Information Flow Security in Dynamic Contexts

Process Algebras exam
Gaudiano Antonio

Riccardo Focardi, Sabina Rossi

Introduction

The problem: Why do we need a «new» security property?

Introduction

Systems are becoming more complex, requiring security considerations like process mobility across different architectures, systems and networks.

Environments can change unpredictably at runtime, exposing processes to new attackers and potentially compromising previously secure programs.

A new security property, Persistent BNDC extends Non-Interference to ensure all reachable states remain secure, even under dynamic environmental changes.

The SPA language

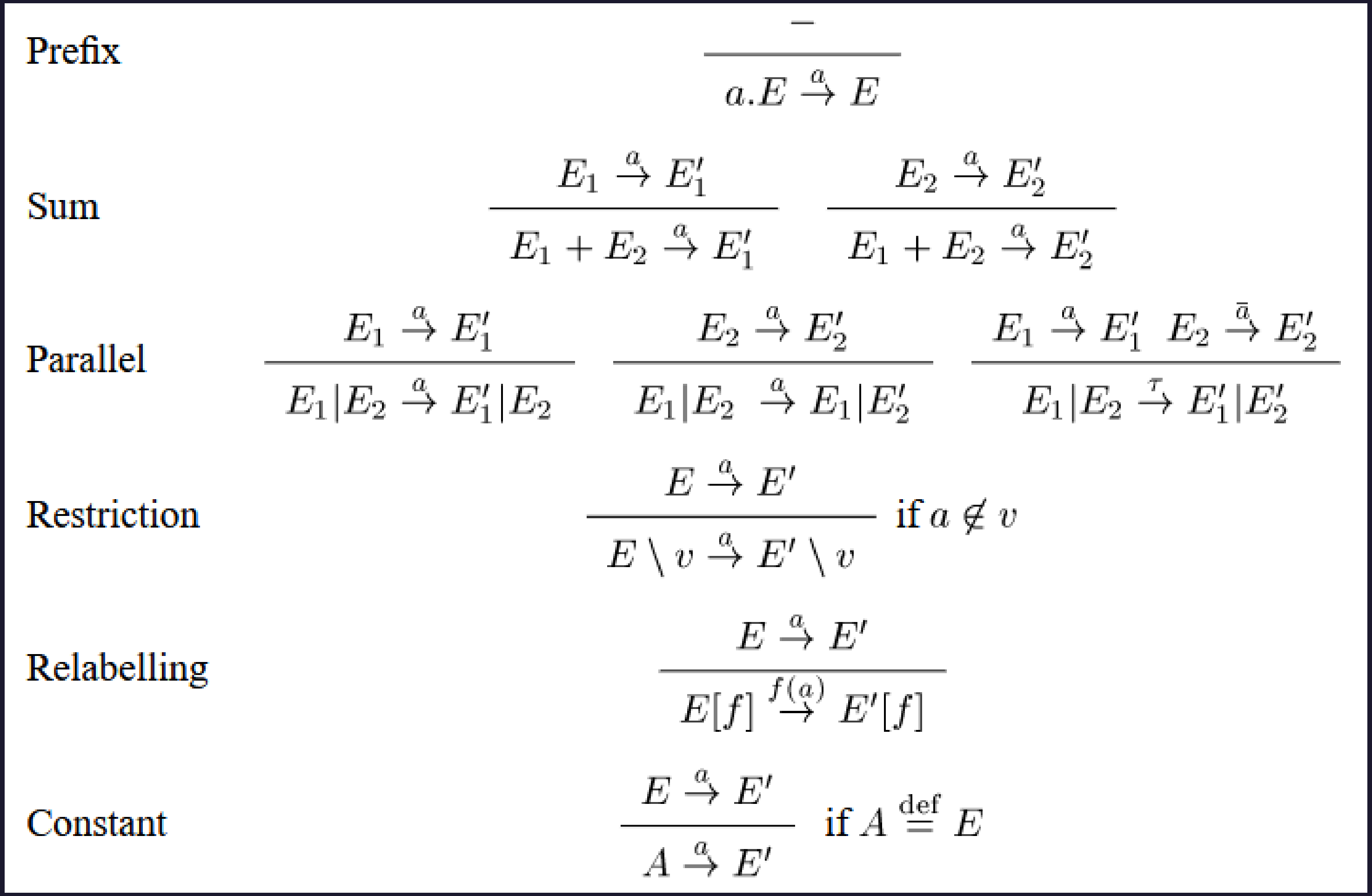
Security Process Algebra language, an extension of Milner’s CCS

The SPA language

- Visible actions: $\mathcal{L} = I \cup O,\ where\ I = \{a, b, \ldots\}\ ,\ O = \{\bar{a}\ , \bar{b}, \ldots\}$
- Complementation function: $\mathcal{L}\ \rightarrow\ \mathcal{L},\ \bar{\bar{a}} = a, \forall a \in \mathcal{L}$
- All actions (visible and internal): $Act =\ \mathcal{L}\ \cup \{\tau\}$
- SPA agents (processes) syntax:

$$E\ \ ::= 0 \mid \alpha.E \mid E + E \mid E|E \mid E\backslash \text{v} \mid E[f] \mid Z$$

SPA operational semantics



Prefix	$\dfrac{-}{a.E \xrightarrow{a} E}$
Sum	$\dfrac{E_1 \xrightarrow{a} E_1'}{E_1 + E_2 \xrightarrow{a} E_1'}$ $\quad$ $\dfrac{E_2 \xrightarrow{a} E_2'}{E_1 + E_2 \xrightarrow{a} E_2'}$
Parallel	$\dfrac{E_1 \xrightarrow{a} E_1'}{E_1 \| E_2 \xrightarrow{a} E_1' \| E_2}$ $\quad$ $\dfrac{E_2 \xrightarrow{a} E_2'}{E_1 \| E_2 \xrightarrow{a} E_1 \| E_2'}$ $\quad$ $\dfrac{E_1 \xrightarrow{a} E_1' \;\; E_2 \xrightarrow{\bar{a}} E_2'}{E_1 \| E_2 \xrightarrow{\tau} E_1' \| E_2'}$
Restriction	$\dfrac{E \xrightarrow{a} E'}{E \setminus v \xrightarrow{a} E' \setminus v}$ if $a \notin v$
Relabelling	$\dfrac{E \xrightarrow{a} E'}{E[f] \xrightarrow{f(a)} E'[f]}$
Constant	$\dfrac{E \xrightarrow{a} E'}{A \xrightarrow{a} E'}$ if $A \overset{\mathrm{def}}{=} E$

Based on LTS (Labeling Transition System), a triple (S, A, →), with the usual notation $S_1 \xrightarrow{a} S_2$

Recall: weak bisimulation

A binary relation $S \subseteq \mathcal{E} x \mathcal{E}$ over agents is a **weak bisimulation** if (E,F) ∈ S implies, for all a ∈ Act:

- whenever E $\overset{a}{\rightarrow}$ E', then there exists F' such that F $\overset{a}{\Rightarrow}$ F' and (E',F') ∈ S;
- whenever F $\overset{a}{\rightarrow}$ F', then there exists E' such that E $\overset{a}{\Rightarrow}$ E' and (E',F') ∈ S;

Two agents E,F ∈ $\mathcal{E}$ are **observation equivalent**, denoted by E ≈ F, if there exist a weak bisimulation S containing the pair (E,F).

Security Properties

The description of the secured system

General specifications

Protect the system from insider threats and Trojan Horse

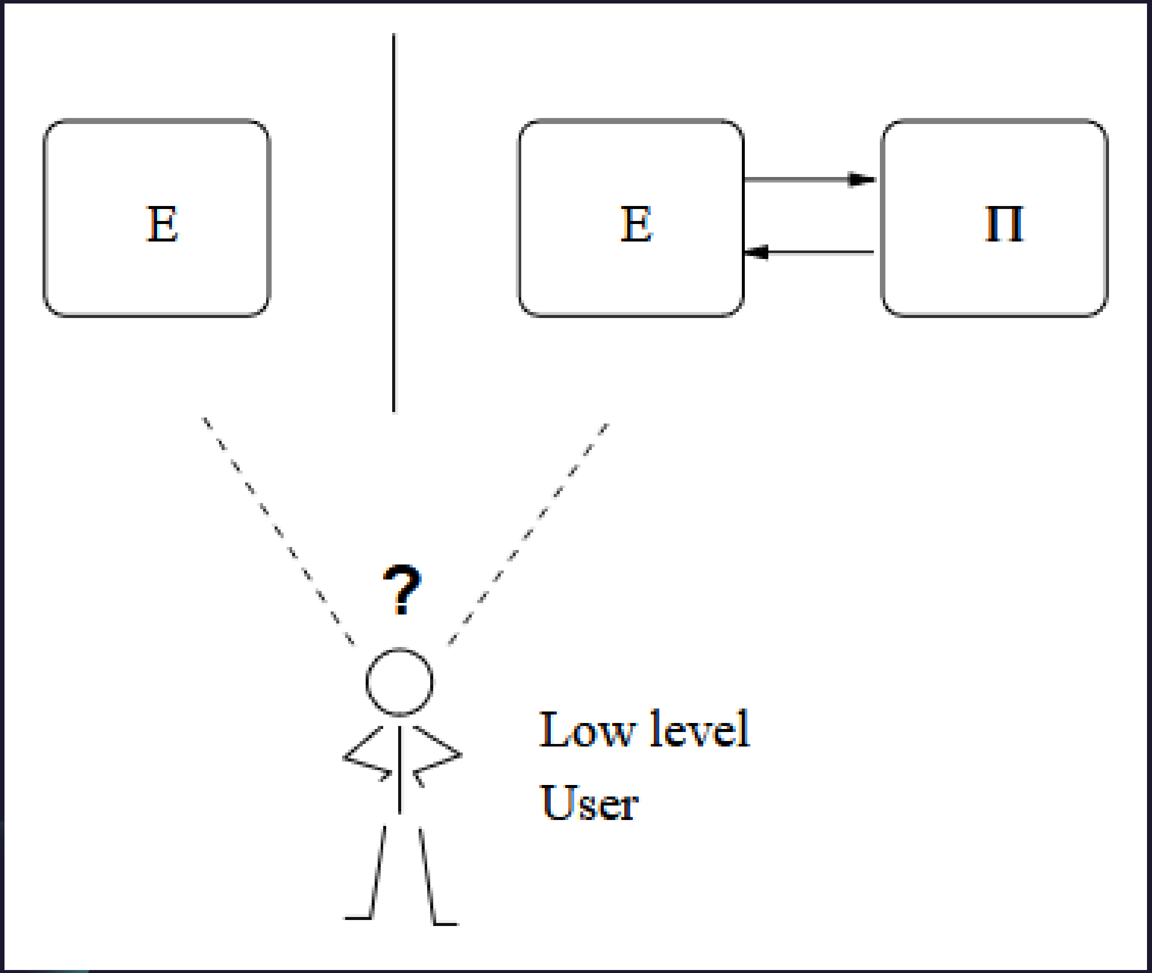
Secure information flow from a classified (high) level of confidentiality to an untrusted (low) one

“No read up, no write down” rule for avoiding declassification attacks

Non Deducibility on Compositions

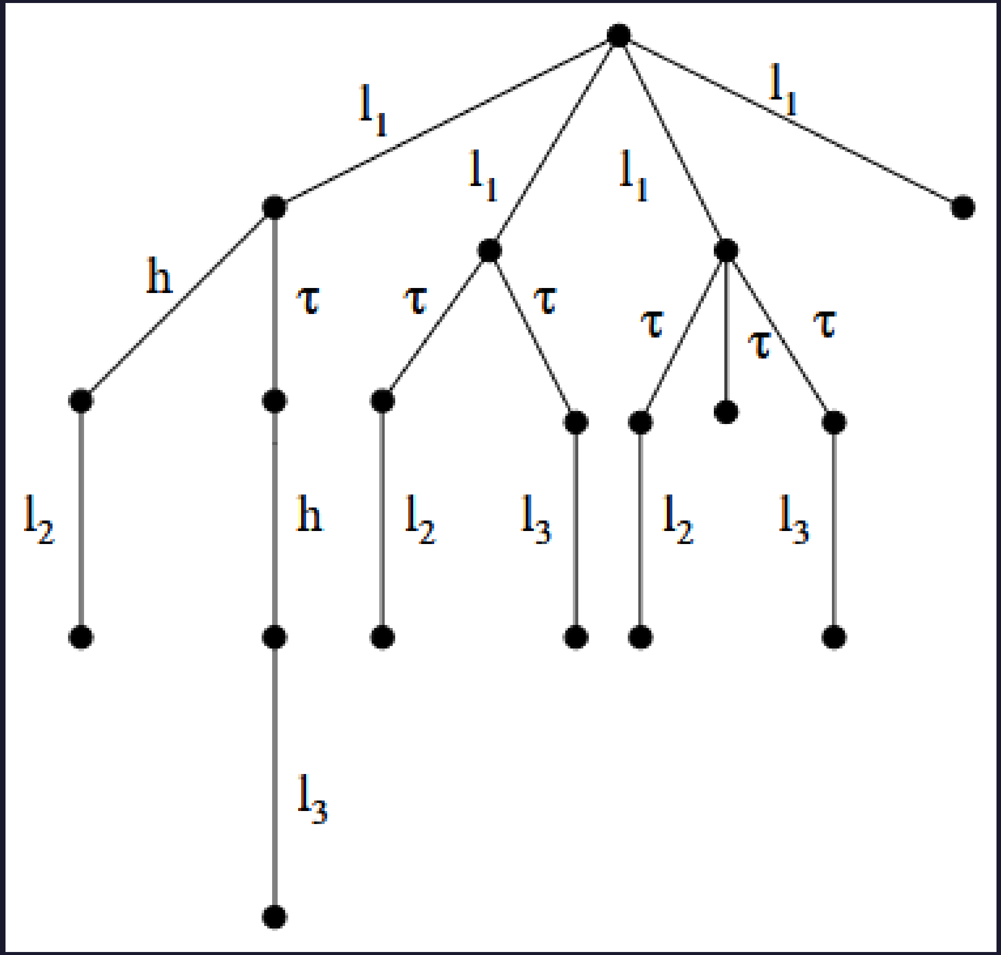
$$E \in BNDC \ iff \ \forall \Pi \in \varepsilon_H, \\ E \setminus Act_H \approx (E|\Pi) \setminus Act_H$$

A low-level user can't distinguish between E and E\Π

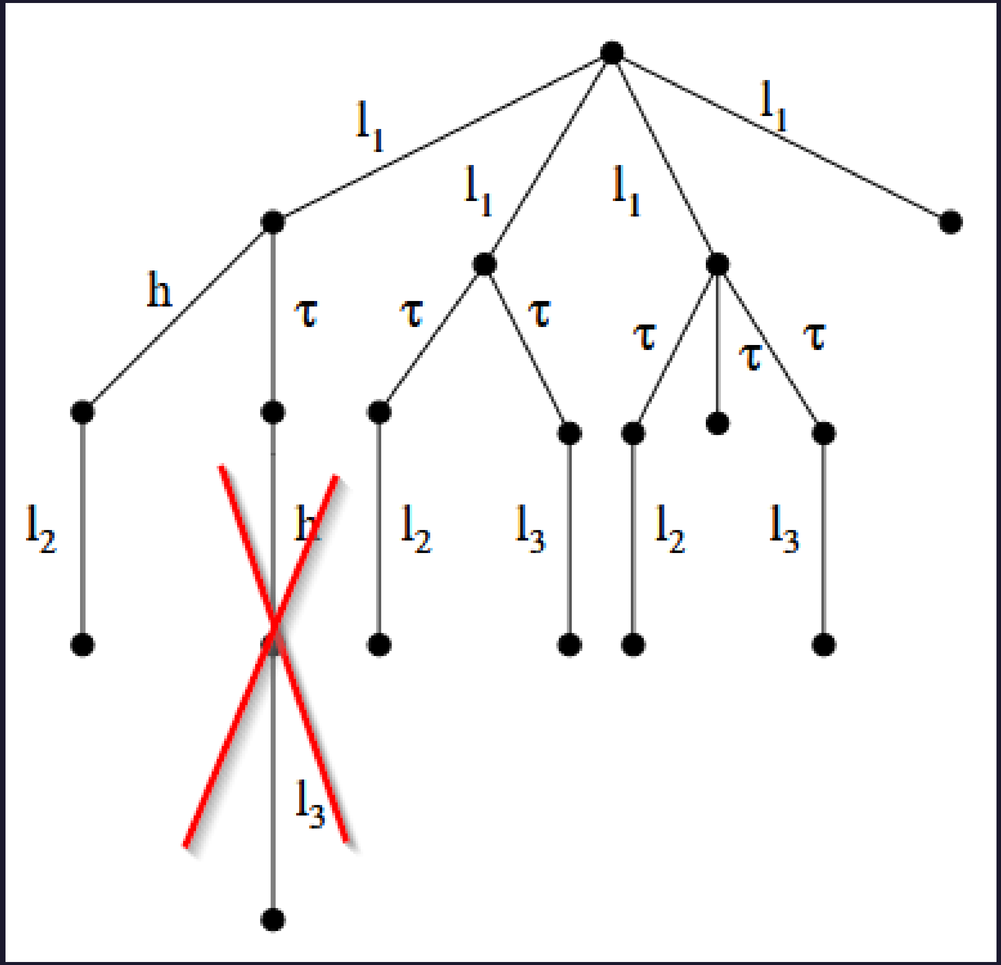


Why BNDC isn’t enough in dynamic contexts

- Let’s consider the process in figure and another high-level process that can be **reconfigured at any time**: h.0 -> 0
- What happens if those processes runs in parallel…?



- The malicious process switches its behaviour to 0, blocking h.
- A low-level observer can notice whether the system executes l_3 or not, so it can **infer the existence of h** checking if the system behaves normally or gets stuck!
- This violates non-interference, meaning low-level users can deduce high-level actions, which is a security flaw.



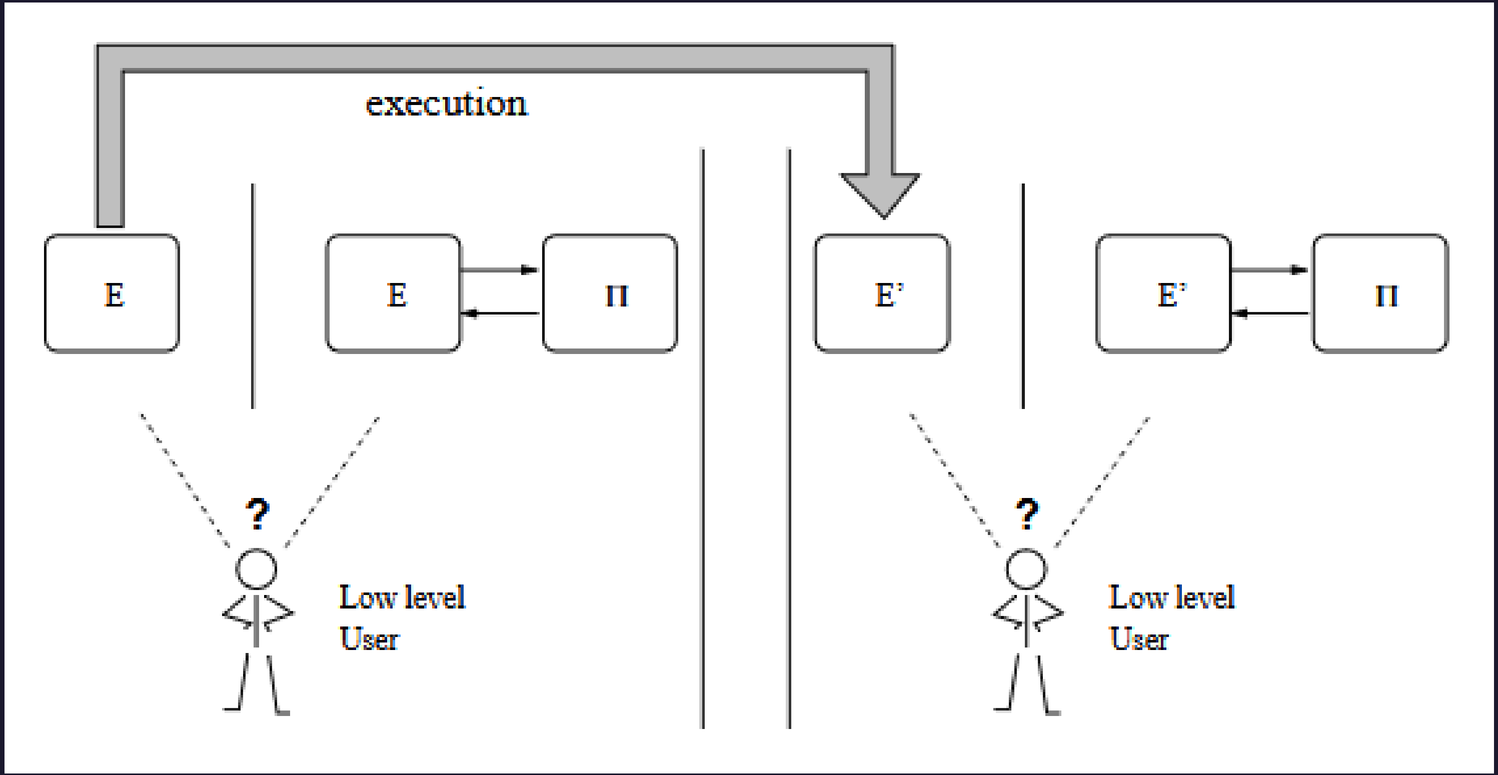
Persistent BNDC

$$E \in P\_BNDC\ iff$$

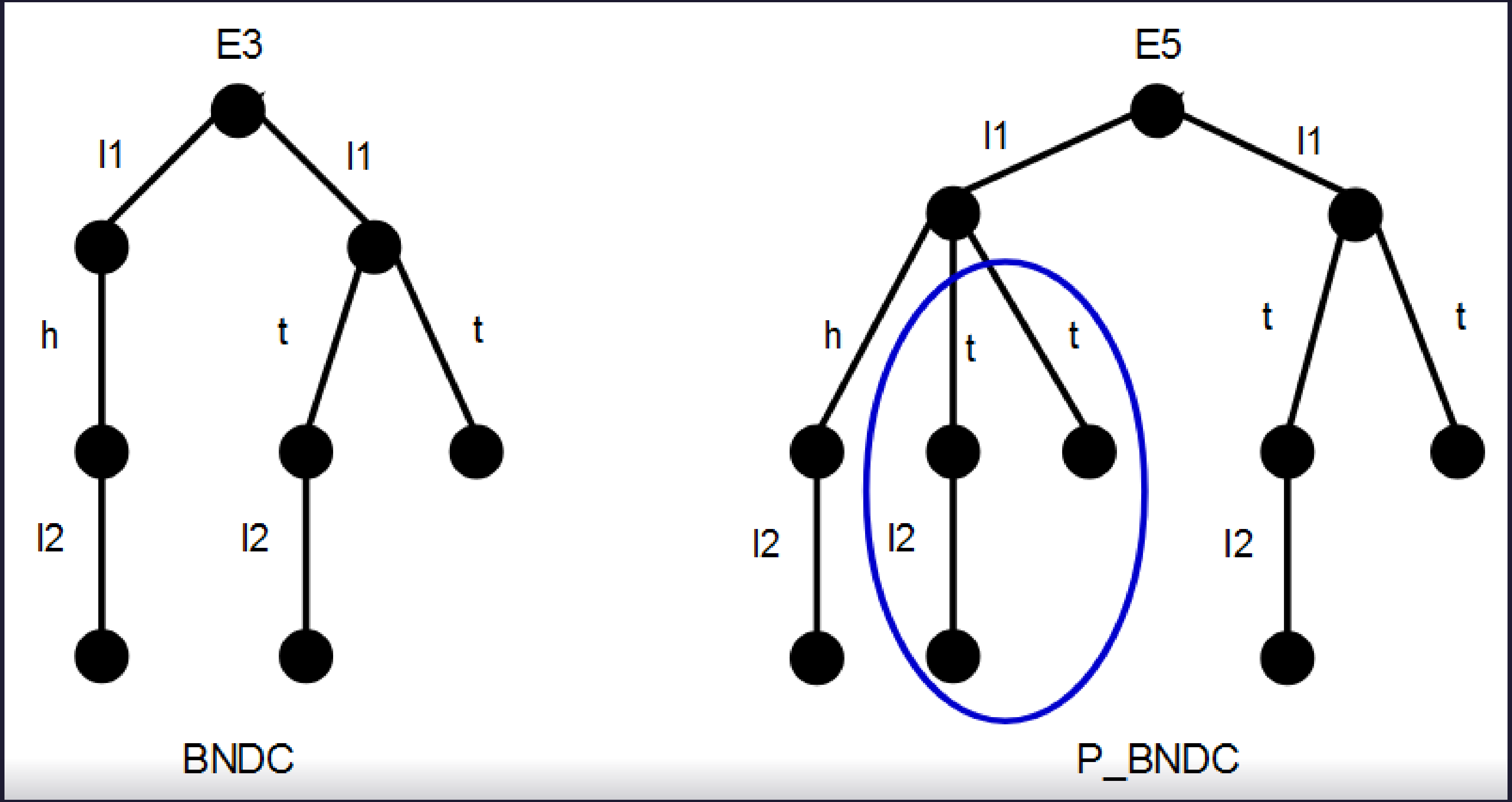
$$\forall\, E'\ reachable\ from\ E\ and\ \forall\, \Pi \in \varepsilon_H,$$

$$E' \setminus Act_H \approx (E'|\Pi) \setminus Act_H$$

A low-level user can't distinguish between E and E\Π **even during execution**



Comparison



P_BNDC Avoids Universal Quantification Over Attackers

Traditional BNDC requires verifying security against **all possible attacker processes**, making it computationally expensive.

P_BNDC replaces this with a condition that must hold at **every reachable state**, simplifying verification.

This approach ensures that security is preserved even if the system evolves dynamically.

P_BNDC Compositionality

P_BNDC is **compositional under parallel composition** (but not under nondeterministic choice), meaning if two processes satisfy P_BNDC individually, their composition also satisfies It.

It is also preserved under **prefixing** (i.e., if a process P is secure, then a.P is also secure).

This allows security verification of complex systems to be **modular**, reducing the need for full system analysis.

Equivalence to SBSNNI (Strong Bisimulation-Based Strong Non-deterministic Non-Interference)

P_BNDC has been formally proven **equivalent** to SBSNNI, a well-established security property in information flow theory.

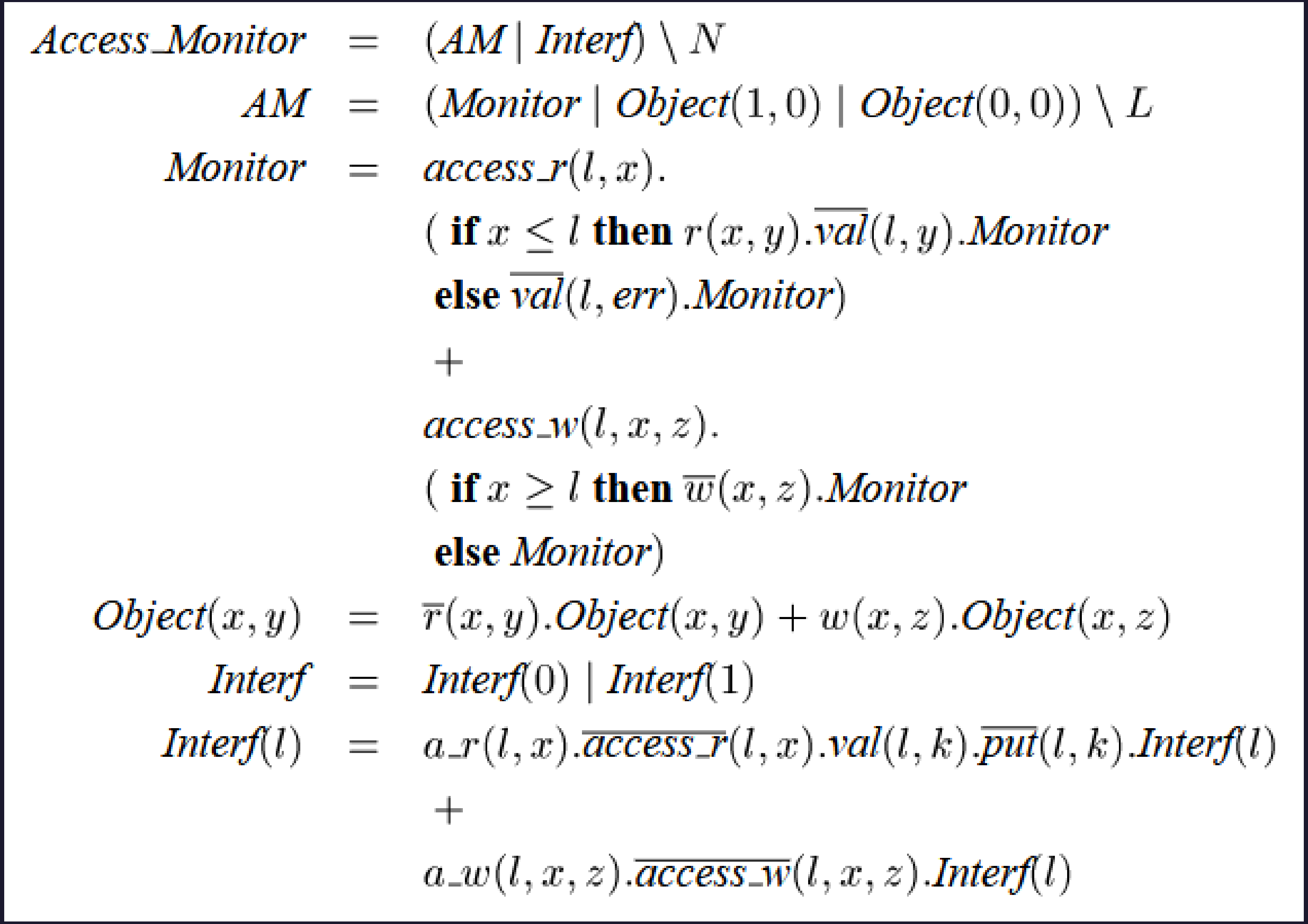
This means that any verification technique applicable to SBSNNI can be used for P_BNDC, facilitating analysis.

The equivalence ensures that **security holds at every execution step**, even in dynamic environments.

An example

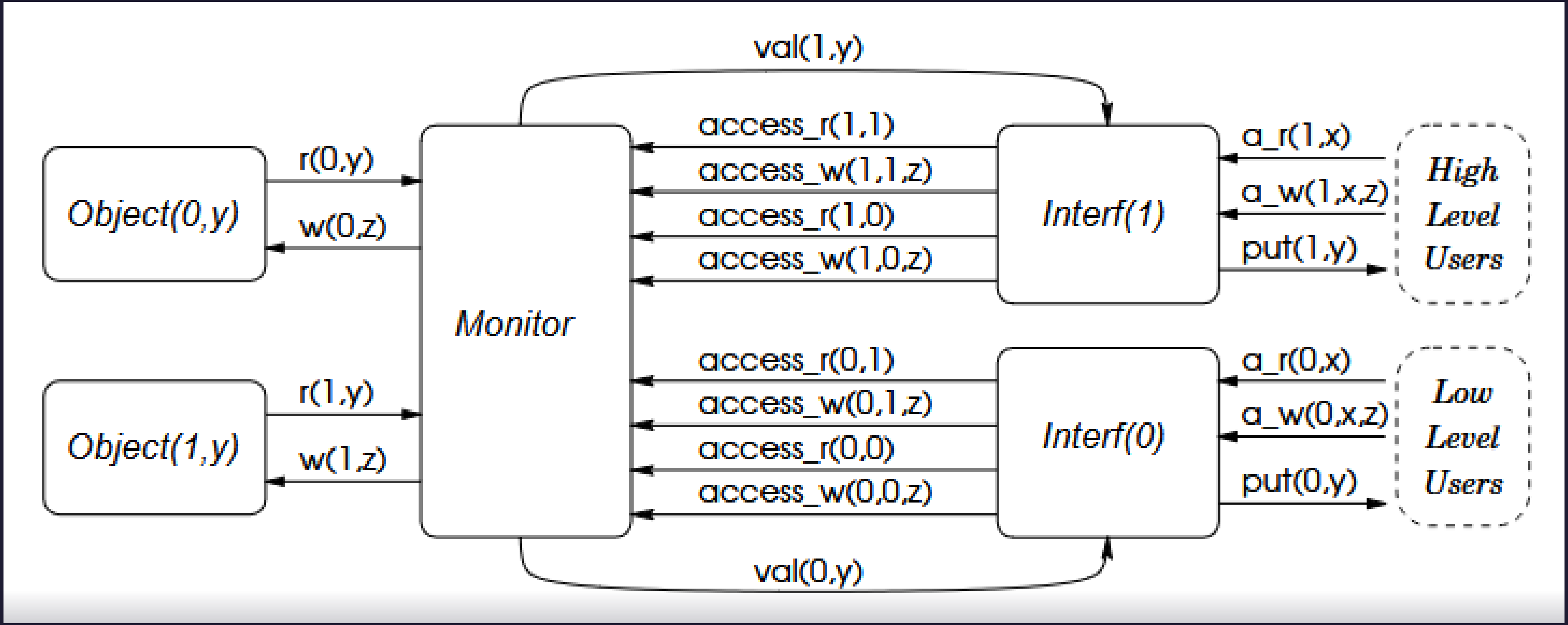
A Multilevel Security Policy «collecting agent»

Access Monitor specifications



$$\begin{array}{rcl}
\mathit{Access\_Monitor} & = & (\mathit{AM} \mid \mathit{Interf}) \setminus N \\
\mathit{AM} & = & (\mathit{Monitor} \mid \mathit{Object}(1,0) \mid \mathit{Object}(0,0)) \setminus L \\
\mathit{Monitor} & = & \mathit{access\_r}(l,x). \\
 & & (\ \textbf{if}\ x \leq l\ \textbf{then}\ r(x,y).\overline{\mathit{val}}(l,y).\mathit{Monitor} \\
 & & \ \textbf{else}\ \overline{\mathit{val}}(l,\mathit{err}).\mathit{Monitor}) \\
 & & \ + \\
 & & \mathit{access\_w}(l,x,z). \\
 & & (\ \textbf{if}\ x \geq l\ \textbf{then}\ \overline{w}(x,z).\mathit{Monitor} \\
 & & \ \textbf{else}\ \mathit{Monitor}) \\
\mathit{Object}(x,y) & = & \overline{r}(x,y).\mathit{Object}(x,y) + w(x,z).\mathit{Object}(x,z) \\
\mathit{Interf} & = & \mathit{Interf}(0) \mid \mathit{Interf}(1) \\
\mathit{Interf}(l) & = & a\_r(l,x).\overline{\mathit{access\_r}}(l,x).\mathit{val}(l,k).\overline{\mathit{put}}(l,k).\mathit{Interf}(l) \\
 & & \ + \\
 & & a\_w(l,x,z).\overline{\mathit{access\_w}}(l,x,z).\mathit{Interf}(l)
\end{array}$$

Access Monitor diagram



Conclusion

Summary and future works

- **Definition**: Security property based on Non-Interference, ensuring protection in dynamic environments.
- **Verification**: Reduced to weak bisimulation up to high-level actions, enabling efficient model-checking.
- **Automation**: Existing security checkers (e.g., SBSNNI) can be improved for better P_BNDC verification.
- **Limitations** & **Future Work**: Not compositional under nondeterministic choice; needs adaptation for mobility-focused languages.

Thanks for the attention!