Factorization of Behavioral Integrity

Ximeng Li(✉), Flemming Nielson, and Hanne Riis Nielson

DTU Compute, Technical University of Denmark, Kongens Lyngby, Denmark
{ximl,fnie,hrni}@dtu.dk

Abstract. We develop a bisimulation-based nonintereference property that describes the allowed dependencies between communication behaviors of different integrity levels. The property is able to capture all possible combinations of integrity levels for the "presence" and "content" of actual communications. Channels of low presence integrity and high content integrity can be used to model the effect of Message Authentication Codes or the consequence of Denial of Service Attacks. In case the distinction between "presence" and "content" is deliberately blurred, the noninterference property specialises to a classical process-algebraic property (called SBNDC). A compositionality result is given to facilitate a structural approach to the analysis of concurrent systems.

1 Introduction

The semantic validation of information flow security [6,15] is achieved with *noninterference* properties [5,7]. Recent proposals of such properties for confidentiality in event-based systems distinguish between the "presence" and "content" of communication events (e.g. [13]).

Consider the simple process $c_1?x.c_2!d$, where some data is received from the confidential channel c_1, and forgotten immediately, with some data d subsequently sent over the public channel c_2. Although the confidential input content is not leaked through the public channel c_2, this process is typically regarded as insecure [2,3,6,8,13,18], in case the input can be occasionally blocked by the environment. The reason is that the "presence" of the confidential input can be leaked through the "presence" of the public output.

When separate confidentiality levels can be assigned for the *presence* and *content* of communication, a more fine-grained analysis can be obtained. Supposing both the "presence" level and the "content" level of c_2 are public, then the content level of c_1 can still be confidential — only the presence level needs to be public, for the process to be secure. However, existing work introduces the constraint that "presence" can be no more confidential than "content" [13,14]: observing the content of a communication implies the knowledge that the communication is happening (present).

By the usually perceived duality [9] between confidentiality and integrity, separating "presence" and "content" applies for integrity as well. Still consider the process $c_1?x.c_2!d$. If both the "presence" and the "content" of communication over c_2 are of high integrity, then only the "presence" of communication

G. Pernul et al. (Eds.): ESORICS 2015, Part II, LNCS 9327, pp. 500–519, 2015.
DOI: 10.1007/978-3-319-24177-7_25

over c_1 needs to be of high integrity as well, the input content can still be of low integrity, for the process to be secure. However, the aforementioned constraint would preclude the use of channels with low presence integrity and high content integrity. Nevertheless, this combination is practically meaningful. When message authentication codes (MAC) are used, a MAC-checker can detect tampered (low integrity) content and choose to accept only high integrity content. As a result, the content, once received by an end user, can be used by her with confidence that no harm will arise. The worry, though, is that the communication allowing to receive that content may not be present. The suspension of this communication may be caused, for example, by message rejection in the MAC-checker due to content corruption. It is therefore sensible to regard the user channel as having low presence integrity and high content integrity.

Our contribution is a novel bisimulation-based noninterference property for integrity, where the *presence* and *content* of communication events are dealt with separately, and *all* combinations of integrity levels for these two dimensions are allowed. The property is shown to degenerate to the classical process-algebraic condition SBNDC (e.g., [6]), and a compositionality result is obtained to facilitate a structural approach to information flow analysis in a concurrent setting.

Our development will be performed in the Quality Calculus [12], a recent extension of the π-calculus. The distinguishing feature of the Quality Calculus is the use of composite binders $\&_q(b_1, ..., b_n)$ that describe the combinations of communications (at the basic input/output binders $b_1, ..., b_n$) that suffice for the computation to proceed. This makes system models more robust, since when faced with a computation environment that does not allow certain communications to be performed, their alternatives could still succeed. Take the composite binder $\&_{1\vee 2}(c_1?x_1, c_2?x_2)$ as an example. This binder is passed immediately after the success of either input. If x_1 has received data d when the binder is passed, then x_1 is bound to $\mathsf{some}(d)$; otherwise to none, which resembles the optional data types used in languages like Standard ML. We can then use the "case construct" of the calculus to model branching decisions based on whether x_1 is bound to $\mathsf{some}(d)$ or none. An example here is $\mathsf{case}\ x_1\ \mathsf{of}\ \mathsf{some}(y) : P_1\ \mathsf{else}\ P_2$ where P_1 and P_2 are two processes.

2 Motivating Examples

Let us give a few examples (in Fig. 1) to frame our mind in terms of *presence* integrity and *content* integrity, and further motivate the noninterference property to be proposed. Channels with two subscripts (L or H, representing their integrity classification) will often be used. The first level describes the presence dimension and the second describes the content dimension. For each subscript, an L (resp. H) will denote low (resp. high) integrity.

Processes 1 and 2 are intuitively secure. In process 1, given the low content integrity and high presence integrity of c_{HL}, the corruption of the input content by an attacker can be "passed on" to the output content, while the input cannot be blocked by the attacker, consequently blocking the output. Hence the low content integrity, high presence integrity of c'_{HL} can be justified in accordance with the integrity classes of c_{HL}. In process 2, any influence on the presence of the input can in turn influence the presence of the output, but cannot by itself corrupt the output content, which also demonstrates the consistency of the integrity classes of c_{LH} and c'_{LH}. On the other hand, process 3 is insecure: the classification of c'_{LH} does not meet the intuition that both the presence of the final output, and its content, can be badly influenced.

One might think that the presence integrity can either be high for all channels, or low for all channels, hence at most one of the classes "high" and "low" is needed. This is not true, as illustrated by the insecurity of process 4, and the security of process 5. In process 4, the content integrity of the output channel cannot be H, since the presence of the input leads to more choices for the output content, some of which may not be possible with the input still blocked.

Given the insecure dependency of high integrity content on low integrity presence in process 4, it becomes interesting to see when certain source channels have the presence level L, which sink channels can still have the content level H without being affected. Process 6 is coded in the Quality Calculus. It is a call to the procedure M whose definition follows. This process is in fact a simple-minded "multiplexer" that directs incoming data from c_{LH} to c'_{LH}, and from c_{LL} to c''_{LL}. Note that if one of the four channels has low presence integrity, then all channels have low presence integrity, since the influence by the presence of communication over one of the channels on the control flow is global. However, c'_{LH} preserves the high content integrity of c_{LH}, despite this pervasive corruption on the "presence" dimension.

1. $c_{HL}?x.c'_{HL}!x$
2. $c_{LH}?x.c'_{LH}!x$
3. $c_{LH}?x_1.c_{LL}?x_2.c'_{LH}!f(x_1,x_2)$,
   where $f(a,b) \neq f(c,d)$
   whenever $a \neq b$ or $c \neq d$
4. $c_{LH}?x_1.c'_{LH}!x_1 \mid c'_{LH}!d$
5. $c_{HH}?x_1.c'_{HH}!x_1 \mid c'_{HH}!d$
6. M where
   $M \triangleq \&_{1\vee 2}(c_{LH}?x_1, c_{LL}?x_2)$.
       $\mathsf{case}\ x_1\ \mathsf{of\ some}(y_1) : c'_{LH}!y_1.M$
       $\mathsf{else\ case}\ x_2\ \mathsf{of\ some}(y_2) : c''_{LL}!y_2.M$
       $\mathsf{else}\ 0$
7. A where
   $A \triangleq \&_{1\vee 2}(c_{LL}?x_1, c'_{LL}?x_2)$.
       $\mathsf{case}\ x_1\ \mathsf{of\ some}(y_1) : c_{HL}!y_1.A$
       $\mathsf{else\ case}\ x_2\ \mathsf{of\ some}(y_2) : c_{HL}!y_2.A$
       $\mathsf{else}\ 0$

Fig. 1. Some example processes

The process 7 is a call to procedure A whose body uses the same predicate $1 \vee 2$, which enables it to source from alternative channels c_{LL} and c'_{LL}. The input content, no matter from which source channel, will be output over the channel c_{HL}. The process is not secure if the environment can block the two inputs at the same time. However, c_{LL} and c'_{LL} might represent sources that are geographically distant or that fail with drastically different causes, which can be modeled by an *environment strategy* (e.g., [10,13]) that always provides at least one of the inputs when the computation proceeds to the composite input binder. The procedure call will be secure under that strategy.

We end this section with a conceptualization of *presence integrity* and *content integrity*, although a more technical characterization comes along with our security property to be presented later.

– Presence integrity: for each i, whether the *existence* of the i-th output/input over channel c in a finite sequence π of communication actions can be influenced by the attacker
– Content integrity: for each i, whether the content of the i-th output/input over channel c can be influenced by the attacker, *in case the input/output exists in a finite sequence* π of communication actions

Note that it is not only whether an input/output on a channel c is eventually available, that matters, but how many times it occurs in each computation sequence, since we are concerned with nonterminating computation and looping behaviors: the processes 6 and 7 in Fig. 1 are such examples.

This paper is structured as follows. In Sect. 3, we present the syntax and semantics of the Quality Calculus. We then present our noninterference condition for behavioral integrity in Sect. 4. Two main theoretical properties of the noninterference condition, including its degeneration to SBNDC, and the compositionality result, are presented in Sect. 5. Further examples are given in Sect. 6, to illustrate the condition and its compositionality. We conclude in Sect. 7, with a few pointers to related work.

3 The Quality Calculus

Syntax. The Quality Calculus [12] has its roots in the π-calculus and CCS, but allows to specify criteria on which communications have to succeed for the computation to continue. This can be expressed by the construct $\&_q(b_1, ..., b_n)$ with predicate q and communication binders b_1,...,b_n. The computation can then continue differently, depending on whether each of these communications has succeeded, using the construct $\mathsf{case}\ e\ \mathsf{of}\ \mathsf{some}(y) : P_1\ \mathsf{else}\ P_2$. The predicate q can refer to any specific binder among b_1, ..., b_n, by its index (1,...,n), and can denote any boolean combination of their evaluation status. Since some previous inputs might be unsuccessful, we allow expressions that are missing data to be evaluated to none, or else to $\mathsf{some}(c)$ with some constant c.

The complete syntax is given in Table 1. Terms t and expressions e are separate syntactical categories that capture the distinction between data and optional data. A constant c is either a channel (**Chn**), or a datum (**Dt**), or both, as we allow $\mathbf{Chn} \cap \mathbf{Dt} \neq \{\}$. For a constant in **Dt**, we also use d (d', etc.) for its denotation. Atomic input binders are of the form $t?x$. Atomic output binders are of the form $t!t'\{x\}$, where the variable x is used as an indicator of whether the output has succeeded, the output content is also bound to x in case it has. We abbreviate $t!t'\{x\}$ to $t!t'$ when such indication provided by x is not needed. With a procedure call $A(\bar{e})$, the procedure A needs to be defined as a process P, with $A(\bar{x}) \triangleq P$. Looping behavior is allowed via recursive procedure calls. The other features not mentioned so far are mostly standard. Although the Quality Calculus does not have a non-deterministic choice operator, an encoding of internal nondeterministic choice (in the style of Hoare's CSP) can be done using composite binders and case constructs, as presented in [12].

Table 1. The syntax of the quality calculus

$t ::= c \mid y \mid f(t_1,\ldots,t_n)$ $e ::= x \mid \mathsf{some}(t) \mid \mathsf{none}$ $b ::= t?x \mid t!t'\{x\} \mid \&_q(b_1,...,b_n)$	$P ::= (\nu c)P \mid P_1 \vert P_2 \mid b.P \mid \mathsf{A}(\bar{e}) \mid 0 \mid$ $\mathsf{case\ e\ of\ some(y)\ :\ P_1\ else\ P_2}$

Semantics. To facilitate the specification of open systems, and the formulation of our security condition, we present a semantics that is parameterized on the computation environment. The tight correspondence of this semantics with the classical semantics [12] of the Quality Calculus is discussed in the appendix.

Processes and binders make transitions together with sequences $\pi \in \Pi$. Each such sequence contains a separator $_\triangle$, which delimits the environment's past actions interacting with the process, and optionally a future communication attempt. Each communication action/attempt is represented by an "abstract binder" $\hat{b} \in AB$ given by the syntax $\hat{b} ::= c?x \mid c?c' \mid c!c' \mid \square$. The abstract binders $c?x$ and $c?c'$ represent a pending input and a completed input, respectively, *of the environment*; on the other hand, $c!c'$ represents either a pending output or a completed output, also *of the environment*. In addition, $\square$ represents the suspension of any communication by one step.

We write $[\pi]_\triangle$ for the prefix of π up to the $_\triangle$ in it, and $\Pi_\triangle$ for the set $\{[\pi]_\triangle \mid \pi \in \Pi\}$. Next introduce environment strategies $\delta : \Pi_\triangle \to 2^{AB} \setminus \{c?c' \vert c, c' \in \mathbf{Chn} \cup \mathbf{Dt}\}$ from the set $\mathbf{Strat}$. For each $\pi \in \Pi_\triangle$, $\delta(\pi)$ gives the set of abstract binders that represent the environment's intended ways of "exercising" the specification for one more step. In case $\delta(\pi)$ is an input abstract binder, it will be of the form $c?x$ rather than $c?c'$, since it represents a pending input.

The transition relation for processes and binders is given in Table 2. We make use of an unspecified evaluation relation $\triangleright$ for terms and expressions. For binders, each transition rule is of the form $\langle b,\ \pi\rangle \xrightarrow{\beta} \langle b',\ \pi'\rangle$, representing that the binder b performs the communication action β ($\beta \neq \tau$) under the environment π and becomes b', turning the environment into π'. The intermediate binder $[c : \mathsf{some}(c')/x]$ is introduced (essentially extending the syntax for binders) to record the completion of the communication of some content c' over channel c, subsequently binding $\mathsf{some}(c')$ to the variable x.

A $[c : \mathsf{some}(c')/x]$ is produced after a transition made by either $t!t'\{x\}$ or $t?x$, given that the content of the output/input is c'. In the case of $t!t'\{x\}$, $\pi_\triangle c?x'$ represents that the environment is expecting some data to be output over channel c (from the process of $t!t'\{x\}$ where t is evaluated to c), and the resulting $\pi.c?c'_\triangle$ represents the completion of this interaction, extending the environment's observational history by $c?c'$. In the case of $t?x$, $\pi_\triangle c!c'$ represents that the environment attempts to output c' over channel c (to the process of $t?x$ where t is evaluated to c), and the resulting $\pi.c!c'_\triangle$ represents the completion of this interaction, growing the environment's observational history by $c!c'$. The transitions of composite binders $\&_q(b_1,...,b_n)$ are simply built on top of those of their sub-binders.

Table 2. The transition relation for processes and binders

$$\frac{\langle b,\ \pi\rangle \xrightarrow{c!c'/c?c'} \langle b',\ \pi'\rangle}{\delta \vdash \langle b.P,\ \pi\rangle \xrightarrow{c!c'/c?c'} \langle P',\ \pi'\rangle} \text{ where } P' = \begin{cases} P\theta & (\text{if } b'::_{\mathtt{tt}}\theta) \\ b'.P & (\text{otherwise}) \end{cases}$$

$$\frac{\delta \vdash \langle P_1,\ {}_\triangle c?x\rangle \xrightarrow{c!c'} \langle P_1', \pi_1'\rangle \quad \delta \vdash \langle P_2,\ {}_\triangle c!c'\rangle \xrightarrow{c?c'} \langle P_2',\ \pi_2'\rangle}{\delta \vdash \langle P_1|P_2,\ \pi\rangle \xrightarrow{\tau} \langle P_1'|P_2',\ \pi\rangle}$$

$$\frac{e \rhd \mathsf{some}(c) \quad \delta \vdash \langle P_1[c/y],\ \pi\rangle \xrightarrow{\beta} \langle P',\ \pi'\rangle}{\delta \vdash \langle CS(e, y, P_1, P_2), \pi\rangle \xrightarrow{\beta} \langle P',\ \pi'\rangle} \qquad \frac{e \rhd \mathsf{none} \quad \delta \vdash \langle P_2,\ \pi\rangle \xrightarrow{\beta} \langle P',\ \pi'\rangle}{\delta \vdash \langle CS(e, y, P_1, P_2), \pi\rangle \xrightarrow{\beta} \langle P',\ \pi'\rangle}$$

$$\frac{\bar{e} \rhd \bar{w} \quad \delta \vdash \langle P[\bar{w}/\bar{x}],\ \pi\rangle \xrightarrow{\beta} \langle P',\ \pi'\rangle}{\delta \vdash \langle \mathsf{A}(\bar{e}),\ \pi\rangle \xrightarrow{\beta} \langle P',\ \pi'\rangle} \text{ if } A(\bar{x}) \triangleq P \qquad \frac{\delta \vdash \langle P_1,\ \pi\rangle \xrightarrow{\beta} \langle P_2,\ \pi'\rangle}{\delta \vdash \langle P_1|P,\ \pi\rangle \xrightarrow{\beta} \langle P_2|P,\ \pi'\rangle}$$

$$\frac{\delta \vdash \langle P,\ \pi\rangle \xrightarrow{\beta} \langle P',\ \pi'\rangle}{\delta \vdash \langle (\nu c)P,\ \pi\rangle \xrightarrow{\beta} \langle (\nu c)P',\ \pi'\rangle} \text{ if } c \notin \mathsf{Ch}(\beta) \qquad \frac{P_1 \equiv P_2 \quad \delta \vdash \langle P_2,\ \pi\rangle \xrightarrow{\beta} \langle P_3,\ \pi'\rangle \quad P_3 \equiv P_4}{\delta \vdash \langle P_1,\ \pi\rangle \xrightarrow{\beta} \langle P_4,\ \pi'\rangle}$$

$$\frac{\neg(\exists c, c', \beta, P', \pi'' : (\beta = c!c' \vee \beta = c?c') \ \wedge\ \delta \vdash \langle P,\ \pi_\triangle \alpha.\pi'\rangle \xrightarrow{\beta} \langle P',\ \pi''\rangle)}{\delta \vdash \langle P,\ \pi_\triangle \alpha.\pi'\rangle \xrightarrow{\Box} \langle P,\ \pi.\Box_\triangle \pi'\rangle}$$

$$\delta \vdash \langle P,\ \pi\rangle \xrightarrow{\mathrm{env}} \langle P,\ \pi.\alpha\rangle \ \text{ if } \pi = [\pi]_\triangle \wedge \alpha \in \delta(\pi)$$

$$\frac{t \rhd c \quad t' \rhd c'}{\langle t!t'\{x\}, \pi_\triangle c?x'\rangle \xrightarrow{c!c'} \langle [c : \mathsf{some}(c')/x], \pi.c?c'_\triangle\rangle} \qquad t!t'\{x\} ::_{\mathbf{ff}} [\mathsf{none}/x] \quad t?x ::_{\mathbf{ff}} [\mathsf{none}/x]$$

$$\frac{t \rhd c}{\langle t?x, \pi_\triangle c!c'\rangle \xrightarrow{c?c'} \langle [c : \mathsf{some}(c')/x], \pi.c!c'_\triangle\rangle} \qquad [c : \mathsf{some}(c')/x] ::_{\mathbf{tt}} [\mathsf{some}(c')/x]$$

$$\frac{\langle b_j,\ \pi\rangle \xrightarrow{\beta} \langle b_j',\ \pi'\rangle}{\langle \&_q(..., b_j, ...),\ \pi\rangle \xrightarrow{\beta} \langle \&_q(..., b_j', ...),\ \pi'\rangle} \qquad \frac{\forall i : b_i ::_{v_i} \theta_i \quad v' = [\![q]\!](\bar{v})}{\&_q(b_1, ..., b_n) ::_{v'} \theta_n \ldots \theta_1}$$

The last couple of rules in Table 2 define the *evaluation* $b ::_v \theta$ of binders b, which is used by the transition rules for processes. Here θ is the substitution produced, recording the optional data bound to variables, and v is a boolean value indicating whether the binder b can already be passed. The basic binders $t!t'\{x\}$ and $t?x$ represent pending (incomplete) communications and hence for the evaluation of both binders, $v = \mathbf{ff}$, and the resulting substitution is $[\mathsf{none}/x]$, representing that no data is received into the variable x. For composite binders $\&_q(...)$, the last evaluation rule in Table 2 uses $[\![q]\!]$, the interpretation of the predicate q, to aggregate the evaluation statuses of the individual sub-binders. As examples, we have $[\![2]\!](v_1, v_2) = v_2$, and $[\![1 \vee 2]\!](v_1, v_2) = v_1 \vee v_2$. On the other hand, the resulting substitution is the composition of all the substitutions resulting from the evaluation of the sub-binders.

For processes, each transition rule of the form $\delta \vdash \langle P,\ \pi\rangle \xrightarrow{\beta} \langle P',\ \pi'\rangle$ governs the transition of process P under environment π into process P', turning the environment into π'. On the other hand, each transition of the form $\delta \vdash \langle P,\ \pi\rangle \xrightarrow{\text{env}} \langle P,\ \pi'\rangle$ represents the advancement of the environment alone. This transition relation is defined assuming a standard structural congruence $\equiv$ (given in detail in Table 3 of the appendix).

For output and input, we start from a process of the form $b.P$. Suppose the binder b makes a transition to the binder b'. In case b' has the evaluation $b' ::_{\mathtt{tt}} \theta$, the execution will embark on the process $P\theta$ — the process P with the substitution θ applied to it. In case b' has the evaluation $b' ::_{\mathtt{ff}} \theta$, we stay with the process $b'.P$, waiting for further communication required by the binder b' before it can be passed. The communication action (either an input or an output) performed by the process $b.P$ is the one performed by b. A synchronization between two processes does not rely on the environment π, and has no impact on it.

The next two rules use the abbreviation $CS(e, y, P_1, P_2)$ for the process $\mathsf{case}\ e\ \mathsf{of}\ \mathsf{some}(y) : P_1\ \mathsf{else}\ P_2$, and describe the execution of the case construct. In case the expression e is evaluated to $\mathsf{some}(c)$, where c is a constant, then the process $P_1[c/y]$ is executed, where the substitution records the binding of y to c. In case e is evaluated to none, then the process P_2 is executed, with no reference to y.

The rules for procedure calls, for parallel composition, for restriction, and for dealing with processes equivalent under $\equiv$, are self-explanatory. Notation-wise, $\mathsf{Ch}(\beta)$ represents the set of channels occurring in β.

The second last transition rule for processes says that when a process P cannot perform any communication action when the environment attempts to use the abstract binder α for the next interaction, we allow P to do a $\square$-step, signaling that there is one step of delay. At the same time, the observational history of the environment is extended by a $\square$, recording the observation of this delay.

The last transition rule says that the environment can make its next interaction attempt when its observational history ends with a $\vartriangle$: it can only "prescribe" the most imminent interaction, without further predication of the future.

We illustrate the semantics in Example 1, where $\delta_{\mathrm{ALL}} = \lambda\pi_{\vartriangle}.\{c!c' | c, c' \in \mathbf{Chn} \cup \mathbf{Dt}\} \cup \{c?x | c \in \mathbf{Chn}\} \cup \{\square\}$ is the strategy that allows the environment to produce any sensible abstract binder with any observation it has.

Example 1. The procedure call M in Fig. 1 has the following transition sequence.

$$
\begin{aligned}
\delta_{\mathrm{ALL}} \vdash \langle M,\ {\vartriangle}\rangle &\xrightarrow{\text{env}} \langle M,\ {\vartriangle} c_{LL}!d\rangle\\
&\xrightarrow{c_{LL}?d} \left\langle \begin{matrix}\mathsf{case\ none\ of\ some}(y_1) : c'_{LH}!y_1.M\\ \mathsf{else\ case\ some}(d)\ \mathsf{of\ some}(y_2) : c''_{LL}!y_2.M\ \mathsf{else}\ 0\end{matrix},\ c_{LL}!d_{\vartriangle}\right\rangle\\
&\xrightarrow{\text{env}} \left\langle \begin{matrix}\mathsf{case\ none\ of\ some}(y_1) : c'_{LH}!y_1.M\\ \mathsf{else\ case\ some}(d)\ \mathsf{of\ some}(y_2) : c''_{LL}!y_2.M\ \mathsf{else}\ 0\end{matrix},\ c_{LL}!d_{\vartriangle} c''_{LL}?x\right\rangle\\
&\xrightarrow{c''_{LL}!d} \langle M,\ c_{LL}!d.c''_{LL}?d_{\vartriangle}\rangle
\end{aligned}
$$

We elaborate slightly on the second step above. According to the transition rules for binders, we have $\langle c_{LL}?x_2,\ {\vartriangle} c_{LL}!d\rangle \xrightarrow{c_{LL}?d} \langle [c_{LL} : \mathsf{some}(d)/x_2],\ c_{LL}!d_{\vartriangle}\rangle$, which gives rise to $\langle \&_{1\vee 2}(c_{LH}?x_1, c_{LL}?x_2),\ {\vartriangle} c_{LL}!d\rangle \xrightarrow{c_{LL}?d} \langle \&_{1\vee 2}(c_{LH}?x_1, [c_{LL} : \mathsf{some}(d)/x_2]),\ c_{LL}!d_{\vartriangle}\rangle$. Using the evaluation rules for binders, we have $\&_{1\vee 2}(c_{LH}?x_1, [c_{LL} : \mathsf{some}(d)/x_2]) ::_{\mathtt{tt}} [\mathsf{none}/x_1][\mathsf{some}(d)/x_2]$. Hence by the transition rule for $b.P$, with b taken to be $\&_{1\vee 2}(c_{LH}?x_1, [c_{LL} : \mathsf{some}(d)/x_2])$, the second transition is derived. $\square$

Hereafter, we will use the more compact $\delta \vdash \langle P,\ \pi\rangle \xrightarrow{\text{env},\beta} \langle P',\ \pi'\rangle$ to represent $\exists \pi_0 : \delta \vdash \langle P,\ \pi\rangle \xrightarrow{\text{env}} \langle P,\ \pi_0\rangle \wedge \delta \vdash \langle P,\ \pi_0\rangle \xrightarrow{\beta} \langle P',\ \pi'\rangle$. We will also use $\mathsf{rch}(P)$ to represent the channel, polarity pairs of all possible communications that can be performed by a derivative of $\langle P,\ \vartriangle\rangle$ under the strategy δ_{ALL}, i.e., $\mathsf{rch}(P) = \{(c,\rho) \mid \exists P', \pi', c' : \delta_{\text{ALL}} \vdash \langle P,\ \vartriangle\rangle \rightarrow^{*} \langle P', \pi'\rangle \xrightarrow{c\rho c'}\}$.

4 Noninterference for Behavioral Integrity

In this section, we build up to our noninterference condition for behavioral integrity. We introduce the *classification mappings* $\mathcal{P}$ and $\mathcal{C}$ to keep track of the presence levels and content levels, respectively, for communication channels. In our definitions and propositions, we tacitly assume that all variables not explicitly quantified are in fact *universally* quantified.

We start by introducing a way of indexing into traces: $\pi @_i^{c,\rho}$ is (n, c') if the i-th communication over channel c with polarity ρ in π is the n-th communication overall in π, and the content of the communication is c'. All the indices start with 0. If the number of communications over c with polarity ρ in π is less than or equal to i, then $\pi @_i^{c,\rho}$ is $\perp$. This is formalized in Definition 1 and illustrated in Example 2.

Definition 1 **($\pi @_i^{c,\rho}$).** $\pi @_i^{c,\rho} = \pi @_{i,0}^{c,\rho}$*, where* $i \geq 0$ *and* $\pi @_{i,0}^{c,\rho}$ *is defined by*

$$\pi @_{i,n}^{c,\rho} = \begin{cases} \perp & (\textit{if } \pi = \epsilon) \\ (n, c') & (\textit{if } \exists \pi' : \pi = c\rho c'.\pi' \wedge i = 0) \\ \pi' @_{i-1,n+1}^{c,\rho} & (\textit{if } \exists c' : \pi = c\rho c'.\pi' \wedge i \neq 0) \\ \pi' @_{i,n+1}^{c,\rho} & (\textit{if } \exists c'', \rho'', c''' : \pi = c''\rho''c'''.\pi' \wedge (c'' \neq c \vee \rho'' \neq \rho)) \end{cases}$$

Example 2. Consider the trace $\pi = c_{LL}!d.c''_{LL}?d_{\vartriangle}$ left by the environment from Example 1. We have $\pi @_0^{c''_{LL},?} = (1, d)$ and $\pi @_i^{c''_{LL},?} = \perp$ whenever $i \geq 1$. □

We define the trace correspondence relation $W_{\mathcal{C}}^{\mathcal{P}}$ as follows, where $|\pi|$ stands for the length of π. The presence and content of communications in traces related by $W_{\mathcal{C}}^{\mathcal{P}}$ are supposed to reflect the integrity levels of their channels.

Definition 2 **($W_{\mathcal{C}}^{\mathcal{P}}$).** $\pi_1\ W_{\mathcal{C}}^{\mathcal{P}}\ \pi_2$ *if and only if* π_1 *and* π_2 *are finite,* $|\pi_1| = |\pi_2|$*, and* $\forall i \geq 0 : \forall c, \rho : \pi_1 @_i^{c,\rho}\ W_{\mathcal{C}(c)}^{\mathcal{P}(c)}\ \pi_2 @_i^{c,\rho}$*.*

In Definition 2, two traces related by $W_{\mathcal{C}}^{\mathcal{P}}$ are required to have the same finite length, and the i-th occurrences of communication over channel c, with polarity ρ, are required to be related by $W_{\mathcal{C}(c)}^{\mathcal{P}(c)}$, for each c and ρ. The latter relation is in turn defined as follows, where $c'_1 \doteq c'_2$ if and only if $c'_1 = c'_2$ or $c'_2 \notin \textbf{Chn} \cup \textbf{Dt}$.

Definition 3 **($W_{l_\mathcal{C}}^{l_\mathcal{P}}$).** $(n_1, c'_1)\ W_{l_\mathcal{C}}^{l_\mathcal{P}}\ (n_2, c'_2)$ *if and only if*

$$\begin{aligned} (n_1, c'_1)\ W_{l_\mathcal{C}}^{l_\mathcal{P}}\ (n_2, c'_2) &\quad \text{iff} \quad (l_\mathcal{P} = H \Rightarrow n_1 = n_2) \wedge (l_\mathcal{C} = H \Rightarrow c'_1 \doteq c'_2) \\ (n_1, c'_1)\ W_{l_\mathcal{C}}^{l_\mathcal{P}}\ \perp &\quad \text{iff} \quad l_\mathcal{P} = L \\ \perp\ W_{l_\mathcal{C}}^{l_\mathcal{P}}\ (n_2, c'_2) &\quad \text{iff} \quad l_\mathcal{P} = L \\ \perp\ W_{l_\mathcal{C}}^{l_\mathcal{P}}\ \perp &\quad \text{iff} \quad \mathsf{true} \end{aligned}$$

It can be seen that for a channel c with high presence integrity, the i-th occurrences of input/output over c need to have the same overall index in their respective traces. On the other hand, for a channel c with high content integrity, the i-th occurrences of input/output over c need to have equivalent content. This corresponds tightly to our description of "presence integrity" and "content integrity" in Sect. 2. The reason that $\doteq$ is used for relating content, instead of $=$, is the potential existence of variables in traces of the form $\dots_{\triangle} c?x$.

We will write $\overset{\widehat{\beta'}}{\Longrightarrow}$ for the weak transition used for standard observational equivalence, i.e., it stands for $(\xrightarrow{\tau})^* \circ \xrightarrow{\beta'} \circ (\xrightarrow{\tau})^*$ when $\beta' \neq \tau$ and for $(\xrightarrow{\tau})^*$ when $\beta' = \tau$. The weak transition will not be used directly in our noninterference property, but encapsulated within the transition $\underset{\beta}{\overset{\widehat{\beta'}}{\Longrightarrow}}$ introduced in Definition 4. It boils down to the weak transition $\overset{\widehat{\beta'}}{\Longrightarrow}$ in case β is a communication with high presence integrity; otherwise τ's are not allowed.

Definition 4 ($\underset{\beta}{\overset{\widehat{\beta'}}{\Longrightarrow}}$)**.** $\underset{\beta}{\overset{\widehat{\beta'}}{\Longrightarrow}} = \begin{cases} \xrightarrow{\beta'} & \text{(if } \beta = \square \vee \exists c, \rho, c' : \beta = c\rho c' \wedge \mathcal{P}(c) = L) \\ \overset{\widehat{\beta'}}{\Longrightarrow} & \text{(otherwise)} \end{cases}$

We then define the notion of δ-bisimulation, where $\delta \in \mathbf{Strat}$ is a strategy.

Definition 5 (**δ-Bisimulation**)**.** *A δ-bisimulation is a symmetric relation R on configurations such that if $\langle P_1, \pi_1\rangle\ R\ \langle P_2, \pi_2\rangle$, $\delta \vdash \langle P_1, \pi_1\rangle \xrightarrow{\text{env},\beta} \langle P_1', \pi_1'\rangle$, $\delta \vdash \langle P_2, \pi_2\rangle \xrightarrow{\text{env}} \langle P_2, \pi_{20}\rangle$, and $\pi_1'\ W_{\mathcal{C}}^{\mathcal{P}}\ \pi_{20}$, then we have*

$$\begin{aligned} \exists P_2', \pi_2', \beta' : \delta \vdash \langle P_2, \pi_{20}\rangle \underset{\beta}{\overset{\widehat{\beta'}}{\Longrightarrow}} \langle P_2', \pi_2'\rangle \ \wedge \\ [\pi_1']_{\triangle}\ W_{\mathcal{C}}^{\mathcal{P}}\ [\pi_2']_{\triangle} \ \wedge\ \langle P_1', [\pi_1']_{\triangle}\rangle\ R\ \langle P_2', [\pi_2']_{\triangle}\rangle. \end{aligned}$$

If two configurations $\langle P_1,\ \pi_1\rangle$ and $\langle P_2,\ \pi_2\rangle$ are related by a δ-bisimulation R, then after $\langle P_1,\ \pi_1\rangle$ interacts with the environment δ for one step, and the environment makes an interaction attempt with $\langle P_2,\ \pi_2\rangle$, such that the interaction and the attempt meet the integrity classes of their channels ($\pi_1'\ W_{\mathcal{C}}^{\mathcal{P}}\ \pi_{20}$), the configuration $\langle P_2,\ \pi_{20}\rangle$ can simulate the interaction made by $\langle P_1,\ \pi_1\rangle$, in a way that meets the integrity classes of the channels involved ($[\pi_1']_{\triangle}\ W_{\mathcal{C}}^{\mathcal{P}}\ [\pi_2']_{\triangle}$).

The definition of δ-Bisimulation introduces two universally quantified transitions, before simulating the first one with an existentially quantified transition. This pattern, previously adopted in [11], is rare in the literature.

We then define δ-bisimilarity ($\underset{\delta}{\sim}$) as the union of all δ-bisimulations (which is itself a δ-bisimulation). Note that δ-bisimilarity is *not* reflexive. In fact, our noninterference condition identifies the δ-security of a process P with the self-relatedness of $\langle P,\ \triangle\rangle$ in $\underset{\delta}{\sim}$, as stated in Definition 6.

Definition 6 (**δ-Security**)**.** *A process P is δ-secure, denoted by $\mathsf{Sec}_\delta(P)$, if and only if $\langle P,\ \triangle\rangle \underset{\delta}{\sim} \langle P,\ \triangle\rangle$.*

To arrive at a better understanding of δ-security, we introduce in Definition 7 the notion of *kernel* δ-bisimulation, which constrains the pairs of observational histories further than δ-bisimulation does. Proposition 1 then says that kernel δ-bisimulations, with a more complex formulation, can be used to characterize δ-security equally well.

Definition 7 (Kernel δ-Bisimulation). *A δ-bisimulation R is said to be a kernel δ-bisimulation if and only if $\langle P_1,\ \pi_1\rangle\ R\ \langle P_2,\ \pi_2\rangle$ implies knl(π_1,π_2), where knl(π_1,π_2) represents $\pi_1\ W^{\mathcal{P}}_{\mathcal{C}}\ \pi_2$, $[\pi_1]_\triangle = \pi_1$, and $[\pi_2]_\triangle = \pi_2$.*

Proposition 1. *There is a δ-bisimulation R such that $\langle P,\ \triangle\rangle\ R\ \langle P,\ \triangle\rangle$, if and only if there is a kernel δ-bisimulation R' such that $\langle P,\ \triangle\rangle\ R'\ \langle P,\ \triangle\rangle$.*

For a δ-secure process P, the implications of the existence of a kernel δ-bisimulation R such that $\langle P,\ \triangle\rangle\ R\ \langle P,\ \triangle\rangle$ are:

1. A communication β with high presence integrity needs to be simulated by a communication over the *same* channel, possibly together with τ's. In case the channel also has high content integrity, the content of the simulating communication should be the same as that of β. If the channel has low content integrity, on the other hand, then the bisimulation should continue under all contents possibly attempted by the environment, that are not necessarily the same as that of β.
2. A communication β with low presence integrity, or a $\square$-transition, is simulated by a communication over a channel also of low presence integrity, or by a $\square$-transition. If the channel of β, say c, has high content integrity, and it is being used for the i-th time with polarity ρ, then the content of β needs to agree with the content of the communication occurring on c with polarity ρ for the i-th time in the second execution, *in case that communication exists.* A similar requirement is imposed on the simulating communication, when its channel has high content integrity.
3. A τ can only be simulated by a (possibly empty) sequence of τ's. This is because when a τ-transition is made from a configuration $\langle P,\ \pi\rangle$, the $\triangle$ in π does not move, which is not the case otherwise. By Proposition 1, it is obvious that $|\pi'_1| = |\pi_{20}|$. Hence $[\pi'_1]_\triangle$ and $[\pi'_2]_\triangle$ will not have the same length and $[\pi'_1]_\triangle\ W^{\mathcal{P}}_{\mathcal{C}}\ [\pi'_2]_\triangle$ will not hold, if a τ is not simulated only by τ's.

The δ_{ALL}-security/insecurity of processes 1-6 in Fig. 1 of Sect. 2 agrees with the claims based on intuition in the same section, with an unconstrained environment. And process 7 is δ_{ALT}-secure, where δ_{ALT} characterizes an environment that provides content over at least one of c_{LL} and c'_{LL} whenever the process is ready for input from these two alternative channels:

$$\delta_{\mathrm{ALT}}(\pi) = \begin{cases} \{c_1!d \mid c_1 \in \{c_{LL}, c'_{LL}\} \wedge d \in \mathbf{Dt}\} & (\text{if } \pi = \triangle \vee \exists \pi', d' : \pi = \pi'.c_{HL}?d'_\triangle) \\ AB & (\text{otherwise}) \end{cases}$$

The construction of the underlying kernel δ_{ALL}-bisimulation $R_\star$ for the δ_{ALL}-security of process 6 is given in the appendix. We demonstrate in Example 3 that some of the requirements of $\langle M,\ \triangle\rangle\ R_\star\ \langle M,\ \triangle\rangle$ are fulfilled, to aid in the reader's intuition.

$$\begin{tikzcd}
\langle M,\ \triangle\rangle \arrow[dd, "{\text{env},\ c_{LH}?d_1}", "1"'] & R_\star & \langle M,\ \triangle\rangle \arrow[d, "2"', "\text{env}"] \\
 & & \langle M,\ \triangle c_{LL}!d_2\rangle \arrow[d, "3"', "c_{LL}?d_2"] \\
\langle c'_{LH}!d_1.M,\ c_{LH}!d_1\triangle\rangle \arrow[dd, "{\text{env},\ c'_{LH}!d_1}", "4"'] & R_\star & \langle c''_{LL}!d_2.M,\ c_{LL}!d_2\triangle\rangle \arrow[d, "5"', "\text{env}"] \\
 & & \langle c''_{LL}!d_2.M,\ c_{LL}!d_2\triangle c''_{LL}?x\rangle \arrow[d, "6"', "c''_{LL}!d_2"] \\
\langle M,\ c_{LH}!d_1.c'_{LH}?d_1\triangle\rangle \arrow[dd, "{\text{env},\ \Box}", "7"'] & R_\star & \langle M,\ c_{LL}!d_2.c''_{LL}?d_2\triangle\rangle \arrow[d, "8"', "\text{env}"] \\
 & & \langle M,\ c_{LL}!d_2.c''_{LL}?d_2\triangle c_{LH}!d_1\rangle \arrow[d, "9"', "c_{LH}?d_1"] \\
\langle M,\ c_{LH}!d_1.c'_{LH}?d_1.\Box\triangle\rangle \arrow[dd, "{\text{env},\ c_{LL}?d'_1}", "10"'] & R_\star & \langle c'_{LH}!d_1.M,\ c_{LL}!d_2.c''_{LL}?d_2.c_{LH}!d_1\triangle\rangle \arrow[d, "11"', "\text{env}"] \\
 & & \langle c'_{LH}!d_1.M,\ c_{LL}!d_2.c''_{LL}?d_2.c_{LH}!d_1\triangle c'_{LH}?x\rangle \arrow[d, "12"', "c'_{LH}!d_1"] \\
\langle c''_{LL}!d'_1.M,\ c_{LH}!d_1.c'_{LH}?d_1.\Box.c_{LL}!d'_1\triangle\rangle & R_\star & \langle M,\ c_{LL}!d_2.c''_{LL}?d_2.c_{LH}!d_1.c'_{LH}?d_1\triangle\rangle
\end{tikzcd}$$

Fig. 2. Partial unfolding of the kernel bisimulation containing $(\langle M,\ \triangle\rangle, \langle M,\ \triangle\rangle)$

Example 3. Figure 2 contains a partial unfolding of $\langle M,\ \triangle\rangle\ R_\star\ \langle M,\ \triangle\rangle$ where $R_\star$ is a kernel δ_{ALL}-bisimulation. For each pair $\langle P_1,\ \pi_1\rangle$ and $\langle P_2,\ \pi_2\rangle$ related by $R_\star$ in Fig. 2, $\pi_1\ W^{\mathcal{P}}_{\mathcal{C}}\ \pi_2$ holds. After transitions 1 and 2, the environment has made the attempt to interact with the process on two different channels c_{LH} and c_{LL}. This is allowed since $c_{LH}!d_{1\triangle}\ W^{\mathcal{P}}_{\mathcal{C}}\ {}_\triangle c_{LL}!d_2$ holds. The process M can indeed perform an input over c_{LH}, resulting in transition 1. This transition needs to be simulated by either an input over c_{LL}, or a $\Box$-transition in case such an input cannot be performed. We are in the former situation and transition 1 is thus simulated by transition 3. Note that according to Definition 4, the simulation of low presence communications should be done without using τ's. This is because such simulation is actually used to introduce interference, rather than to demonstrate resilience to it. And τ-transitions are conventionally used to weaken the requirement for a process to be resilient to interference. We then direct our attention to transitions 7, 8 and 9. The environment intentionally resists communication with the process in transition 7. On the other hand, it attempts to feed some content over c_{LH} to the process through transition 8. That content is restricted to d_1 since only then it holds that $c_{LH}!d_1.c'_{LH}?d_1.\Box_\triangle\ W^{\mathcal{P}}_{\mathcal{C}}\ c_{LL}!d_2.c''_{LL}?d_{2\triangle}c_{LH}!d_1$. Intuitively, the input over c_{LH} is blocked for a while in the second execution, but it needs to happen with the same content d_1 since the channel has high content integrity. For transitions 10, 11 and 12, the attempt of the environment to input from the process over channel c'_{LH} in transition 11 is satisfied with the content d_1, resulting in transition 12. The latter transition is a legitimate simulation of transition 10 since the content d_1 is the same as that of transition 4 — the corresponding communication over c'_{LH} in the first execution. $\Box$

A total order can be built on the set **Strat** of environment strategies, characterizing their relative aggressiveness (Definition 8), which has its impact on the strength of the security condition (Theorem 1).

Definition 8 (Aggressiveness of Environments). *Environment δ_2 is said to be more aggressive than δ_1, denoted $\delta_1 \leq \delta_2$, if $\forall \pi \in \Pi_\triangle : \delta_1(\pi) \subseteq \delta_2(\pi)$.*

Theorem 1 (Monotonicity). *δ-bisimilarity is anti-monotonic in δ, i.e., for all δ_1, δ_2 such that $\delta_1 \leq \delta_2$, it holds that $\underset{\delta_2}{\sim} \subseteq \underset{\delta_1}{\sim}$.*

This monotonicity result may look counter-intuitive since δ appears to be used both positively and negatively in Definition 5. However, $\delta \vdash \langle P_2, \pi_{20}\rangle \overset{\widehat{\beta'}}{\underset{\beta}{\Longrightarrow}} \langle P_2', \pi_2'\rangle$ if and only if $\delta' \vdash \langle P_2, \pi_{20}\rangle \overset{\widehat{\beta'}}{\underset{\beta}{\Longrightarrow}} \langle P_2', \pi_2'\rangle$ for all $\delta' \in \mathbf{Strat}$. In other words, δ is not actually used in the derivation of the transition sequence from $\langle P_2,\ \pi_{20}\rangle$.

Corollary 1. *The permissiveness of δ-security is anti-monotonic in δ, i.e.,*

$$\forall \delta_1, \delta_2 \in \mathbf{Strat} : \delta_1 \leq \delta_2 \wedge \mathsf{Sec}_{\delta_2}(P) \ \Rightarrow\ \mathsf{Sec}_{\delta_1}(P).$$

We will discuss deeper theoretical properties of our security condition in Sect. 5, focusing on δ_{ALL}-security. It will be seen that the most pessimistic assumption about the environment, captured by its most aggressive strategy δ_{ALL}, is in line with classical process-algebraic conditions like SBNDC, and also facilitates the compositional verification of the security property.

5 Theoretical Properties

Connection with SBNDC. We reformulate SBNDC [6] using the classical semantics of the Quality Calculus, and with respect to the environment $\mathcal{I}$: $\mathbf{Chan} \to \{H, L\} \cup \{\bot\}$ that gives the presence level of a channel only when its presence level and content level are the same, i.e., $\mathcal{I}(c) = \begin{cases} \mathcal{P}(c) & (\text{if } \mathcal{P}(c) = \mathcal{C}(c)) \\ \bot & (\text{otherwise}) \end{cases}$. The aim of $\mathcal{I}$ is to obtain the integrity class of each channel when its presence integrity and content integrity are the same (using $\mathcal{C}(c)$ instead of $\mathcal{P}(c)$ in the definition of $\mathcal{I}(c)$ would have the same effect).

We also introduce the notation $\overline{loi}$ to represent the list of low integrity channels, i.e., $\overline{loi} = \{c \mid \mathcal{I}(c) = L\}$, and use $\mathsf{ch}(\beta)$ to denote the channel used by the (non-τ) communication action β. The reformulation is then given in Definition 9, where $\approx$ is the standard observational equivalence. The intuitive interpretation is that before and after each low integrity communication, a process is required to have the same high integrity behaviors. Then the central result of this subsection, that δ_{ALL}-security coincides with SBNDC when the same integrity levels are always used for both "presence" and "content", is given in Theorem 2.

Definition 9 (SBNDC). *$P \in SBNDC$ if for all P', P'', communication β, such that $P \to^* P'$, $P' \xrightarrow{\beta} P''$, and $\mathcal{I}(\mathsf{ch}(\beta)) = L$, we have $(\nu\, \overline{loi})P' \approx (\nu\, \overline{loi})P''$.*

Theorem 2 (Degeneration). *For all processes P, if $\forall c, \rho$ s.t. $(c, \rho) \in \mathsf{rch}(P)$: $\mathcal{P}(c) = \mathcal{C}(c)$, then $\mathsf{Sec}_{\delta_{\mathrm{ALL}}}(P)$ if and only if $P \in SBNDC$.*

To build up to a proof of Theorem 2, we recast SBNDC in the form of *self-bisimilarity*. The underlying bisimulation is the $\triangleright$-bisimulation of Definition 10.

Definition 10 ($\triangleright$-Bisimulation). *A symmetric relation R on processes qualifies as a $\triangleright$-bisimulation if $P_1\ R\ P_2$ implies:*
for all P_1' and β such that $P_1 \xrightarrow{\beta} P_1'$, there exists P_2' such that

- *if $\mathcal{I}(\mathsf{ch}(\beta)) = L$, then $P_2' \equiv P_2$ and $P_1'\ R\ P_2'$,*
- *if $\mathcal{I}(\mathsf{ch}(\beta)) = H$ or $\beta = \tau$, then $P_2 \overset{\widehat{\beta}}{\Longrightarrow} P_2'$ and $P_1'\ R\ P_2'$.*

We define $\triangleright$-bisimilarity as the union of all $\triangleright$-bisimulations. The $\triangleright$ used here symbolizes the triangular structure created by the simulation of a low integrity communication by *inaction*, as required in Definition 10. It can be shown that self-$\triangleright$-bisimilarity coincides with self-δ_{ALL}-bisimilarity when the presence levels and content levels are the same for all channels whose uses are reachable.

Lemma 1. *Suppose P is such that $\forall c, \rho$ s.t. $(c, \rho) \in \mathsf{rch}(P) : \mathcal{P}(c) = \mathcal{C}(c)$. Then $P \in SBNDC \iff P \sim_{\triangleright} P$, and $P \sim_{\triangleright} P \iff \langle P,\ \vartriangle\rangle \underset{\delta_{\mathrm{ALL}}}{\sim} \langle P,\ \vartriangle\rangle$.*

The degeneration result presented above demonstrates that the notion of δ-security is in fact well-based on the classical process-algebraic noninterference properties, and SBNDC, as one of those properties, actually has the implicit assumption of the most aggressive environment.

Compositionality. Compositionality is a desirable property for the verification of noninterference properties. The security of a parallel composition can be directly obtained from that of its constituents, in case full compositionality is enjoyed by a noninterference condition. However, δ-security is not fully compositional. Nevertheless, this is key to spotting the insecurity of the example process 4 given in Sect. 2, since the processes $c_{LH}?x_1.c'_{LH}!x_1$ and $c'_{LH}!d$ are themselves δ_{ALL}-secure. We then discuss the *sufficient conditions* required for δ_{ALL}-security to be compositional.

A process P is *deterministic* with respect to output over a channel c, denoted by $\mathsf{det}(P, c)$, if

$$\delta_{\mathrm{ALL}} \vdash \langle P,\ \vartriangle\rangle \to^* \langle P',\ \pi'\rangle \wedge (\forall i \in \{1,2\} : \delta_{\mathrm{ALL}} \vdash \langle P',\ \pi'\rangle \xrightarrow{c!c_i'} \langle P_i',\ \pi_i'\rangle) \Rightarrow c_1' = c_2'.$$

We then have the following theorem for the compositionality of δ_{ALL}-security.

Theorem 3 (Compositionality). *If $\mathsf{Sec}_{\delta_{\mathrm{ALL}}}(P_1)$, and $\mathsf{Sec}_{\delta_{\mathrm{ALL}}}(P_2)$, then we have $\mathsf{Sec}_{\delta_{\mathrm{ALL}}}((\nu\bar{c'})(P_1|P_2))$, provided that for all $i \in \{1,2\}$ and channel c:*

$$\mathcal{P}(c) \not\sqsupseteq \mathcal{C}(c) \wedge (c, \rho_1) \in \mathsf{rch}(P_i) \wedge (c, \rho_2) \in \mathsf{rch}(P_{3-i}) \Rightarrow \rho_1 \neq \rho_2 \wedge \mathsf{det}(P_i, c) \wedge c \in \{\bar{c'}\}.$$

In words, Theorem 3 says that given two processes P_1 and P_2 that are both δ_{ALL}-secure, the process $(\nu\bar{c'})(P_1|P_2)$ is δ_{ALL}-secure, provided that

1. No LH-channels are used by both P_1 and P_2 with the same polarity (note that the process 4 given in Sect. 2 does not meet this requirement), and

2. For each LH-channel c used by P_1 and P_2 with different polarities, P_1 and P_2 must be deterministic with respect to output on c, and c must be among the constants over which there is a top-level restriction; thus the input side always sources from the output side, never from the environment.

Corollary 2. *Suppose* $\forall c, \rho : (c, \rho) \in \mathsf{rch}(P) \Rightarrow \mathcal{P}(c) \sqsupseteq \mathcal{C}(c)$*. Then* $\mathsf{Sec}_{\delta_{\mathrm{ALL}}}(P_1|P_2)$ *can be deduced from* $\mathsf{Sec}_{\delta_{\mathrm{ALL}}}(P_1)$ *and* $\mathsf{Sec}_{\delta_{\mathrm{ALL}}}(P_2)$*.*

The results presented above help elucidate the points below.

1. If δ-security had been fully compositional, it would not have uncovered certain insecure dependencies of high integrity content on low integrity presence.
2. The notion of δ-security is fully compositional for processes that do not make use of LH-channels.

6 Further Examples and Discussion

We have associated with LH-channels the meaning: communications over these channels can be blocked by the attacker, but with uninfluenced contents when they finally happen. So far the abstract environment has been assumed to be able to induce these channels. In this section, we present a concrete process in the Quality Calculus that can accomplish the same task. We then make the multiplexer process presented in Sect. 2 obtain its input from this process, to illustrate our compositionality result.

On LH-Channels. We illustrate that LH-channels can be induced from channels that are LL and HH by a concrete process. The procedure $SINK$ in Fig. 3 mimics the potential congestion of the high integrity data source c_{HH} using a queue: output of the oldest element suspended in the queue is attempted through the sink channel c_{LH} only when the low integrity switch c_{LL} is on. Recall that the $\&_2(\_,\_)$ can be passed if and only if the second communication is successful.

```
$SINK \triangleq \&_2(c_{LL}?x_1, c_{HH}?x_2).$
          case $x_2$ of some$(y_2)$ :
          $(\nu c_{\mathrm{f}})(c_{\mathrm{i}}!(y_2, c_{\mathrm{f}}).c_{\mathrm{f}}?x_{\mathrm{f}}.$
            case $x_1$ of some$(y_1)$ :
              $(\nu c_{\mathrm{e}}, c_{\mathrm{r}})(c_{\mathrm{p}}!(c_{\mathrm{e}}, c_{\mathrm{r}}).c_{\mathrm{r}}?x_3[y_3].$
                $\&_2(c_{LH}!y_3\{x_3'\}, c_{\mathrm{HH}}'?x_{\mathrm{t}}).$
                case $x_3'$ of some$(y_3')$ :
                  $(\nu c_{\mathrm{e}}', c_{\mathrm{r}}')(c_{\mathrm{d}}!(c_{\mathrm{e}}', c_{\mathrm{r}}').c_{\mathrm{r}}?x_4[y_4].SINK)$
                else $SINK)$
            else $c_{\mathrm{HH}}'?x_{\mathrm{t}}.SINK)$
          else $0$
```

Fig. 3. The "Realization" of sink channels with low presence integrity and high content integrity

The channels c_{i}, c_{d}, and c_{p} are interfaces for the operations "insert" ("enque"), "delete" ("deque"), and "peek" (the non-destructive inspection of the oldest element) of the queue specified by the procedure QUE (adapted from the priority queue in [17]) in the appendix. The procedure $SINK$ waits on the input over c_{HH} for the composite binder on the first line to be passed. When that happens, the input data over c_{HH} is enqued, with the completion of the "enque" operation signaled on c_f. If the input over c_{LL} was also successful, then outputting the head of the queue is attempted, with a high integrity timeout supposed to come over c'_{HH}. If the output is successful before the timeout, then the data item of the output is deleted from the queue. In the "peek" and "deque" operations, the channels c_e and c'_e are sent to the queue for the latter to signal back whether it is already an empty queue. In our case the non-emptiness of the queue is an invariant and hence neither c_e nor c'_e is subsequently used. The process $(\nu c_\mathrm{i}, c_\mathrm{d}, c_\mathrm{p})(\mathit{SINK} \mid \mathit{QUE}(\mathsf{some}(c_\mathrm{i}), \mathsf{some}(c_\mathrm{d}), \mathsf{some}(c_\mathrm{p})))$ is δ_ALL-secure.

Compositionality. We now consider making the multiplexer process (process 6 in Fig. 1) source from the channel c_{LH} in Fig. 3. Let $\mathit{SRC} \triangleq (\nu c_\mathrm{i}, c_\mathrm{d}, c_\mathrm{p})$ $(\mathit{SINK} \mid \mathit{QUE}(\mathsf{some}(c_\mathrm{i}), \mathsf{some}(c_\mathrm{d}), \mathsf{some}(c_\mathrm{p})))$. The process under consideration is (νc_{LH}) $(\mathit{SRC} | M)$. It is not difficult to see that $\mathsf{det}(\mathit{SRC}, c_{LH})$ and $\mathsf{det}(M, c_{LH})$ hold. Hence we can deduce the validity of $\mathsf{Sec}_{\delta_\mathrm{ALL}}((\nu c_{LH})(\mathit{SRC}|M))$ by Theorem 3 and the δ_ALL-security of SRC and M.

Confidentiality. We are in a position to further explain having developed our theory for integrity, rather than confidentiality. It has been illustrated by the example in Fig. 3 that a *concrete process* can influence the presence of communication over a sink channel of it, without influencing the communication content. For confidentiality, a channel $c_\star$ with high presence confidentiality and low content confidentiality would correspond to our channel with LH-integrity. Assuming the existence of $c_\star$ and developing the same theory would not be problematic. However, it is difficult to come up with a possibilistic process that leaks the content of $c_\star$ properly, without leaking the presence of communication over it, unless other channels also with confidential presence and public content are used. Hence the meaning of "confidential presence, public content" would be harder to justify as opposed to "low integrity presence, high integrity content".

7 Conclusion

We have studied the integrity of communication behaviors in process-algebraic systems from the viewpoint of information flow control. A fine-grained, bisimulation-based noninterference property is proposed: the *presence* and *content* of communications have separate integrity levels, and *all* combinations of integrity levels for both dimensions are allowed. When identical levels are always used for both dimensions, the property coincides with the classical process-algebraic property SBNDC (e.g., [6]), demonstrating faithful inheritance from known frameworks. A compositionality result is obtained, facilitating modular flow analysis of concurrent processes.

Our recasting of SBNDC as self-$\rhd$-bisimilarity may reflect the insights behind existing work [3] in bridging language-based and process-algebraic security, but may be the first direct reformulation of BNDC-like properties as self-bisimilarity. This gives another perspective on the secure semantics induced by SBNDC.

Clarkson et al. [4] dimensions *quantitative integrity* in terms of information *suppression* and *contamination*, where dissimilarity of integrity to confidentiality is also examined: information suppression has no confidentiality counterpart.

It would not be difficult to adapt δ-security to support the use of *downgrading* [16], which relaxes information flow constraints. This can be done along the directions of [1]. Another interesting line of future work is the design of information flow type systems supporting δ-security.

Acknowledgement. We would like to thank the ProSec research group at Chalmers University of Technology, especially Andrei Sabelfeld, Willard Rafnsson and David Sands, for useful feedback on this work at an earlier stage.

A Appendix

Structural Congruence. The structural congruence is the smallest congruence relation satisfying the rules in Table 3. In Table 3, the α-equivalence of two processes is denoted by $\equiv_\alpha$, and $fc(P)$ gives the set of free constants of the process P and can be defined in a straightforward manner.

Table 3. The structural congruence

$P_1 \| P_2 \equiv P_2 \| P_1$	$(\nu c_1)(\nu c_2)P \equiv$
$P_1 \| (P_2 \| P_3) \equiv (P_1 \| P_2) \| P_3$	$(\nu c_2)(\nu c_1)P$ (if $c_1 \neq c_2$)
$P \mid 0 \equiv P$	$(\nu c)(P_1 \| P_2) \equiv ((\nu c)P_1) \| P_2$
$(\nu c)P \equiv P$ (if $c \notin fc(P)$)	(if $c \notin fc(P_2)$)
$P \equiv_\alpha P' \Rightarrow P \equiv P'$	$P_1 \equiv P_2 \Rightarrow (\nu c)P_1 \equiv (\nu c)P_2$

Semantics Without Environment. The "classical" semantics [11] of the Quality Calculus is given in Table 4. The transitions made by processes are of the form $P \xrightarrow{\beta} P'$, where β is a communication action or a τ. The correspondence between the two semantics is given in Lemma 2, where $\mathsf{ch}(\beta) = \begin{cases} c \ \ (\text{if } \beta = c!c' \vee \beta = c?c') \\ \bot \ (\text{if } \beta = \tau \vee \beta = \square) \end{cases}$, $\rho(\beta) = \begin{cases} \rho_0 \ (\text{if} \exists c, c' : \beta = c\rho_0 c') \\ \bot \ (\text{otherwise}) \end{cases}$, $\widetilde{!} = ?$ and $\widetilde{?} = !$.

Lemma 2. *For all processes P, P', actions β_1, β_2, ..., and β_n such that there is at most one $i \in \{1, ..., n\}$ for which $\beta_i \neq \tau$, and $\forall i \in \{1, ..., n\} : \beta_i \neq \square$, histories π such that $\pi = [\pi]_\triangle$, and π_0 such that $\forall i \in \{1, ..., n\} : \beta_i \neq \tau \Rightarrow \pi_0 = \pi.\mathsf{ch}(\beta_i)\widetilde{\rho(\beta_i)}c'$ for some c', the following are equivalent:*

1. $P \overset{\beta_1 \ldots \beta_n}{\Longrightarrow} P'$, *and*
2. $\delta_{\text{ALL}} \vdash \langle P,\ \pi \rangle \xrightarrow{\text{env}} \langle P,\ \pi_0 \rangle \wedge \exists \pi_0' : \delta_{\text{ALL}} \vdash \langle P,\ \pi_0 \rangle \overset{\beta_1 \ldots \beta_n}{\Longrightarrow} \langle P',\ \pi_0' \rangle$.

Proof. Both directions can be shown by induction on the length of the corresponding sequences of semantic derivation. $\square$

Table 4. The transition relation for processes

$$\frac{b \xrightarrow{c!c'} b'}{b.P \xrightarrow{c!c'} P'} \text{ where } P' = \begin{cases} P\theta & (\text{if } b'{::}_{\mathrm{tt}}\theta) \\ b'.P & (\text{if } b'{::}_{\mathrm{ff}}\theta) \end{cases} \qquad \frac{b \xrightarrow{c?c'} b'}{b.P \xrightarrow{c?c'} P'} \text{ where } P' = \begin{cases} P\theta & (\text{if } b'{::}_{\mathrm{tt}}\theta) \\ b'.P & (\text{if } b'{::}_{\mathrm{ff}}\theta) \end{cases}$$

$$\frac{P_1 \xrightarrow{c!c'} P_1' \quad P_2 \xrightarrow{c?c'} P_2'}{P_1|P_2 \xrightarrow{\tau} P_1'|P_2'}$$

$$\frac{e \rhd \mathsf{some}(c) \quad P_1[c/y] \xrightarrow{\beta} P'}{\mathsf{case}\ e\ \mathsf{of}\ \mathsf{some}(y) : P_1\ \mathsf{else}\ P_2 \xrightarrow{\beta} P'} \qquad \frac{e \rhd \mathsf{none} \quad P_2 \xrightarrow{\beta} P'}{\mathsf{case}\ e\ \mathsf{of}\ \mathsf{some}(y) : P_1\ \mathsf{else}\ P_2 \xrightarrow{\beta} P'}$$

$$\frac{\bar{e} \rhd \bar{w} \quad P[\bar{w}/\bar{x}] \xrightarrow{\beta} P'}{\mathsf{A}(\bar{e}) \xrightarrow{\beta} P'} \text{ if } A(\bar{x}) \triangleq P \qquad \frac{P \xrightarrow{\beta} P'}{(\nu c)P \xrightarrow{\beta} (\nu c)P'} \text{ if } c \notin \mathsf{Ch}(\beta)$$

$$\frac{P_1 \xrightarrow{\beta} P_2}{P_1|P \xrightarrow{\beta} P_2|P} \qquad \frac{P_1 \equiv P_2 \quad P_2 \xrightarrow{\beta} P_3 \quad P_3 \equiv P_4}{P_1 \xrightarrow{\beta} P_4}$$

$$\frac{t \rhd c \quad t' \rhd c'}{t!t'\{x\} \xrightarrow{c!c'} [c : \mathsf{some}(c')/x]} \qquad t!t'\{x\}{::}_{\mathrm{ff}}\,[\mathsf{none}/x] \quad t?x{::}_{\mathrm{ff}}\,[\mathsf{none}/x]$$

$$\frac{t \rhd c}{t?x \xrightarrow{c?c'} [c : \mathsf{some}(c')/x]} \qquad [c : \mathsf{some}(c')/x]{::}_{\mathrm{tt}}\,[\mathsf{some}(c')/x]$$

$$\frac{b_j \xrightarrow{\beta} b_j'}{\&_q(\ldots, b_j, \ldots) \xrightarrow{\beta} \&_q(\ldots, b_j', \ldots)} \qquad \frac{\forall i : b_i {::}_{v_i} \theta_i \quad v' = \llbracket q \rrbracket(\bar{v})}{\&_q(b_1, \ldots, b_n) {::}_{v'} \theta_n \ldots \theta_1}$$

We introduce the notation $\pi \downarrow C$ for π ending with $\triangle$, and C a set of channels, to represent the order-preserving sequence of all communications on channels within C in π, and abbreviate $\pi \downarrow \{c\}$ as $\pi \downarrow c$ where c is a channel.

$\delta_{\mathbf{ALL}}$-Security of Process 6. We construct the binary relation R_{sym} that is the symmetric closure of the following relation R. Below, $\phi(\pi_1, \pi_2)$ stands for

$$\begin{aligned} &knl(\pi_1, \pi_2) \ \wedge \\ &\forall i \in \{1,2\} : \forall c_{\mathrm{a}}, c_{\mathrm{b}} : c_{LH}!c_{\mathrm{a}} \text{ is followed immediately by } c'_{LH}\rho c_{\mathrm{b}} \text{ in } \pi_i \downarrow \{c_{LH}, c'_{LH}\} \\ &\qquad\qquad \implies \rho = ? \wedge c_{\mathrm{a}} = c_{\mathrm{b}}. \end{aligned}$$

In addition, $CSs(e_1, e_2)$ stands for

$$\begin{aligned} &\mathsf{case}\ e_1\ \mathsf{of}\ \mathsf{some}(y_1) : c'_{LH}!y_1.M \\ &\mathsf{else}\ \mathsf{case}\ e_2\ \mathsf{of}\ \mathsf{some}(y_2) : c''_{LL}!y_2.M\ \mathsf{else}\ 0 \end{aligned}$$

$$\begin{aligned} R = {} & \{(\langle M,\ \pi_1\rangle, \langle M,\ \pi_2\rangle) \mid \phi(\pi_1,\pi_2)\} \cup \\ & \{(\langle CSs(\mathsf{some}(c'_{\mathrm{a}}), \mathsf{none}),\ \pi_1\rangle, \langle M,\ \pi_2\rangle) \mid \pi_1 = \ldots c_{LH}!c'_{\mathrm{a}\triangle} \wedge \phi(\pi_1,\pi_2)\} \cup \\ & \{(\langle CSs(\mathsf{none}, \mathsf{some}(c'_{\mathrm{a}})),\ \pi_1\rangle, \langle M,\ \pi_2\rangle) \mid \pi_1 = \ldots c_{LL}!c'_{\mathrm{a}\triangle} \wedge \phi(\pi_1,\pi_2)\} \cup \\ & \{(\langle CSs(\mathsf{some}(c'_{\mathrm{a}}), \mathsf{none}),\ \pi_1\rangle, \langle CSs(\mathsf{some}(c'_{\mathrm{b}}), \mathsf{none}),\ \pi_2\rangle) \mid \\ & \qquad \pi_1 = \ldots c_{LH}!c'_{\mathrm{a}\triangle} \wedge \pi_2 = \ldots c_{LH}!c'_{\mathrm{b}\triangle} \wedge \phi(\pi_1,\pi_2)\} \cup \\ & \{(\langle CSs(\mathsf{none}, \mathsf{some}(c'_{\mathrm{a}})),\ \pi_1\rangle, \langle CSs(\mathsf{none}, \mathsf{some}(c'_{\mathrm{b}})),\ \pi_2\rangle) \mid \\ & \qquad \pi_1 = \ldots c_{LL}!c'_{\mathrm{a}\triangle} \wedge \pi_2 = \ldots c_{LL}!c'_{\mathrm{b}\triangle} \wedge \phi(\pi_1,\pi_2)\} \cup \\ & \{(\langle CSs(\mathsf{some}(c'_{\mathrm{a}}), \mathsf{none}),\ \pi_1\rangle, \langle CSs(\mathsf{none}, \mathsf{some}(c'_{\mathrm{b}})),\ \pi_2\rangle) \mid \\ & \qquad \pi_1 = \ldots c_{LH}!c'_{\mathrm{a}\triangle} \wedge \pi_2 = \ldots c_{LL}!c'_{\mathrm{b}\triangle} \wedge \phi(\pi_1,\pi_2)\} \end{aligned}$$

It can be shown that R_{sym} is a δ_{ALL}-bisimulation relating $\langle M,\ \triangle\rangle$ to itself.

Queue Specification. We adapte the priority queue discussed in [17] to be a FIFO queue specified in Fig. 4. A peek operation that returns but does not remove the head of the queue is added.

$$
\begin{array}{l}
QUE(x_{\mathrm{i}}, x_{\mathrm{d}}, x_{\mathrm{p}}) \triangleq (\nu c_{\mathrm{g}})(E(x_{\mathrm{i}}, x_{\mathrm{d}}, x_{\mathrm{p}}, \mathsf{some}(c_{\mathrm{g}})) \mid G(\mathsf{some}(c_{\mathrm{g}})))\\[1em]
G(x_{\mathrm{g}}[c_{\mathrm{g}}]) \triangleq c_{\mathrm{g}}?(x_{\mathrm{i}}, x_{\mathrm{d}}, x_{\mathrm{p}}).E(x_{\mathrm{i}}, x_{\mathrm{d}}, x_{\mathrm{p}}, x_{\mathrm{g}}) \mid G(x_{\mathrm{g}})\\[1em]
E(x_{\mathrm{i}}[c_{\mathrm{i}}], x_{\mathrm{d}}[c_{\mathrm{d}}], x_{\mathrm{p}}[c_{\mathrm{p}}], x_{\mathrm{g}}[c_{\mathrm{g}}]) \triangleq\\
\quad c_{\mathrm{i}}?(x, x_{\mathrm{f}})[\_, c_{\mathrm{f}}].\\
\qquad (\nu c'_{\mathrm{i}}, c'_{\mathrm{d}}, c'_{\mathrm{p}})(c_{\mathrm{g}}!(c'_{\mathrm{i}}, c'_{\mathrm{d}}, c'_{\mathrm{p}}).c_{\mathrm{f}}!\checkmark.F(x_{\mathrm{i}}, x_{\mathrm{d}}, x_{\mathrm{p}}, x, \mathsf{some}(c'_{\mathrm{i}}), \mathsf{some}(c'_{\mathrm{d}}), \mathsf{some}(c'_{\mathrm{p}}), x_{\mathrm{g}}))\\
\quad + c_{\mathrm{d}}?(x_{\mathrm{e}}, x_{\mathrm{r}})[c_{\mathrm{e}}, \_].c_{\mathrm{e}}!\checkmark.E(x_{\mathrm{i}}, x_{\mathrm{d}}, x_{\mathrm{p}}, x_{\mathrm{g}})\\
\quad + c_{\mathrm{p}}?(x'_{\mathrm{e}}, x'_{\mathrm{r}})[c'_{\mathrm{e}}, \_].c'_{\mathrm{e}}!\checkmark.E(x_{\mathrm{i}}, x_{\mathrm{d}}, x_{\mathrm{p}}, x_{\mathrm{g}})\\[1em]
F(x_{\mathrm{i}}[c_{\mathrm{i}}], x_{\mathrm{d}}[c_{\mathrm{d}}], x_{\mathrm{p}}[c_{\mathrm{p}}], x_{\mathrm{k}}[c_{\mathrm{k}}], x'_{\mathrm{i}}[c'_{\mathrm{i}}], x'_{\mathrm{d}}[c'_{\mathrm{d}}], x'_{\mathrm{p}}[c'_{\mathrm{p}}], x_{\mathrm{g}}) \triangleq\\
\quad c_{\mathrm{i}}?(x, x_{\mathrm{f}})[y, c_{\mathrm{f}}].\\
\qquad (\nu c'_{\mathrm{f}})(c'_{\mathrm{i}}!(y, c'_{\mathrm{f}}) \mid c'_{\mathrm{f}}?x'.c_{\mathrm{f}}!\checkmark.F(x_{\mathrm{i}}, x_{\mathrm{d}}, x_{\mathrm{p}}, x_{\mathrm{k}}, x'_{\mathrm{i}}, x'_{\mathrm{d}}, x'_{\mathrm{p}}, x_{\mathrm{g}})) +\\
\quad c_{\mathrm{d}}?(x_{\mathrm{e}}, x_{\mathrm{r}})[\_, c_{\mathrm{r}}].\\
\qquad (\nu c'_{\mathrm{e}}, c'_{\mathrm{r}})(c'_{\mathrm{d}}!(c'_{\mathrm{e}}, c'_{\mathrm{r}}) \mid\\
\qquad (c'_{\mathrm{e}}?x''.c_{\mathrm{r}}!c_{\mathrm{k}}.E(x_{\mathrm{i}}, x_{\mathrm{d}}, x_{\mathrm{p}}, x_{\mathrm{g}}) + c'_{\mathrm{r}}?x'''.c_{\mathrm{r}}!c_{\mathrm{k}}.F(x_{\mathrm{i}}, x_{\mathrm{d}}, x_{\mathrm{p}}, x''', x'_{\mathrm{i}}, x'_{\mathrm{d}}, x'_{\mathrm{p}}, x_{\mathrm{g}}))) +\\
\quad c_{\mathrm{p}}?(x''_{\mathrm{e}}, x''_{\mathrm{r}})[\_, c''_{\mathrm{r}}].c''_{\mathrm{r}}!c_{\mathrm{k}}.F(x_{\mathrm{i}}, x_{\mathrm{d}}, x_{\mathrm{p}}, x_{\mathrm{k}}, x'_{\mathrm{i}}, x'_{\mathrm{d}}, x'_{\mathrm{p}}, x_{\mathrm{g}})
\end{array}
$$

Fig. 4. Specification of FIFO queue

Sketch of Proof for Compositionality (Theorem 3). Define $\widetilde{\pi}$ to be the order-preserving sequence of all actions in π with all the polarities ρ changed to $\tilde{\rho}$. For convenience we rename the P_1 and P_2 in the precondition of Theorem 3 into P_1° and P_2°, and the list $\bar{c'}$ into $\bar{c^{\circ}}$.

Construct the binary relation R as:

$$
\begin{aligned}
R = \{(\langle P_1,\ \pi_1\rangle, \langle P_2,\ \pi_2\rangle) \mid\ & \exists P_{11}, P_{12}, P_{21}, P_{22}, \pi_{11}, \pi_{12}, \pi_{21}, \pi_{22} :\\
& \psi(P_1, P_2, P_{11}, P_{12}, P_{21}, P_{22}, \pi_1, \pi_2, \pi_{11}, \pi_{12}, \pi_{21}, \pi_{22})\},
\end{aligned}
$$

where $\psi(P_1, P_2, P_{11}, P_{12}, P_{21}, P_{22}, \pi_1, \pi_2, \pi_{11}, \pi_{12}, \pi_{21}, \pi_{22})$ is the conjunction of the following clauses:

$$\forall i \in \{1, 2\} : P_i \equiv (\nu \bar{c^{\circ}})(P_{1i} | P_{2i}) \tag{1}$$

$$\forall j, i \in \{1, 2\} : \exists \pi' : \delta_{\mathrm{ALL}} \vdash \langle P_j^{\circ},\ \triangle\rangle \longrightarrow^{*} \langle P_{ji},\ \pi'\rangle \tag{2}$$

$$\forall j \in \{1, 2\} : \langle P_{j1},\ \pi_{j1}\rangle \underset{\delta_{\mathrm{ALL}}}{\sim} \langle P_{j2},\ \pi_{j2}\rangle \tag{3}$$

$$\mathit{knl}(\pi_1, \pi_2) \tag{4}$$

$$\forall j \in \{1, 2\} : \mathit{knl}(\pi_{j1}, \pi_{j2}) \tag{5}$$

$$
\begin{aligned}
&\forall c\ s.t.\ \mathcal{P}(c) = L \wedge \mathcal{C}(c) = H : \forall i \in \{1, 2\} : \\
&\quad ((\exists \rho : (c, \rho) \in \mathsf{rch}(P_1) \wedge (c, \tilde{\rho}) \notin \mathsf{rch}(P_2)) \Rightarrow \pi_i \downarrow c = \pi_{1i} \downarrow c \wedge \pi_{2i} \downarrow c = \epsilon) \wedge\\
&\quad ((\exists \rho : (c, \rho) \in \mathsf{rch}(P_2) \wedge (c, \tilde{\rho}) \notin \mathsf{rch}(P_1)) \Rightarrow \pi_i \downarrow c = \pi_{2i} \downarrow c \wedge \pi_{1i} \downarrow c = \epsilon) \wedge\\
&\quad ((\exists \rho : (c, \rho) \in \mathsf{rch}(P_1) \wedge (c, \tilde{\rho}) \in \mathsf{rch}(P_2)) \Rightarrow \pi_{2i} \downarrow c = \widetilde{\pi_{1i} \downarrow c} \wedge \pi_i \downarrow c = \epsilon)
\end{aligned}
\tag{6}
$$

We show that R qualifies as a δ_{ALL}-bisimulation. Then $\mathsf{Sec}_{\delta_{\text{ALL}}}((\nu\bar{c^\circ})(P_1^\circ|P_2^\circ))$ will follow, since it holds that

$$\psi((\nu\bar{c^\circ})(P_1^\circ|P_2^\circ),(\nu\bar{c^\circ})(P_1^\circ|P_2^\circ),P_1^\circ,P_1^\circ,P_2^\circ,P_2^\circ,\vartriangle,\vartriangle,\vartriangle,\vartriangle,\vartriangle,\vartriangle),$$

and we thus have $\langle(\nu\bar{c^\circ})(P_1^\circ|P_2^\circ),\ \vartriangle\rangle\ R\ \langle(\nu\bar{c^\circ})(P_1^\circ|P_2^\circ),\ \vartriangle\rangle$.

We omit further details. $\square$

References

1. Bossi, A., Piazza, C., Rossi, S.: Modelling downgrading in information flow security. In: 17th IEEE Computer Security Foundations Workshop, (CSFW-17 2004), 28–30 June 2004, Pacific Grove, CA, USA, p. 187 (2004)
2. Capecchi, S., Castellani, I., Dezani-Ciancaglini, M., Rezk, T.: Session types for access and information flow control. In: Gastin, P., Laroussinie, F. (eds.) CONCUR 2010. LNCS, vol. 6269, pp. 237–252. Springer, Heidelberg (2010)
3. Castellani, I.: State-oriented noninterference for CCS. Electron. Notes Theor. Comput. Sci. **194**(1), 39–60 (2007)
4. Clarkson, M.R., Schneider, F.B.: Quantification of integrity. In: Proceedings of the 23rd IEEE Computer Security Foundations Symposium, CSF (2010)
5. Cohen, E.S.: Information transmission in computational systems. In: SOSP, pp. 133–139 (1977)
6. Focardi, R., Gorrieri, R.: Classification of security properties. In: Focardi, R., Gorrieri, R. (eds.) FOSAD 2000. LNCS, vol. 2171, pp. 331–396. Springer, Heidelberg (2001)
7. Goguen, J.A, Meseguer, J.: Security policies and security models. In: IEEE Symposium on Security and Privacy, pp. 11–20 (1982)
8. Kobayashi, N.: Type-based information flow analysis for the pi-calculus. Acta Inf. **42**(4–5), 291–347 (2005)
9. Montagu, B., Pierce, B.C., Pollack, R.: A theory of information-flow labels. In: 2013 IEEE 26th Computer Security Foundations Symposium, New Orleans, LA, USA, 26–28 June 2013, pp. 3–17 (2013)
10. Muller, S., Chong, S.: Towards a practical secure concurrent language. In: Proceedings of the 27th Annual ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications, OOPSLA 2012, pp. 57–74 (2012)
11. Nielson, H.R., Nielson, F.: Safety versus security in the quality calculus. In: Liu, Z., Woodcock, J., Zhu, H. (eds.) Theories of Programming and Formal Methods. LNCS, vol. 8051, pp. 285–303. Springer, Heidelberg (2013)
12. Nielson, H.R., Nielson, F., Vigo, R.: A calculus for quality. In: Păsăreanu, C.S., Salaün, G. (eds.) FACS 2012. LNCS, vol. 7684, pp. 188–204. Springer, Heidelberg (2013)
13. Rafnsson, W., Hedin, D., Sabelfeld, A.: Securing interactive programs. In: 25th IEEE Computer Security Foundations Symposium, CSF 2012 (2012)
14. Sabelfeld, A., Mantel, H.: Static confidentiality enforcement for distributed programs. In: Hermenegildo, M.V., Puebla, G. (eds.) SAS 2002. LNCS, vol. 2477, pp. 376–394. Springer, Berlin Heidelberg (2002)
15. Sabelfeld, A., Myers, A.C.: Language-based information-flow security. IEEE J. Sel. Areas Commun. **21**(1), 5–19 (2003)

16. Sabelfeld, A., Sands, D.: Declassification: dimensions and principles. J. Comput. Secur. **17**(5), 517–548 (2009)
17. Sangiorgi, D., Walker, D.: The Pi-Calculus - A Theory of Mobile Processes. Cambridge University Press, UK (2001)
18. van Bakel, S., Vigliotti, M.G.: Note on a simple type system for non-interference. CoRR, abs/1109.4843 (2011)