Complexity of Fairness Constraints for the Dolev-Yao Attacker Model

Jan Cederquist
SQIG-IT and DEI-IST
Instituto Superior Técnico, Tagus Park
2780-990 Porto Salvo, Portugal
jan.cederquist@ist.utl.pt

Mohammad Torabi Dashti
ETH Zürich
Universitätstrasse 6
8092 Zürich, Switzerland
torabidm@inf.ethz.ch

ABSTRACT

Liveness properties do, in general, not hold in the Dolev-Yao attacker model, unless we assume that certain communication channels are resilient, i.e. they do not lose messages. The resilient channels assumption can be seen as a fairness constraint for the Dolev-Yao attacker model. Here we study the complexity of expressing such fairness constraints for the most common interpretation of the Dolev-Yao model, in which the attacker is the communication medium. We give reference models which describe how resilient channels behave, with unbounded and bounded communication buffers. Then we show that, for checking liveness security requirements, any fairness constraint that makes this common interpretation of the Dolev-Yao model sound and complete w.r.t. the unbounded (resp. bounded) reference model is not an ω-regular (resp. locally testable) language. These results stem from the complexity of precisely capturing the behavior of resilient channels, and indicate that verification of liveness security requirements in this interpretation of the Dolev-Yao model cannot be automated efficiently.

1. INTRODUCTION

Many security requirements studied in the literature can be encoded as safety properties, cf. [19]. Nonetheless, in this paper we are concerned with security requirements which are liveness properties. *Timeliness* for contract signing protocols is such a requirement. Timeliness stipulates that each honest participant of the protocol is able to unilaterally (i.e. with no help from the opponent) terminate the protocol [5].

In message passing settings, distributed algorithms do not satisfy any (non-trivial) liveness property if transmitted messages can be destroyed, cf. Gray's *generals paradox*. Therefore, distributed algorithms which aim at liveness properties rely upon communication channels that guarantee some level of fairness, e.g. channels are *fair lossy* [17]. Similarly, contract signing protocols, which aim at timeliness, assume that (a subset of) the communication channels in the system are *resilient*. A resilient channel guarantees to *eventually* deliver all the messages that are sent to it [5].

Assuming that a channel is resilient does not prevent the attacker from writing to, and reading from, the channel, or delaying messages and changing the order between them. But, it does prevent the attacker from destroying communicated messages. This is incongruent with the (standard) Dolev-Yao attacker model [10], in which the attacker can stop the communication permanently. Resilient channels are however justified by noting that in practice two principals who are willing to communicate can eventually establish a (fair lossy) channel, despite the attacker's obstructions. For instance, in wireless networks, given that jamming is only locally sustainable, the principals can always move to an area where they can send and receive messages. Ultimately if the principals fail to properly establish a channel over computer networks, they can resort to other communication means, e.g. postal services, which are reliable and protected by law.

Physical fairness assumptions, such as resilient channels, must be reflected in the formalizations of the security protocols that aim at liveness goals. In particular, the resilient channels assumption can be seen as a *fairness constraint* for the attacker model. Informally, a fairness constraint is a filter, which excludes certain executions of the model from further analyzes. These excluded executions represent "unrealistic" behaviors in the model that do not reflect the actual system under study. Intuitively, a fairness constraint that formalizes the resilient channels assumption would filter out all the executions in which the attacker destroys messages over a resilient channel.

When adding fairness constraints to attacker models, some level of caution is necessary for checking liveness properties. Consider, for instance, the situation when the agent *Alice* wants to send a message m to the agent *Bob*. *Bob*, on the other hand, wants to receive the message $h(m)$, where h is a public hash function. When *Bob* receives $h(m)$ he performs a certain action α. We are interested in the following liveness property: α eventually occurs. Now we add the Dolev-Yao attacker and *fair scheduling assumption* to the system. The attacker learns m, and the fair scheduling assumption "forces" the attacker to eventually generate $h(m)$ and send it to *Bob*, who will then perform α. This is clearly unsound for the aforementioned liveness property, since *Bob* would never perform α in the system without the attacker.

In general, given an attacker model A, one can ask for fairness assumptions that, if added to A, results in a sound system (w.r.t. a certain reference model). This is the central question of our paper. More precisely, we focus on a particular interpretation (or abstraction) of the Dolev-Yao

SAC'11 March 21-25, 2011, TaiChung, Taiwan.

attacker model, referred to as **I** in the following, and study the complexity of expressing fairness constraints (seen as formal languages) which if added to **I** would correctly reflect the resilient channels assumption. In interpretation **I**, the attacker plays the role of the communication media. Our study is targeted at this interpretation because it has been extensively used in the existing algorithms and tools for automatic verification of (safety) security properties; see our related work, below.

In interpretation **I**, *send* actions of honest participants are modeled by adding the transmitted messages to the knowledge of the attacker, and *receive* actions of honest participants correspond to checking whether the messages, which are to be received, belong to the closure of the attacker's knowledge under certain deduction rules (which often reflect the *ideal cryptography* assumption). Therefore, the attacker does not perform any "actions" explicitly; all the attacker does is captured via modifying and checking its knowledge set. This feature of interpretation **I** is of paramount significance to automatic verification of security protocols, as the attacker is tightly coupled with the honest participants and would therefore generate only the messages that can be possibly consumed by the participants [8, 20, 7]. This reduces the (infinite) space of the messages that must be searched for establishing or refuting existence of flaws in security protocols. We remark that the algorithms of [8, 20, 7] are not tailored towards checking security requirements which can be expressed as liveness properties.

Contributions.

Our study defines a *reference model* that is meant to reflect the actual hostile environment in which security protocols are executed; namely, where the communication buffer is external to the attacker process, albeit being fully controlled by it. In our reference model, hereafter called R_n, there is a bound $n \in \mathbb{N}_+ \cup \{\infty\}$ on the capacity of the communication buffer, with $\mathbb{N}_+ = \mathbb{N} \setminus \{0\}$. We consider two distinct cases: (1) when $n = \infty$, i.e. the model R_∞, the communication media is an unbounded buffer, and (2) when $n \in \mathbb{N}_+$, the size of the communication buffer is finite. The resilient channels assumption can be added to the reference model by imposing a standard *fair scheduling* constraint, which accepts an execution as fair iff all the honest processes have fully progressed through their local algorithms. [1]

Then, we investigate a common interpretation (or, abstraction) of the Dolev-Yao model, namely interpretation **I**, in which the attacker *is* the communication media. We ask which fairness constraints can be added to interpretation **I** so that the resulting model would be *sound* and *complete* w.r.t. the reference model, for checking liveness security requirements. Soundness, informally, means that any liveness property that holds in the interpretation also holds in the reference model, and completeness means that any liveness property that holds in the reference model also holds in the interpretation. Soundness and completeness of interpretation **I** under a fairness assumption would entail that the fairness assumption correctly reflects the behavior of resilient channels in **I**, according to the reference model. These notions are made precise in the paper.

We prove that if a fairness constraint makes the interpretation **I** sound and complete w.r.t. the reference model with unbounded communication buffers (i.e. R_∞), then it is *not* an *ω-regular language*. Expressing such fairness constraints would therefore fall beyond temporal logics, such as LTL and CTL*, which are typically used for specifying fairness constraints in automated verification algorithms and tools. Similarly, we show that if a fairness constraint makes the interpretation **I** sound and complete w.r.t. the reference model with bounded communication buffers (e.g. R_1), then it is *not* a *locally-testable language*. The Locally testable languages is a proper subset of star-free regular languages [18], which in turn are as expressive as LTL confined to finite sequences [11].

On the positive side, the very same feature of the interpretation **I**, which results in the high complexity of fairness constraints, plays a favorable role in verification of protocols with bounded buffers. Namely, we prove that if a liveness property holds in a protocol with buffer capacity n, it will also hold when buffers have capacity larger than n. The result allows deriving general security guarantees from verification of small bounded models. These notions are made precise in the paper.

Our study is orthogonal to the ideal cryptography assumption, often associated to the Dolev-Yao threat model.

Related work.

Formal verification of liveness in security protocols has been mostly limited to properties of the family of *optimistic fair exchange* protocols, which subsumes contract signing protocols. Detecting non-termination in these protocols is challenging, e.g. see [13] for a subtle attack undermining the termination of a protocol proposed by Zhou et al., and see [5] for identifying non-termination problems in various existing protocols.

Notable examples of automated symbolic verification of liveness properties of optimistic fair exchange protocols are Kremer and Raskin's game-based semantics [15], Kähler, Küsters and Truderung's decision procedure for certain game-based properties (comprising those formalized in [15]) for a large class of security protocols [14], Armando, Carbone and Compagna's encoding of contract signing requirements in LTL [4], and Wei and Heather's work [24] on (semi-automated) verification of optimistic fair exchange protocols in the PVS proof checker using the stable failures semantics of CSP. These works all devise fairness constraints which are applicable to a particular setting, and do not consider the expressive complexity of fairness constraints required for the Dolev-Yao attacker model, in general. Liveness properties have been omitted from symbolic verifications in a number of other works, e.g. Bella and Paulson's [6] formalization of non-repudiation protocols in Isabelle/HOL, and Abadi and Blanchet's verification of a certified email protocol [1].

In the computational setting, Cortier, Küsters and Warinschi [9] study fair scheduling for branching security protocols, e.g. optimistic contract signing protocols, with branching properties. Since the attacker is a legitimate (and indispensable) participant in these protocols, a scheduler external to the attacker is considered. A fair scheduler process can stop only when no other process (honest, attacker, buffers,

[1] Modeling resilient channels (as fairness constraints in the Dolev-Yao attacker) cannot be simplified to "each submitted message should be received". This is because if, in an execution a (malicious) process submits a message which is not deliverable, i.e. no process would receive it, then that execution should not be excluded by the fairness constraint from further analyses.

etc.) can take any further action. As the scheduler does not "know" whether a process can take actions or not, before giving it the turn, it follows that there exist protocols for which no fair scheduler can be found.

Structure of the paper.

Preliminary notions are introduced in Section 2. In Section 3, we define the reference models. Interpretation **I** is formalized in Section 4. Section 5 concerns the complexity of fairness constraints for interpretation **I**. There the main results of the paper are proved. Section 6 focuses on sound abstractions for bounded communication buffers in the presence of the attacker. Section 7 concludes the paper.

2. PRELIMINARIES

Properties, safety and liveness.

For a finite set of symbols Σ, we write Σ^* for the set of all finite sequences of elements of Σ, containing the empty sequence ϵ. We let $\Sigma^+ = \Sigma^* \setminus \{\epsilon\}$. An infinite sequence σ on Σ is a total function $\sigma : \mathbb{N} \to \Sigma$. The set of all infinite sequences on Σ is denoted Σ^ω. The concatenation of two sequences $\sigma \in \Sigma^*$ and $\sigma' \in \Sigma^* \cup \Sigma^\omega$ is denoted by $\sigma \cdot \sigma'$. For $\sigma \in \Sigma^*$ and $\sigma' \in \Sigma^* \cup \Sigma^\omega$, we write $\sigma \leq \sigma'$ if σ is a prefix of σ', i.e. $\exists \sigma'' \in \Sigma^* \cup \Sigma^\omega.\ \sigma' = \sigma \cdot \sigma''$.

A *property* on Σ^ω is a subset of Σ^ω. [2] Properties can be divided into two general classes: safety properties (stating that something bad will never occur) and liveness properties (stipulating that something good will eventually occur). Formally, a property ϕ is a *safety* property on Σ^ω iff ϕ can be violated in finite time: $\forall \sigma \in \Sigma^\omega.\ \sigma \notin \phi \implies \exists \sigma' \leq \sigma. \forall \sigma'' \in \Sigma^\omega.\ \sigma' \cdot \sigma'' \notin \phi$. A property ϕ is a *liveness* property on Σ^ω iff any finite trace can be extended so that it satisfies ϕ: $\forall \sigma \in \Sigma^*. \exists \sigma' \in \Sigma^\omega.\ \sigma \cdot \sigma' \in \phi$. Any property can be written as the intersection of a safety and a liveness property [3].

A property $\phi \subseteq \Sigma^\omega$ is said to *saturate* an equivalence relation E on Σ^ω iff, for any equivalence class c_E of the relation, either $c_E \subseteq \phi$ or $c_E \cap \phi = \emptyset$. That is, a property saturates an equivalence relation, if it is a property of the equivalence classes of the relation. The notion of saturation is later used to define when a property is not sensitive to (finite) occurrences of certain symbols.

Processes and protocols.

A security protocol consists of a finite number of honest processes, and an attacker which comprises all corrupted parties and controls the communication media. Below, we give a simple syntax and semantics to honest processes, which is a subset of basic CCS process algebra with prefix iteration [2]. Various attacker models are formalized in the subsequent sections.

Fix a finite alphabet of actions A, not containing the silent action τ, and set $A_\tau = A \cup \{\tau\}$. We assume that A can be partitioned into three disjoint subsets; $\mathsf{atom} = \{a_1, \cdots, a_n\}$, $\mathsf{atom}' = \{a' \mid a \in \mathsf{atom}\}$ and $\mathsf{atom}_\mathbf{A} = \{i(a), d(a) \mid a \in \mathsf{atom}\}$. Intuitively, $a \in \mathsf{atom}$ denotes a send action and a' is the corresponding receive action. The actions $i(a)$ and $d(a)$ are attacker-specific and they are not directly visible to honest processes. Informally, they denote inserting/deleting messages to/from the communication buffer. This formalization amounts to a finite fixed set of messages being communicated among participants. We thus use the terms *message* and *action symbol* interchangeably.

[2]This notion of "first order trace properties" does not cover, e.g., information flow control requirements which concern sets of traces, hence being "higher order".

$$\frac{}{\alpha \cdot P \xrightarrow{\alpha} P} \qquad \frac{}{\alpha^* \cdot P \xrightarrow{\alpha} \alpha^* \cdot P} \qquad \frac{P \xrightarrow{\alpha'} P'}{\alpha^* \cdot P \xrightarrow{\alpha'} P'}$$

Figure 1: Operational semantics for regular processes

Remark 1. *Formal specification languages for security protocols typically allow variable binding to facilitate receiving an unbounded number of possible messages. These are deliberately omitted in our formalization, in order to simplify the presentation. Using a simple model for honest processes benefits us in two ways: First, it greatly simplifies our formalization, and helps to present the main ideas of the paper without unnecessary cluttering. Second, any specification language used for realistic security protocols needs to include the minimal expressiveness of our setting. Therefore, the complexity results discussed here persist when such richer languages are used.*

A labeled transition system (LTS) is a tuple $(\mathbf{P}, P^0, \to)$, where $\mathbf{P}$ is a set of states (or processes), $P^0 \in \mathbf{P}$ is the initial state, and $\to \in \mathbf{P} \times A \times \mathbf{P}$ is the transition relation. We write $P \xrightarrow{a} Q$ for $(P, a, Q) \in \to$. This notation is extended to $P \xrightarrow{\xi} Q$ for $\xi \in A^+$. An LTS is *deterministic*, iff for any $P \in \mathbf{P}$ and $a \in A$, we have $P \xrightarrow{a} Q \wedge P \xrightarrow{a} Q' \implies Q = Q'$. An *execution* of $P \in \mathbf{P}$ is an infinite sequence of elements of A that results from unrolling P w.r.t. the transition relation $\to$. As a convention, terminating executions (ending in a deadlock state) are extended to infinite executions by repeating τ at the end of the execution, cf. [3]. For instance, if $P \xrightarrow{a} Q$, such that $\neg \exists Q', a.\ Q \xrightarrow{a} Q'$, then $a \cdot \tau^\omega$ is an execution of P, where τ^ω is an infinite sequence of τs. An execution *belongs* to $L = (\mathbf{P}, P^0, \to)$ iff it is an execution of P^0.

The set of *regular processes* is defined inductively as: $\mathbf{0}$ is a regular process; if P is a regular process, then so are $\alpha \cdot P$ and $\alpha^* \cdot P$, with $\alpha \in \mathsf{atom} \cup \mathsf{atom}'$. The semantics of a regular process is an LTS defined according to the rules given in Figure 1, where P and P' are place-holders for processes, and α and α' are place-holders for actions. Note that $\mathbf{0}$ can perform no actions, and no action from $\mathsf{atom}_\mathbf{A}$ appears in regular processes. A process P and its corresponding LTS are used interchangeably throughout the paper.

Remark 2. *The operational semantics for regular processes is a mixture of* small step *and* big step *rules. This avoids using the silent action τ in the semantics of regular processes, and moreover enables us to define regular processes with non-terminating behaviors. Namely any process of the form $a^* \cdot \mathbf{0}$ admits only the infinite execution a^ω, cf. [2]. Indeed, contract signing protocols typically assume* trustees *who must reply to an unbounded number of (dispute resolution) requests [5].*

Next we introduce a notion of *honest* processes. A protocol describes a finite set of honest processes $\mathcal{H} = \{P_1, \cdots, P_n\}$

satisfying the following conditions: (1) Any honest process $P_i \in \mathcal{H}$ is a deterministic regular process. (2) For any two processes $P_i, P_j \in \mathcal{H}$, with $i \neq j$, the set of actions appearing in P_i and P_j are disjoint. We focus on protocols consisting of two honest processes. The results can however be extended to any number of honest processes.

A process P *satisfies* the property ϕ, denoted $P \models \phi$, iff the set of executions of P is a subset of ϕ. A process P_1 is a *sound* abstraction for process P_2 w.r.t. property ϕ iff $P_1 \models \phi \implies P_2 \models \phi$. Process P_1 is a *complete* abstraction for process P_2 w.r.t. property ϕ iff $P_2 \models \phi \implies P_1 \models \phi$.

The Dolev-Yao (DY) attacker model.

The attacker model envisaged by Dolev and Yao [10] intercepts and remembers all messages that have been communicated. It can decompose messages, and compose new messages from its knowledge and inject them into the communication media; in particular, it can decrypt, encrypt and sign messages, if it knows the corresponding key. It can also remove or delay messages in favor of others being communicated.

We make minimal assumptions on inference capabilities of the DY attacker. For a set $X \subseteq \mathsf{atom}$, let $C(X)$ be the set of actions that the attacker can perform (or, the set of messages that the attacker can construct) knowing X. We do not explicitly specify C. Instead, we require $C : 2^{\mathsf{atom}} \to 2^{\mathsf{atom}}$ to conform to the following conditions for any $X, Y \subseteq \mathsf{atom}$: (1) *Proximity*: $X \subseteq C(X)$ and (2) *Monotonicity*: $X \subseteq Y \implies C(X) \subseteq C(Y)$. We also assume that there is a message λ in $C(\emptyset)$, that does not appear in honest processes. This λ may denote a random dummy message generated by the attacker (as in Example 1, below).

Fairness constraints.

Enforcing fairness constraints is in general unavoidable when checking liveness properties. Below, we define a fairness constraint that selects only those executions of an LTS L in which, if an honest process is ready to perform an action infinitely often, then it also executes that action infinitely often. This, intuitively, means that no individual process is indefinitely ignored.

Take the deterministic LTS $L = (\mathbf{P}, P^0, \to)$. Given an execution σ in L, write $\mathbf{S}(\sigma)$ for the unique sequence of states that σ visits (uniqueness of $\mathbf{S}(\sigma)$ follows from L being deterministic). To refer to the i^{th} action of σ, we write σ_i, and similarly $\mathbf{S}(\sigma)_i$ is the i^{th} state in the sequence $\mathbf{S}(\sigma)$. For $P \in \mathbf{P}$, the set of actions that are *enabled* at P is $en(P) = \{a \in A \mid \exists P' \in \mathbf{P}.\ P \xrightarrow{a} P'\}$.

Definition 1 (Fair scheduling $\mathbf{F}$). *For a deterministic LTS $L = (\mathbf{P}, P^0, \to)$, and σ an execution of L, we say σ is a* fair *execution according to $\mathbf{F}$ iff*

$$\forall a \in A \setminus \mathsf{atom}_{\mathbf{A}}.\ \{i \mid a \in en(\mathbf{S}(\sigma)_i)\} \text{ is infinite} \implies \{i \mid a = \sigma_i\} \text{ is infinite}$$

Given a process P and property ϕ, we write $P \models_{\mathbf{F}} \phi$, iff all executions of P that are fair according to $\mathbf{F}$ belong to ϕ. The definitions of soundness and completeness are correspondingly extended.

Due to Definition 1, the honest processes must reach maximal progress in any execution which is fair according to $\mathbf{F}$. But, this is not the case for the attacker, as actions by the attacker are not constrained by $\mathbf{F}$. Indeed, if the attacker is forced to take actions (and reach maximal progress), then she can corner herself by "talking too much", so to speak; see Example 1. This notion is made precise in Theorem 1.

$$\frac{P \xrightarrow{a} P'}{P|\mathbb{B}_n, \mathbb{DY}|Q \xrightarrow{a} P'|\mathbb{B}_n \cup \{a\}, \mathbb{DY} \cup \{a\}|Q}$$

$$\frac{P \xrightarrow{a'} P' \qquad a \in \mathbb{B}_n}{P|\mathbb{B}_n, \mathbb{DY}|Q \xrightarrow{a'} P'|\mathbb{B}_n \setminus \{a\}, \mathbb{DY}|Q}$$

$$\frac{a \in C(\mathbb{DY})}{P|\mathbb{B}_n, \mathbb{DY}|Q \xrightarrow{i(a)} P|\mathbb{B}_n \cup \{a\}, \mathbb{DY}|Q}$$

Figure 2: Operational semantics for reference model R_n, with $n \in \mathbb{N}_+ \cup \{\infty\}$.

Remark 3. *As name clashes between actions of honest processes never occur, our definition of fair scheduling corresponds to the* strong *notion of fairness in [12].*

3. ASYNCHRONOUS COMMUNICATION VIA BUFFERS

A *buffer* is an asynchronous channel through which processes can communicate. We model a buffer $\mathbb{B}_n$ as a function that maps elements of atom to $\mathbb{N}$, where $n \in \mathbb{N}_+ \cup \{\infty\}$ denotes the *capacity* of $\mathbb{B}_n$. For an action symbol $a \in \mathsf{atom}$, $\mathbb{B}_n(a)$ shows how many instances of a are stored in the buffer. The capacity of the buffer places an upper-bound on how many instances of a can be stored at a time, for any $a \in \mathsf{atom}$. That is, $\forall a \in \mathsf{atom}.\ \mathbb{B}_n(a) \leq n$. Remark that the "actual" size of buffer $\mathbb{B}_n$ is bounded by $n \cdot |\mathsf{atom}|$. The empty buffer $\emptyset_n$ satisfies $\forall a \in \mathsf{atom}.\ \emptyset_n(a) = 0$, for any n.

For action symbol $a \in \mathsf{atom}$, the notation $\mathbb{B}_n \cup \{a\}$ stands for the function that coincides with $\mathbb{B}_n$ on any $b \neq a$, and $\mathbb{B}_n \cup \{a\}(a) = \min(\mathbb{B}_n(a) + 1, n)$. Similarly, $\mathbb{B}_n \setminus \{a\}(a) = \max(\mathbb{B}_n(a) - 1, 0)$, and for $b \neq a$ we have $\mathbb{B}_n \setminus \{a\}(b) = \mathbb{B}_n(b)$. Note that these operations agree with the notion of capacity, described above. Buffer $\mathbb{B}_n$ is *bounded* if $n \in \mathbb{N}_+$; it is *unbounded* if $n = \infty$. As a convention, we write $\mathbb{B}$ for $\mathbb{B}_\infty$. An unbounded buffer $\mathbb{B}$ is a multiset of elements of atom.

We model the Dolev-Yao attacker as an entity separated from the communication buffer, who can use its observations to derive new messages and put them in the buffer.

Definition 2 (Reference model R_n). *The asynchronous composition of two honest processes P and Q, through a buffer with capacity n, in the presence of the Dolev-Yao attacker, denoted by $P\ |\mathbb{B}_n, \mathbb{DY}|\ Q$, is an LTS defined by the rules of Figure 2 (symmetric rules for Q are omitted for brevity). There, $a \in \mathsf{atom}$ and $a' \in \mathsf{atom}'$, $\mathbb{B}_n$ represents the communication buffer that is external to the attacker process, and the set $\mathbb{DY}$ denotes attacker's knowledge (i.e. attacker's observations). The action $i(a)$ denotes* injecting *the symbol a into the buffer $\mathbb{B}_n$.*

We remark that if P and Q are honest processes, then $P|\mathbb{B}_n, \mathbb{DY}|Q$ describes a deterministic LTS. Moreover, in the reference model, buffers do not lose messages, unless they are full. They do not duplicate messages; they may however delay or reorder messages.

$$\frac{P \xrightarrow{a} P'}{P[\mathbb{B}]Q \xrightarrow{a} P'[\mathbb{B} \cup \{a\}]Q} \qquad \frac{P \xrightarrow{a'} P' \qquad a \in C(\mathbb{B})}{P[\mathbb{B}]Q \xrightarrow{a'} P'[\mathbb{B}]Q}$$

Figure 3: Operational semantics for interpretation I

REMARK 4. *The model $P|\mathbb{B}_n, \mathbb{DY}|Q$ can easily be extended to express systems in which* some *of the communication channels are not resilient [5]: The attacker can destroy messages communicated over such channels. This is obtained by adding the following rule to the rules of Figure 2.*

$$\frac{a \in \mathbb{B}_n \qquad \neg res(a)}{P|\mathbb{B}_n, \mathbb{DY}|Q \xrightarrow{d(a)} P|\mathbb{B}_n \setminus \{a\}, \mathbb{DY}|Q}$$

Here, a stands for any action in atom*, and the predicate $res(a)$ indicates whether a is communicated via a resilient channel or not: $res(a) = \mathsf{T}$ or $res(a) = \mathsf{F}$.*

4. THE DOLEV-YAO ATTACKER AS AN AUGMENTED BUFFER

The Dolev-Yao attacker can be seen as a buffer which can create messages.

DEFINITION 3 (INTERPRETATION **I**). *The asynchronous composition of two honest processes P and Q, through the Dolev-Yao attacker, denoted by $P[\mathbb{B}]Q$, is an LTS defined by the rules of Figure 3 (symmetric rules for Q are omitted for brevity). Here, a and a' stand for any action in* atom *and* atom$'$ *respectively. The set $\mathbb{B}$ is meant to represent the knowledge collected by the attacker.*

To keep the presentation consistent with Definition 2, one can safely assume that $\mathbb{B}$ in Definition 3 is a multiset.

A feature of interpretation **I** of the Dolev-Yao attacker, which is often exploited in automated verification tools, is that honest participants' receive actions are combined with message composition checks on the attacker's knowledge, and honest participants' send actions are modelled by adding the transmitted messages to the attacker's knowledge. The attacker does not explicitly perform any "action" in this model; all the attacker does is captured via checking or modifying its knowledge set.

5. FAIRNESS CONSTRAINTS FOR INTERPRETATION I

The attacker model given in Definition 3 can exhibit undesired behaviors when used along with fair scheduling. This is demonstrated in the following example.

EXAMPLE 1. *Let $P_1 = a_1 \cdot \mathbf{0}$ and $P_2 = a'_1 \cdot a'_1 \cdot a_2 \cdot \mathbf{0}$, where $a_1, a_2 \in$ atom and $a_1, a_2 \notin C(\emptyset)$. We consider the liveness property ϕ which contains only those elements of A^ω_τ in which a_2 eventually appears. Consider $P_1[\emptyset]P_2$, and observe that the only execution of this process is $a_1 \cdot a'_1 \cdot a'_1 \cdot a_2 \cdot \tau^\omega$, due to the proximity of C. This execution is indeed a fair execution, since after performing a_2 no action is enabled. The infinite repetition of τ is also merely an artifact of our way of representing terminating executions. We conclude $P_1[\emptyset]P_2 \models_{\mathbf{F}} \phi$. But this result is absurd, as a_2 does not occur when no attacker is present. In particular, the execution $a_1 \cdot a'_1 \cdot i(\lambda)^\omega$ belongs to $P_1|\emptyset_n, \emptyset|P_2$ (remark that to all intents and purposes, $i(\lambda)^\omega$ can safely be substituted here with τ^ω), and it is indeed a fair execution according to Definition 1 as it cannot be extended by any (non-attacker) participants. The liveness property ϕ does thus not hold in $P_1|\emptyset_n, \emptyset|P_2$, under **F**.*

Example 1 shows that the attacker model of Definition 3 can corner herself when checking liveness under **F**, and therefore *attacks* can be missed. An attack for a security requirement ϕ is an execution which witnesses ϕ does not hold in the model. The following theorem is immediate.

THEOREM 1. *Checking liveness using the attacker model $P[\mathbb{B}]Q$ is not sound w.r.t. the model $P|\mathbb{B}_n, \mathbb{DY}|Q$, under fairness constraint **F**:*

$$P[\mathbb{B}]Q \models_{\mathbf{F}} \phi \not\Longrightarrow P|\mathbb{B}_n, \mathbb{DY}|Q \models_{\mathbf{F}} \phi$$

Example 1 puts the result of Theorem 1 in the context of cryptographic protocols.

EXAMPLE 2 (REALIZATION OF EXAMPLE 1). *Note that in Example 1, with $P_2 = a'_1 \cdot a'_3 \cdot a_2 \cdot \mathbf{0}$, the same unsoundness result is obtained, if $a_3 \in C(a_1)$. Now, assume a_1 corresponds to P_1 putting a secret number $n \in \mathbb{N}$ in the buffer. Let us assume* hash *is a cryptographic secure hash function and a'_3 denotes the fact that P_2 expects to receive* hash(n)*. Clearly, if there is no attacker in the system, P_1 and P_2 cannot communicate, and thus a_2 never happens. However, when the attacker is present, knowing n, the attacker can construct* hash(n)*. Therefore, if the fair scheduling constraint is added to the model, then the attacker will serve as an interface which facilitates the conversation between P_1 and P_2. This feature of the attacker process is desired when checking safety properties. When it comes to liveness, nonetheless, it may result in that some attacks are overlooked.*

The interpretation $P_1[\mathbb{B}]P_2$ is the one that is most commonly used for specifying the DY attacker in formal verification tools (see the related work). Below, we investigate whether interpretation $P_1[\mathbb{B}]P_2$ can be sound and complete w.r.t. $P_1|\mathbb{B}_n, \mathbb{DY}|P_2$ by enforcing perhaps a carefully crafted fairness constraint. To answer this question, we distinguish two cases: (1) $n = \infty$, and (2) $n \in \mathbb{N}_+$.

5.1 Unbounded buffers

Here, we are interested in characterizing any fairness constraint $\mathbf{F}^\dagger$ such that

$$P[\mathbb{B}]Q \models_{\mathbf{F}^\dagger} \phi \iff P|\mathbb{B}, \mathbb{DY}|Q \models_{\mathbf{F}} \phi \qquad (1)$$

where P and Q are honest processes, and ϕ is any liveness property. Below, we show that any fairness constraint $\mathbf{F}^\dagger$ which satisfies Formula 1 is non-regular. This indicates that specifying $\mathbf{F}^\dagger$ falls beyond the expressive power of propositional μ-calculus, and in particular that of temporal logics such as LTL and CTL* which are typically used for specifying fairness constraints in standard automated verification algorithms and tools, cf. [23]. Employing such fairness constraints would thus defy efficient automatic verification of liveness security requirements in interpretation **I**.

We continue with some definitions. Given an execution σ in P we write $\hat{\sigma}$ for the sequence of pairs $(\sigma_i, en(\mathbf{S}(\sigma)_i))$,

for $i \in \mathbb{N}$. Clearly $\hat{\sigma} \in (A \times 2^A)^\omega$. We consider a fairness constraint as being a property on $(A \times 2^A)^\omega$. A subset of Σ^ω, for finite Σ, is an *ω-regular language* iff it is a finite union of sets $U \cdot V^\omega$, where $U, V \subseteq \Sigma^*$ are regular subsets of Σ^* [23]. We say a fairness constraint is *regular* iff it constitutes an ω-regular language. It is straightforward to prove that fair scheduling (Definition 1) is regular on $(A \times 2^A)^\omega$, by giving *Streett automata* (see [23]) that accepts $\mathbf{F}$. Our proof of $\mathbf{F}^\dagger$ not being regular depends solely on non-regularity of finite prefixes of elements of $\mathbf{F}^\dagger$.

THEOREM 2. *$\mathbf{F}^\dagger$ is not regular over $(A \times 2^A)^\omega$.*

PROOF. Consider two processes $P = a_1^n \cdot a_2 \cdot \mathbf{0}$ and $Q = a_2' \cdot {a'_1}^m \cdot a_3 \cdot \mathbf{0}$, with $n, m \in \mathbb{N}_+$, such that $a_1, a_2 \notin C(\emptyset)$ and $a_3 \notin C(\{a_1, a_2\})$ ($\alpha^0 = \epsilon$ and $\alpha^{i+1} = \alpha \cdot \alpha^i$, for $i \in \mathbb{N}$). Let $\phi \subseteq A_\tau^\omega$ be the liveness property that contains all executions in which a_3 occurs. Note that with $m \leq n$, $P|\emptyset, \emptyset|Q \models_{\mathbf{F}} \phi$, while $m > n$ results in $P|\emptyset, \emptyset|Q \not\models_{\mathbf{F}} \phi$. Observe that the executions of $P[\emptyset]Q$ are of the form $\sigma = a_1^n \cdot a_2 \cdot a_2' \cdot {a'_1}^m \cdot a_3 \cdot \tau^\omega$. This results in $\hat{\sigma} = (a_1, \{a_1\})^{n-1} \cdot (a_1, \{a_2\}) \cdot (a_2, \{a_2'\}) \cdot (a_2', \{a_1'\}) \cdot (a_1', \{a_1'\})^{m-1} \cdot (a_1', \{a_3\}) \cdot (a_3, \emptyset) \cdot (\tau, \emptyset)^\omega$, which has to belong to $\mathbf{F}^\dagger$ iff $m \leq n$. The expression above can be rewritten as $\hat{\sigma} = x^{n-1} \cdot y \cdot z^{m-1} \cdot w \cdot \tau^\omega$, for appropriate x, y, z, w, and with overloading τ. Therefore, the set $\mathbf{F}^\dagger$ must include $U \cdot \tau^\omega$, with $U = \{x^n \cdot y \cdot z^m \cdot w \mid m \leq n\}$. The set U is not a regular subset of $(A \times 2^A)^*$.

Now, assume, toward a contradiction, that $\mathbf{F}^\dagger$ is an ω-regular language. Then, there must exist a set U' such that $U' \cdot \tau^\omega \subseteq \mathbf{F}^\dagger$, and $\mathbf{U} = U \cup U'$ is a regular subset of $(A \times 2^A)^*$. We claim this would change the behavior of $\mathbf{F}^\dagger$ for the above processes as well. Below, we "pump down" the set $\mathbf{U}$ to show that indeed if $\mathbf{U}$ is regular, then $\mathbf{F}^\dagger$ does not satisfy Formula 1. Take $x^p \cdot y \cdot z^p \cdot w \in \mathbf{U}$, where p is the pumping length of the regular set $\mathbf{U}$. According to the pumping lemma, $\exists q.\ x^q \cdot y \cdot z^p \cdot w \in \mathbf{U}$, with $q < p$. This contradicts the allowed behavior of $\mathbf{F}^\dagger$, mentioned above, for witness processes P and Q. Therefore, $\mathbf{U}$ is not a regular set in general, and therefore $\mathbf{F}^\dagger$ is not regular. □

The proof of Theorem 2 intuitively states that fairness constraint $\mathbf{F}^\dagger$ inevitably involves counting. Such constraints can be realized, e.g., using pushdown automata, or using a counting operator. For instance, given a counting operator #, we can specify that elements of atom which are enabled infinitely often, and elements of atom' which are *positively* enabled infinitely often, are taken infinitely often in a fair execution. Here $a' \in \mathsf{atom}'$ is positively enabled in execution σ at $\mathbf{S}(\sigma)_i$ iff $a' \in en(\mathbf{S}(\sigma)_i)$ and $\#(a) - \#(a') > 0$, where $\#(\alpha)$ is the number of occurrences of α in the prefix of σ up to σ_i, i.e. the sequence $\sigma_1 \cdots \sigma_i$.

REMARK 5. *The Dolev-Yao attacker model can be (interpreted, and) formalized in various ways. We focus on interpretation* **I** *in which messages are channeled through the attacker process, because it has been used in many tools and algorithms for automated formal verification of security protocols. One can however think of other interpretations, e.g. the one given below, that can be proved to be sound and complete (with constraint* $\mathbf{F}$*), under certain conditions, w.r.t. the reference models.*

$$\frac{P_1 \xrightarrow{a} P_1'}{P_1 \langle\!\langle \mathbb{B} \rangle\!\rangle P_2 \xrightarrow{a} P_1' \langle\!\langle \mathbb{B} \cup \{a\} \rangle\!\rangle P_2}$$

$$\frac{P_1 \xrightarrow{a'} P_1' \qquad a \in \mathbb{B}}{P_1 \langle\!\langle \mathbb{B} \rangle\!\rangle P_2 \xrightarrow{a'} P_1' \langle\!\langle \mathbb{B} \setminus \{a\} \rangle\!\rangle P_2}$$

$$\frac{a \in C(\mathbb{B})}{P_1 \langle\!\langle \mathbb{B} \rangle\!\rangle P_2 \xrightarrow{i(a)} P_1 \langle\!\langle \mathbb{B} \cup \{a\} \rangle\!\rangle P_2}$$

Here, P_1 and P_2 are regular processes, and the multiset $\mathbb{B}$ represents the attacker knowledge as well as the content of the communication buffer.

Remark that the interpretation $P_1 \langle\!\langle \mathbb{B} \rangle\!\rangle P_2$ does not require complex fairness constraints for soundness and completeness (as mentioned above, the fair scheduling $\mathbf{F}$ is sufficient for this purpose), and moreover it is more concise than $P_1 | \mathbb{B}_n, \mathbb{DY} | P_2$, namely the communication buffer and and attacker knowledge set coincide in $P_1 \langle\!\langle \mathbb{B} \rangle\!\rangle P_2$. However, in interpretation $P_1 \langle\!\langle \mathbb{B} \rangle\!\rangle P_2$, receive actions of the participants are not coupled with computing the closure of the attacker knowledge (cf. Section 1), and moreover the knowledge of the attacker, captured by the multiset $\mathbb{B}$ in this case, is not monotonic. Monotonicity of attacker's knowledge is a key factor in decidability results for secrecy in cryptographic protocols, e.g. see [21] and [20]. These features, therefore, severely limit the applicability of interpretation $P_1 \langle\!\langle \mathbb{B} \rangle\!\rangle P_2$ for automated formal analysis.

REMARK 6. *For verifying safety properties which are* insensitive *to attacker actions, all the interpretations introduced in the paper are sound and complete w.r.t. the reference models. The notion of insensitivity is made precise in Section 6.*

5.2 Bounded buffers

In this section we investigate whether the interpretation $P_1[\mathbb{B}]P_2$ can be made sound and complete w.r.t. $P_1 | \mathbb{B}_n, \mathbb{DY} | P_2$, with $n \in \mathbb{N}_+$, by enforcing a suitable fairness constraint. To simplify the presentation, we consider only the case where $n = 1$, i.e. $\mathbb{B}_n$ is implemented as a set. We are interested in any fairness constraint $\mathbf{F}_r$ such that

$$P_1[\mathbb{B}]P_2 \models_{\mathbf{F}_r} \phi \iff P_1 | \mathbb{B}_1, \mathbb{DY} | P_2 \models_{\mathbf{F}} \phi \qquad (2)$$

where P_1 and P_2 are honest processes, and ϕ is any liveness property. The non-regularity argument of the previous section does not hold with bounded buffers. Nevertheless, $\mathbf{F}_r$ turns out to be a complicated condition. In particular, $\mathbf{F}_r$ would not be "insensitive to addition and deletion of prefixes", a feature often attributed to fairness constraints [22]. Below, we use the notion of *local testability* [18] to capture this intuition. Informally, a property is locally testable if one can determine whether an execution belongs to the property or not, by observing the execution only via a window of bounded width. We proceed with some definitions.

Given a finite sequence $\sigma \in \Sigma^*$, with Σ being a finite set of symbols and $k \in \mathbb{N}_+$, define $\mathbf{L}_k(\sigma)$, $\mathbf{R}_k(\sigma)$ and $\mathbf{i}_k(\sigma)$ as, respectively, the left-end segment of σ of length k, the right-end segment of σ of length k, and the set of interior segments of σ of length k. When the length of σ is less than or equal to k, then $\mathbf{L}_k(\sigma) = \mathbf{R}_k(\sigma) = \sigma$ and $\mathbf{i}_k(\sigma) = \emptyset$. We write $\sigma \equiv_k \sigma'$, for $\sigma, \sigma' \in \Sigma^*$, iff $\mathbf{L}_k(\sigma) = \mathbf{L}_k(\sigma')$, $\mathbf{R}_k(\sigma) = \mathbf{R}_k(\sigma')$ and $\mathbf{i}_k(\sigma) = \mathbf{i}_k(\sigma')$. For infinite sequences $\sigma, \sigma' \in \Sigma^\omega$, we write $\sigma \equiv_k \sigma'$ iff $\exists \sigma_0, \sigma_0' \in \Sigma^*, \theta \in \Sigma^\omega$ such that $\sigma = \sigma_0 \cdot \theta$ and $\sigma' = \sigma_0' \cdot \theta$, and $\sigma_0 \equiv_k \sigma_0'$. A property $\psi \subseteq \Sigma^\omega$, is *$k$-testable*, for $k \in \mathbb{N}_+$, iff ψ saturates $\equiv_k$, and

a property ψ is *locally testable* iff it is k-testable for some $k \in \mathbb{N}_+$. Below, we show that fairness constraint $\mathbf{F}_r$, seen as a property on $(A \times 2^A)^\omega$, is not locally testable.

THEOREM 3. *$\mathbf{F}_r$ is not locally testable.*

PROOF. Assume, towards a contradiction, that $\mathbf{F}_r$ is locally testable. Therefore, $\mathbf{F}_r$ is k-testable, for some $k > 0$. Let $P_1 = a_1^k \cdot a_2 \cdot a_1^k \cdot a_2' \cdot a_1^k \cdot a_2' \cdot a_1^k \cdot a_3 \cdot \mathbf{0}$, and $P_2 = \mathbf{0}$ with $a_1, a_2, a_3 \in \mathsf{atom}, a_1, a_2 \notin C(\emptyset)$ and $a_3 \notin C(\{a_1, a_2\})$. Consider the liveness property ϕ containing only executions in which a_3 appears. Note that $P_1|\emptyset_1, \emptyset|P_2 \not\models_{\mathbf{F}} \phi$. Since $P_1[\emptyset_1]P_2$ is sound under $\mathbf{F}_r$ (see Formula 2), the terminating execution of $P_1[\emptyset_1]P_2$, i.e. $a_1^k{\cdot}a_2{\cdot}a_1^k{\cdot}a_2'{\cdot}a_1^k{\cdot}a_2'{\cdot}a_1^k{\cdot}a_3{\cdot}\tau^\omega$, is considered unfair according to $\mathbf{F}_r$. Now, consider $Q_1 = a_1^k \cdot a_2 \cdot a_1^k{\cdot}a_2'{\cdot}a_1^k{\cdot}a_3{\cdot}\mathbf{0}$, and $Q_2 = \mathbf{0}$. Observe that $Q_1|\emptyset_1, \emptyset|Q_2 \models_{\mathbf{F}} \phi$. Due to Formula 2, the terminating execution of $Q_1[\emptyset_1]Q_2$, i.e. $a_1^k{\cdot}a_2{\cdot}a_1^k{\cdot}a_2'{\cdot}a_1^k{\cdot}a_3{\cdot}\tau^\omega$, is considered fair according to $\mathbf{F}_r$. Note that $a_1^k{\cdot}a_2{\cdot}a_1^k{\cdot}a_2'{\cdot}a_1^k{\cdot}a_3 \equiv_k a_1^k{\cdot}a_2{\cdot}a_1^k{\cdot}a_2'{\cdot}a_1^k{\cdot}a_2'{\cdot}a_1^k{\cdot}a_3$. This is because instances of a_2 and a_2' are placed at least k points apart. These sequences are indistinguishable for any k-testable property; hence a contradiction. □

Non-locality of $\mathbf{F}_r$ indicates that $\mathbf{F}_r$ cannot be reduced to a simple local (per state) conflict resolution. This inversely affects the efficiency of automated verification algorithms using $\mathbf{F}_r$. We remark that the fair scheduling constraint $\mathbf{F}$ is locally testable, intuitively because no finite prefix of σ affects whether σ is fair according to $\mathbf{F}$ or not.

6. SOUND ABSTRACTIONS FOR BOUNDED BUFFERS

In the presence of the DY attacker, if a liveness property holds with a certain buffer capacity, it also holds in any system with a larger buffer capacity. This is intuitively because the DY attacker can duplicate messages and hence emulate large buffers using small buffers. Proving this observation formally however requires limiting the liveness properties under consideration, as shown in Theorem 4. The theorem allows deriving general security guarantees from verification of a bounded model.

We proceed with some definitions. For two LTSs $L_1 = (\mathbf{P}_1, P_1^0, \rightarrow_1)$ and $L_2 = (\mathbf{P}_2, P_2^0, \rightarrow_2)$, a binary relation $\Upsilon_\chi \subseteq \mathbf{P}_1 \times \mathbf{P}_2$ is a *branching simulation preorder*, w.r.t. $\chi \subseteq A$, if whenever $P_1 \Upsilon_\chi P_2$ and action $\alpha \in A$, $P_1 \xrightarrow{\alpha}_1 P_1'$ implies that there exist $\hat{P}_2, P_2' \in \mathbf{P}_2$ and $\xi \in \chi^*$ such that $P_2 \xrightarrow{\xi}_2 \hat{P}_2$ and $\hat{P}_2 \xrightarrow{\alpha}_2 P_2'$ with $P_1 \Upsilon_\chi \hat{P}_2$ and $P_1' \Upsilon_\chi P_2'$. We use the notation $L_1 \preceq_\chi L_2$ iff there exists a branching simulation preorder Υ_χ such that $P_1^0 \Upsilon_\chi P_2^0$. Intuitively, if $L_1 \preceq_\chi L_2$ then L_2 simulates L_1; the simulation is modulo χ. The elements of χ are "hidden" in $\preceq_\chi$, and when using this relation, χ consists of attacker actions which are not directly visible to honest processes.

For a finite sequences of actions σ, we write $f_B(\sigma)$ for the sequence that is found by removing all elements of $B \subseteq A$ from σ. That is, f_B is a function that filters out elements of B. For two infinite sequences of actions σ, σ', we write $\sigma \sim_B \sigma'$ iff $\exists \theta, \theta' \in A^*, \alpha \in A^\omega$. $f_B(\theta) = f_B(\theta')$ and $\sigma = \theta{\cdot}\alpha$ and $\sigma' = \theta'{\cdot}\alpha$. Obviously, $\sim_B$, with $B \subseteq A$, is an equivalence relation. We say a property is *insensitive* to the actions of the attacker if it saturates $\sim_{\mathsf{atom}_\mathbf{A}}$.

THEOREM 4. *For any $m \in \mathbb{N}_+, n \in \mathbb{N}_+ \cup \{\infty\}$, and any liveness property ϕ that saturates $\sim_{\mathsf{atom}_\mathbf{A}}$, if $m \leq n$, then $P_1|\mathbb{B}_m, \mathbb{DY}|P_2 \models_{\mathbf{F}} \phi \implies P_1|\mathbb{B}_n, \mathbb{DY}|P_2 \models_{\mathbf{F}} \phi$.*

PROOF. Let $L_m = P_1|\mathbb{B}_m, \mathbb{DY}|P_2$ and $L_n = P_1|\mathbb{B}_n, \mathbb{DY}|P_2$. Suppose that $L_n \not\models_{\mathbf{F}} \phi$. To prove the theorem, we need to show $L_m \not\models_{\mathbf{F}} \phi$. Let σ be an execution in L_n that violates ϕ, under the fairness constraint $\mathbf{F}$. Suppose that $L_n \preceq_\chi L_m$, with $\chi = \mathsf{atom}_\mathbf{A}$, (we will show this later). Then the translation of σ in L_n to some simulation σ' in L_m is immediate. Now, if σ violates ϕ and ϕ saturates $\sim_{\mathsf{atom}_\mathbf{A}}$, then σ' violates ϕ as well. To show that σ' is fair in L_m according to $\mathbf{F}$ we note that in the branching simulation preorder the set of actions enabled at σ' is the same as that of σ. Therefore, as σ is a fair schedule in L_n, we conclude that σ' is a fair schedule in L_m.

Now we show that $L_n \preceq_\chi L_m$, with $\chi = \mathsf{atom}_\mathbf{A}$ holds. Note that the converse relation $L_m \preceq L_n$ is obvious. Define the relation Υ_χ as $(P_1|X, Y|P_2)\Upsilon_\chi(P_1|Z, W|P_2)$ iff $\forall a \in \mathsf{atom}$. $Z(a) \leq X(a) \wedge Y = W$, where X and Z are functions from atom to $\{0, \cdots, n\}$ and to $\{0, \cdots, m\}$, respectively, and $Y, W \subseteq \mathsf{atom}$. Observe that when the buffers are the same and the attacker's initial knowledge is equivalent in L_m and L_n, this relation holds.

To see that Υ_χ is indeed a branching simulation preorder, take some $a \in \mathsf{atom}$ and suppose $P_1|X, Y|P_2 \xrightarrow{a} P_1'|X', Y'|P_2$. Clearly, $X'(a) = \min(X(a) + 1, n)$ and $Y' = Y \cup \{a\}$. Corresponding to this we have $P_1|Z, W|P_2 \xrightarrow{a} P_1'|Z', W'|P_2$, where $Z'(a) = \min(Z(a) + 1, m)$ and $W' = W \cup \{a\}$. Since $Z(a) \leq X(a)$ and $m \leq n$, it follows that $(P_1'|X', Y'|P_2)\Upsilon(P_1'|Z', W'|P_2)$.

Now suppose $P_1|X, Y|P_2 \xrightarrow{a'} P_1'|X', Y'|P_2$, for some $a' \in \mathsf{atom}'$. Clearly, $X'(a) = \max(X(a) - 1, 0)$, and $Y' = Y$. If $P_1|Z, W|P_2$ can perform a', then an argument like above works. There are however cases where L_m cannot perform a' because there is no a left in $\mathbb{B}_m$ (while there are still as in $\mathbb{B}_n$). In this situation we introduce an attacker action $i(a)$ to σ'. Namely, we define $\xi = i(a)$ in case $X(a) > 0$ and $Z(a) = 0$. This attacker action is possible due to the proximity property of C. Then, $P_1|Z, W|P_2 \xrightarrow{\xi} P_1|Z', W|P_2$. Due to the precondition $X(a) > 0 \wedge Z(a) = 0$, we get $Z'(a) = \min(Z(a) + 1, m) \leq X(a)$. From this state, a' can be performed. The cases for the attacker's internal actions in these two systems are identical, as the bounded communication buffer does not affect the $\mathbb{DY}$ set. We conclude that $L_n \preceq_\chi L_m$. □

We remark that Theorem 4 hinges upon the attacker's ability to duplicate messages. This has two implications. First, Theorem 4 holds in all models that allow faulty communication links which duplicate messages. For instance, various message relay defects, Byzantine process failures, and the DY model are sources of such link failures. Second, the theorem in fact falls apart when messages cannot be arbitrarily duplicated, e.g. when the DY attacker is not present in the model. For instance, take $P_1 = a_1 \cdot a_1 \cdot a_1' \cdot a_1' \cdot a_2 \cdot \mathbf{0}$, and $P_2 = \mathbf{0}$, with $a_1, a_2, a_3 \in \mathsf{atom}$. Let ϕ be the liveness property containing all executions in which if a_2 occurs, a_3 occurs as well. Note that composition of P_1 and P_2 via asynchronous buffer $\mathbb{B}_1$ satisfies ϕ, while in their composition via buffer $\mathbb{B}_2$ property ϕ does not hold.

7. CONCLUDING REMARKS

We study an interpretation of the Dolev-Yao attacker model $P[\mathbb{B}]Q$ in which messages are channeled through the attacker process, and compare it to a reference model $P|\mathbb{B}, \mathbb{DY}|Q$ in which the communication media, albeit being controlled by the attacker, are external to the attacker process. The central question of the paper is to find *suitable* fairness constraints for checking liveness properties in the model $P[\mathbb{B}]Q$. Here, a fairness constraint F is called suitable if any liveness property that holds in the model $P[\mathbb{B}]Q$ under constraint F also holds in the model $P|\mathbb{B}, \mathbb{DY}|Q$ under the standard maximal progress fairness constraint, and vice versa.

We formalize fairness constraints as formal languages, and show that no ω-regular language is a suitable fairness constraint if the communication buffers are unbounded in the model $P|\mathbb{B}, \mathbb{DY}|Q$. Similarly, we show that no locally testable language is a suitable fairness constraint if the communication buffers in the model $P|\mathbb{B}, \mathbb{DY}|Q$ are bounded. These results indicate that $P[\mathbb{B}]Q$ is not favorable when it comes to efficient automatic verification of liveness security requirements, because of high complexity of the fairness constraints.

On the positive side, the very same feature of the Dolev-Yao attacker model which results in high complexity of fairness constraints plays a favorable role in verification of protocols with bounded buffers. Namely, we show that, in the presence of the attacker, any liveness property that holds in a protocol with buffer capacity n, also holds when buffers have capacity $m \geq n$. The result allows deriving general security guarantees from verification of small bounded models. We remark that this result (1) hinges upon attacker's "contributions" to the protocol, i.e. it fails to hold in asynchronous systems in general, absent of attacker, and (2) holds only for liveness properties that are "insensitive" to the actions of the attacker. We precisely define the class of such properties.

We expect that with simple attacker models (e.g. $P[\mathbb{B}]Q$) one can still find *simple* suitable fairness constraints (e.g. standard maximal progress constraint), for verifying liveness security requirements, by adding reasonable assumptions on the behaviors of honest participants. Finding such assumptions for the attacker models that are commonly used calls for further research.

The results obtained in this paper are not bound to the Dolev-Yao attacker model. We expect them to also hold, with minor adjustments, in other models which allow faulty communication links to duplicate messages. For instance, various message relay defects, and Byzantine failures are sources of such link failures. Extending our results to more general settings is left for future work.

Acknowledgements

We are grateful to Eugen Zalinescu for his comments on the paper. Jan Cederquist has been supported by FCT via KLog, PTDC/MAT/68723/2006. Mohammad Torabi Dashti has been supported by AVANTSSAR, FP7-ICT-2007-1 Project no. 216471.

8. REFERENCES

[1] M. Abadi and B. Blanchet. Computer-assisted verification of a protocol for certified email. *Sci. Comput. Program.*, 58(1-2):3–27, 2005.

[2] L. Aceto, R. van Glabbeek, W. Fokkink, and A. Ingólfsdóttir. Axiomatizing prefix iteration with silent steps. *Inf. Comput.*, 127(1):26–40, 1996.

[3] B. Alpern and F. Schneider. Defining liveness. *Inf. Proc. Let.*, 21(4):181–185, 1985.

[4] A. Armando, R. Carbone, and L. Compagna. LTL model checking for security protocols. In *CSF '07*, pages 385–396. IEEE CS, 2007.

[5] N. Asokan, V. Shoup, and M. Waidner. Optimistic fair exchange of digital signatures. *IEEE J. on Selected Areas in Communications*, 18(4):593–610, 2000.

[6] G. Bella and L. Paulson. Accountability protocols: Formalized and verified. *ACM Trans. Inf. Syst. Secur.*, 9(2):138–161, 2006.

[7] B. Blanchet. An efficient cryptographic protocol verifier based on Prolog rules. In *CSFW '01*, page 82. IEEE Computer Society, 2001.

[8] E. Clarke, S. Jha, and W. Marrero. Verifying security protocols with Brutus. *ACM Trans. Softw. Eng. Methodol.*, 9(4):443–487, 2000.

[9] V. Cortier, R. Küsters, and B. Warinschi. A cryptographic model for branching time security properties. In *ESORICS*, volume 4734 of *LNCS*, pages 422–37, 2007.

[10] D. Dolev and A. Yao. On the security of public key protocols. *IEEE Trans. on Information Theory*, IT-29(2):198–208, 1983.

[11] E. Emerson. Temporal and modal logic. In Leeuwen [16], pages 995 – 1072.

[12] N. Francez. *Fairness*. Springer, 1986.

[13] S. Gürgens, C. Rudolph, and H. Vogt. On the security of fair non-repudiation protocols. *Int. J. Inf. Sec.*, 4(4):253–262, 2005.

[14] D. Kähler, R. Küsters, and T. Truderung. Infinite state AMC-model checking for cryptographic protocols. In *LICS '07*, pages 181–192. IEEE CS, 2007.

[15] S. Kremer and J. Raskin. A game-based verification of non-repudiation and fair exchange protocols. In *CONCUR '01*, volume 2154 of *LNCS*, pages 551–565, 2001.

[16] J. van Leeuwen, editor. *Handbook of theoretical computer science (vol. B): Formal models and semantics*. MIT, 1990.

[17] N. Lynch. *Distributed Algorithms*. Morgan Kaufmann Publishers, 1996.

[18] R. McNaughton and S. Papert. *Counter-Free Automata*. MIT Press, 1971.

[19] C. Meadows. Ordering from satan's menu: A survey of requirements specification for formal analysis of cryptographic protocols. *Sci. Comput. Program.*, 50(1-3):3–22, 2004.

[20] J. Millen and V. Shmatikov. Constraint solving for bounded-process cryptographic protocol analysis. In *CCS '01*, pages 166–175. ACM Press, 2001.

[21] M. Rusinowitch and M. Turuani. Protocol insecurity with finite number of sessions is NP-complete. In *CSFW '01*, page 174. IEEE CS, 2001.

[22] A. Sistla. Safety, liveness and fairness in temporal logic. *FAC*, 6(5):495–512, 1994.

[23] W. Thomas. Automata on infinite objects. In Leeuwen [16], pages 133–191.

[24] K. Wei and J. Heather. A theorem-proving approach to verification of fair non-repudiation protocols. In *FAST '06*, volume 4691 of *LNCS*, pages 202–219, 2007.