Available online at www.sciencedirect.com

ScienceDirect

Electronic Notes in Theoretical Computer Science 242 (2009) 99–112

Electronic Notes in Theoretical Computer Science

www.elsevier.com/locate/entcs

Security Abstractions and Intruder Models (Extended Abstract)

Michele Bugliesi [1] and Riccardo Focardi [2]

Dipartimento di Informatica
Università Ca' Foscari di Venezia

Abstract

Process algebraic specifications of distributed systems are increasingly being targeted at identifying security primitives well-suited as high-level programming abstractions, and at the same time adequate for security analysis and verification. Drawing on our earlier work along these lines [5], we investigate the expressive power of a core set of security and network abstractions that provide high-level primitives for the specifications of the honest principals in a network as well as the lower-level adversarial primitives that must be assumed available to an attacker.
We analyze various bisimulation equivalences for security, arising from endowing the intruder with (*i*) different adversarial capabilities and (*ii*) increasingly powerful control on the interaction among the distributed principals of a network. By comparing the relative strength of the bimimulation equivalences we obtain a direct measure of the discriminating power of the intruders, hence of the expressiveness of the corresponding models.

Keywords: Process algebras, bisimulation for security, intruder model

1 Introduction

The challenges in achieving security in distributed systems often create a tension between two conflicting requirements. On the one side, security concerns call for detailed formal specifications of the safeguards built against the threats to which the systems are exposed. On the other side, programming the systems needs techniques and reasoning methods that abstract away from such details to focus on the expected functional properties.

In the literature on process calculi, this tension has generated a range of approaches, with two extremes. At one end, we find specifications that draw on low-level cryptographic primitives as in the spi calculus [3] or in the applied-pi calculus [1]. At the other end lie specifications based on the pi calculus [11], which assume

[1] Email: michele@dsi.unive.it

[2] Email: focardi@dsi.unive.it

1571-0661 © 2009 Elsevier B.V. Open access under CC BY-NC-ND license.

doi:10.1016/j.entcs.2009.06.015

very abstract, and hard-to-implement, mechanisms to secure communications by hiding them on private channels. A more recent line of research [2,9,4,6,7] follows a different approach, aimed at identifying security primitives well-suited as high-level programming abstractions, and at the same time adequate for security analysis and verification in adversarial setting.

Drawing on our initial ideas in [5], in the present paper we further assess the adequacy of our approach by investigating the expressive power of our security and network abstractions. In particular, we analyze various bisimulation equivalences for security, associated with a variety of intruder models. The models arise from endowing the intruders with (i) different adversarial capabilities and (ii) increasingly powerful control on the interaction among the distributed principals of a network. The bisimulation equivalences, in turn, provide a direct measure of the discriminating power of the intruders, hence of the expressiveness of the corresponding models.

The starting point is the asynchronous pi-calculus with security abstractions we defined in [5]. In this model, the intruder has the capability to interfere in all network interactions: it can forge its own traffic, intercept all messages and forward them back to the network, possibly replicating them. However, similarly to what happens in the Dolev-Yao model for cryptographic protocols, it cannot learn any secret message and cannot forge any authenticated transmission. For this intruder, we give a sound characterization of strong barbed equivalence in terms of strong asynchronous bisimulation. Also, we show that asynchronous and synchronous bisimilarity coincide.

We then extend our network abstractions with a new primitive that enables the intruder to silently eavesdrop on network traffic (without necessarily intercepting it). We show that the new capability adds no discriminating power to the intruder, in that it does not affect the security equivalences (either synchronous or asynchronous). On the other hand, eavesdropping turns out to be strictly less powerful than intercepting.

As a further, and final step, we look at the notion of intruder adopted in [4], that corresponds to what is sometimes referred to as the *man-in-the-middle* intruder. In this new model two principals may never engage in a synchronization directly as it is customary for the semantics of traditional process calculi (and as we assume in the initial model). Instead, all exchanges take place with the intruder's intervention. We show, somewhat surprisingly, that this additional control on the interactions on the network does not change the notion of equivalence, hence does not add discriminating power to the intruder.

Plan. Sections 2 and 3 review the calculus from [5]. Sections 4 and 5 discuss the results for the Dolev-Yao intruder of [5]. Section 6 contrasts this model with the man-in-the-middle intruder of [4]. Section 7 concludes the presentation.

2 Security and Network Abstractions

We start with a brief review of the calculus of security and network abstractions from [5]. We presuppose two countable sets $\mathbf{N}$ and $\mathbf{V}$ of names and variables, respectively, and let $a - q$ range over names, $, w, x, y, z$ over variables and t, u, v over $\mathbf{N} \cup \mathbf{V}$. Names enable communication, but serve also as identities: for instance, $\overline{b}\langle a : \tilde{n}\rangle$ indicates an output to b originating from a, while $b(a : \tilde{x})$ denotes an input performed by b of a message from a. Tuples are indicated by a tilde, as in $\tilde{n}, \tilde{x}, \tilde{v}$.

2.1 High-Level Principals

The syntax of the high-level calculus is below.

$$\begin{array}{rlll}
H, K & ::= & \overline{u}\langle \underline{a} : \tilde{v}\rangle^{\circ} & \text{(Output)} \\
& \mid & a(\underline{v} : \tilde{y})^{\circ}.H & \text{(Input)} \\
& \mid & \mathbf{0} & \text{(Null)} \\
& \mid & H|K & \text{(Parallel)} \\
& \mid & \mathsf{if}\ u = v\ \mathsf{then}\ H\ \mathsf{else}\ K & \text{(Conditional)} \\
& \mid & A\langle \tilde{u}\rangle & \text{(Definition)} \\
& \mid & (\boldsymbol{\nu} a)H & \text{(Restriction)}
\end{array}$$

We use $\underline{v}$ as short for the name or variable v, or the distinguished name $-$ associated with an *anonymous* identity. The null, parallel composition and conditional forms are just as in the pi-calculus. $A\langle \tilde{u}\rangle$ is the process defined by a (possibly recursive) definition $A(\tilde{x}) \stackrel{\text{def}}{=} H$ (A may only occur guarded in H). The restriction $(\boldsymbol{\nu} a)H$ has the familiar pi-calculus syntax but weaker scoping rules (see below). As to communication, we have four output forms, depending whether $\underline{a}$ is a or $-$ and whether $\circ$ is $\bullet$ or the empty character, as explained next: $u\langle - : \tilde{v}\rangle$ denotes a *plain* output, a communication primitive that conveys no security guarantee; $u\langle a : \tilde{v}\rangle$ denotes a public, but *authentic* output, which certifies the originator of the message, and ensures that the message cannot be replayed; $u\langle - : \tilde{v}\rangle^{\bullet}$ denotes a *secret* transmission, providing guarantees that only the intended receiver will have access to the message payload; finally, $u\langle a : \tilde{v}\rangle^{\bullet}$ denotes a *secure* transmission, combining the guarantees of the authentic and secret modes. In sum, the secret outputs protect from message disclosure, while authentic outputs protect against replication and forging. On the other hand, an opponent may intercept all outputs, and then selectively forward them back to the network.

The input forms have dual semantics: $a(v : \tilde{y})^{\circ}.H$ denotes an input, which consumes a message sent on a from v or $-$, binding $\tilde{y}$ to the tuple of names that form the payload. The input prefix is thus a binder for the variable $\tilde{y}$, whose scope is H: instead, v must be instantiated at the time the input prefix is ready to fire. As for output, $\circ$ signals the secrecy mode and v the authenticity one. Inputs and outputs must agree on the transmission mode to synchronize.

Like in the *local* pi-calculus [10], we make a clear distinction between the input and output capabilities for communication, and we disallow the transmission of the former. Note, to this regard, that input prefixes are built around names (not variables) in the channel position. Similarly, we require a name in the sender position of authentic messages. Taken together, these constraints guarantees that a process H never gets to dynamically *impersonate* a new identity, in the following sense:

Definition 2.1 [Impersonation] A process H impersonates an identity a iff H uses a as the subject of $a(\underline{u} : \tilde{y})^{\circ}.H$, or as the source of an authentic (public or secret) output, as in $\overline{u}\langle a : \tilde{v}\rangle^{\circ}$.

2.2 Networks and Intruders

Networks provide the low-level counterpart of the high-level calculus we just discussed. In addition, they make it possible to express the capabilities of an attacker. The syntax is given below: within networks, names are partitioned into two sets $\mathbf{N}_t$ and $\mathbf{N}_u$ of *trusted* and *untrusted* identities, respectively. By convention, we assume that α-renaming respects this partition.

$$\begin{array}{rll}
M, N, O ::= & \overline{u}\langle \underline{a} : \tilde{v} \parallel \tilde{t}\rangle^{\circ} & \text{(Low Output)} \\
\mid & a(\underline{u} : \tilde{y} \parallel \tilde{z})^{\circ}.M & \text{(Low Input)} \\
\mid & \mathbf{0} \mid M|N \mid A\langle\tilde{u}\rangle \mid (\boldsymbol{\nu} a)N \mid \mathsf{if}\ u = v\ \mathsf{then}\ M\ \mathsf{else}\ N & \\
\mid & \dagger z(x : \tilde{y} \parallel \tilde{w})^{\circ}_{i}.M & \text{(Intercept)} \\
\mid & \mathbf{!}i & \text{(Forward/Replay)}
\end{array}$$

The first two productions introduce the network-level primitives for input and output and are subject to the same restrictions about the use of names as in the high-level syntax. The novelty is in the additional components $\tilde{t}$ of the output messages: these represent the network view of the payload, i.e. the view of the payload given to an external observer of the message, and are bound to the variables $\tilde{z}$ in the input prefix upon synchronization. The last two productions define the adversarial primitives. The intercept prefix $\dagger z(x : \tilde{y} \parallel \tilde{w})^{\circ}_{i}.M$ enables an adversary to intercept all network messages. The prefix is a binder for the name i and all its component variables, with scope M: intercepting the output $\overline{b}\langle a : \tilde{m} \parallel \tilde{n}\rangle^{\circ}$ creates a copy of the message indexed by the fresh name i and binds z to the target b, x to a and $\tilde{w}$ to $\tilde{n}$. As to $\tilde{y}$, the binding depends on the secrecy mode of the message and on the trust status of the identity b. In particular, if the message is secret and $b \in \mathbf{N}_t$ then $\tilde{y}$ gets bound to $\tilde{n}$, otherwise $\tilde{y}$ is bound to $\tilde{m}$. Notice (i) that intercepting a secret message directed to a trusted principal does not break the secrecy of the payload, and (ii) that a message can be intercepted even if it is directed to a restricted identity, as in $(\boldsymbol{\nu} b)b\langle a : \tilde{m} \parallel \tilde{n}\rangle^{\circ}$. The cached copies of the intercepted messages may be manipulated by way of the replay/forward form $\mathbf{!}i$ that uses the index i to forward a copy back to the network, or to produce a replica (in case the original messages was not authenticated).

Definition 2.2 [WELL-FORMED NETWORKS] We say that plain output $\overline{u}\langle - : \tilde{v} \parallel \tilde{t}\rangle$ is well-formed iff $\tilde{v} = \tilde{t}$; a secret/secure $\overline{u}\langle \underline{a} : \tilde{v} \parallel \tilde{t}\rangle^{\bullet}$ or authentic $\overline{u}\langle a : \tilde{v} \parallel \tilde{t}\rangle$ output is well-formed iff $|\tilde{v}| = |\tilde{t}|$. A network N is well-formed iff it is closed (has no free variable) and all of its outputs are well-formed.

The network view of a message depends on the transmission mode: it coincides with the payload in plain outputs, while it is a fresh tuple of names in each authentic and/or secret output. The correspondence between message formats is established by the following translation of high-level principals H into their network level counterparts $[H]$. We only give the clauses for the communication forms (the remaining clauses are defined homomorphically). As discussed in [5], in a cryptographic implementation, the chosen format may be realized by means of a time-dependent signature, in an authentic message, and by a randomized encryption in a secret output.

$$\begin{array}{lll}
[\overline{u}\langle - : \tilde{v}\rangle] & \triangleq \overline{u}\langle - : \tilde{v} \parallel \tilde{v}\rangle & \\
[\overline{u}\langle a : \tilde{v}\rangle] & \triangleq (\boldsymbol{\nu}\tilde{c})\overline{u}\langle a : \tilde{v} \parallel \tilde{c}\rangle & (|\tilde{v}| = |\tilde{c}|) \\
[\overline{u}\langle \underline{a} : \tilde{v}\rangle^{\bullet}] & \triangleq (\boldsymbol{\nu}\tilde{c})\overline{u}\langle \underline{a} : \tilde{v} \parallel \tilde{c}\rangle^{\bullet} & (|\tilde{v}| = |\tilde{c}|) \\
[b(\underline{u} : \tilde{x})^{\circ}.H] & \triangleq b(\underline{u} : \tilde{x} \parallel \tilde{y})^{\circ}.[H] & (|\tilde{x}| = |\tilde{y}| \wedge \tilde{y} \cap \mathit{fv}(H) = \emptyset)
\end{array}$$

The partition on the set of identities makes it possible to identify, within a network, the trusted components from the intruder.

Definition 2.3 [TRUSTED PROCESSES VS INTRUDERS] A network process N is *trusted* iff $N \equiv [H]$, with H high-level principal, it only impersonates identities in the set $\mathbf{N}_t$ and only creates fresh names in the same set. A network process N is an *opponent/intruder* iff it only impersonates identities in the set $\mathbf{N}_u$ and only creates fresh names in the same set.

Throughout, we reserve the letters P and Q to range over the class of trusted processes, and their run-time derivatives.

3 Barbed Equivalence

The dynamics of the calculus is given in Table 1 in terms of reduction and structural congruence. To formalize the dynamics of networks, we need a special form to represent the copies of the messages stored upon interception. We introduce the new form below as part of what we call run-time network configurations:

$$M, N, O ::= \ldots \text{as in Section } 2 \ldots \mid b\langle a : \tilde{m} \parallel \tilde{c}\rangle^{\circ}_{i}$$

Structural congruence

(Struct Par Comm)	$M \mid N \equiv N \mid M$	
(Struct Par Assoc)	$(N \mid N') \mid N'' \equiv N \mid (N' \mid N'')$	
(Struct Par Zero)	$N \mid \mathbf{0} \equiv N$	
(Struct Res Zero)	$(\nu a)\mathbf{0} \equiv \mathbf{0}$	
(Struct Res Comm)	$(\nu a)(\nu b)N \equiv (\nu b)(\nu a)N$	
(Struct Res Par)	$M \mid (\nu a)N \equiv (\nu a)(M \mid N)$	when $a \notin fn(M)$
(Struct Rec)	$A\langle\tilde{w}\rangle \equiv N\{\tilde{w}/\tilde{x}\}$	if $A(\tilde{x}) \overset{\text{def}}{=} N$ and $\lvert\tilde{w}\rvert = \lvert\tilde{x}\rvert$
(Struct If True)	$\mathsf{if}\ a = a\ \mathsf{then}\ M\ \mathsf{else}\ N\ \equiv\ M$	
(Struct If False)	$\mathsf{if}\ a = b\ \mathsf{then}\ M\ \mathsf{else}\ N \equiv\ N$	when $a \neq b$
(Struct Equiv)	$M \equiv M,\ M \equiv N$ implies $N \equiv M$, $N \equiv N'$ and $N' \equiv N''$ imply $N \equiv N''$	
(Struct Cong)	$N \equiv N'$ implies $(\boldsymbol{\nu} n)N \equiv (\boldsymbol{\nu} n)N'$ and $N \mid N'' \equiv N' \mid N''$	

Reduction

In the (Intercept) rule $i \notin \{b, \underline{a}, \tilde{m}, \tilde{c}\}$, σ is the substitution $\{b/z,\ \underline{a}/x,\ \tilde{p}/\tilde{y},\ \tilde{c}/\tilde{w}\}$, and the $\tilde{p}$ are as follows: if $\circ = \bullet$ and $b \in \mathbf{N}_t$ then $\tilde{p} = \bar{\tilde{c}}$ else $\tilde{p} = \tilde{m}$.

(Struct)
$$\begin{array}{c} M \equiv M' \quad M' \longrightarrow N' \quad N' \equiv N \\[1em] M \longrightarrow N \end{array}$$

(Res)
$$\begin{array}{c} N \longrightarrow N' \\[1em] (\nu a)N \longrightarrow (\nu a)N' \end{array}$$

(Par)
$$\begin{array}{c} M \longrightarrow M' \\[1em] M \mid N \longrightarrow M' \mid N \end{array}$$

$$\begin{array}{lrcl}
\text{(Intercept)} & \bar{b}\langle \underline{a} : \tilde{m} \parallel \tilde{c}\rangle^{\circ} \ \mid\ \dagger z(x : \tilde{y} \parallel \tilde{w})_i.N & \longrightarrow & (\boldsymbol{\nu} i)(\bar{b}\langle \underline{a} : \tilde{m} \parallel \tilde{c}\rangle^{\circ}_{i} \ \mid\ N\sigma) \\
\text{(Comm)} & \bar{b}\langle \underline{a} : \tilde{m} \parallel \tilde{c}\rangle^{\circ} \ \mid\ b(\underline{a} : \tilde{y} \parallel \tilde{z})^{\circ}.N & \longrightarrow & N\{\tilde{m}/\tilde{y},\ \tilde{c}/\tilde{z}\} \\
\text{(Forward)} & \bar{b}\langle a : \tilde{m} \parallel \tilde{c}\rangle^{\circ}_{i} \ \mid\ \mathbf{!}i & \longrightarrow & \bar{b}\langle a : \tilde{m} \parallel \tilde{c}\rangle^{\circ} \\
\text{(Replay)} & \bar{b}\langle - : \tilde{m} \parallel \tilde{c}\rangle^{\circ}_{i} \ \mid\ \mathbf{!}i & \longrightarrow & \bar{b}\langle - : \tilde{m} \parallel \tilde{c}\rangle^{\circ}_{i} \ \mid\ \bar{b}\langle - : \tilde{m} \parallel \tilde{c}\rangle^{\circ}
\end{array}$$

Table 1
Reduction Semantics

The index i attached to the cached copy is associated univocally with the intercept that created the copy, as shown in the (Intercept) rule. Notice, in the same rule, that the bindings created depend on the structure and, more specifically, on the secrecy of the intercepted message, as explained earlier on. As for the remaining reductions, (Comm) is the usual synchronization rule, while (Forward) and (Reply) formalize the semantics of the adversarial form $\mathbf{!}i$. Notice in particular that a non-authentic message is replicated, while an authentic one is not (the cached copy is erased).

The semantics of the calculus is completed by a notion of contextual equality based on reduction barbed congruence [8]. We first define the observation predicate, as usual in terms of barbs.

Definition 3.1 [Barb] We write $N{\downarrow}b$ whenever $N \equiv (\boldsymbol{\nu}\tilde{n})(b\langle \ldots \rangle^{\circ} \mid N')$ and $b \notin \tilde{n}$

Definition 3.2 [Intruder Equivalence] A symmetric relation $\mathscr{R}$ on (run-time) networks is (i) *barb preserving* if $M\mathscr{R}N$ and $M \downarrow b$ imply $N \downarrow b$; (ii) *reduction closed* if $M\mathscr{R}N$ and $M \longrightarrow M'$ imply $N \longrightarrow N'$ with $M'\mathscr{R}N'$; (iii) *contextual* if $M\mathscr{R}N$

implies $M \,|\, O \mathscr{R} N \,|\, O$ for all (closed) intruder O and $(\boldsymbol{\nu}\tilde{n})M\mathscr{R}(\boldsymbol{\nu}\tilde{n})N$ for all names $\tilde{n} \in \mathbf{N}_u$. Intruder equivalence $\simeq_I$ is the largest equivalence relation that is reduction closed, barb-preserving and contextual.

Notice that we define observation in terms of strong bisimulation: this is a consequence of the fact that all interactions occur on the network. Also note that we restrict to *adversarial* contexts, as our intention is to find a reasoning method specifically targeted at the analysis of security-centric properties. Accordingly, we define two processes equivalent if they cannot be distinguished by any opponent/intruder that observes them and/or actively interacts with them, reading, intercepting, forwarding and replaying the messages exchanged, or forging new ones.

4 Labelled transitions and bisimilarity

We give an alternative formulation of the semantics of networks, based on labelled transitions. The labelled transitions are organized in two sets. A first set, in Table 2, collects the transitions that correspond to the reduction semantics of Section 3. In most cases the transitions are either standard, or constitute the direct counterpart of the corresponding reductions in Table 1. The two (Output Intercepted) transitions deserve more attention. First notice that, when the receiver is trusted, the label exhibits different information depending on the secrecy mode of the output. Secondly, observe that the transitions leave in their residual a cached copy of the message emitted: this reflects the effect of an interaction with a surrounding context that tests the presence of an output by intercepting it. A further remark is in order on the difference between the two rules that govern scope extrusion. The difference is best understood if we take the view that a channel name comprises the two identities of the end-points it connects: the source and the destination. Under this interpretation the (Open) rule states that the channel name is not extruded, as in the pi-calculus, while the (Open Intercepted) opens the scope in accordance with the reduction semantics by which intercepting a message discloses the identity of the receiver (as well of the sender) even though restricted. The following, standard result connects the reductions with the silent actions in the labelled transition semantics.

Lemma 4.1 (Harmony)

- *If* $M \xrightarrow{\alpha} M'$ *and* $M \equiv N$ *then* $N \xrightarrow{\alpha} N'$ *and* $M' \equiv N'$
- $N \longrightarrow N'$ *if and only if* $N \xrightarrow{\tau} \equiv N'$.

A second set of labelled transitions, in Definition 4.2, provide the observable counterpart of the labelled transitions of Table 2. They are obtained by the transitions in in Table 2 by filtering away all the transitions that involve the adversarial forms (intercept and forward/reply) as well all the transitions that may not be observed by an opponent by virtue of the restriction the opponent suffers on the use of the trusted identities of a network.

Common Transitions

(Input)

$$\frac{}{b(\underline{a} : \tilde{y} \parallel \tilde{w})^{\circ}.N \xrightarrow{b(\underline{a}:\tilde{m}\parallel\tilde{c})^{\circ}} N\{\tilde{m}/\tilde{y},\ \tilde{c}/\tilde{w}\}}$$

(Output)

$$\overline{b}\langle\underline{a} : \tilde{m} \parallel \tilde{c}\rangle^{\circ} \xrightarrow{\overline{b}\langle\underline{a}:\tilde{m}\parallel\tilde{c}\rangle^{\circ}} \mathbf{0}$$

(Secret Output Intercepted)

$$b \in \mathbf{N}_t \qquad i \notin \{b, \underline{a}, \tilde{m}, \tilde{c}\}$$

$$\overline{b}\langle\underline{a} : \tilde{m} \parallel \tilde{c}\rangle^{\bullet} \xrightarrow{(i)\dagger\overline{b}\langle\underline{a}:\tilde{c}\parallel\tilde{c}\rangle_i^{\bullet}} \overline{b}\langle\underline{a} : \tilde{m} \parallel \tilde{c}\rangle_i^{\bullet}$$

(Output Intercepted)

$$\frac{b \notin \mathbf{N}_t \text{ or } \circ \neq \bullet \qquad i \notin \{b, \underline{a}, \tilde{m}, \tilde{c}\}}{\overline{b}\langle\underline{a} : \tilde{m} \parallel \tilde{c}\rangle^{\circ} \xrightarrow{(i)\dagger\overline{b}\langle\underline{a}:\tilde{m}\parallel\tilde{c}\rangle_i^{\circ}} \overline{b}\langle\underline{a} : \tilde{m} \parallel \tilde{c}\rangle_i^{\circ}}$$

(Open)

$$\frac{N \xrightarrow{(\tilde{p})\overline{b}\langle\underline{a}:\tilde{m}\parallel\tilde{c}\rangle^{\circ}} N' \quad n \in \{\tilde{m}, \tilde{c}\} - \{b, \underline{a}, \tilde{p}\}}{(\boldsymbol{\nu} n)N \xrightarrow{(n,\tilde{p})\overline{b}\langle\underline{a}:\tilde{m}\parallel\tilde{c}\rangle^{\circ}} N'}$$

(Open Intercepted)

$$\frac{N \xrightarrow{(\tilde{p},i)\dagger\overline{b}\langle\underline{a}:\tilde{m}\parallel\tilde{c}\rangle_i^{\circ}} N' \quad n \in \{b, \underline{a}, \tilde{m}, \tilde{c}\} - \{\tilde{p}, i\}}{(\boldsymbol{\nu} n)N \xrightarrow{(n,\tilde{p},i)\dagger\overline{b}\langle\underline{a}:\tilde{m}\parallel\tilde{c}\rangle_i^{\circ}} N'}$$

(Replay/Forward)

$$\mathbf{!}i \xrightarrow{\langle i\rangle} \mathbf{0}$$

(Intercept)

$$\frac{}{\dagger z(x : \tilde{y} \parallel \tilde{w})_i^{\circ}.N \xrightarrow{\dagger b(\underline{a}:\tilde{p}\parallel\tilde{c})_i^{\circ}} N\{b/z,\ \underline{a}/x,\ \tilde{p}/\tilde{y},\ \tilde{c}/\tilde{w}\}}$$

(Restr)

$$N \xrightarrow{\alpha} N' \quad n \notin n(\alpha)$$

$$(\boldsymbol{\nu} n)N \xrightarrow{\alpha} (\boldsymbol{\nu} n)N'$$

(Cond)

$$\frac{(a = b \wedge M \xrightarrow{\alpha} N) \vee (a \neq b \wedge M' \xrightarrow{\alpha} N)}{\textsf{if } a = b \textsf{ then } M \textsf{ else } M' \xrightarrow{\alpha} N}$$

(Par)

$$M \xrightarrow{\alpha} M' \quad bn(\alpha) \cap fn(N) = \emptyset$$

$$M \mid N, N \mid M \xrightarrow{\alpha} M' \mid N, N \mid M'$$

(Rec)

$$N\{\tilde{w}/\tilde{x}\} \xrightarrow{\alpha} N' \quad A(\tilde{x}) \stackrel{\text{def}}{=} N$$

$$A\langle\tilde{w}\rangle \xrightarrow{\alpha} N'$$

Intruder

(Synch)

$$\frac{M \xrightarrow{(\tilde{p})\alpha} M' \quad N \xrightarrow{\alpha} N' \quad \tilde{p} \cap fn(N) = \emptyset}{M \mid N \xrightarrow{\tau} (\boldsymbol{\nu}\tilde{p})(M' \mid N')}$$

(Co-replay)

$$\frac{}{\overline{b}\langle - : \tilde{m} \parallel \tilde{c}\rangle_i^{\circ} \xrightarrow{(i)} \overline{b}\langle - : \tilde{m} \parallel \tilde{c}\rangle_i^{\circ} \ \mid\ \overline{b}\langle - : \tilde{m} \parallel \tilde{c}\rangle^{\circ}}$$

(Co-forward)

$$\overline{b}\langle a : \tilde{m} \parallel \tilde{c}\rangle_i^{\circ} \xrightarrow{(i)} \overline{b}\langle a : \tilde{m} \parallel \tilde{c}\rangle^{\circ}$$

Table 2
Labelled Transition Semantics

Definition 4.2 [Observable LTS] We say that a network has an observable transition, noted $N \overset{\alpha}{\longmapsto} N'$, if and only if it may be derived by the following rules:

$$\begin{array}{c} N \xrightarrow{\alpha} N' \\[1em] N \overset{\alpha}{\longmapsto} N' \end{array} \qquad \alpha \notin \left\{ \begin{array}{ll} b(a : \tilde{m} \parallel \tilde{c})^{\circ} & a \in \mathbf{N}_t \\ (\tilde{p})\bar{b}\langle \underline{a} : \tilde{m} \parallel \tilde{c} \rangle^{\circ} & b \in \mathbf{N}_t \\ \dagger b(\underline{a} : \tilde{m} \parallel \tilde{c})^{\circ}_{i}, \langle i \rangle & \end{array} \right\}$$

The notions of synchronous and asynchronous bisimulation arise as expected from the observable labelled transitions.

Definition 4.3 [Intruder Bisimularity] Let $\mathscr{R}$ be a symmetric relation over networks. $\mathscr{R}$ is a *bisimulation* if whenever $M \mathscr{R} N$ and $M \overset{\alpha}{\longmapsto} M'$ with $bn(\alpha) \cap fn(N) = \emptyset$ there exists N' such that $N \overset{\alpha}{\longmapsto} N'$ and $M' \mathscr{R} N'$.

$\mathscr{R}$ is an *asynchronous bisimulation* if whenever $M \mathscr{R} N$ and $M \overset{\alpha}{\longmapsto} M'$ with $bn(\alpha) \cap fn(N) = \emptyset$ one has: (i) if α is not an input, then $N \overset{\alpha}{\longmapsto} N'$ and $M' \mathscr{R} N'$; (ii) if α is an input, then $N \overset{\alpha}{\longmapsto} N'$ and $M' \mathscr{R} N'$ or $N \overset{\tau}{\longmapsto} N'$ and $M' \mathscr{R} N' \mid \overline{\alpha}$.

Bisimilarity $\sim$ is the largest bisimulation, and *asynchronous bisimilarity* $\sim_a$ is the largest asynchronous bisimulation.

By adapting the proof in [5] we can show that $\sim_a$ implies $\simeq_I$. Hence $\sim_a$ is a sound, purely coinductive, proof technique for $\simeq_I$. In the next two theorems, we implicitly assume processes to be derivatives of trusted processes.

Theorem 4.4 (Soundness) $\sim_a \subseteq \simeq_I$

A further, more interesting relationship exists between the asynchronous and synchronous versions of bisimilarity. Indeed, the ability to intercept all traffic makes asynchronous bisimilarity just as powerful as as synchronous bisimilarity.

Theorem 4.5 (Asynchronous vs synchronous Bisimilarity) $\sim_a = \sim$

Proof sketch. Clearly $\sim \subseteq \sim_a$ because, by definition, a synchronous bisimulation is also an asynchronous bisimulation. To prove the reverse inclusion, we show that $\mathscr{R} = \{(P, Q) \mid P \sim_a Q\}$ is a bisimulation. The crux of the proof is to show that whenever $P \sim_a Q$, the input actions by P (resp. Q) may always be matched by corresponding input actions by Q (resp. P). We give the outline, leaving the somewhat elaborate details to the full version of the paper.

Assume $P \overset{\beta}{\longmapsto} P'$ and let β be an input action. Then Q may match this move by a τ transition: however, this transition is not entirely silent, because, to make the τ step, Q must emit an output, which can be intercepted, hence observed, by the environment. Let then Q' be the process reached by Q via the output (intercepted) transition. Since $P \sim_a Q$, P must have a corresponding transition to a $P' \sim_a Q'$. At this stage, we still have $P' \overset{\beta}{\longmapsto} P''$ for some P'', and we can repeat the same reasoning we just made, now on Q' rather than Q. The reasoning may be repeated only a finite number of times, unless P and Q have infinitely many outputs ready to fire, which cannot happen, as replication, and recursion, may only occur guarded in our processes. Thus, after a finite number of steps, it must be the case that Q responds to the β transition by P with a corresponding β transition. The proof follows by showing that, indeed, Q may respond to the input action at the very first step, as the input action β commutes with all the intervening outputs involved in the *bisimulation game* we just illustrated. □

The proof breaks if we lift the restriction that recursion be guarded: indeed, not only the proof breaks, but the result itself is false. Here is a counter-example. Let $*R$ denote the replicated version of R, defined by the unguarded recursive definition $*R \stackrel{\mathrm{def}}{=} *R \mid R$. Consider

$$\begin{array}{lcl} P & \stackrel{\mathrm{def}}{=} & *\,\overline{a}\langle - : m\rangle \mid *\,a(- : x).\overline{a}\langle - : x\rangle \mid b(- : x).\overline{b}\langle - : x\rangle \\ Q & \stackrel{\mathrm{def}}{=} & *\,\overline{a}\langle - : m\rangle \mid *\,a(- : x).\overline{a}\langle - : x\rangle \end{array}$$

Clearly $P \not\approx Q$, because there is no way for Q to match the input transition available for P on b. On the other hand, the two processes cannot be distinguished in the asynchronous version of -bisimilarity as P's move on b, $P \xrightarrow{b(-:n)} Q \mid \overline{b}\langle - : n\rangle$, may be matched by Q via a τ-transition that takes Q back to itself (thanks to the presence of the replicated output).

The result breaks similarly in the presence of a choice operator $P_1 + P_2$, with the usual semantics $P_1 + P_2 \xrightarrow{\alpha} P'$ if $P_1 \xrightarrow{\alpha} P'$ or $P_2 \xrightarrow{\alpha} P'$. The counterexample is given by the following processes, where $a(- : x)*$ denotes the guarded recursive process $Q \stackrel{\mathrm{def}}{=} a(- : x).Q$.

$$\begin{array}{lcl} P' & \stackrel{\mathrm{def}}{=} & a(- : x)* \mid (\overline{a}\langle - : x\rangle + b(- : x).\overline{b}\langle - : x\rangle) \\ Q' & \stackrel{\mathrm{def}}{=} & a(- : x)* \mid \overline{a}\langle - : x\rangle \end{array}$$

Clearly $P' \not\approx Q'$, because there is no way for Q' to match the input transition available for P' on b. On the other hand, the two processes cannot be distinguished in the asynchronous version of bisimilarity as $P' \xrightarrow{b(-:n)} a(- : x)* \mid \overline{b}\langle - : x\rangle$, may be matched by $Q' \xrightarrow{\tau} a(- : x)*$.

5 Eavesdroppers

We continue our analysis on the strength of our security abstractions by extending the set of adversarial forms to include an eavesdrop primitive.

$$M, N ::= \ldots \text{ (as in Section 2)} \ldots \mid\ ?z(x : \tilde{y} \parallel \tilde{w})^{\circ}_{i}.M$$

Like the intercept prefix, $?z(x : \tilde{y} \parallel \tilde{w})^{\circ}_{i}.M$ is a binder for the name i and for all of its component variables, with scope M. The semantics is given in Table 3: the reductions and labelled transitions follow the exact same rationale as the corresponding

Reduction: as in the intercept rules, σ is the substitution $\{b/z,\ \underline{a}/x,\ \tilde{p}/\tilde{y},\ \tilde{c}/\tilde{w}\}$, and the $\tilde{p}$ are as follows: if $\circ = \bullet$ and $b \in \mathbf{N}_t$ then $\tilde{p} = \tilde{c}$ else $\tilde{p} = \tilde{m}$. Moreover, in the (Eavesdrop) rule, $i \notin \{b, \tilde{m}, \tilde{c}\}$.

$$\text{(Eavesdrop Auth)}\quad \overline{b}\langle a : \tilde{m} \parallel \tilde{c}\rangle^{\circ} \mid ?z(x : \tilde{y} \parallel \tilde{w})^{\circ}_{i}.N \longrightarrow \overline{b}\langle \underline{a} : \tilde{m} \parallel \tilde{c}\rangle^{\circ} \mid (\boldsymbol{\nu} i)N\sigma$$

$$\text{(Eavesdrop)}\quad \overline{b}\langle - : \tilde{m} \parallel \tilde{c}\rangle^{\circ} \mid ?z(x : \tilde{y} \parallel \tilde{w})^{\circ}_{i}.N \longrightarrow \overline{b}\langle - : \tilde{m} \parallel \tilde{c}\rangle^{\circ} \mid (\boldsymbol{\nu} i)(\overline{b}\langle - : \tilde{m} \parallel \tilde{c}\rangle^{\circ}_{i} \mid N\sigma)$$

Labelled Transitions:

(Output Eavesdropped)

$$\frac{b \notin \mathbf{N}_t \text{ or } \circ \neq \bullet \qquad i \notin \{b, \tilde{m}, \tilde{c}\}}{\overline{b}\langle - : \tilde{m} \parallel \tilde{c}\rangle^{\circ} \xrightarrow{(i)?\overline{b}\langle - : \tilde{m} \parallel \tilde{c}\rangle^{\circ}_{i}} \overline{b}\langle - : \tilde{m} \parallel \tilde{c}\rangle^{\circ}_{i} \mid \overline{b}\langle - : \tilde{m} \parallel \tilde{c}\rangle^{\circ}}$$

(Output Eavesdropped Auth)

$$\frac{b \notin \mathbf{N}_t \text{ or } \circ \neq \bullet \qquad i \notin \{b, a, \tilde{m}, \tilde{c}\}}{\overline{b}\langle a : \tilde{m} \parallel \tilde{c}\rangle^{\circ} \xrightarrow{(i)?\overline{b}\langle a : \tilde{m} \parallel \tilde{c}\rangle^{\circ}_{i}} \overline{b}\langle a : \tilde{m} \parallel \tilde{c}\rangle^{\circ}}$$

(Secret Output Eavesdropped)

$$\frac{b \in \mathbf{N}_t \qquad i \notin \{b, \tilde{m}, \tilde{c}\}}{\overline{b}\langle - : \tilde{m} \parallel \tilde{c}\rangle^{\bullet} \xrightarrow{(i)?\overline{b}\langle - : \tilde{c} \parallel \tilde{c}\rangle^{\bullet}_{i}} \overline{b}\langle - : \tilde{m} \parallel \tilde{c}\rangle^{\bullet}_{i} \mid \overline{b}\langle - : \tilde{m} \parallel \tilde{c}\rangle^{\bullet}}$$

(Secret Output Eavesdropped Auth)

$$\frac{b \in \mathbf{N}_t \qquad i \notin \{b, a, \tilde{m}, \tilde{c}\}}{\overline{b}\langle a : \tilde{m} \parallel \tilde{c}\rangle^{\bullet} \xrightarrow{(\boldsymbol{\nu} i)?\overline{b}\langle a : \tilde{c} \parallel \tilde{c}\rangle^{\bullet}_{i}} \overline{b}\langle a : \tilde{m} \parallel \tilde{c}\rangle^{\bullet}}$$

(Open Eavesdropped)

$$\frac{N \xrightarrow{(\tilde{p}, i)?\overline{b}\langle \underline{a} : \tilde{m} \parallel \tilde{c}\rangle^{\circ}_{i}} N' \quad n \in \{b, \underline{a}, \tilde{m}, \tilde{c}\} - \{\tilde{p}, i\}}{(\boldsymbol{\nu} n)N \xrightarrow{(n, \tilde{p}, i)?\overline{b}\langle \underline{a} : \tilde{m} \parallel \tilde{c}\rangle^{\circ}_{i}} N'}$$

(Eavesdrop)

$$\frac{}{?z(x : \tilde{y} \parallel \tilde{w})_{i}.N \xrightarrow{?b(\underline{a} : \tilde{p} \parallel \tilde{c})^{\circ}_{i}} N\{b/z,\ \underline{a}/x,\ \tilde{p}/\tilde{y},\ \tilde{c}/\tilde{w}\}}$$

Table 3
Semantics of Eavesdropping

rules for the intercept primitive, with the differences (i) that eavesdropping does not consume the output, and hence (ii) that it does not create a copy in case the output is authentic.

In the rest of this section we analyze the import of eavesdropping on the discriminating power of the intruder. In that direction we let $(\sim^{\kappa})_{\kappa \subseteq \{\dagger, ?, !\}}$ denote the family of bisimilarity relationships associated with the corresponding set of adversarial primitives. Similarly, we define the set $(\sim^{\kappa}_{a})_{\kappa \subseteq \{\dagger, ?, !\}}$ for the asynchronous setting, and look at the relative strength of (some of) the equivalences in these sets.

Our first result shows that eavesdropping does not give any additional discriminatory power.

Theorem 5.1 $\sim^{\dagger !}_{a} = \sim^{\dagger ? !}_{a}$, *and similarly* $\sim^{\dagger !} = \sim^{\dagger ? !}$.

Proof sketch. Clearly, $\sim^{\dagger ? !}_{a} \subseteq \sim^{\dagger !}_{a}$. For the reverse inclusion, we show that eavesdropping can be encoded in terms of intercept and forward/reply. The same technique applies in the proof for the synchronous relationships. □

Next, we show that eavesdropping is strictly less powerful than intercepting.

Theorem 5.2 $\sim^{\dagger !}_{a} \subsetneq \sim^{? !}_{a}$ *and similarly* $\sim^{\dagger !} \subsetneq \sim^{? !}$.

Proof sketch. Clearly $\sim^{\dagger ? !}_{a} \subseteq \sim^{? !}_{a}$, which implies $\sim^{\dagger !}_{a} \subseteq \sim^{? !}_{a}$ by Theorem 5.1 above. The exact same reasoning applies in the synchronous case. That the inclusions are strict follows by the following counter-example, which applies uniformly to the

Reductions:

$$\text{(Forward)}\quad \bar{b}\langle a : \tilde{m} \parallel \tilde{c}\rangle_i^\circ \mid\ !i \mid b(a : \tilde{y} \parallel \tilde{z}).N \longrightarrow N\{\tilde{m}/\tilde{y},\ \tilde{c}/\tilde{z}\}$$

$$\text{(Replay)}\quad \bar{b}\langle - : \tilde{m} \parallel \tilde{c}\rangle_i^\circ \mid\ !i \mid b(- : \tilde{y} \parallel \tilde{z}).N \longrightarrow \bar{b}\langle - : \tilde{m} \parallel \tilde{c}\rangle_i^\circ \mid N\{\tilde{m}/\tilde{y},\ \tilde{c}/\tilde{z}\}$$

Labelled Transitions

$$\text{(Co-replay)}\quad \frac{N \xrightarrow{b(-:\tilde{m}\|\tilde{c})^\circ} N'}{\bar{b}\langle - : \tilde{m} \parallel \tilde{c}\rangle_i^\circ \mid N \xrightarrow{(i)^*} \bar{b}\langle - : \tilde{m} \parallel \tilde{c}\rangle_i^\circ \mid N'}$$

$$\text{(Co-forward)}\quad \begin{array}{c} N \xrightarrow{b(a:\tilde{m}\|\tilde{c})^\circ} N' \\ \bar{b}\langle a : \tilde{m} \parallel \tilde{c}\rangle_i^\circ \mid N \xrightarrow{(i)^*} N' \end{array}$$

Table 4
Man-in-the-middle reductions and labelled transitions

synchronous and asynchronous cases. Let $a \in \mathbf{N}_t$, and take the following two processes:

$$P \overset{\text{def}}{=} \bar{a}\langle - : m\rangle \mid \bar{a}\langle - : m\rangle \mid *\, a(- : x).\bar{a}\langle - : x\rangle$$

$$Q \overset{\text{def}}{=} \bar{a}\langle - : m\rangle \mid *\, a(- : x).\bar{a}\langle - : x\rangle$$

P and Q are easily distinguished by using the intercept prefix to count the outputs in the two processes. On the other hand, $P \sim^{?!} Q$ as this counting in not possible for an eavesdropper, because (i) eavesdropping does not consume the messages, and (ii) a is a trusted name, hence the intruder cannot impersonate a to input on a. □

6 Men in the middle

We continue our analysis by looking at the intruder model adopted in [4], which we adapt to our calculus. In this new model, two principals may never engage in a synchronization directly as in our initial semantics in Section 3. Instead, all exchanges require the mediation of the intruder which intercepts all outputs and then delivers them to the processes in the exact moment they are ready to consume them. As a result, the reduction relation arises from the relation defined in Table 1 by dropping the (Comm) rule and by replacing the (Forward) and (Replay) rules with two rules in Table 4.

A corresponding modification is required on the labelled transition semantics to mimic the form of three-way synchronization induced by the new rules of reduction. In particular, the new labelled transitions arise from those defined in Table 2 by (i) replacing the rules (Co-reply) and (Co-forward) with the two rules in Table 4, and (ii) by dropping the (Synch) rule in when α and α are input and output actions, respectively (thus effectively disabling direct synchonization between trusted processes). The observable LTS for the new semantics is derived exactly as we did in Definition 4.2. We let $\overset{*}{\sim}$ be the bisimilarity relationship associated with the new LTS: notice that, having disabled all τ-actions on trusted processes, the relations of asynchronous and synchronous bisimilarity on trusted processes collapse to the same relation $\overset{*}{\sim}$.

At a first look, the new equivalence $\overset{*}{\sim}$ would appear finer than $\sim$ due to the tighter control the new intruder can exercise over the interaction between the principals of a network. As it turns out, however, this additional control does not add any discriminating power.

Theorem 6.1 *On trusted processes,* $\overset{*}{\sim} = \sim$.

Proof sketch. We prove the two inclusions separately, by coinduction. The inclusion $\overset{*}{\sim} \subseteq \sim$ is subtle, as we need to identify the 'right' pairs of run-time processes to be included in the candidate relation. In that direction, we define processes P and Q *cache-consistent* iff $P \equiv (\boldsymbol{\nu}\tilde{p})(\hat{P} \mid \overline{b}\langle \underline{a} : \tilde{m} \parallel \tilde{c}\rangle_i^{\circ})$ if and only if $Q \equiv (\boldsymbol{\nu}\tilde{q})(\hat{Q} \mid \overline{b}\langle \underline{a} : \tilde{m}' \parallel \tilde{c}\rangle_i^{\circ})$, with $\tilde{m} = \tilde{m}'$ whenever $\circ \neq \bullet$ or $b \notin \mathbf{N}_t$. Now we define the relation

$$\mathscr{R} = \{(P, Q) \mid P \overset{*}{\sim} Q \textit{ and } P, Q \textit{ are cache-consistent}\}$$

and show that $\mathscr{R}$ is a $\sim$-bisimulation. This is enough to prove the claim because any two trusted processes are trivially cache-consistent (they have no cached copies). The proof shows that re-emitting a cached message when no process is ready to input it does not given any discriminating power. It derives from the following closure property for $\overset{*}{\sim}$ under the co-reply/co-forward transitions $\overset{(i)}{\longmapsto}$ from the original system:

> Let P and Q be cache-consistent processes. If $P \overset{*}{\sim} Q$ and $P \overset{(i)}{\longmapsto} P'$ then $Q \overset{(i)}{\longmapsto} Q'$ and $P' \overset{*}{\sim} Q'$.

The reverse implication follows by coinduction, using the relation

$$\mathscr{R} = \{(P, Q) \mid P \sim Q\}$$

The subtlety here is to show that whenever $P \sim Q$ and P consumes a forward or a replica of a cached message, then the same must happen on Q. Again, this is a consequence of the discriminating power provided by intercept: were Q not to respond as expected, an observer would be able to tell Q from P based on the presence of an output that cannot be observed in P. □

The hypothesis that P and Q be trusted processes (hence cache-consistent) is crucial for the proof. Indeed, the theorem is false for arbitrary run-time configurations. For instance $\overline{b}\langle - : m \parallel m\rangle_i \overset{*}{\sim} \mathbf{0}$, as neither process has any transition; on the other hand, clearly, $\overline{b}\langle - : m \parallel m\rangle_i \not\sim \mathbf{0}$ as the process on the left has an (i)-transition, while $\mathbf{0}$ clearly has not.

7 Conclusions

We have investigated a core set of abstractions for distributed communication, and assessed their adequacy for security verification by analyzing their interplay with a class of different intruder models. Our results show that the abstractions are robust, in that the observational equivalences they yield are preserved under the different observations available with the different adversarial primitives and interaction models which we have considered.

In its present form, the framework is targeted at secrecy and authentication. Future work includes expending it to account for more advanced properties required in modern network applications such as e-commerce protocols and electronic voting.

Acknowledgement

Work partially supported by MIUR Project *SOFT: Security-Oriented Formal Techniques.*

References

[1] Abadi, M. and C. Fournet, *Mobile values, new names, and secure communication*, in: *POPL 2001: The 28th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, London*, 2001, pp. 104–115.

[2] Abadi, M. and C. Fournet, *Private authentication*, Theor. Comput. Sci. **322** (2004), pp. 427–476.

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

[4] Adão, P. and C. Fournet, *Cryptographically sound implementations for communicating processes*, in: M. Bugliesi, B. Preneel, V. Sassone and I. Wegener, editors, *ICALP (2)*, Lecture Notes in Computer Science **4052** (2006), pp. 83–94.

[5] Bugliesi, M. and R. Focardi, *Language based secure communication*, in: *Proceedings of the 21st IEEE Computer Security Foundations Symposium, CSF 2008, Pittsburgh, Pennsylvania, 23-25 June 2008* (2008), pp. 3–16.

[6] Corin, R., P.-M. Deniélou, C. Fournet, K. Bhargavan and J. J. Leifer, *Secure implementations for typed session abstractions*, in: *20th IEEE Computer Security Foundations Symposium, CSF 2007, 6-8 July 2007, Venice, Italy* (2007), pp. 170–186.

[7] Fournet, C. and T. Rezk, *Cryptographically sound implementations for typed information-flow security*, in: *Proceedings of the 35th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2008, San Francisco, California, USA, January 7-12, 2008* (2008), pp. 323–335.

[8] Honda, K. and N. Yoshida, *On reduction-based process semantics*, Theor. Comput. Sci. **151** (1995), pp. 437–486.

[9] Laud, P., *Secrecy types for a simulatable cryptographic library*, in: V. Atluri, C. Meadows and A. Juels, editors, *ACM Conference on Computer and Communications Security* (2005), pp. 26–35.

[10] Merro, M. and D. Sangiorgi, *On asynchrony in name-passing calculi*, Mathematical Structures in Computer Science **14** (2004), pp. 715–767.

[11] Milner, R., J. Parrow and D. Walker, *A calculus of mobile processes, Parts I and II*, Information and Computation **100** (1992), pp. 1–77.