A Process Algebra for (Delimited) Persistent Stochastic Non-Interference

Andrea Marin[1] Carla Piazza[2] Sabina Rossi[1]

[1] Università Ca' Foscari Venezia, Italy

[2] Università degli Studi di Udine, Italy

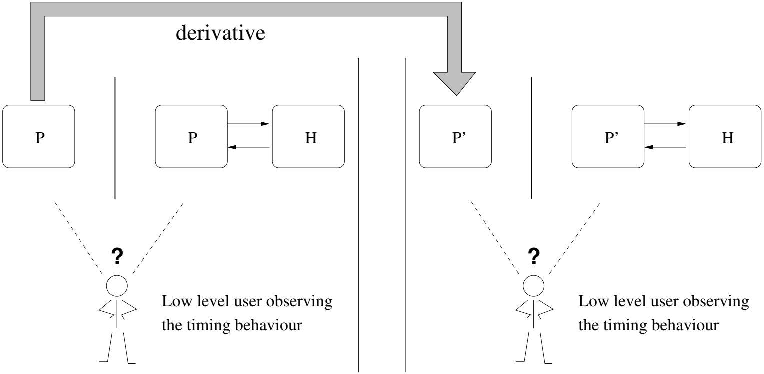
CILC 2019 - VALUETOOLS 2019 and QEST 2019

PERSISTENT STOCHASTIC NON-INTERFERENCE

THE CONTEXT

- Non-Interference aims at protecting sensitive data from undesired accesses
- Goguen-Meseguer'82: information does not flow from high (confindential) to low (public) if the high behavior cannot be observed at low level
- Few results deal with time behaviour and Non-Interference
- Persistency: Non-Interference has to be guaranteed in all the states of the system, if processes migrate during execution

Intuitively



Delimited Persistent Stochastic Non-Interference

Motivation - I

- Non-Interference could be too demanding. It does not allow any information flow
- Delimited: mechanisms for downgrading or declassifying information from high to low are necessary
- Downgrading of information has to be performed by a trusted component

Delimited Persistent Stochastic Non-Interference

Motivation - II

- Once a process has been designed, it is necessary to check whether it satisfies Delimited Non-Interference or not
- If the process is not secure, it is necessary to modify it
- We look for a language which defines only secure processes

Delimited Persistent Stochastic Non-Interference

Contribution

- We introduce Persistent Stochastic Non-Interference (PSNI) Delimited Persistent Stochastic Non-Interference (D_PSNI) over Performance Evaluation Process Algebra (PEPA)
- We define process algebras for PSNI and D_PSNI processes
- Our process algebras denote equivalence relations that are
  - stronger than lumpability (bisimulation)
  - linearly verifiable w.r.t. the syntax of the process

Outline of the Talk

- Performance Evaluation Process Algebra (PEPA)
- Observation Equivalence: Lumpable Bisimilarity
- Persistent Stochastic Non-Interference (PSNI)
- Delimited Persistent Stochastic Non-Interference (D_PSNI)
- Unwinding and Compositionality: two secure process algebras
- Example and Conclusions

PEPA - Syntax and Semantics

Definition - PEPA Syntax

Let $\mathcal{A}$ be a set of actions with $\tau \in \mathcal{A}$
Let $\alpha \in \mathcal{A}$, $A \subseteq \mathcal{A}$, and $r \in \mathbb{R} \cup \{\top\}$

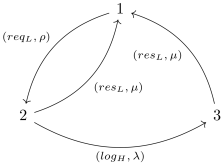
$$\begin{array}{rcl} S & ::= & \mathbf{0} \mid (\alpha, r).S \mid S + S \mid X \\ P & ::= & P \underset{A}{\bowtie} P \mid P/A \mid P \setminus A \mid S \end{array}$$

Each variable X is associated to a definition $X \stackrel{\text{def}}{=} P$

Definition - PEPA Semantics

It defines Labeled Continuous Time Markov Chains

Example



$$
\begin{aligned}
X_1 &= (req_L, \rho).X_2 \\
X_2 &= (res_L, \mu).X_1 + (log_H, \lambda).X_3 \\
X_3 &= (res_L, \mu).X_1
\end{aligned}
$$

PEPA - Semantics for Synchronization

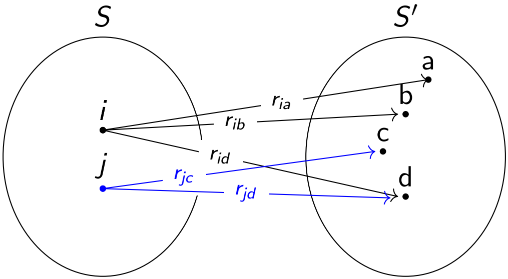
$$\frac{P \xrightarrow{(\alpha,r)} P'}{P \underset{A}{\bowtie} Q \xrightarrow{(\alpha,r)} P' \underset{A}{\bowtie} Q}\ (\alpha \notin A) \qquad \frac{Q \xrightarrow{(\alpha,r)} Q'}{P \underset{A}{\bowtie} Q \xrightarrow{(\alpha,r)} P \underset{A}{\bowtie} Q'}\ (\alpha \notin A)$$

$$\frac{P \xrightarrow{(\alpha,r_1)} P' \quad Q \xrightarrow{(\alpha,r_2)} Q'}{P \underset{L}{\bowtie} Q \xrightarrow{(\alpha,R)} P' \underset{A}{\bowtie} Q'} \qquad (\alpha \in A)$$

where $\quad R = \dfrac{r_1}{r_\alpha(P)} \dfrac{r_2}{r_\alpha(Q)} \min(r_\alpha(P), r_\alpha(Q))$

Observation Equivalence

Lumpability on the CTMC



$$r_{ia} + r_{ib} + r_{id} = r_{jc} + r_{jd}$$

Users cannot distinguish lumpable bisimilar PEPA components

Observation Equivalence

Definition - Lumpable bisimilarity

It is the largest equivalence relation $\approx_l$ such that if $P \approx_l Q$, then for all α and for each S equivalence class

- either $\alpha \neq \tau$,
- or $\alpha = \tau$ and $P, Q \notin S$,

it holds

$$\sum_{P' \in S,\ P \xrightarrow{(\alpha, r_\alpha)} P'} r_\alpha = \sum_{Q' \in S,\ Q \xrightarrow{(\alpha, r_\alpha)} Q'} r_\alpha$$

Properties

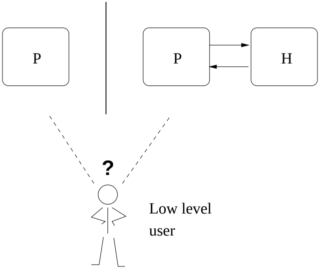
It is *contextual*, *action preserving*, and induces a *lumpability*

Non-Interference

A general definition [Focardi-Gorrieri'95]

$P \in NI$ iff $\forall$ high level process H, $(P|\,0) \sim^{low} (P|H)$

where $\sim^{low}$ denotes a low level observation equivalence



Stochastic Non-Interference (SNI)

- We partition the actions into $\mathcal{L}$ (low), $\mathcal{H}$ (high), $\{\tau\}$ (sinch.)
- High level processes can only perform high level actions
- Low level users can only perform/observe low level actions

Definition - SNI

$P \in SNI$ iff $\forall$ high level PEPA component H

$$(P \underset{\mathcal{H}}{\bowtie} 0) \sim^{low} (P \underset{\mathcal{H}}{\bowtie} H)$$

Low level observation $\sim^{low}$

It is $\approx_l$ without observing actions in $\mathcal{H}$

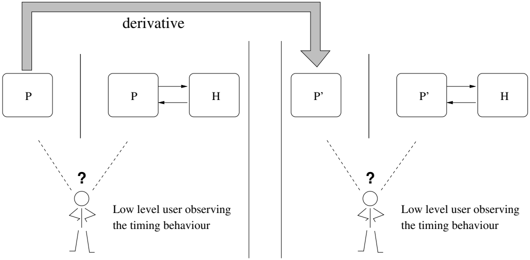
$$(P \underset{\mathcal{H}}{\bowtie} 0)/\mathcal{H} \approx_l (P \underset{\mathcal{H}}{\bowtie} H)/\mathcal{H}$$

PERSISTENT SNI (PSNI)

DEFINITION - PSNI

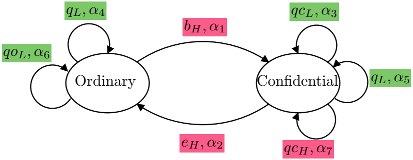
$$P \in PSNI \text{ iff } \forall \text{ derivative } P' \text{ of } P$$

$$P' \in SNI$$

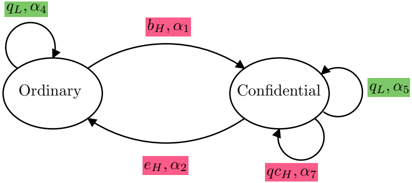


Toy Example: Unsecure Vs Secure System

Unsecure



Secure iff $\alpha_4 = \alpha_5$



DELIMITED PSNI (D_PSNI)

- We partition the actions into $\mathcal{L}$, $\mathcal{H}$, $\mathcal{D}$ (downgrading), $\{\tau\}$
- Downgrading actions specify the behavior of a trusted component that allows delimited flows from high to low
- Low level users can only perform/observe low level actions

DEFINITION - D_PSNI

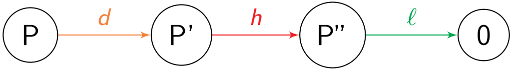
$P \in D\_PSNI$ iff $\forall$ derivative P' of P

$\forall$ high level PEPA component H

$$((P' \underset{\mathcal{H}}{\bowtie} 0)/\mathcal{H}) \setminus \mathcal{D} \approx_l ((P' \underset{\mathcal{H}}{\bowtie} H)/\mathcal{H}) \setminus \mathcal{D}$$

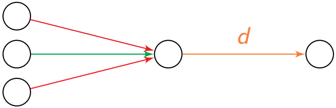
The importance of persistence

Example



P satisfies the condition, while P' does not

Intuitively



- The *d* action *downgrades* the high incoming actions
- It does not downgrade subsequent high actions

Let us ...

... focus on *PSNI*

Luckily, as for the secure process algebra, *D_PSNI* is mainly a technical generalization

Properties

Theorem - Unwinding

$P \in PSNI$ iff $\forall$ derivative P' of P,

$$P' \xrightarrow{(h,r)} P'' \text{ implies } P' \setminus \mathcal{H} \approx_l P'' \setminus \mathcal{H}$$

PROPERTIES

THEOREM - UNWINDING

$$P \in PSNI \text{ iff } \forall \text{ derivative } P' \text{ of } P,$$

$$P' \xrightarrow{(h,r)} P'' \text{ implies } P' \setminus \mathcal{H} \approx_l P'' \setminus \mathcal{H}$$

- This allows to explicitly identify the *dangerous* situations
- Whenever a high level action is performed we impose syntactic conditions that ensure $\approx_l$

PROPERTIES

THEOREM - COMPOSITIONALITY I

Let $P, P_i \in PSNI$, Q be a PEPA component, and $A \subseteq \mathcal{A} \setminus \{\tau\}$
The following processes are $PSNI$

- $\mathbf{0}$
- $Q \backslash \mathcal{H}$, $Q \backslash \mathcal{L}$, $Q/\mathcal{H}$, and $Q/\mathcal{L}$
- $(\ell, r).P$ with $\ell \in \mathcal{L} \cup \{\tau\}$
- P/A and $P \setminus A$
- $P_i \underset{A}{\bowtie} P_j$

Properties

Theorem - Compositionality I

Let $P, P_i \in PSNI$, Q be a PEPA component, and $A \subseteq \mathcal{A} \setminus \{\tau\}$
The following processes are $PSNI$

- $\mathbf{0}$
- $Q \backslash \mathcal{H}$, $Q \backslash \mathcal{L}$, $Q/\mathcal{H}$, and $Q/\mathcal{L}$
- $(\ell, r).P$ with $\ell \in \mathcal{L} \cup \{\tau\}$
- P/A and $P \setminus A$
- $P_i \underset{A}{\bowtie} P_j$

Remark

These are consequences of PEPA broadcasting synchronization rules and are not true in other process algebra (e.g., CCS like)

PROPERTIES

THEOREM - COMPOSITIONALITY II

Let $P, P_i \in PSNI$, Q be a PEPA component, and $A \subseteq \mathcal{A} \setminus \{\tau\}$

- X_c, X_c' are $PSNI$ where

$$X_c \stackrel{def}{=} \sum_{i \in I} (\ell_i, r_i).P_i + \sum_{k \in K} (\ell_k, r_k).X_k + \sum_{j \in J} (h_j, r_j).X_c \setminus H_j + \sum_{m \in M} (h_m, r_m).X_c'$$

$$X_c' \stackrel{def}{=} \sum_{i \in I} (\ell_i, r_i).P_i + \sum_{k \in K} (\ell_k, r_k).X_k$$

Properties

Theorem - Compositionality II

Let $P, P_i \in PSNI$, Q be a PEPA component, and $A \subseteq \mathcal{A} \setminus \{\tau\}$

- X_c, X'_c are $PSNI$ where

$$X_c \stackrel{def}{=} \sum_{i \in I} (\ell_i, r_i).P_i + \sum_{k \in K} (\ell_k, r_k).X_k + \sum_{j \in J} (h_j, r_j).X_c \setminus H_j + \sum_{m \in M} (h_m, r_m).X'_c$$

$$X'_c \stackrel{def}{=} \sum_{i \in I} (\ell_i, r_i).P_i + \sum_{k \in K} (\ell_k, r_k).X_k$$

Remark

- This is a trade-off between readability and expressivity
- How much can we improve? See *Some of My Favourite Results in Classic Process Algebra* by L. Aceto

PSNI Process Algebra

Definition - $\mathcal{C}_{PSNI}$

Let Q be PEPA component and $A \subseteq \mathcal{A} \setminus \{\tau\}$
$\mathcal{C}_{PSNI}$ is defined by the following grammar:

$$S ::= \mathbf{0} \mid Q \setminus \mathcal{H} \mid Q \setminus \mathcal{L} \mid (\ell, r).S \mid X$$
$$P ::= S \mid P/A \mid P \setminus A \mid P \underset{A}{\bowtie} P$$

where X has a recursive definition of the form

$$X \stackrel{def}{=} \sum_{i \in I} (\ell_i, r_i).S_i + \sum_{j \in J} (h_j, r_j).X \setminus H_j + \sum_{m \in M} (h_m, r_m).X'$$
$$X' \stackrel{def}{=} \sum_{i \in I} (\ell_i, r_i).S_i$$

PSNI Process Algebra

Definition - $\mathcal{C}_{PSNI}$

Let Q be PEPA component and $A \subseteq \mathcal{A} \setminus \{\tau\}$
$\mathcal{C}_{PSNI}$ is defined by the following grammar:

$$S ::= \mathbf{0} \mid Q \setminus \mathcal{H} \mid Q \setminus \mathcal{L} \mid (\ell, r).S \mid X$$
$$P ::= S \mid P/A \mid P \setminus A \mid P \underset{A}{\bowtie} P$$

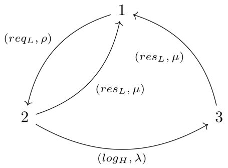
where X has a recursive definition of the form

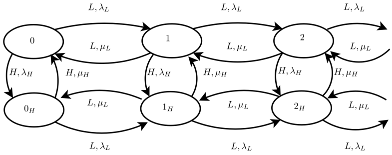
$$X \stackrel{def}{=} \sum_{i \in I} (\ell_i, r_i).S_i + \sum_{j \in J} (h_j, r_j).X \setminus H_j + \sum_{m \in M} (h_m, r_m).X'$$
$$X' \stackrel{def}{=} \sum_{i \in I} (\ell_i, r_i).S_i$$

Remark

- We can also define infinite state processes
- We can generalize to a process algebra for *D_PSNI*

Toy Examples: $\mathcal{C}_{PSNI}$ Systems





Conclusion

- A general framework for PSNI and D_PSNI has been presented
- The use of Contextual Lumpability guarantees that the steady state distribution is not influenced by the high level behavior
- Two process algebras that allow to define processes secure by construction have been introduced

Questions

- Can we find a *complete process algebra*?
- How is it related to efficient computation of lumpability/bisimulation?