_u^v\). For the last property \(\text{OC}_{\text{dotp}}\) we choose \(\alpha = \epsilon\).

For showing that (2) implies (1), we assume that \(M\) is an insecure system. Therefore, it exists \(u \in D\), \(a \in A\), \(\alpha \in A^*\) such that \(\text{dom}(a) \neq \sim_s u\), \(\alpha\) satisfies the property (1) in \(s \cdot a\) and \(\text{obs}_u(s \cdot \alpha) \neq \text{obs}_u(s \cdot \alpha')\). We set \(v = \text{dom}(a)\) and let \(\preceq_u^v\) a relation that satisfies the properties \(\text{LR}_{\text{dotp}}\) and \(\text{SC}_{\text{dotp}}\). From property \(\text{LR}_{\text{dotp}}\) follows \((s, s \cdot a) \in \preceq_u^v\). Since \(\alpha\) satisfies the property (1), the property \(\text{SC}_{\text{dotp}}\) give that \((s, \alpha, s \cdot a) \in \preceq_u^v\). Therefore, the relation \(\preceq_u^v\) does not satisfy the property \(\text{OC}_{\text{dotp}}\).

For proofing the direction from (1) to (3), we consider a \text{dotP}-secure system \(M\). We assume that it exists \(u \in D\), \(a \in A\), \(\alpha \in A^*\) and \(s \in S\) such that \(\text{dom}(a) \notin \text{dsource}(a, u, s)\) and \(\text{obs}_u(s \cdot a) \neq \text{obs}_u(s \cdot \alpha)\) and the length of \(\alpha\) is minimal for all choices of \(u, s, a\) and \(\alpha\).

Assume it exists \(b \in A\), \(\beta, \beta' \in A^*\) with \(\alpha = \beta b \beta'\) and \(\text{dom}(b) = \text{dom}(a)\) and \(\text{dom}(a) \rightarrow_{s, a} \beta\). Then it exists some set \(V \subseteq D\) with

\[
\text{dsource}(a \beta b \beta', u, s) = V \cup \text{dsource}(b \beta', u, s \cdot a \beta) = V \cup \{\text{dom}(a)\} \cup \text{dsource}(\beta', u, s \cdot a \beta b) .
\]

This contradicts the assumption that \(\text{dom}(a) \notin \text{dsource}(a \alpha, u, s)\). The missing direction from (3) to (1) is obvious.

□
B.4 Proof of Theorem 5.5

Proof. We first consider the dipurge-characterization and then the unwinding conditions.

1. We first show that dIP-security implies the dipurge-characterization. Hence indirectly assume that the system is dIP-secure, and indirectly assume that the dipurge-condition is not satisfied. Then there exists a state $s$, an agent $u$, and sequences $\alpha$ and $\beta$ with \( \text{dipurge}(\alpha, u, s) = \text{dipurge}(\beta, u, s) \), and $\text{obs}_u(s \cdot \alpha) \neq \text{obs}_u(s \cdot \beta)$. We choose $\alpha$ and $\beta$ such that $|\alpha| + |\beta|$ is minimal among all such examples. Clearly, if both $\alpha$ and $\beta$ start with an action that is transmitted to $u$, then this action must be the same: If $\alpha = a\alpha'$ with $\text{dom}(a) \in \text{disources}(aa', u, s)$ and $\beta = b\beta'$ with $\text{dom}(b) \in \text{disources}(b\beta', u, s)$, then $\text{dipurge}(\alpha, u, s)$ starts with $a$, and $\text{dipurge}(\beta, u, s)$ starts with $b$. It thus follows that $a = b$, and hence we could use the state $s' = s \cdot a$ and the sequences $\alpha'$ and $\beta'$ as a counter-example, which contradicts the minimality of $\alpha$ and $\beta$. Hence we can, without loss of generality, assume that $\alpha = a\alpha'$ for some $a$ with $\text{dom}(a) \notin \text{disources}(aa', u, s)$. It thus follows that $\text{dipurge}(\alpha', u, s) = \text{dipurge}(\alpha, u, s) = \text{dipurge}(\beta, u, s)$. Since the system is secure, we also have $\text{obs}_u(s \cdot \alpha') = \text{obs}_u(s \cdot a\alpha') = \text{obs}_u(s \cdot \alpha) \neq \text{obs}_u(s \cdot \beta)$, and hence we again obtain a contradiction to the minimality of $\alpha$ and $\beta$ (with choosing $\alpha'$ instead of $\alpha$).

We now show the converse, i.e., that the dipurge-characterization implies dIP-security. Hence assume that the system satisfies the dipurge-condition. To show interference security, let $\text{dom}(a) \notin \text{disources}(aa', u, s)$ for some agent $u$ and state $s$, we show that $\text{obs}_u(s \cdot a\alpha) = \text{obs}_u(s \cdot \alpha)$. Note that since $\text{dom}(a) \notin \text{disources}(aa', u, s)$, it follows that $\text{dipurge}(aa', u, s) = \text{dipurge}(\alpha, u, s)$. Hence from the prerequisites of the theorem it follows that $\text{obs}_u(s \cdot a\alpha) = \text{obs}_u(s \cdot \alpha)$ as required.

2. We prove that the unwinding condition is also equivalent to dIP-security. First assume that there is a dynamic intransitive unwinding $(\mathcal{D}' \subseteq \mathcal{D})$ for $M$ with respect to $(\neg s)_{s \in S}$. We show that the system is dIP-secure. For this it suffices to show that if $\text{dom}(a) \notin \text{disources}(aa', u, s)$, then $s \cdot a\alpha \not\leq_{D'} s \cdot \alpha$ for some set $D'$ with $u \in D'$. For each prefix $\alpha'$ of $\alpha$, let $D_{\alpha'}$ be defined as

$$D_{\alpha'} = \{ v \in D \mid \text{dom}(a) \notin \text{disources}(aa', v, s) \} .$$

Clearly, if $\alpha'$ is a prefix of $\alpha''$, then $D_{\alpha''} \subseteq D_{\alpha'}$. Since $u \in D_{\alpha}$, it suffices to show that $s \cdot a\alpha' \not\leq_{D_{\alpha'}} s \cdot \alpha'$ for all prefixes $\alpha'$ of $\alpha$. We show the claim by induction. For $\alpha' = \epsilon$, the claim follows from dLR\textsuperscript{dIP}, since $\text{dom}(a) \not\rightarrow_s u$. Hence assume that $\alpha' = \beta b$ for some sequence $\beta$ and action $b$. By induction, we have that $s \cdot a\beta \not\leq_{D_{\beta}} s \cdot \beta$, where $D_{\beta}$ contains all agents $v$ with $\text{dom}(a) \notin \text{disources}(a\beta, v, s)$. Now let $u \in D_{\alpha'}$, it then also follows that $u \in D_{\beta}$. Let $D'$ be defined as in the condition dSC\textsuperscript{dIP}. Since the condition implies $s \cdot a\beta \not\leq_{D'} s \cdot \beta b$, it suffices to show that $u \in D'$. Clearly this is the case if $\text{dom}(b) \in D_{\beta}$, i.e., if $D_{\beta} = D'$. Hence assume this is not the case, by definition of $D_{\beta}$ it then follows that $\text{dom}(a) \in \text{disources}(a\beta, \text{dom}(b), s)$. Since $\text{dom}(a) \notin \text{disources}(aa', u, s)$, it follows that $s \cdot a\alpha' \not\leq_{D_{\alpha'}} s \cdot \alpha'$, as required.
disources(aβb, u, s), this implies that dom(b) \not\succ_{s-a} u, hence u \in D' follows in this case as well.

For the other direction, assume that the system is secure. We define s \preceq_{D'} t if there is a state \hat{s}, an action a and a sequence \alpha, such that s = \hat{s} \cdot a\alpha, t = \hat{s} \cdot \alpha, and for all u \in D', we have dom(a) \notin disources(a\alpha, u, \hat{s}). We claim that this defines a dynamic intransitive unwinding for M with respect to (\rightarrow_s)_{s \in S}.

Since the system is dIP-secure, the condition dOCdIP is obviously satisfied. The condition dLRdIP follows from the fact that if dom(a) \not\succ_s u, then dom(a) \notin disources(a, u, s). It remains to show dSCdIP. Hence let s \preceq_{D'} t, and let \hat{s}, a and \alpha be chosen with the above properties. Let b be an action, and let D'' be the set resulting from applying dSCdIP. It remains to show that for each u \in D'', we have dom(a) \notin disourses(aob, u, \hat{s}). First assume that dom(b) \in D', it then follows from the definition of \preceq_{D'} that dom(a) \notin disources(a\alpha, dom(b), \hat{s}), and hence dom(a) \notin disourses(aob, u, \hat{s}). On the other hand, if dom(b) \notin D', then from u \in D'', we know that dom(b) \not\succ_{\hat{s}-a\alpha} u, and hence from dom(a) \notin disources(a\alpha, u, \hat{s}) (since u \in D') and disources(aob, u, \hat{s}) = disources(a\alpha, u, \hat{s}), it follows that dom(a) \notin disourses(aob, u, \hat{s}) as required.

\[ \square \]

B.5 Proof of Theorem \[5.6\]

**Theorem B.1.** Checking whether a system is not dIP-secure can be done in NP.

**Proof.** The algorithm simply guesses the corresponding values of a, u, s, and \alpha, and verifies that these satisfy obs_u(s \cdot a\alpha) \neq obs_u(s \cdot \alpha) and dom(a) \notin disourses(a\alpha, u, s) in the straightforward way. To show that this gives an NP-algorithm, it suffices to show that the length of \alpha can be bounded polynomially in the size of the system. We show that if the system is insecure, then \alpha can be chosen with |\alpha| \leq |S|^2.

To show this, let \alpha be a path of minimal length satisfying the above. Let F_s and F_{s \cdot a} be the finite state machines obtained when starting the system in the states s and s \cdot a, respectively, and let F = F_s \times F_{s \cdot a}, with initial state (s, s \cdot a). Clearly, in F, we have (s, s \cdot a) \cdot \alpha = (s \cdot \alpha, s \cdot a\alpha). If |\alpha| \geq |S|^2, then \alpha visits a state from F twice, i.e., \alpha contains a nontrivial loop. Such a loop can be removed from \alpha without changing the states that are reached. Clearly, removing a loop does not add information flow, hence the thus-obtained \alpha' also satisfies the prerequisites for \alpha, which is a contradiction to \alpha’s minimality.

\[ \square \]

**Theorem B.2.** For every security definition that is at least as strict as information-flow-security and at least as permissive as interference-security, the problem to determine whether a given system is insecure is NP-hard under \leq m reductions.

27
We reduce from the 3-colorability problem for graphs. Let a graph $G$ with vertices $u_1, \ldots, u_n$ and edges $(u_1^1, u_2^2), \ldots, (u_1^m, u_2^m)$ be given. We construct a system $M^G$ as follows:

- for each vertex $u$, there is an agent $u$ with actions $u = 0$, $u = 1$, and $u = 2$, and there are agents $u \neq 0$, $u \neq 1$, $u \neq 2$, each having exactly one action, which for simplicity we denote with the agent’s name. Additionally, there is an agent $h$ with a single action $h$, and an agent $L$ with a single action $L$.

- for each vertex $u$, we construct a subsystem $C(u)$ (see Figure 9), that models the choice of coloring of $u$ in the graph. In $C(u)$ and all following systems, all transitions that are not explicitly indicated in the graphical representation loop in the corresponding state.

- for each edge $(u, v)$, we construct a subsystem $E(u, v)$ (see Figure 11), which enforces that the colors of $u$ and $v$ must be different. The edges labelled with a transition of the form $u \neq i$ represent two consecutive edges, the first one with the transition $u \neq i$, and the second one labelled with the transition $u \neq j$, where the policy is repeated between the two transitions.

- the system $M^G$ is now designed as shown in Figure 10. We denote the leftmost state with $s_0$. The unlabelled arrows between the different $C(u)$ and $E(u, v)$-nodes express that the final node of one is the starting node of the other. The subsystems $C'(u)$ and $E'(u, v)$ are defined in the same way as $C(u)$ and $E(u, v)$, except that here, in all states we have policies that allow interference between any two agents. With last, we denote the final state of $E(u_1^m, u_2^m)$, and with last', the final state of $E'(u_1^m, u_2^m)$. We define the observation functions as follows: \( \text{obs}_L(\text{last'}) = 1 \), and for all other combinations of agent $u$ and state $s$, \( \text{obs}_u(s) = 0 \).

The main property of $M^G$ is that it is possible to find a path $h\alpha$ from $s_0$ to last that does not transmit $h$ to $L$ if and only if $G$ is 3-colorable:

**Definition B.3.** A path $h\alpha$ is hiding, if $\text{dom}(h) \notin \text{disources}(h\alpha, L, s_0)$, and $s_0 \cdot h\alpha = \text{last}$.

Intuitively, the subsystem $C(u)$ forces the agent $u$ to “choose” a color $i \in \{0, 1, 2\}$, by performing the action $u = i$. For each edge $(u, v)$ or $(v, u)$ in which $u$ is
involved, the agent $u$ later repeats the same transition in the subsystem $E(u, v)$ (or $E(v, u)$). These systems ensure that no two agents that are connected with an edge can choose the same color—if they do, then a dead-end is reached. To ensure that agents are consistent in their choice of colors (i.e., choose the same color in later $E(u, v)$-systems as in the $C(u)$ system, and consequently chooses the same color for each $E(u, v)$-system), we use the following construction: When agent $u$ chooses color $i$ in $C(u)$, the agent $u \neq i$ “receives” interference from $h$. If the agent $u$ later claims to have a color different from $i$, then the only available path is one that allows an interference between $u \neq i$ and $L$, which transmits the information about $h$ to $L$.

**Lemma B.4.** There is a hiding path if and only if $M^G$ is 3-colorable.

**Proof.** First assume that $G$ is 3-colorable, hence let $c: \{u_1, \ldots, u_n\} \rightarrow \{0, 1, 2\}$ be a coloring function such that for all edges $(u, v) \in E$, we have that $c(u) \neq c(v)$. We construct the path $aa$ as the unique path from $s_0 \cdot a$ to $last$ that starts with $L$, does not use loops in any state, and where each agent $u$ chooses the action $u=c(u)$ whenever the current state has more than one non-looping actions. Since $c$ is a 3-coloring, this path does not hit a dead-end in any of the $E(u, v)$-systems, and in particular, reaches the state $last$. Due to the construction of the path, whenever a transaction $u\neq i$ is performed, the action $u_{=i}$ has never been performed on the path, and thus $u\neq i$ has not received $h$. Hence none of the agents interfering with $L$ has received the action $h$, and thus $\text{dom}(h) \notin \text{disources}(aa, L, s_0)$, i.e., $aa$ is hiding.

For the other direction, assume that there is a hiding path $aa$. Without loss of generality, we can assume that $aa$ does not use any actions that loop in the current state. Since $aa$ is hiding, we know that $s_0 \cdot aa = last$, in particular, every subsystem $C(u)$ and $E(u, v)$ is passed when following $aa$ from $s_0$. We can thus define a coloring $c: \{u_1, \ldots, u_n\} \rightarrow \{0, 1, 2\}$ by $c(u) = i$, where $i$ is the unique value such that at the start of $C(u)$, the action $u_{=i}$ is performed by $u$. We claim that this is a 3-coloring of $G$.

For this, first observe that on $aa$, no action $u_{=j}$ is performed for $j \neq c(u)$: Due to the above, no looping action is performed. Now observe that after the performance of $u_{=c(u)}$ in $C(u)$, the agent $u_{\neq c(u)}$ has received the $h$-event. Now after a later performance of the action $u_{=j}$, every path that proceeds to $last$ uses
a transition \( u \neq e(u) \) in a state where \( u \neq c(u) \rightarrow L \), which is a contradiction to the assumption that \( ha \) is hiding.

We now show that for each edge \((u, v)\) of \( G \), we have that \( c(u) \neq c(v) \). Since \( a \alpha \) is hiding, \( a \alpha \) passes through the subsystem \( E(u, v) \). Due to the above, in this subsystems the actions \( u = e(u) \) and \( v = e(v) \) are performed at the relevant states. If \( c(u) \) and \( c(v) \) were equal, this would reach a dead-end state, which is a contradiction, as \( a \alpha \) is hiding, and hence \( s_0 \cdot a \alpha = \text{last} \). \( \Box \)

Since \( M^G \) can clearly be constructed from \( G \) in logarithmic space, the following lemma now proves Theorem B.2.

**Lemma B.5.** — If \( G \) is 3-colorable, then \( M^G \) is not information-flow-secure (and hence not secure with respect to the definition under consideration),

— If \( G \) is not 3-colorable, then \( M^G \) is interference-secure (and hence secure with respect to the definition under consideration).

**Proof.** First assume that \( G \) is 3-colorable. By Lemma B.4, there is a hiding path \( ha \). In particular, \( s_0 \cdot ha = \text{last} \). Since the action \( h \) loops in the state \( s_0 \cdot h \), we can without loss of generality assume that \( \alpha \) does not start with \( h \), and hence \( s_0 \cdot \alpha = \text{last}' \). Since \( ha \) is hiding, we know that \( \text{dom}(h) \notin \text{disources}(ha, L, s_0) \). Since in \( s_0 \), there is no outgoing edge from \( h \), we also know that \( \text{dom}(h)^{s_0} \cap \text{disources}(\alpha, L, s_0) = \emptyset \). Since \( \text{obs}_L(\text{last}) \neq \text{obs}_L(\text{last}') \), it follows that the \( M^G \) is not information-flow-secure.

Now assume that \( G \) is not 3-colorable, and indirectly assume that \( M^G \) is not interference-secure. Since \( L \) is the only agent whose observation function is not constant, this implies that there is a state \( s \), an action \( a \), and a sequence \( \alpha \) such that \( \text{dom}(a) \notin \text{disources}(a \alpha, L, s) \) and \( \text{obs}_L(s \cdot a \alpha) \neq \text{obs}_L(s \cdot \alpha) \). Since \( \text{last}' \) is the only state with an observation different from \( 0 \), we know that \( \text{last}' \in \{ s \cdot a \alpha, s \cdot \alpha \} \). In particular, \( s \) is an ancestor of \( \text{last}' \) in \( M^G \). Since \( \text{dom}(a) \notin \text{disources}(a \alpha, L, s) \), we know that in particular, \( \text{dom}(a) \not\supseteq \text{obs}_L(s \cdot \alpha) \). Since the only ancestor state of \( \text{last}' \) in which the information-flow policy is not the complete relation is \( s_0 \), we know that \( s = s_0 \). Since in \( s_0 \), all agents except for \( h \) may interfere with \( L \), we also know that \( a = h \). Since \( s_0 \cdot ha \neq \text{last}' \) for any \( \alpha \), we know that \( s_0 \cdot \alpha = \text{last}' \). From the design of \( M^G \), it follows that \( s_0 \cdot ha = \text{last} \). Since \( h \notin \text{disources}(ha, L, s_0) \), it follows that \( ha \) is hiding, and thus Lemma B.4 implies that \( G \) is 3-colorable as required. \( \Box \)

**B.6 Proof of Corollary 5.7**

**Proof.** It clearly suffices to provide an FPT algorithm. Such an algorithm can be obtained by the standard dynamic programming approach, by first creating a table with an entry for every choice \( s, t \) and \( D' \), that indicates whether \( s \preceq_{D'} t \) has already been established. The size of the table is \( 2|D| \cdot |S|^2 \). Now initialize the table with \( |S| \cdot |A| \) operations (using the dLR\textsuperscript{dIP} property), and use the dSC\textsuperscript{dIP} condition to add entries to the table until no changes are performed anymore. Then the condition dOC\textsuperscript{dIP} can be verified by checking, for each
Fig. 11. Subsystem $E(u, v)$
agent $u$, and each set $D'$ for which $u \in D'$, whether for all $s \in D'$, we have $\text{obs}_u(s) = \text{obs}_u(t)$. For each choice of $u$ and $D'$, this requires $|S|^2$ accesses to the table. Since the access to the table can be implemented in time $2^{|D|} \cdot \text{poly}|M|$, this completes the proof. □

B.7 Proof of Proposition 5.9

Proof. First assume that $M$ is secure with respect to the transitive definition, and let $\text{dom}(a) \notin \text{dissources}(aa, u, s)$. In particular, it follows that $\text{dom}(a) \not\sim_s u$. Since $M$ is secure, it follows that $\text{obs}_u(s \cdot aa) = \text{obs}_u(s \cdot a)$ (note that this direction is true in every system).

Now assume that $M$ is secure with respect to the intransitive definition, and let $\text{dom}(a) \not\sim_s u$, and let $\alpha$ be an arbitrary action sequence. Since $M$ is essentially transitive and $\text{dom}(a) \not\sim_s u$, we know that $\text{dom}(a) \notin \text{dissources}(aa, u, s)$. Since $M$ is secure with respect to the intransitive definition, it follows that $\text{obs}_u(s \cdot aa) = \text{obs}_u(a \cdot \alpha)$ as required. □

B.8 Proof of Theorem 5.12

Proof. 1. Clearly, if $M$ is not dIP-secure with respect to $(\sim_s)_{s \in S}$, then $M$ is also not dIP-secure with respect to $(\sim'_s)_{s \in S}$. Using induction, we can assume that $(\sim'_s)_{s \in S}$ arose from $(\sim_s)_{s \in S}$ by removing a single redundant edge $e$. Assume that $M$ is not dIP-secure with respect to $(\sim'_s)_{s \in S}$. Hence there are $a \in A, \alpha \in A^*, s \in S, u \in D$ such that $\text{dom}(a) \notin \text{disources}(aa, u, s)$ (with respect to $(\sim'_s)_{s \in S}$) and $\text{obs}_u(s \cdot aa) \neq \text{obs}_u(s \cdot \alpha)$. Since $M$ is dIP-secure, we know that $\text{dom}(a) \in \text{disources}(aa, u, s)$ (with respect to $(\sim_s)_{s \in S}$). In particular, we know that $s \cdot aa \not\sim_s u \cdot a$. It follows that $e$ is not redundant, a contradiction.

2. For all agents $u$ and all states $s$, define $\text{obs}_u^N(s) = [s]_{\sim_u}$, i.e., the equivalence class of $s$ with respect to $\sim_u$ (where $\sim_u$ refers to the original system $M$). The system $N$ is dIP-secure, since $M$ is dIP-secure. Since $e$ is not redundant, there exist $a, \alpha, s, u$ such that $s \cdot aa \not\sim_u s \cdot \alpha$, and $\text{dom}(a) \notin \text{disources}(aa, u, s)$ (with respect to the policy $(\sim'_s)_{s \in S}$). Since $\text{obs}_u^N(s \cdot aa) = [s \cdot aa]_{\sim_u} \neq [s \cdot \alpha]_{\sim_u} = \text{obs}_u^N(s \cdot \alpha)$, it follows that $N$ is not dIP-secure.

3. It easily follows from the definition that if $(\sim'_s)_{s \in S}$ is obtained from $(\sim_s)_{s \in S}$ by removing a set $E$ of edges that are redundant in $(\sim_s)_{s \in S}$, then a remaining edge is redundant in $(\sim'_s)_{s \in S}$ if and only if it is redundant in $(\sim_s)_{s \in S}$. The reason for this is that removing redundant edges does not change the relation $\sim_u$.

□

B.9 Proof of Theorem 5.15

The proof of this theorem, since it highlights an interesting difference between static and dynamic intransitive noninterference: It can easily be shown
Clearly if such

(see EvdMSW11) that if a system is not IP-secure, then there exist a “wit-
ness” for the insecurity consisting of a state $s$, an agent $u$, an action $a$, and a
sequence $\alpha$ such that

1. $\text{dom}(a) \notin \text{sources}(aa, u)$ and $\text{obs}_u(s \cdot aa) \neq \text{obs}_u(s \cdot \alpha)$ (i.e., these values
demonstrate insecurity of the system), and
2. $\alpha$ contains no $b$ with $\text{dom}(a) \rightarrow_s \text{dom}(b)$.

Intuitively, this means that to verify insecurity, it suffices to consider se-
quen
ces in which the “secret” action $a$ is not transmitted even one step.
This feature is crucial for the polynomial-time algorithm to verify IP-security
in EvdMSW11. In the dynamic setting, the situation is different, the above-
mentioned property does not hold. This is in fact the key reason why no “small”
unwinding for dIP-security exists, and why the verification problem is NP-hard.
However, in systems with a uniform policy, we again can prove an analogous
property, even though the proof is more complicated than for the static setting:

**Lemma B.6.** Let $M$ be a system with a policy that is intransitively uniform.
Then $M$ is interference-insecure if and only if there are $a$, $u$, $s$, and $\alpha$ with $\text{dom}(a) \notin \text{sources}(aa, u, s)$, $\text{obs}_u(s \cdot \alpha) \neq \text{obs}_u(s \cdot aa)$, and no $b$ with $\text{dom}(a) \rightarrow_s \text{dom}(b)$. We consider three cases.

- **Assume** $\text{obs}_u(s \cdot a\beta b') \neq \text{obs}_u(s \cdot a\beta \beta')$. Note that $\text{dom}(b) \notin \text{sources}(b', u, s \cdot a\beta)$. Hence choosing $s' = s \cdot a\beta$, $a' = b$, and $\alpha' = b'$ is a contradiction to the minimality of $\alpha$.

- **Assume** $\text{obs}_u(s \cdot b\beta') \neq \text{obs}_u(s \cdot b\beta')$. To show that this again is a contradiction to the minimality of $\alpha$ (starting in the state $s \cdot b$), it suffices to show that $\text{dom}(b) \notin \text{sources}(b', u, s \cdot b \beta)$. Hence indirectly assume that $\text{dom}(b) \in \text{sources}(b \beta', u, s \cdot b \beta)$, and let $\gamma$ be a minimal prefix of $b \beta'$ such that there is some agent $v$ with

- $\text{dom}(b) \in \text{sources}(\gamma, v, s \cdot b \beta)$,
- $\text{dom}(a) \notin \text{sources}(a \beta \gamma, v, s)$.

Since choosing $v = u$ and $\gamma = \beta'$ satisfies these conditions, such a minimal
$\gamma$ exists. Again, consider the point where $v$ “learns” that $a$ was performed,
i.e., let $\gamma = \pi \gamma'$ with

- $\text{dom}(b) \in \text{sources}(\pi, \text{dom}(c), s \cdot b \beta)$, and
- $\text{dom}(c) \rightarrow_{s, b \beta} v$.

Since $\text{dom}(a) \notin \text{sources}(a \cdot b \gamma, v, s)$, and $\pi$ is a prefix of $\gamma$, the prerequisites
of the lemma imply that $v_{s \cdot a \beta}^\pi = v_{s \cdot a \beta}^\gamma$, in particular, $\text{dom}(c) \rightarrow_{s \cdot a \beta \pi} v$. Since $\text{dom}(a) \notin \text{sources}(a \beta \gamma, v, s)$, this implies

$\text{dom}(a) \notin \text{sources}(a \beta \pi, \text{dom}(c), s)$,

hence we have a contradiction to the minimality of $\gamma$. 

33
Assume \( \text{obs}_u(s \cdot a \beta b') = \text{obs}_u(s \cdot a \beta') \) and \( \text{obs}_u(s \cdot \beta b') = \text{obs}_u(s \cdot \beta') \). Since \( \text{obs}_u(s \cdot a \beta b') \neq \text{obs}_u(s \cdot b \beta') \), this implies \( \text{obs}_u(s \cdot a \beta') \neq \text{obs}_u(s \cdot \beta') \). To obtain a contradiction to the minimality of \( \alpha \), it suffices to show that \( \text{dom}(a) \notin \text{disources}(a \beta', u, s) \). Hence indirectly assume that \( \text{dom}(a) \in \text{disources}(a \beta', u, s) \), and let \( \gamma \) be a minimal prefix of \( \beta' \) such that there is an agent \( v \) with

- \( \text{dom}(a) \notin \text{disources}(a \beta \gamma, v, s) \), and
- \( \text{dom}(a) \in \text{disources}(a \beta \gamma, v, s) \).

Since choosing \( v = u \) and \( \gamma = \beta' \) satisfies these conditions, such a minimal \( \gamma \) exists. Now consider the step where \( v \) “learns” \( a \), which clearly happens inside \( \gamma \) (as \( \text{dom}(a) \notin \text{disources}(a \beta b \gamma, v, s) \)). Hence \( \gamma = \pi \pi' \) with

- \( \text{dom}(a) \in \text{disources}(a \beta \pi, \text{dom}(c), s) \), and
- \( \text{dom}(c) \rightarrow_s a \beta \pi, v \).

Since \( \text{dom}(a) \notin \text{disources}(a \beta b \gamma, v, s) \), we have \( \text{dom}(b) \notin \text{disources}(b \gamma, v, s \cdot a \beta) \). Since \( \pi \) is a prefix of \( \gamma \), this implies \( \text{dom}(b) \notin \text{disources}(b \pi, v, s \cdot a \beta) \).

The conditions of the lemma this imply that \( u_{s \cdot a \beta \pi}^{\uparrow} = u_{s \cdot a \beta \pi}^{\uparrow} \). In particular, this implies \( \text{dom}(c) \rightarrow_s a \beta \pi, v \). Since \( \text{dom}(a) \notin \text{disources}(a \beta b \gamma, v, s) \), this implies \( \text{dom}(a) \notin \text{disources}(a \beta \pi, \text{dom}(c), s) \), which is a contradiction to the minimality of \( \gamma \).

We now show a similar fact which allows us to easily verify whether a policy is intransitively uniform: To verify uniformity, it again suffices to consider action sequences in which the “secret” action is not even transmitted a single step. This is shown in the following Lemma:

**Lemma B.7.** If a policy for a system is not intransitively uniform, there is an agent \( u \), an action \( a \), a sequence \( \alpha \), and a state \( s \) such that

1. \( \text{dom}(a) \notin \text{disources}(a \alpha \alpha, u, s) \),
2. \( u_{s \cdot a \alpha \alpha}^{\uparrow} \neq u_{s \cdot a \alpha}^{\uparrow} \),

and contains no \( b \) with \( \text{dom}(a) \rightarrow_s \text{dom}(b) \).

**Proof.** Choose \( u, a, s, \) and \( \alpha \) such that \( |\alpha| \) is minimal, and indirectly assume that \( \alpha = \beta b' \) for some sequences \( \beta \) and \( \beta' \), where \( \text{dom}(a) \rightarrow_s \text{dom}(b) \). Note that this implies

\[
\text{dom}(b) \notin \text{disources}(b \beta', u, s \cdot a \beta),
\]

which we will use throughout the proof. We consider three cases:

- **Assume that** \( u_{s \cdot a \beta b'}^{\uparrow} \neq u_{s \cdot a \beta}^{\uparrow} \). We choose \( s' = s \cdot a \beta, \ a' = b, \) and \( \alpha' = \beta' \).

This is a contradiction to the minimality of \( \alpha \), since \( |\alpha'| < |\beta'| \).

- **Assume that** \( u_{s \cdot \beta b'}^{\uparrow} \neq u_{s \cdot \beta}^{\uparrow} \). We choose \( s' = s \cdot \beta, \ a' = b, \) and \( \alpha = \beta' \) and obtain a contradiction in the same way as in the above case. For this, it suffices to prove that \( \text{dom}(b) \notin \text{disources}(b \beta', u, s \cdot \beta) \). Hence assume indirectly that \( \text{dom}(b) \in \text{disources}(b \beta', u, s \cdot \beta) \). Let \( \gamma \) be a minimal prefix of \( b \beta' \) such that there is an agent \( v \) with
Proof. 1. First assume that there is a uniform dynamic intransitive unwind-
ing satisfying \( uPC^{dIP} \), \( uSC^{dIP} \), and \( uLR^{dIP} \), and indirectly assume that the policy is not intransitively uniform. Due to Lemma 5.7 there exist \( a, u, s \), and \( \alpha \) such that \( \text{dom}(a) \notin \text{discourses}(a\alpha, u, s) \), \( u^{s-a\alpha} \neq u^{s-\alpha} \), and \( \alpha \) does not contain any \( b \) with \( \text{dom}(a) \Rightarrow_s \text{dom}(b) \). Let \( v = \text{dom}(a) \). Let \( \sim_{s,v}^{s} \) be an equivalence relation satisfying \( uPC^{dIP} \), \( uSC^{dIP} \), and \( uLR^{dIP} \). It suffices to show that \( s \cdot a\alpha \sim_{s,v}^{s} s \cdot \alpha \) to obtain a contradiction to \( uPC^{dIP} \).

Using these lemmas, we can now prove Theorem 5.15:

\[
\begin{align*}
&\text{dom}(b) \in \text{discourses}(\gamma, v, s \cdot \beta), \\
&\text{dom}(a) \notin \text{discourses}(\alpha \beta \gamma, v, s).
\end{align*}
\]

Since \( \gamma = b\beta \) and \( v = u \) satisfies these conditions, such a minimal choice of \( \gamma \) and \( v \) exists. Now consider the position where \( v \) “learns” \( b \), i.e., let \( \gamma = \pi \alpha \), such that the action \( c \) transmits the \( b \)-action to \( v \), i.e., we have that

\[
\begin{align*}
&\text{dom}(b) \in \text{discourses}(\pi, \text{dom}(c), s \cdot \beta), \\
&\text{dom}(c) \Rightarrow_{s,\beta \gamma} v.
\end{align*}
\]

Note that \( \pi \) is a proper prefix of \( \gamma \). Since \( \text{dom}(a) \notin \text{discourses}(a\beta \gamma, v, s) \), it follows that \( \text{dom}(a) \notin \text{discourses}(a\beta \pi, v, s) \). Hence we know by the minimality of \( \alpha \) that \( v^1_{s,\beta \pi} = v^1_{s-a\beta \pi} \). In particular, \( \text{dom}(c) \Rightarrow_{s-a\beta \pi} v \). We now have the following:

\[
\begin{align*}
&\text{Due to the above, we know that dom}(b) \in \text{discourses}(\pi, \text{dom}(c), s \cdot \beta), \\
&\text{since dom}(a) \notin \text{discourses}(a\beta \gamma, v, s), \text{ we know that } \text{dom}(a) \notin \text{discourses}(a\beta \pi, \text{dom}(c), s).
\end{align*}
\]

Since \( \pi \) is a proper prefix of \( \gamma \), this is a contradiction to the minimality of \( \gamma \).

\[
\text{Assume that } u^1_{s-a\beta \beta} = u^1_{s-a\beta \beta'} \text{ and } u^1_{s-a\beta \beta'} = u^1_{s-a\beta \beta}. \text{ Since } u^1_{s-a\beta \beta'} \neq u^1_{s-a\beta \beta}, \text{ it then follows that } u^1_{s-a\beta \beta'} \neq u^1_{s-a\beta \beta}. \text{ It suffices to show that } \text{dom}(a) \notin \text{discourses}(a\beta \beta', u, s), \text{ we then have a contradiction to the minimality of } \alpha. \text{ Hence indirectly assume that } \text{dom}(a) \in \text{discourses}(a\beta \beta', u, s). \text{ Let } \gamma \text{ be a minimal prefix of } \beta' \text{ such that there is some } v \text{ such that}
\]

\[
\begin{align*}
&\text{dom}(a) \notin \text{discourses}(a\beta \beta \gamma, v, s), \\
&\text{dom}(a) \in \text{discourses}(a\beta \beta, v, s).
\end{align*}
\]

Since \( \gamma = \beta' \) and \( v = u \) satisfy these conditions, such a minimal choice exists. Similarly as before, look at the action where \( a \) is forwarded to \( v \), i.e., let \( \gamma = \pi \alpha \) such that

\[
\begin{align*}
&\text{dom}(a) \in \text{discourses}(a\beta \pi, \text{dom}(c), s), \\
&\text{dom}(c) \Rightarrow_{s-a\beta \pi} v.
\end{align*}
\]

Since \( \text{dom}(a) \notin \text{discourses}(a\beta \beta \gamma, v, s) \) and \( \text{dom}(a) \Rightarrow_{s} \text{dom}(b) \), it follows that \( \text{dom}(b) \notin \text{discourses}(b\gamma, v, s-a\beta) \). Since \( \pi \) is a prefix of \( \gamma \), this implies \( \text{dom}(b) \notin \text{discourses}(b\pi, v, s \cdot a\beta) \). The minimality of \( \alpha \) implies that \( v^1_{s-a\beta \pi} = v^1_{s-a\beta \pi} \), in particular, \( \text{dom}(c) \Rightarrow_{s-a\beta \pi} v \). Since \( \text{dom}(a) \notin \text{discourses}(a\beta \beta \pi, v, s) \), we obtain

\[
\begin{align*}
&\text{dom}(a) \notin \text{discourses}(a\beta \beta \pi, \text{dom}(c), s), \\
&\text{from the above, we know that } \text{dom}(a) \in \text{discourses}(a\beta \pi, \text{dom}(c), s).
\end{align*}
\]

This contradicts the minimality of \( \gamma \), since \( \pi \) is a proper prefix of \( \gamma \).

\[\square\]
Clearly, $\text{dom}(a) \not\sim_a s \cdot u$, hence $u\text{LR}^{\text{dIP}}$ implies $s \sim_s^{\text{dom}(a)} s \cdot a$, i.e., $s^u s \cdot a$. Note that for all $a'$ appearing in $\alpha$, we have that $\text{dom}(a) \not\sim_a s \cdot \alpha$. Hence applying $u\text{SC}^{\text{dIP}}$ for each $a'$, we obtain $s \cdot a \alpha \sim_s^{\text{dom}(a)} s \cdot \alpha$ as required.

For the converse, assume that for all $\text{dom}(a) \notin \text{disources}(a \alpha, u, s)$, we have that $u_{s, a}^\alpha = u_{s, a}^\alpha$, and let $s_0$ be a state, and let $v$ and $u$ be agents. We define

$s \sim_u^{s_0, v} t$ iff for all sequences $\alpha$ that contain no $b$ with $v \rightarrow s_0$

\[
\text{dom}(b), \quad u_{s, a}^{\alpha, s} = u_{s, a}^{\alpha, s}.
\]

Clearly, $\sim_u^{s_0, v}$ is an equivalence relation and satisfies $u\text{PC}^{\text{dIP}}$ (choose $\alpha = \epsilon$). For showing $u\text{SC}^{\text{dIP}}$, let $s \sim_u^{s_0, v} t$, and let $v \not\rightarrow_{s_0} \text{dom}(a)$. To show the required condition $s \cdot a \sim_u^{s_0, v} t \cdot a$, let $\alpha$ be a sequence containing no $b$ with $v \rightarrow_{s_0} b$. Since $v \not\rightarrow_{s_0} \text{dom}(a)$, the sequence $a \alpha$ satisfies the same condition, and hence from $s \sim_u^{s_0, v} t$, it follows that $u_{s, a}^{\alpha, s} = u_{s, a}^{\alpha, s}$ as required.

Finally, consider $u\text{LR}^{\text{dIP}}$. Let $\text{dom}(a) \not\sim_s u \cdot a$. To show that $s \sim_s^{u, \text{dom}(a)} s \cdot a$, let $\alpha$ be such that no $b$ with $\text{dom}(a) \rightarrow_{s_0} \text{dom}(b)$ appears in $\alpha$, we need to show that $u_{s, a}^{\alpha, s} = u_{s, a}^{\alpha, s}$. This follows from the prerequisites, since clearly, $\text{dom}(a) \notin \text{disources}(a \alpha, u, s)$.

2. (a) Assume that the system is noninterference-secure, let $s_0$ be a state, and let $v$ and $u$ be agents. We define:

$s \sim_u^{s_0, v} t$ iff for all sequences $\alpha$ that contain no $b$ with $v \rightarrow s_0$

\[
\text{dom}(b), \quad \text{obs}_u(s \cdot a) = \text{obs}_u(t \cdot a).
\]

Clearly, $\sim_u^{s_0, v}$ is an equivalence relation and satisfies $u\text{OC}^{\text{dIP}}$ (choose $\alpha = \epsilon$). For showing $u\text{SC}^{\text{dIP}}$, let $s \sim_u^{s_0, v} t$, and let $a \in A$ with $v \not\rightarrow_{s_0} \text{dom}(a)$. We need to show that for all $\alpha$ containing no $b$ with $v \rightarrow_{s_0} \text{dom}(b)$, we have $\text{obs}_u(s \cdot a \alpha) = \text{obs}_u(t \cdot a \alpha)$. This trivially follows from $s \sim_u^{s_0, v} t$, since $\alpha' = a \alpha$ also does not contain a $b$ with $v \rightarrow_{s_0} \text{dom}(b)$.

Finally, consider $u\text{LR}^{\text{dIP}}$. Let $\text{dom}(a) \not\sim_s u \cdot a$. We need to show that $s \sim_s^{u, \text{dom}(a)} s \cdot a$. Hence let $\alpha$ be a sequence containing no $b$ with $\text{dom}(a) \rightarrow_{s_0} \text{dom}(b)$. We need to show that $\text{obs}_u(s \cdot a) = \text{obs}_u(s \cdot a \alpha)$. Since the system is noninterference-secure, it suffices to show that $\text{dom}(a) \notin \text{disources}(a \alpha, u, s)$. This follows trivially since $\text{dom}(a) \not\sim_s u$, and $\alpha$ does not contain any $b$ with $\text{dom}(a) \rightarrow_{s_0} \text{dom}(b)$.

(b) Assume that the system is not dIP-secure. Due to Lemma B.6, there is a state $s$, an agent $u$, an action $a$, and a sequence $\alpha$ with $\text{dom}(a) \notin \text{disources}(a \alpha, u, s)$, $\text{obs}_u(s \cdot a) \neq \text{obs}_u(s \cdot a \alpha)$, and $\alpha$ does not contain any $b$ with $\text{dom}(a) \rightarrow_{s_0} \text{dom}(b)$. Let $v = \text{dom}(a)$, and let $\sim_u^{s_0, v}$ be an equivalence relation on $S$ that satisfies $u\text{OC}^{\text{dIP}}$, $u\text{SC}^{\text{dIP}}$, and $u\text{LR}^{\text{dIP}}$. It suffices to show that $s \alpha \sim_u^{s_0, v} s \cdot a \alpha$. Clearly we have that $v \not\sim_u s \cdot a$. Therefore, (recall that $v = \text{dom}(a)$), $u\text{LR}^{\text{dIP}}$ implies $s \sim_u^{s_0, v} s \cdot a$. Note that for all $b \in \alpha$, we have that $\text{dom}(a) \not\sim_s \text{dom}(b)$. Hence applying $u\text{SC}^{\text{dIP}}$ repeatedly, we obtain $s \cdot a \alpha \sim_u^{s_0, v} s \cdot a$, which completes the proof.

\[\Box\]
References

BDRS08. C. Boettcher, R. DeLong, J. Rushby, and W. Sifre. The MILS component integration approach to secure information sharing. In Proc. 27th IEEE/AIAA Digital Avionics Systems Conference, pages 1.C.2–1–1.C.2–14, October 2008.

BS09. Niklas Broberg and David Sands. Flow-sensitive semantics for dynamic information flow policies. In Stephen Chong and David A. Naumann, editors, PLAS, pages 101–112. ACM, 2009.

Den76. D. E. Denning. A lattice model of secure information flow. Communications of the ACM, 19(5):263–243, 1976.

DF99. R. G. Downey and M. R. Fellows. Parameterized Complexity. Monographs in Computer Science. Springer, 1999.

EvdMSW11. Sebastian Eggert, Ron van der Meyden, Henning Schnoor, and Thomas Wilke. The complexity of intransitive noninterference. In IEEE Symposium on Security and Privacy, pages 196–211. IEEE Computer Society, 2011.

GM82. J.A. Goguen and J. Meseguer. Security policies and security models. In Proc. IEEE Symp. on Security and Privacy, pages 11–20, Oakland, 1982.

GM84. J.A. Goguen and J. Meseguer. Unwinding and inference control. In IEEE Symp. on Security and Privacy, 1984.

HY87. J.T. Haigh and W.D. Young. Extending the noninterference version of MLS for SAT. IEEE Trans. on Software Engineering, SE-13(2):141–150, Feb 1987.

Les06. R. Leslie. Dynamic intransitive noninterference. Proc. IEEE International Symposium on Secure Software Engineering, 2006.

MS04. H. Mantel and D. Sands. Controlled declassification based on intransitive noninterference. In Proc. Asian Symp. on Programming Languages and Systems, volume 3302 of LNCS, pages 129–145. Springer-Verlag, November 2004.

Rus92. J. Rushby. Noninterference, transitivity, and channel-control security policies. Technical Report CSL-92-02, SRI International, Dec 1992.

vdM07. Ron van der Meyden. What, indeed, is intransitive noninterference? In Joachim Biskup and Javier Lopez, editors, European Symposium On Research In Computer Security (ESORICS), volume 4734 of Lecture Notes in Computer Science, pages 235–250. Springer, 2007.

vdMZ10. R. van der Meyden and C. Zhang. A comparison of semantic models for noninterference. Theoretical Computer Science, 411(47):4123–4147, October 2010.

ZM04. Lantian Zheng and Andrew C. Myers. Dynamic security labels and noninterference (extended abstract). In Theodosis Dimitrakos and Fabio Martinelli, editors, Formal Aspects in Security and Trust, pages 27–40. Springer, 2004.