Ownership Types for the Join Calculus.

M. Patrignani D. Clarke D. Sangiorgi

Katholieke Universiteit Leuven

Università di Bologna

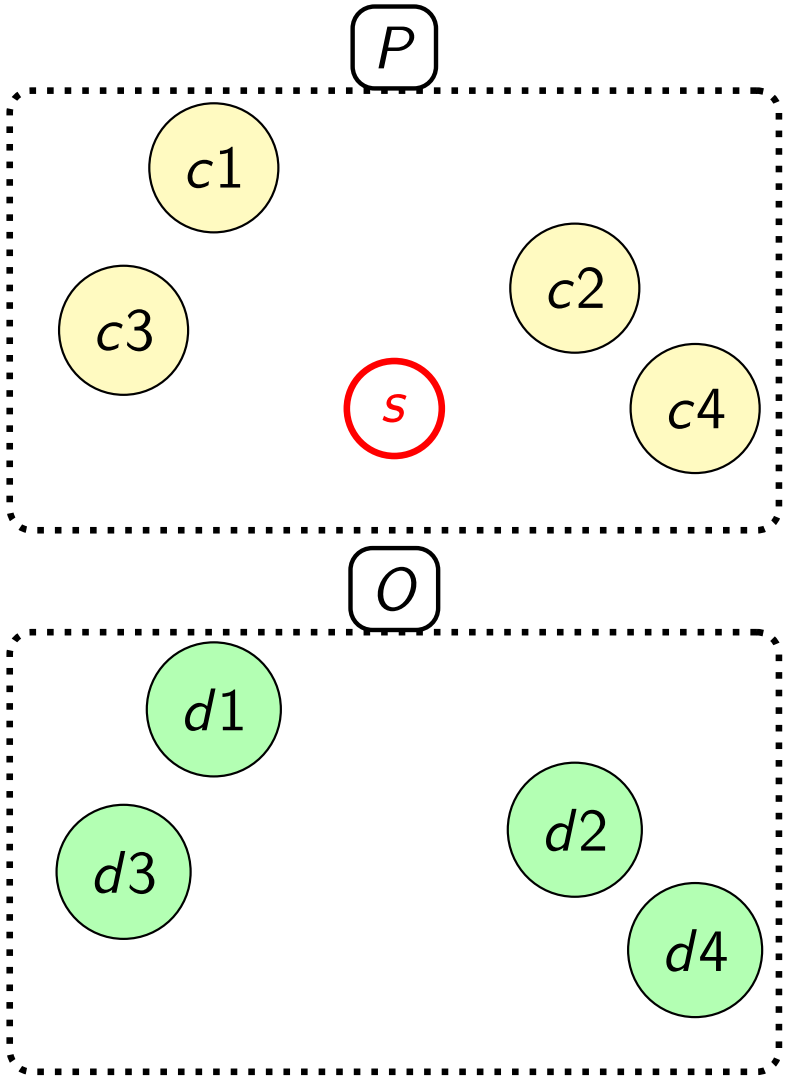
March 7, 2012

Outline

Outline

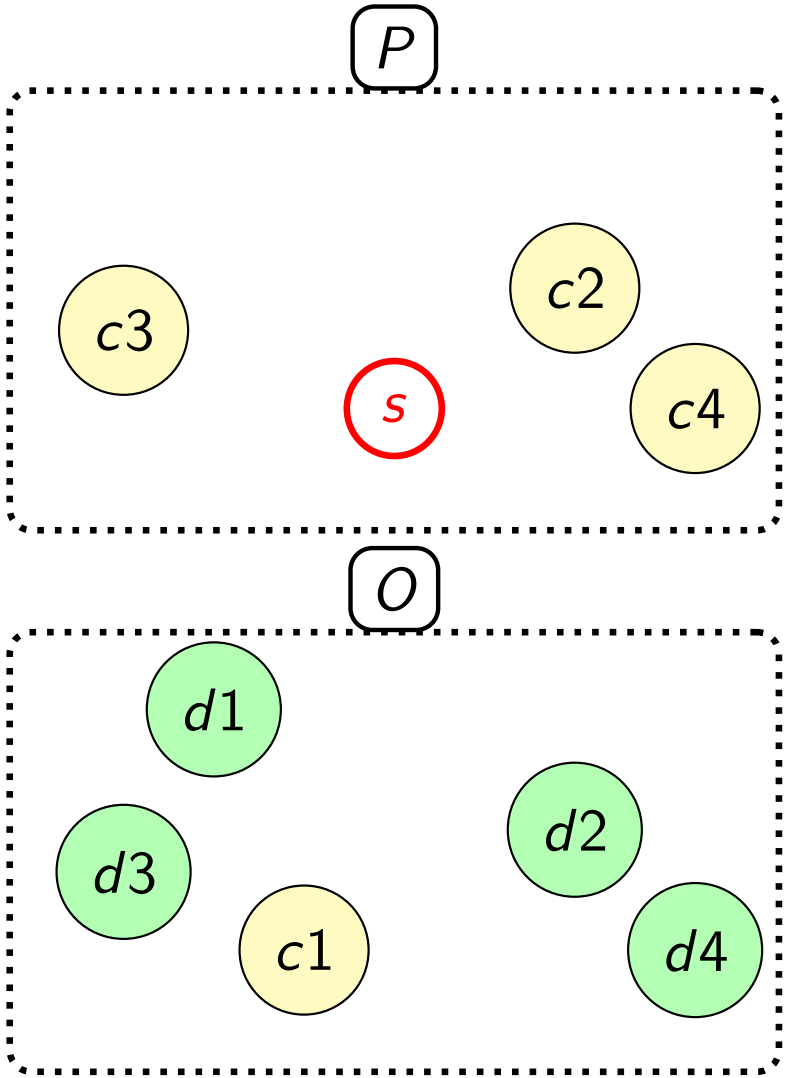
Introduction

- Channels are free to wander around, too free to do so.



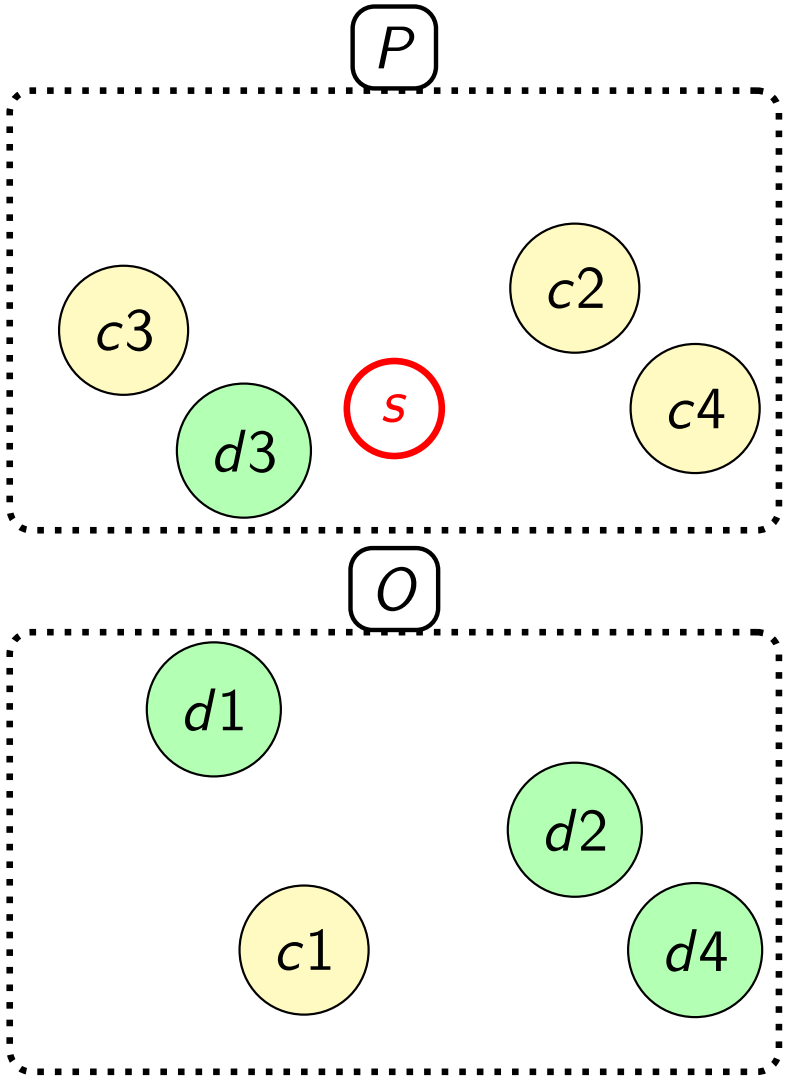
Introduction

- Channels are free to wander around, too free to do so.



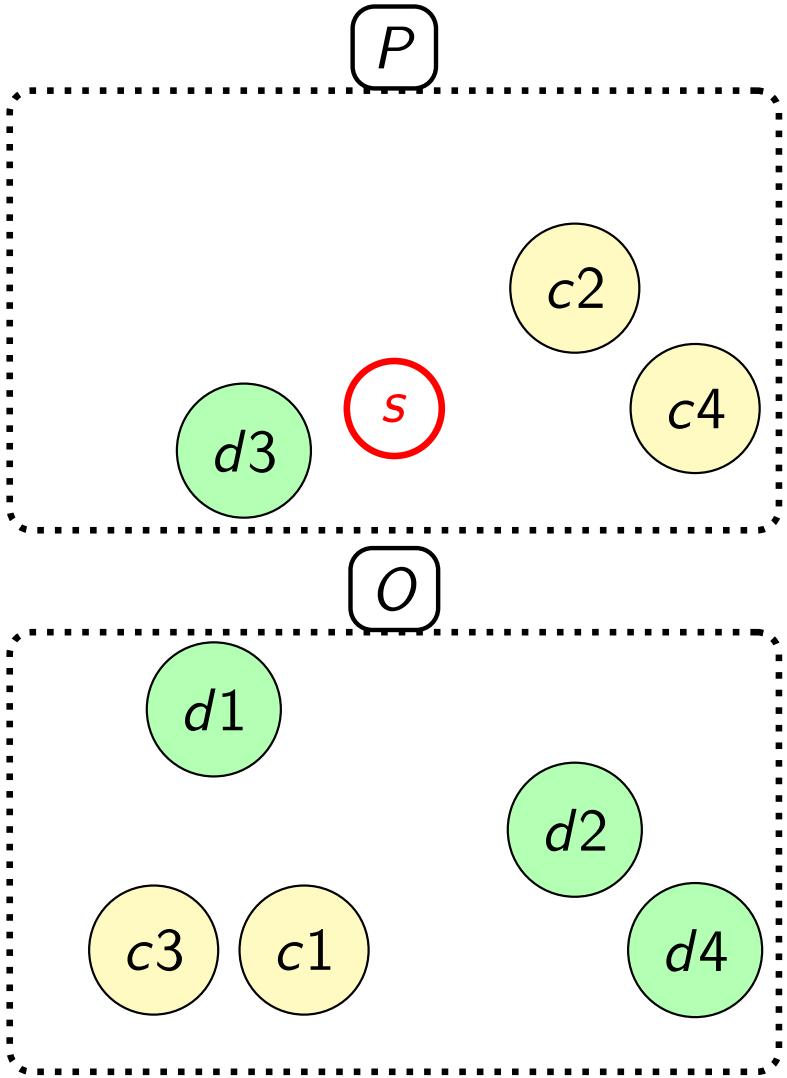
Introduction

- Channels are free to wander around, too free to do so.



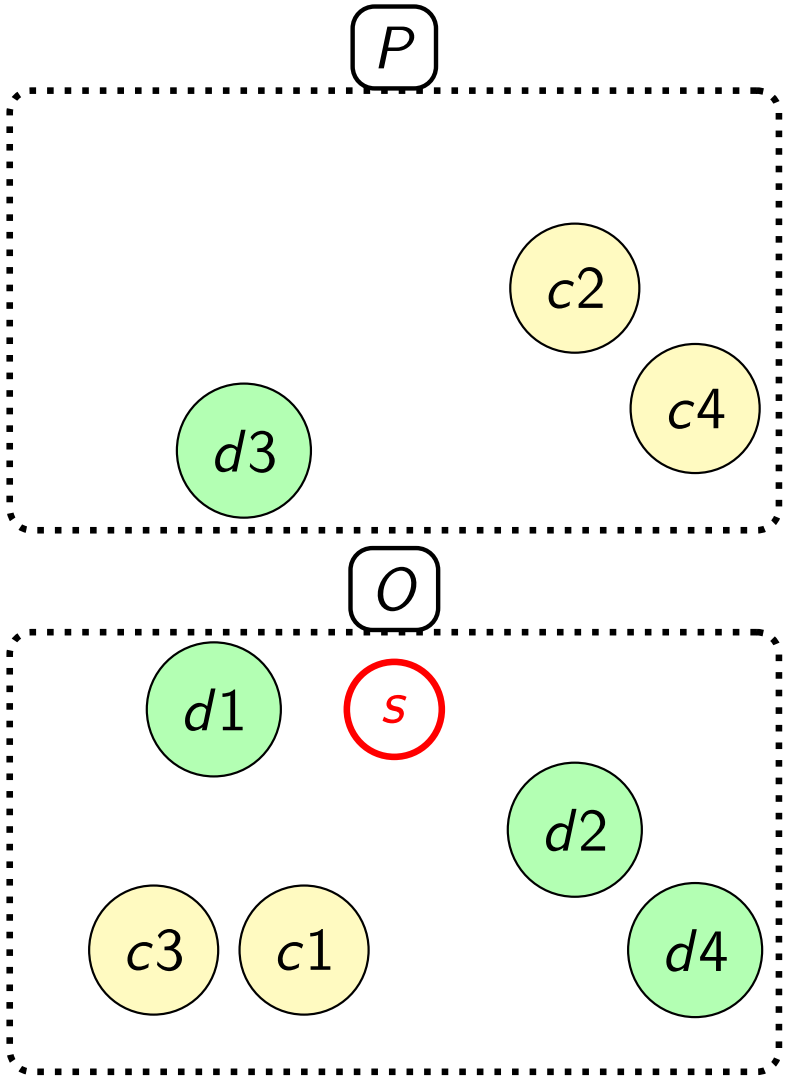
Introduction

- Channels are free to wander around, too free to do so.
- Channels may define secrets.



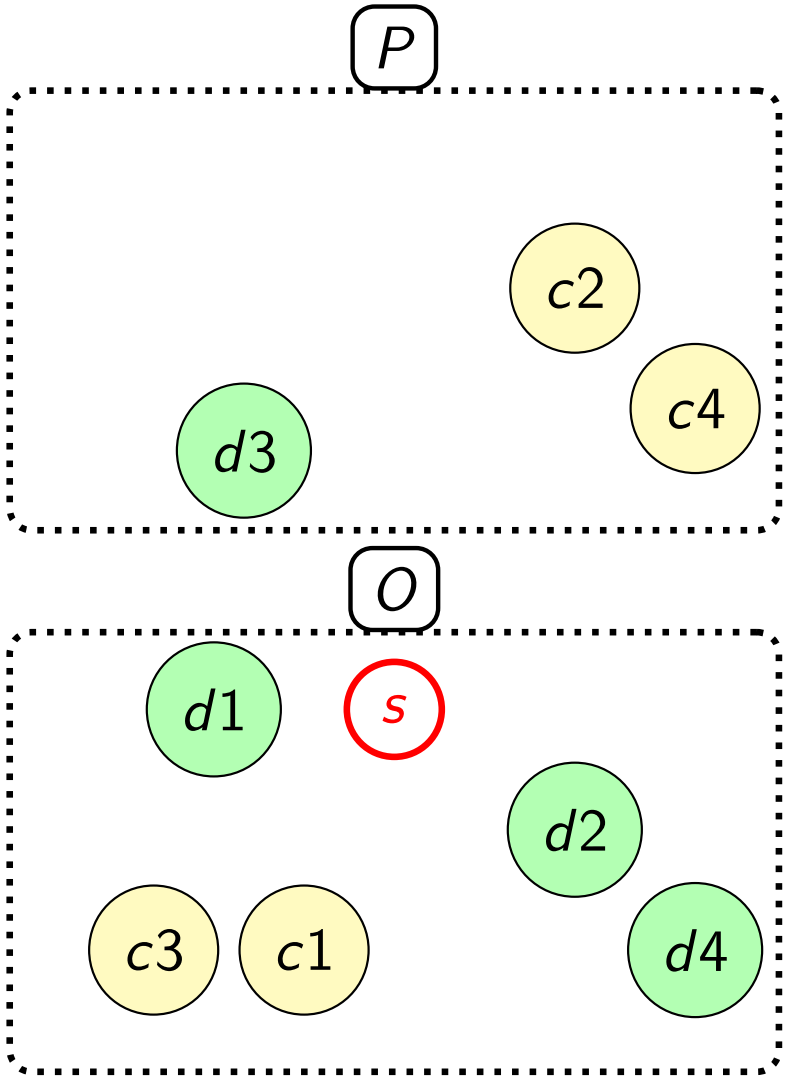
Introduction

- Channels are free to wander around, too free to do so.
- Channels may define secrets.
- Problem: leakage of secret.



Introduction

- Channels are free to wander around, too free to do so.
- Channels may define secrets.
- Problem: leakage of secret.
- Solution: enforcing fine-grained control on channels, a typing discipline that enforces a notion of encapsulation borrowed from OO: Ownership Types.



Outline

Untyped Join Calculus: Syntax

$$P = \textbf{def}\ D\ \textbf{in}\ P \mid P \mid P \mid x\langle \overline{u} \rangle \mid \emptyset$$

$$D = D \wedge D \mid J \rhd P \mid \top$$

$$J = J \mid J \mid x\langle \overline{y} \rangle$$

$$P = \textbf{def}\ \ x\langle a \rangle \mid\ y\langle\rangle \rhd \emptyset \ \wedge\ \top\ \textbf{in}\ \ x\langle y \rangle \mid y\langle\rangle$$

Untyped Join Calculus: Syntax

$$P = \textbf{def}\ D\ \textbf{in}\ P \mid P \mid P \mid x\langle \overline{u} \rangle \mid \emptyset$$

$$D = D \wedge D \mid J \rhd P \mid \top$$

$$J = J \mid J \mid x\langle \overline{y} \rangle$$

$$P = \textbf{def}\ x\langle a \rangle \mid y\langle\rangle \rhd \emptyset \wedge \top\ \textbf{in}\ x\langle y \rangle \mid y\langle\rangle$$

Untyped Join Calculus: Syntax

$$P = \textbf{def}\ D\ \textbf{in}\ P \mid P \mid P \mid x\langle \overline{u} \rangle \mid \emptyset$$

$$D = D \wedge D \mid \mathrm{J} \rhd \mathrm{P} \mid \top$$

$$J = \mathrm{J} \mid \mathrm{J} \mid x\langle \overline{y} \rangle$$

$$P = \textbf{def}\ x\langle a \rangle \mid y\langle\rangle \rhd \emptyset \wedge \top\ \textbf{in}\ x\langle y \rangle \mid y\langle\rangle$$

Untyped Join Calculus: Syntax

$$P = \textbf{def}\ D\ \textbf{in}\ P \mid P \mid P \mid x\langle \overline{u} \rangle \mid \emptyset$$

$$D = D \wedge D \mid J \rhd P \mid \top$$

$$J = J \mid J \mid x\langle \overline{y} \rangle$$

$$P = \textbf{def}\ x\langle a \rangle \mid y\langle\rangle \rhd \emptyset \wedge \top\ \textbf{in}\ x\langle y \rangle \mid y\langle\rangle$$

Untyped Join Calculus: Syntax

$$P = \textbf{def}\ D\ \textbf{in}\ P \mid P \mid P \mid x\langle \overline{u} \rangle \mid \emptyset$$

$$D = \mathrm{D} \wedge \mathrm{D} \mid J \rhd P \mid \top$$

$$J = J \mid J \mid x\langle \overline{y} \rangle$$

$$P = \textbf{def}\ x\langle a \rangle \mid y\langle\rangle \rhd \emptyset \wedge \top\ \textbf{in}\ x\langle y \rangle \mid y\langle\rangle$$

Untyped Join Calculus: Syntax

$$P = \textbf{def}\ D\ \textbf{in}\ P \quad | \quad P \mid P \quad | \quad x\langle \overline{u} \rangle \quad | \quad \emptyset$$

$$D = D \wedge D \quad | \quad J \rhd P \quad | \quad \top$$

$$J = J \mid J \quad | \quad x\langle \overline{y} \rangle$$

$$P = \textbf{def}\ \ x\langle a \rangle |\ y\langle\rangle \rhd \emptyset \ \wedge\ \top \ \textbf{in}\ \ x\langle y \rangle \mid y\langle\rangle$$

Outline

Untyped Join Calculus: Semantics

$$\mathcal{D} \Vdash \mathcal{P}$$

Untyped Join Calculus: Semantics

$$\mathcal{D} \Vdash \mathcal{P}$$

$$\frac{J \stackrel{\circ}{=}_{\sigma} \mathcal{J}}{D \wedge J \rhd P \wedge D' \Vdash \mathcal{J} \longrightarrow D \wedge J \rhd P \wedge D' \Vdash P\sigma} \textit{ R-BETA}$$

- $\stackrel{\circ}{=}$ states that J and $\mathcal{J}$ must have the same structure
- σ is the substitution of received variables of J with those of $\mathcal{J}$

Outline

A Secret-leaking Process

x is a secret of our code.

$$
\begin{array}{lll}
P = \textbf{def}\ o\langle a\rangle & & \triangleright\ a\langle\rangle \mid y\langle a\rangle \\
\quad \wedge & x\langle\rangle \mid y\langle b\rangle & \triangleright\ \texttt{export b} \\
\quad \textbf{in} & o\langle x\rangle &
\end{array}
$$

A Secret-leaking Process

x is a secret of our code.

$$
\begin{array}{lll}
P = \textbf{def}\ o\langle a\rangle & & \triangleright\ a\langle\rangle \mid y\langle a\rangle \\
\quad \wedge & x\langle\rangle \mid y\langle b\rangle & \triangleright\ \texttt{export b} \\
\quad \textbf{in} & o\langle x\rangle &
\end{array}
$$

$$\emptyset \Vdash P \ \underline{\quad S-DEF \quad}$$

A Secret-leaking Process

x is a secret of our code.

$$
\begin{array}{lll}
P = \textbf{def}\ o\langle a\rangle & & \rhd\ a\langle\rangle \mid y\langle a\rangle \\
\quad \wedge & x\langle\rangle \mid y\langle b\rangle & \rhd\ \texttt{export b} \\
\quad \textbf{in} & o\langle x\rangle &
\end{array}
$$

$$
\emptyset \Vdash P \ \frac{S-DEF}{}
$$

$$
o\langle a\rangle \rhd a\langle\rangle \mid y\langle a\rangle \wedge x\langle\rangle \mid y\langle b\rangle \rhd \texttt{export b} \ \Vdash o\langle x\rangle
$$

A Secret-leaking Process

x is a secret of our code.

$$
\begin{array}{lll}
P = \textbf{def}\ o\langle a\rangle & & \rhd\ a\langle\rangle \mid y\langle a\rangle \\
\quad \wedge & x\langle\rangle \mid y\langle b\rangle & \rhd\ \texttt{export b} \\
\quad \textbf{in} & o\langle x\rangle &
\end{array}
$$

$$
\emptyset \Vdash P \xrightarrow{S-DEF}
$$

$$
o\langle a\rangle \rhd a\langle\rangle \mid y\langle a\rangle \wedge x\langle\rangle \mid y\langle b\rangle \rhd \texttt{export b} \Vdash o\langle x\rangle \xrightarrow{R-BETA}
$$

A Secret-leaking Process

x is a secret of our code.

$$P = \textbf{def}\ o\langle a\rangle \quad \triangleright\ a\langle\rangle \mid y\langle a\rangle$$
$$\wedge\quad x\langle\rangle \mid y\langle b\rangle \triangleright \texttt{export b}$$
$$\textbf{in}\quad o\langle x\rangle$$

$$\emptyset \Vdash P \xrightarrow{S-DEF}$$
$$o\langle a\rangle \triangleright a\langle\rangle|y\langle a\rangle \wedge x\langle\rangle \mid y\langle b\rangle \triangleright \texttt{export b} \Vdash o\langle x\rangle \xrightarrow{R-BETA}$$
$$o\langle a\rangle \triangleright a\langle\rangle \mid y\langle a\rangle \wedge x\langle\rangle \mid y\langle b\rangle \triangleright \texttt{export b} \Vdash (a\langle\rangle|y\langle a\rangle)\ [x/a]$$

A Secret-leaking Process

x is a secret of our code.

$$P = \textbf{def}\ o\langle a\rangle \quad \triangleright\ a\langle\rangle \mid y\langle a\rangle$$
$$\wedge\quad x\langle\rangle \mid y\langle b\rangle \triangleright \texttt{export b}$$
$$\textbf{in}\quad o\langle x\rangle$$

$$\emptyset \Vdash P \xrightarrow{S-DEF}$$
$$o\langle a\rangle \triangleright a\langle\rangle | y\langle a\rangle \wedge x\langle\rangle \mid y\langle b\rangle \triangleright \texttt{export b} \Vdash o\langle x\rangle \xrightarrow{R-BETA}$$
$$o\langle a\rangle \triangleright a\langle\rangle \mid y\langle a\rangle \wedge x\langle\rangle \mid y\langle b\rangle \triangleright \texttt{export b} \Vdash x\langle\rangle | y\langle x\rangle$$

A Secret-leaking Process

x is a secret of our code.

$$
\begin{array}{lll}
P = \textbf{def}\ o\langle a\rangle & & \rhd\ a\langle\rangle \mid y\langle a\rangle \\
\quad \wedge & x\langle\rangle \mid y\langle b\rangle & \rhd\ \texttt{export b} \\
\quad \textbf{in} & o\langle x\rangle &
\end{array}
$$

$$\emptyset \Vdash P \xrightarrow{S-DEF}$$

$$o\langle a\rangle \rhd a\langle\rangle \mid y\langle a\rangle \wedge x\langle\rangle \mid y\langle b\rangle \rhd \texttt{export b} \Vdash o\langle x\rangle \xrightarrow{R-BETA}$$

$$o\langle a\rangle \rhd a\langle\rangle \mid y\langle a\rangle \wedge x\langle\rangle \mid y\langle b\rangle \rhd \texttt{export b} \Vdash x\langle\rangle \mid y\langle x\rangle$$

$$o\langle a\rangle \rhd a\langle\rangle \mid y\langle a\rangle \wedge \boxed{x\langle\rangle \mid y\langle b\rangle} \rhd \texttt{export b} \Vdash \boxed{x\langle\rangle \mid y\langle x\rangle} \xrightarrow{R-BETA}$$

A Secret-leaking Process

x is a secret of our code.

$$
\begin{array}{lll}
P = \textbf{def}\ o\langle a\rangle & & \rhd\ a\langle\rangle \mid y\langle a\rangle \\
\quad \wedge & x\langle\rangle \mid y\langle b\rangle & \rhd\ \texttt{export b} \\
\quad \textbf{in} & o\langle x\rangle &
\end{array}
$$

$$\emptyset \Vdash P \xrightarrow{S-DEF}$$

$$o\langle a\rangle \rhd a\langle\rangle \mid y\langle a\rangle \wedge x\langle\rangle \mid y\langle b\rangle \rhd \texttt{export b} \Vdash o\langle x\rangle \xrightarrow{R-BETA}$$

$$o\langle a\rangle \rhd a\langle\rangle \mid y\langle a\rangle \wedge x\langle\rangle \mid y\langle b\rangle \rhd \texttt{export b} \Vdash x\langle\rangle \mid y\langle x\rangle$$

$$o\langle a\rangle \rhd a\langle\rangle \mid y\langle a\rangle \wedge \boxed{x\langle\rangle \mid y\langle b\rangle} \rhd \texttt{export b} \Vdash \boxed{x\langle\rangle \mid y\langle x\rangle} \xrightarrow{R-BETA}$$

$$o\langle a\rangle \rhd a\langle\rangle \mid y\langle a\rangle \wedge x\langle\rangle \mid y\langle b\rangle \rhd \texttt{export b} \Vdash (\texttt{export b})[x/b]$$

A Secret-leaking Process

x is a secret of our code.

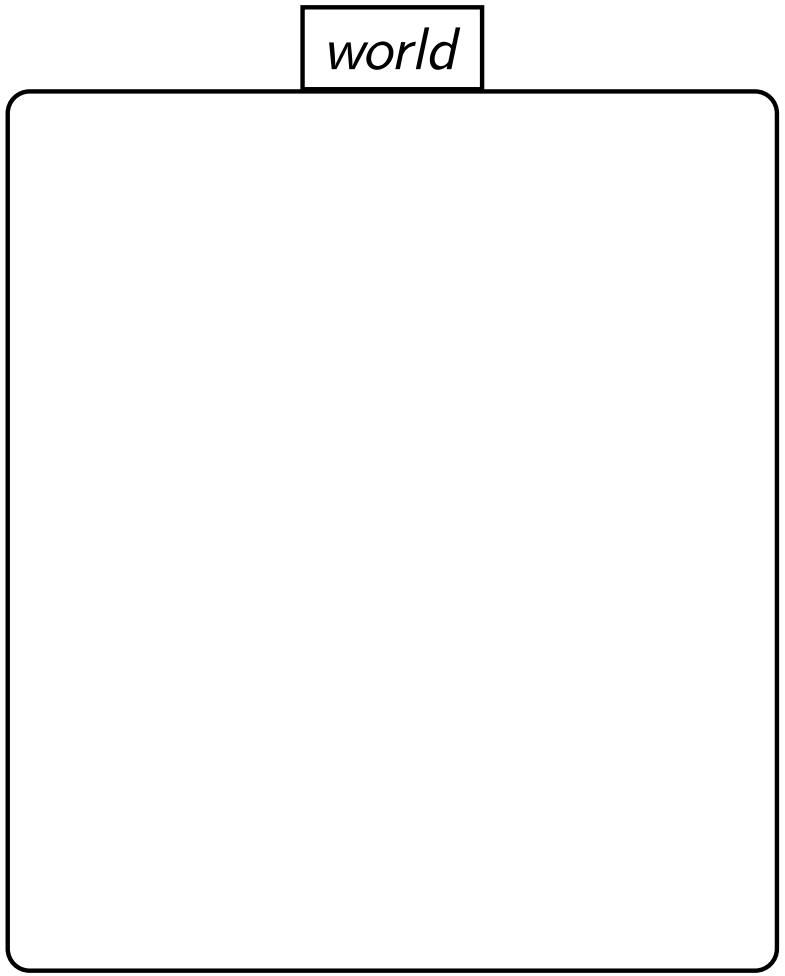
$$P = \textbf{def}\ o\langle a\rangle \quad \triangleright\ a\langle\rangle \mid y\langle a\rangle$$
$$\wedge\quad x\langle\rangle \mid y\langle b\rangle \ \triangleright\ \texttt{export b}$$
$$\textbf{in}\quad o\langle x\rangle$$

$$\emptyset \Vdash P \xrightarrow{S-DEF}$$
$$o\langle a\rangle \triangleright a\langle\rangle \mid y\langle a\rangle \wedge x\langle\rangle \mid y\langle b\rangle \triangleright \texttt{export b} \Vdash o\langle x\rangle \xrightarrow{R-BETA}$$
$$o\langle a\rangle \triangleright a\langle\rangle \mid y\langle a\rangle \wedge x\langle\rangle \mid y\langle b\rangle \triangleright \texttt{export b} \Vdash x\langle\rangle \mid y\langle x\rangle$$
$$o\langle a\rangle \triangleright a\langle\rangle \mid y\langle a\rangle \wedge \boxed{x\langle\rangle \mid y\langle b\rangle} \triangleright \texttt{export b} \Vdash \boxed{x\langle\rangle \mid y\langle x\rangle} \xrightarrow{R-BETA}$$
$$o\langle a\rangle \triangleright a\langle\rangle \mid y\langle a\rangle \wedge x\langle\rangle \mid y\langle b\rangle \triangleright \texttt{export b} \Vdash \texttt{export x}$$

Outline

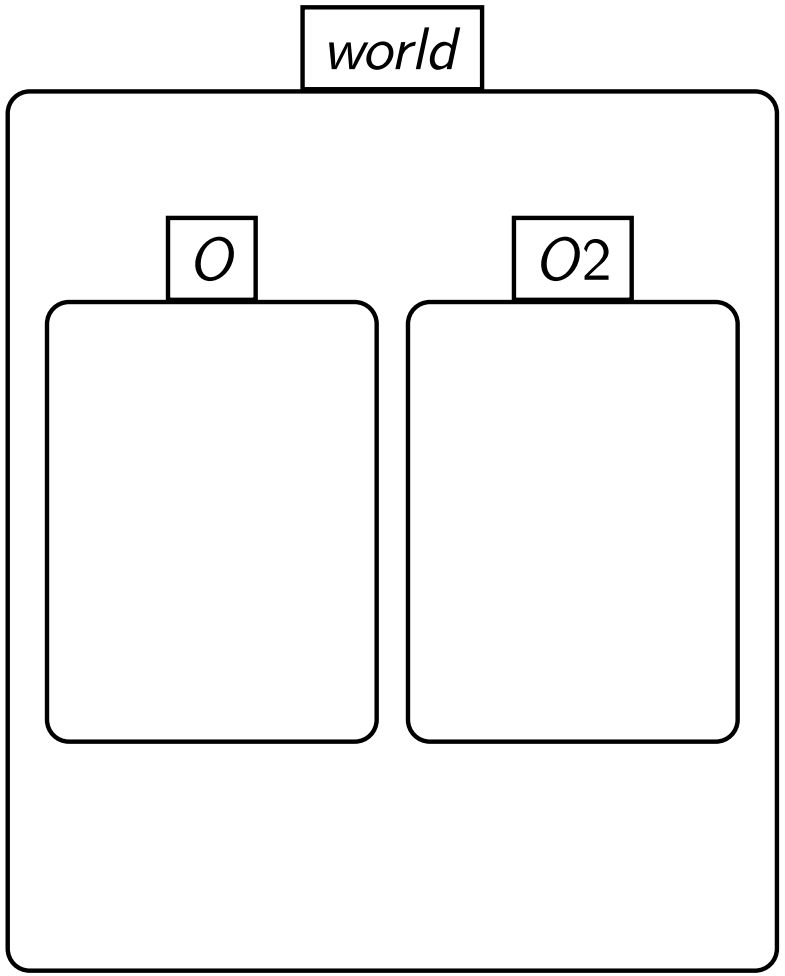
Ownership Types

- The system has a general box called *world*.



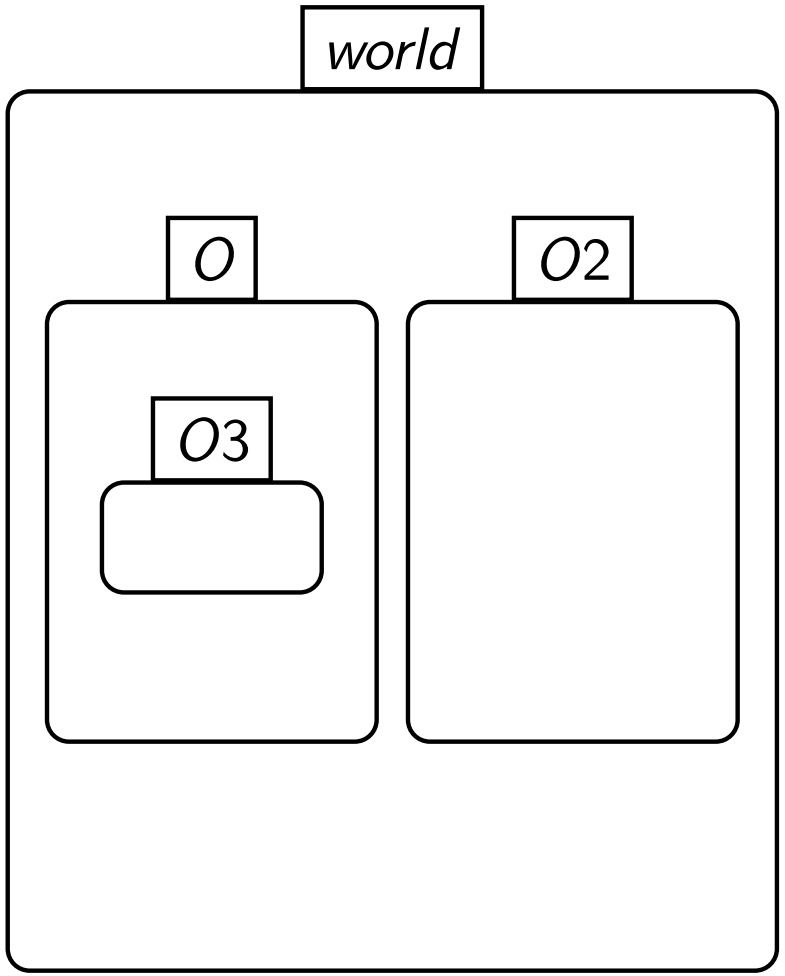
Ownership Types

- The system has a general box called *world*.
- Every object *O* implicitly defines a box, which is its representation.



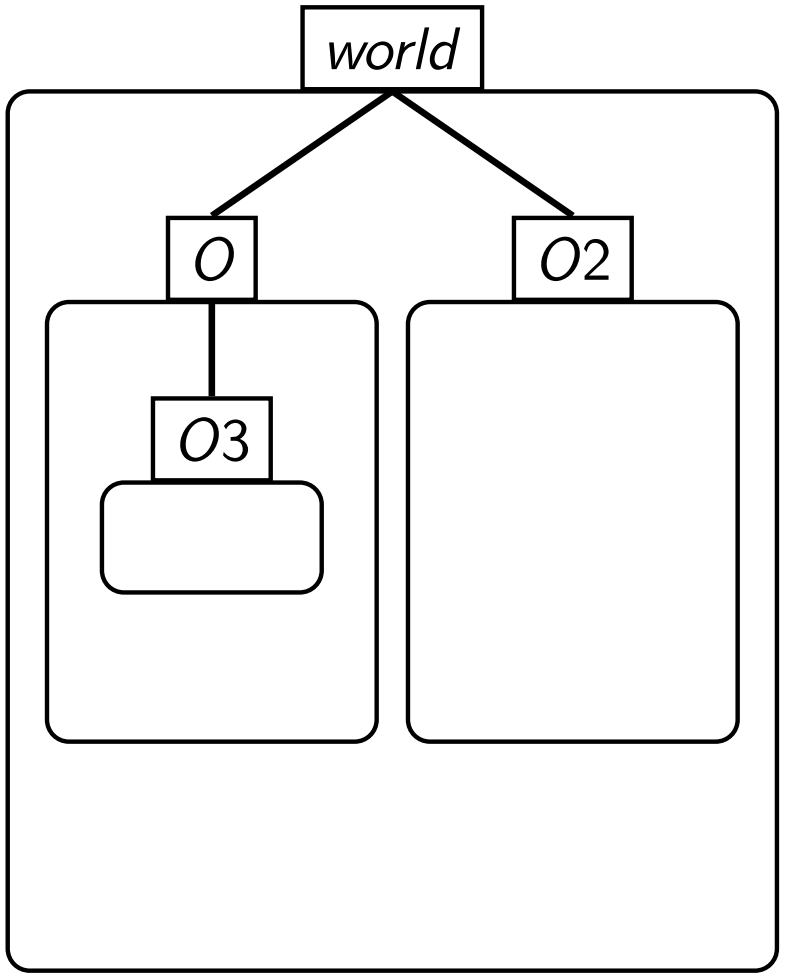
Ownership Types

- The system has a general box called *world*.
- Every object *O* implicitly defines a box, which is its representation.
- Every object *O* must specify in which box it must be placed: *O*'s owner.



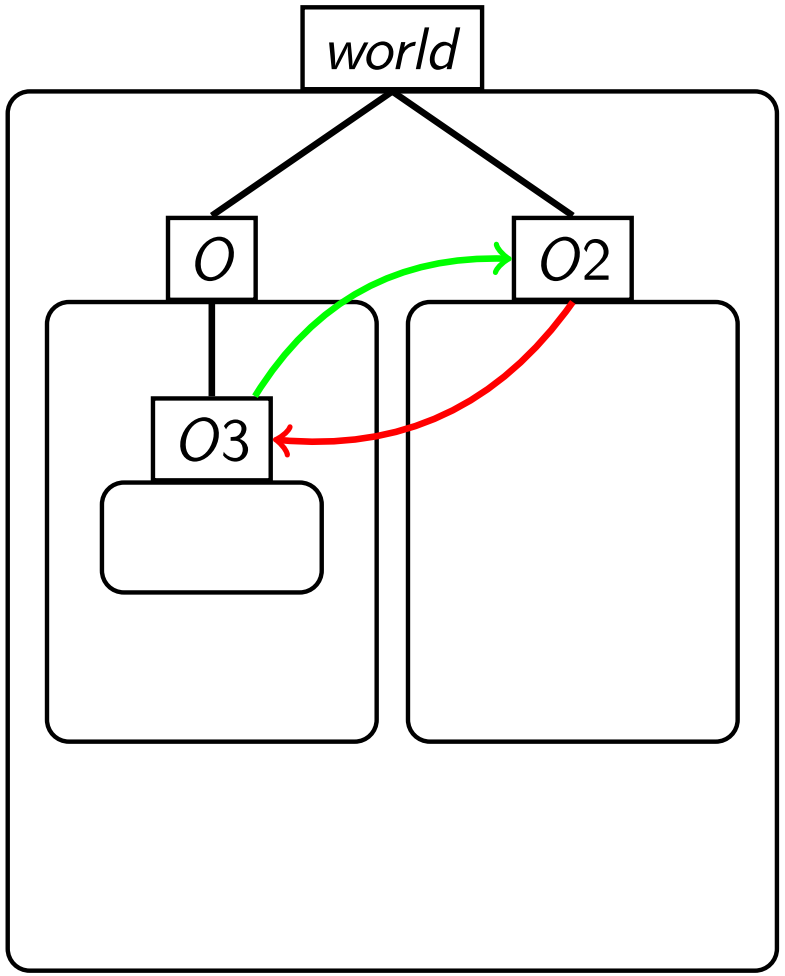
Ownership Types

- The system has a general box called *world*.
- Every object O implicitly defines a box, which is its representation.
- Every object O must specify in which box it must be placed: O's owner.
- The nest of boxes specify the ownership tree.



Ownership Types

- The system has a general box called *world*.
- Every object O implicitly defines a box, which is its representation.
- Every object O must specify in which box it must be placed: O's owner.
- The nest of boxes specify the ownership tree.
- **Owners-as-dominators property**: references cannot cross the box boundaries from outside to inside.



Outline

A Typed Join Calculus: J_{OT}

$$J = J \mid J \qquad T = \begin{matrix} o \\ o \end{matrix} \langle\!| \overline{t} |\!\rangle \qquad o = x$$

$$\mid \; x\langle \overline{y} \rangle : \mathsf{T} \qquad\qquad\qquad\qquad \mid \; \mathit{world}$$

$$t = \exists \alpha . T \qquad\qquad \mid \; \alpha$$

Types keep track of:

- o: the owner.
- o: the representation.
- $\overline{t}$: the types of parameters $\overline{y}$.

Existential types provide flexibility, they are implicitly used.

Outline

A Secrecy-preserving Process

x is a secret of our code.

$$
\begin{array}{lll}
P & = \textbf{def}\ o\langle a\rangle & \triangleright\ a\langle\rangle \mid y\langle a\rangle \\
& \quad \wedge\ \ x\langle\rangle \qquad\quad \mid y\langle b\rangle & \triangleright\ \texttt{export b} \\
& \quad \textbf{in}\ \ o\langle x\rangle &
\end{array}
$$

A Secrecy-preserving Process

x is a secret of our code.

$$P_T = \textbf{def}\ o\langle a\rangle : \boxed{{}^{world}_{o}\langle\!|\exists\alpha.\ {}^{o}_{\alpha}\langle\!|\,|\!\rangle\,|\!\rangle} \quad \triangleright\ a\langle\rangle \mid y\langle a\rangle$$

$$\wedge\quad x\langle\rangle : \boxed{{}^{o}_{x}\langle\!|\,|\!\rangle} \mid y\langle b\rangle : \boxed{{}^{world}_{y}\langle\!|\exists\alpha.\ {}^{world}_{\alpha}\langle\!|\,|\!\rangle\,|\!\rangle} \quad \triangleright\ \texttt{export b}$$

$$\textbf{in}\quad o\langle x\rangle$$

A Secrecy-preserving Process

x is a secret of our code.

$$P_T = \textbf{def}\ o\langle a\rangle : \boxed{{}^{world}_{o}\langle\!|\exists\alpha.\ {}^{o}_{\alpha}\langle\!|\,|\!\rangle\,|\!\rangle} \quad \triangleright\ a\langle\rangle \mid y\langle a\rangle$$

$$\wedge\quad x\langle\rangle : \boxed{{}^{o}_{x}\langle\!|\,|\!\rangle} \mid y\langle b\rangle : \boxed{{}^{world}_{y}\langle\!|\exists\alpha.\ {}^{world}_{\alpha}\langle\!|\,|\!\rangle\,|\!\rangle} \quad \triangleright\ \texttt{export b}$$

$$\textbf{in}\quad o\langle x\rangle$$

WRONG!! ILL-TYPED!!

A Secrecy-preserving Process

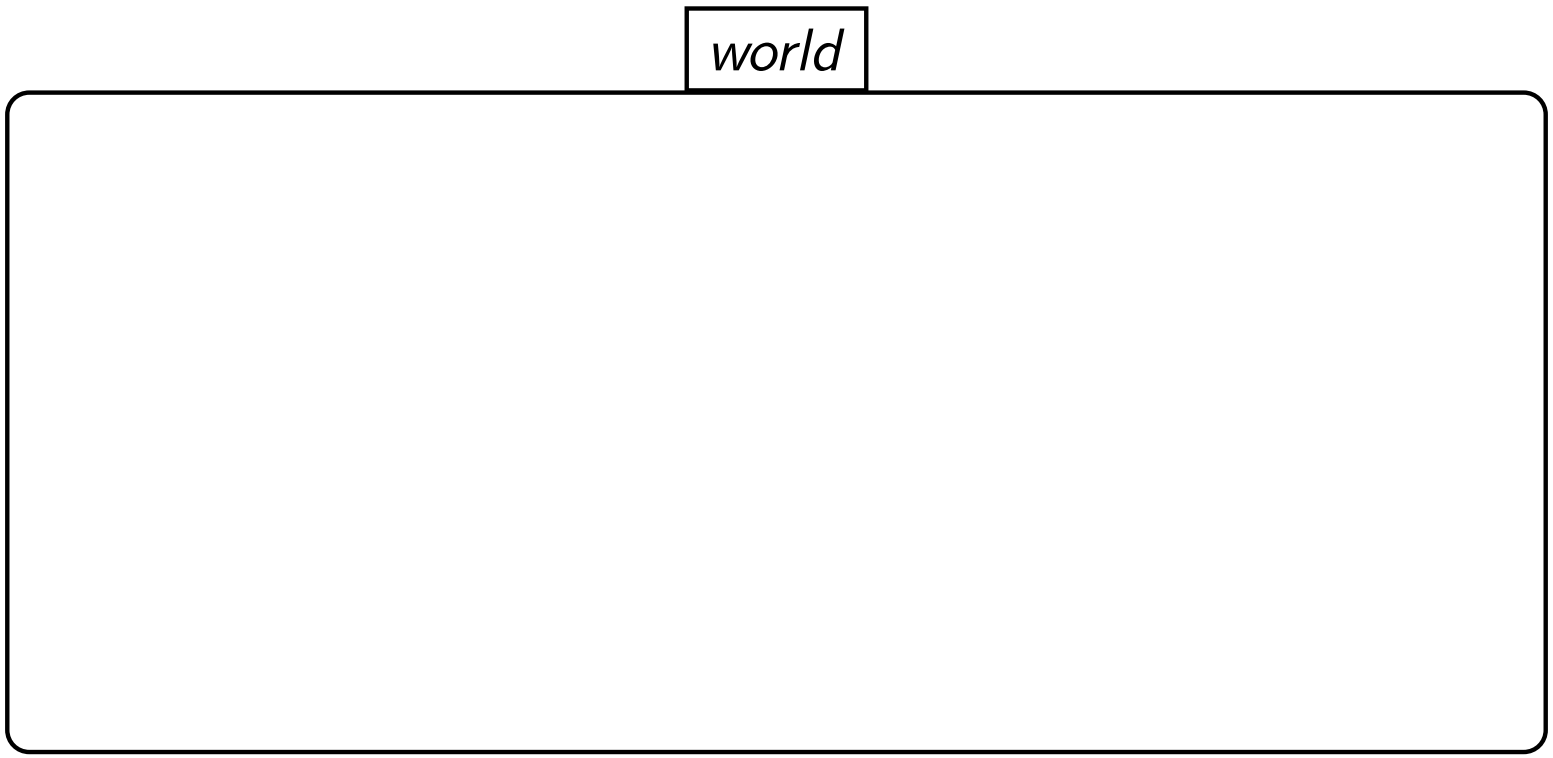
x is a secret of our code.

$$P_T = \textbf{def}\ o\langle a\rangle : \boxed{{}^{world}_{o}\langle\!|\exists\alpha.\ {}^{o}_{\alpha}\langle\!|\,|\!\rangle\,|\!\rangle} \quad \triangleright\ a\langle\rangle \mid y\langle a\rangle$$

$$\wedge\quad x\langle\rangle : \boxed{{}^{o}_{x}\langle\!|\,|\!\rangle} \mid y\langle b\rangle : \boxed{{}^{world}_{y}\langle\!|\exists\alpha.\ {}^{world}_{\alpha}\langle\!|\,|\!\rangle\,|\!\rangle} \quad \triangleright\ \texttt{export b}$$

$$\textbf{in}\quad o\langle x\rangle$$

WRONG!! ILL-TYPED!!



A Secrecy-preserving Process

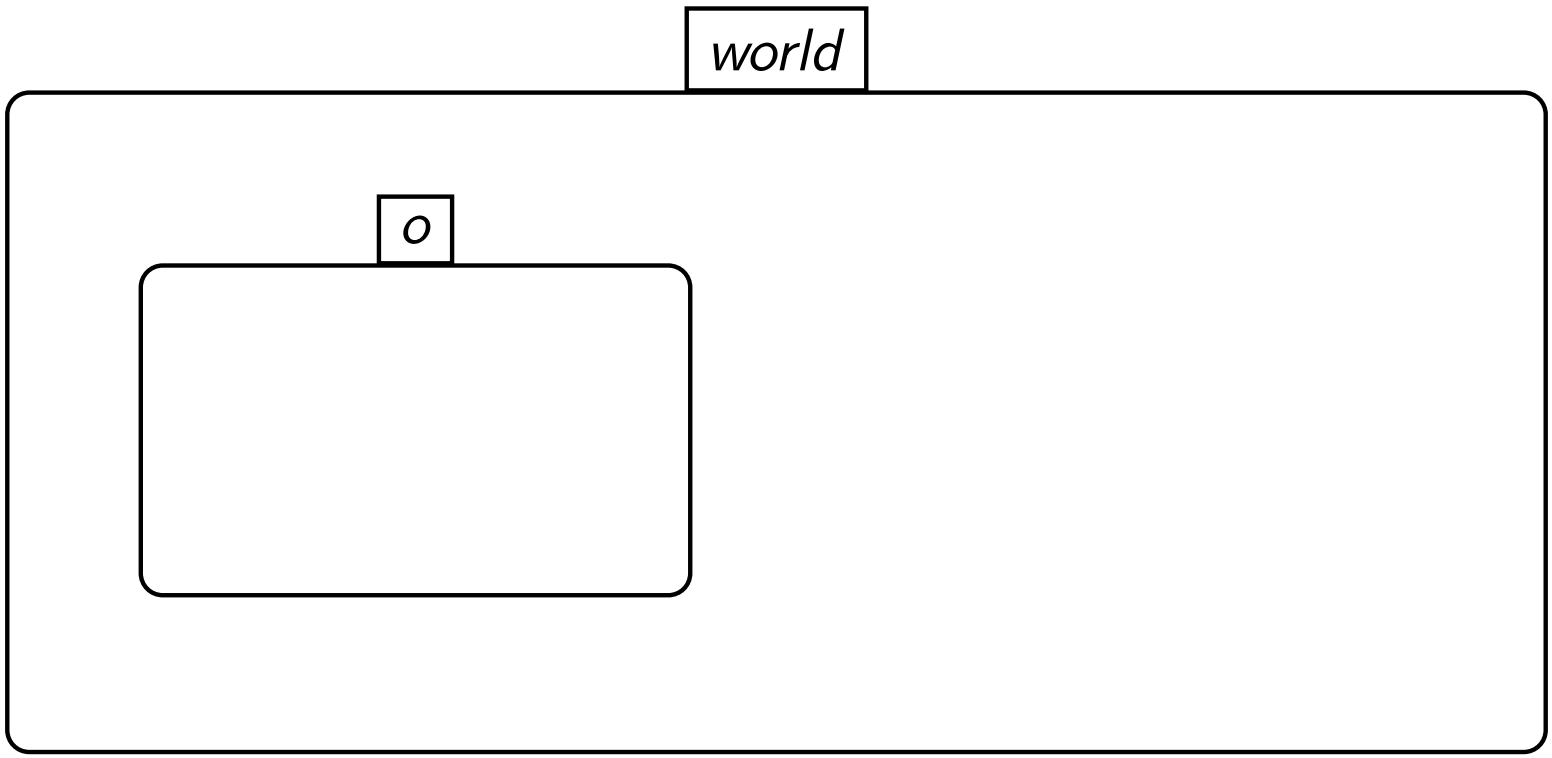
x is a secret of our code.

$$P_T = \textbf{def}\ o\langle a\rangle : \boxed{{}^{world}_{o}\langle\!|\exists\alpha.\ {}^{o}_{\alpha}\langle\!|\,|\!\rangle\,|\!\rangle} \quad \triangleright\ a\langle\rangle \mid y\langle a\rangle$$

$$\wedge\quad x\langle\rangle : \boxed{{}^{o}_{x}\langle\!|\,|\!\rangle} \mid y\langle b\rangle : \boxed{{}^{world}_{y}\langle\!|\exists\alpha.\ {}^{world}_{\alpha}\langle\!|\,|\!\rangle\,|\!\rangle} \quad \triangleright\ \texttt{export b}$$

$$\textbf{in}\quad o\langle x\rangle$$

WRONG!! ILL-TYPED!!



A Secrecy-preserving Process

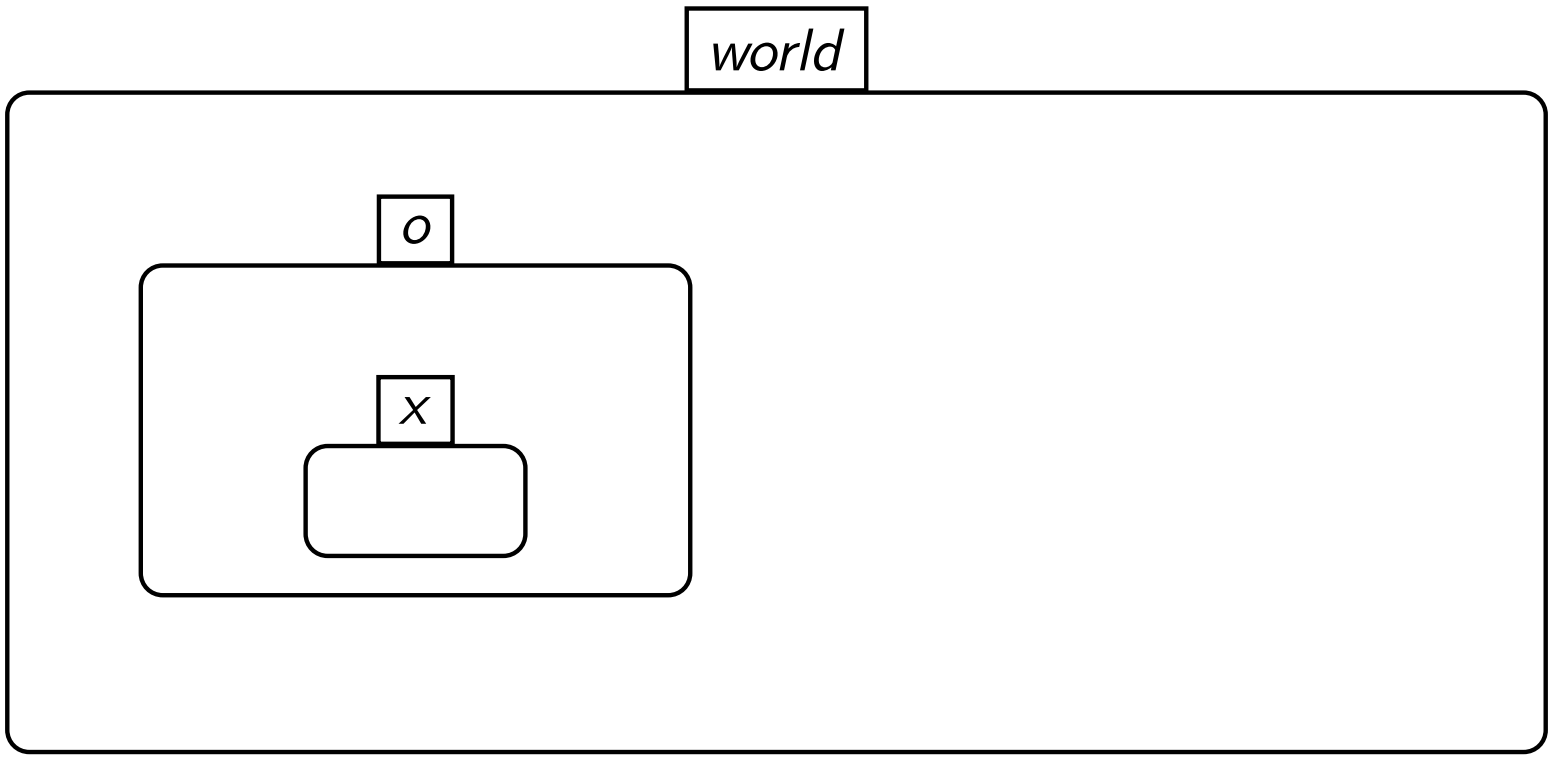
x is a secret of our code.

$$P_T = \textbf{def}\ o\langle a\rangle : \boxed{{}^{world}_{o}\langle\!|\exists\alpha.\ {}^{o}_{\alpha}\langle\!|\,|\!\rangle\,|\!\rangle} \quad \triangleright\ a\langle\rangle \mid y\langle a\rangle$$

$$\wedge\quad x\langle\rangle : \boxed{{}^{o}_{x}\langle\!|\,|\!\rangle} \mid y\langle b\rangle : \boxed{{}^{world}_{y}\langle\!|\exists\alpha.\ {}^{world}_{\alpha}\langle\!|\,|\!\rangle\,|\!\rangle} \quad \triangleright\ \texttt{export b}$$

$$\textbf{in}\quad o\langle x\rangle$$

WRONG!! ILL-TYPED!!



A Secrecy-preserving Process

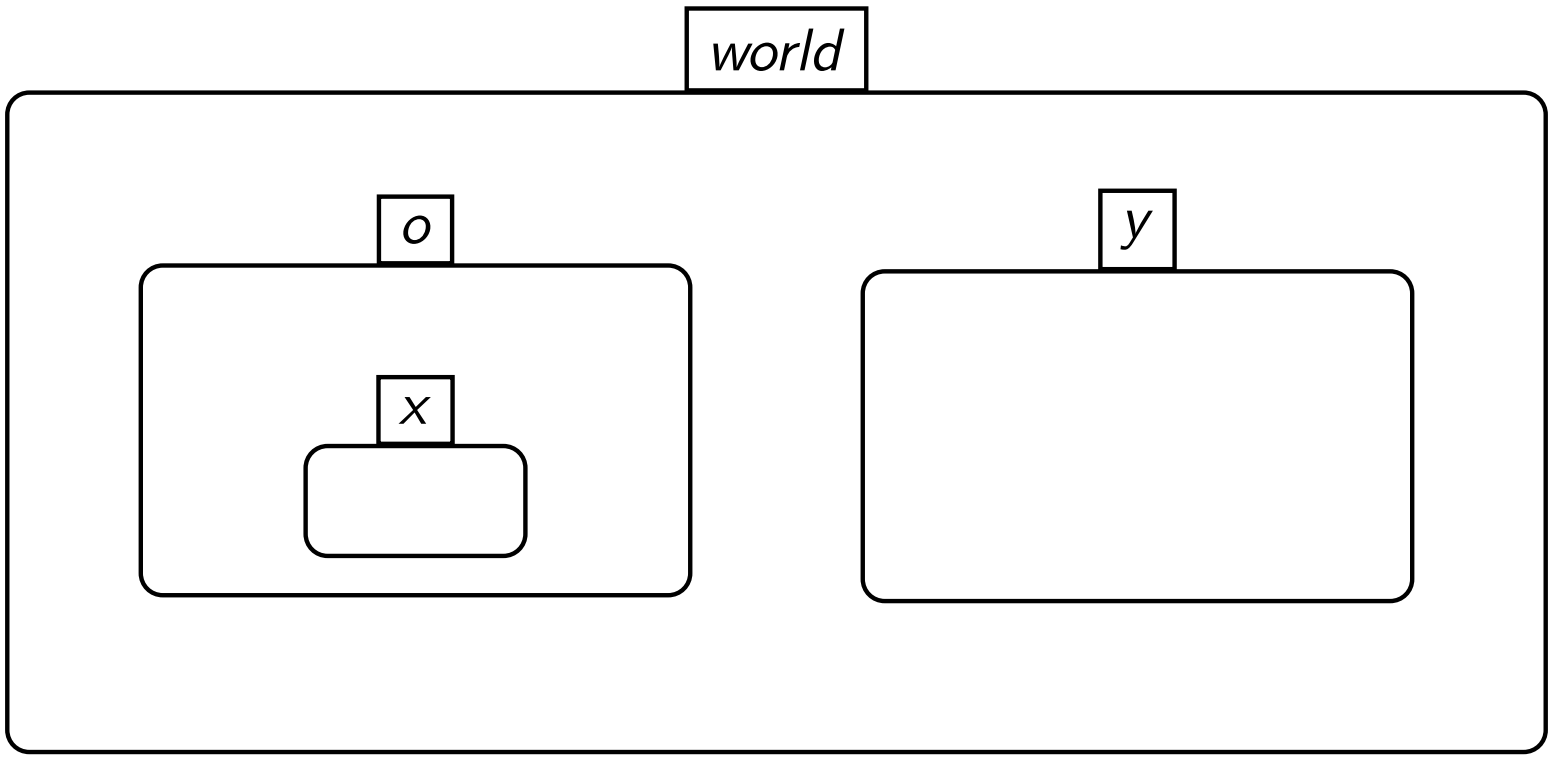
x is a secret of our code.

$$P_T = \textbf{def}\ o\langle a\rangle : \boxed{{}^{world}_{o}\langle\!|\exists\alpha.\ {}^{o}_{\alpha}\langle\!|\,|\!\rangle\,|\!\rangle} \quad \triangleright\ a\langle\rangle \mid y\langle a\rangle$$

$$\wedge\quad x\langle\rangle : \boxed{{}^{o}_{x}\langle\!|\,|\!\rangle} \mid y\langle b\rangle : \boxed{{}^{world}_{y}\langle\!|\exists\alpha.\ {}^{world}_{\alpha}\langle\!|\,|\!\rangle\,|\!\rangle} \quad \triangleright\ \texttt{export b}$$

$$\textbf{in}\quad o\langle x\rangle$$

WRONG!! ILL-TYPED!!



A Secrecy-preserving Process

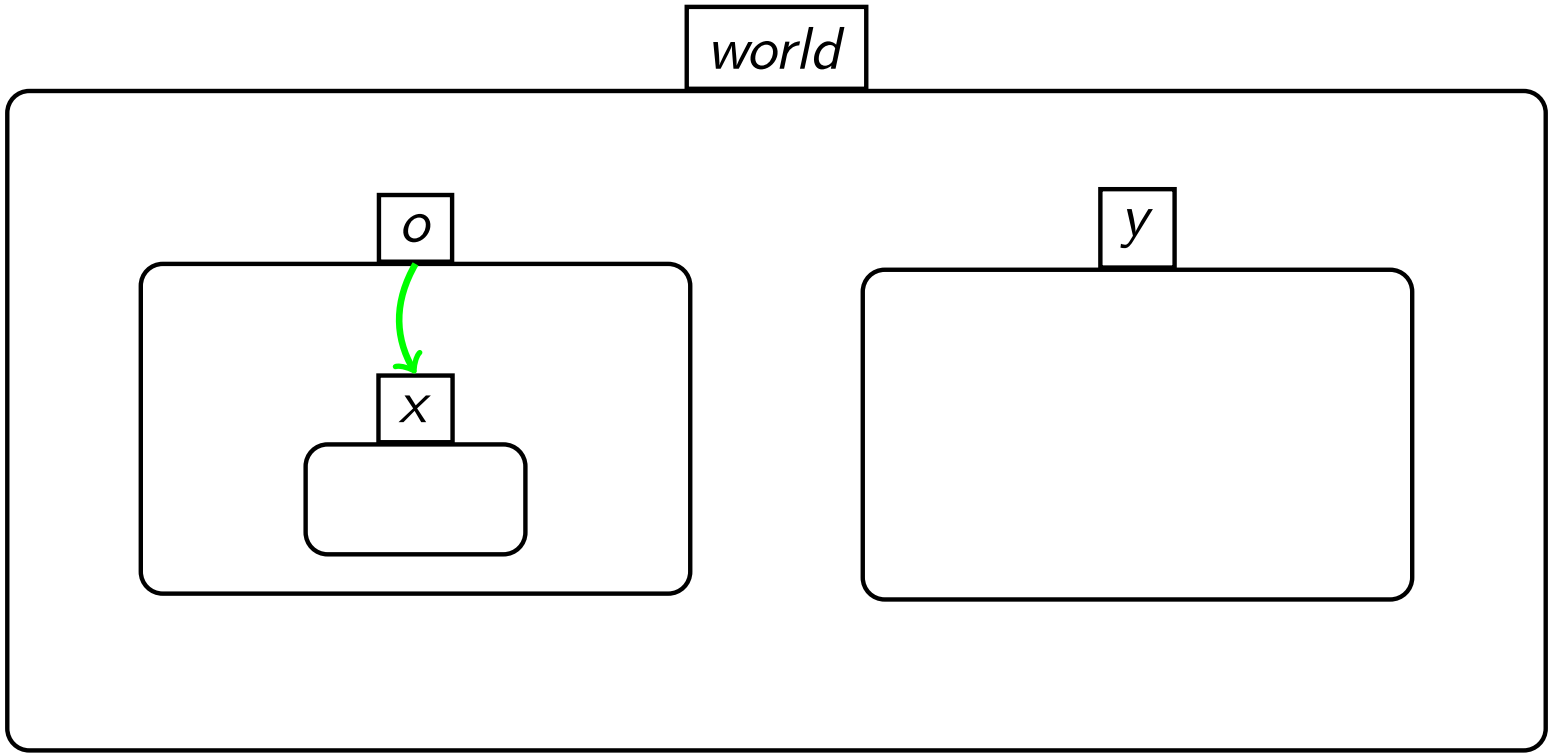
x is a secret of our code.

$$P_T = \textbf{def}\ o\langle a\rangle : \boxed{{}^{world}_{o}\langle\!|\exists\alpha.\ {}^{o}_{\alpha}\langle\!|\,|\!\rangle\,|\!\rangle} \triangleright\ a\langle\rangle \mid y\langle a\rangle$$

$$\wedge\quad x\langle\rangle : \boxed{{}^{o}_{x}\langle\!|\,|\!\rangle} \mid y\langle b\rangle : \boxed{{}^{world}_{y}\langle\!|\exists\alpha.\ {}^{world}_{\alpha}\langle\!|\,|\!\rangle\,|\!\rangle} \triangleright\ \texttt{export b}$$

$$\textbf{in}\quad o\langle x\rangle$$

WRONG!! ILL-TYPED!!



A Secrecy-preserving Process

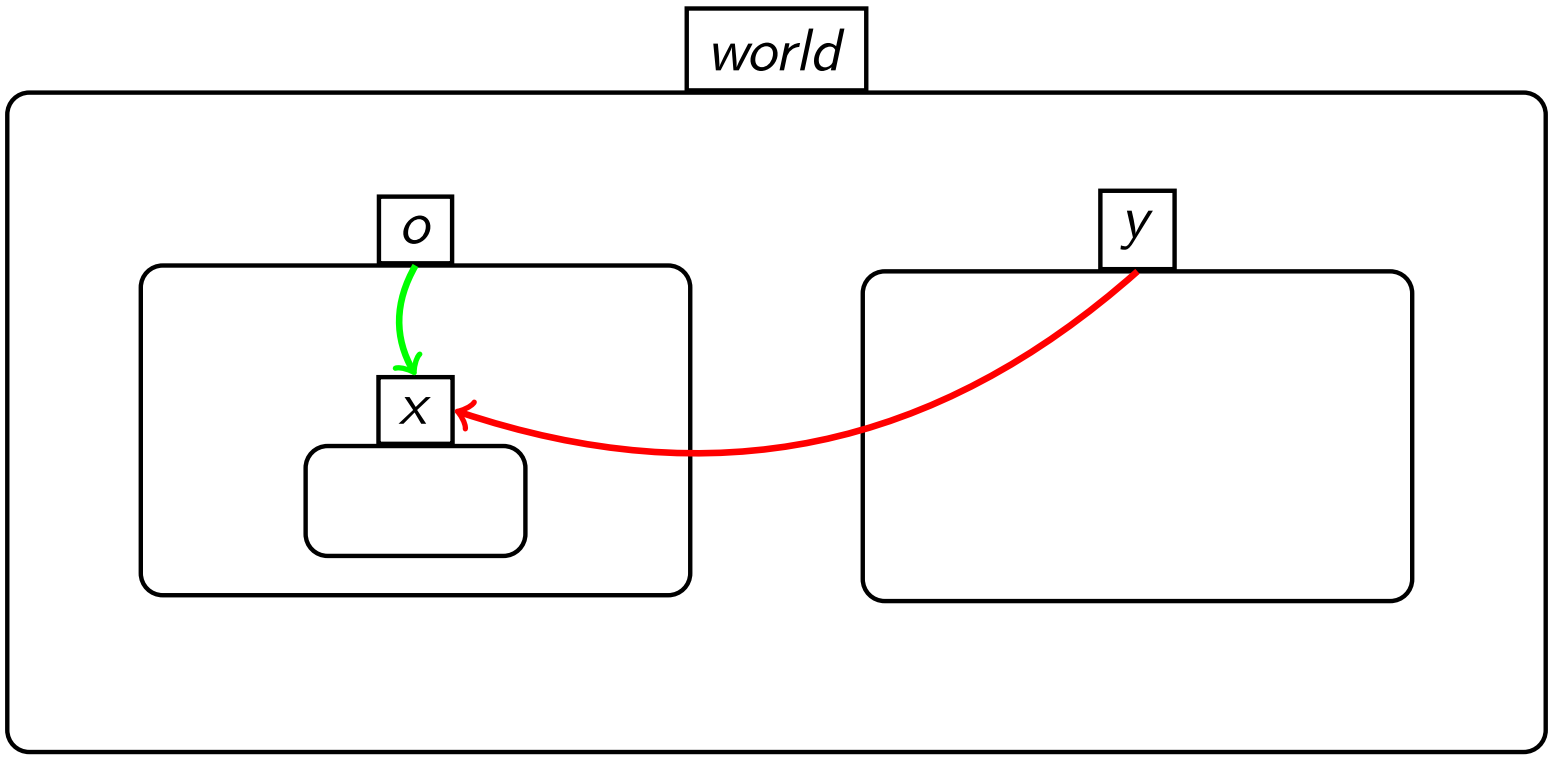
x is a secret of our code.

$$P_T = \textbf{def}\ o\langle a\rangle : \overset{world}{o}\langle\!|\exists\alpha.\ \overset{o}{\alpha}\langle\!|\,|\!\rangle\,|\!\rangle \quad \triangleright\ a\langle\rangle \mid y\langle a\rangle$$

$$\wedge\quad x\langle\rangle : \overset{o}{x}\langle\!|\,|\!\rangle \mid y\langle b\rangle : \overset{world}{y}\langle\!|\exists\alpha.\ \overset{world}{\alpha}\langle\!|\,|\!\rangle\,|\!\rangle \quad \triangleright\ \texttt{export b}$$

$$\textbf{in}\quad o\langle x\rangle$$

WRONG!! ILL-TYPED!!



Outline

An Untyped Opponent

- A malicious and cheating entity is modeled as an untyped opponent.

[1] Luca Cardelli, Giorgio Ghelli, and Andrew D. Gordon. Secrecy and group creation. *Inf. Comput.*, 196(2):127–155, 2005.

An Untyped Opponent

- A malicious and cheating entity is modeled as an untyped opponent.
- We prove that after erasing type annotation from a well typed process P_O, it can interact with an untyped one P_U and preserve the secrets.

[1] Luca Cardelli, Giorgio Ghelli, and Andrew D. Gordon. Secrecy and group creation. *Inf. Comput.*, 196(2):127–155, 2005.

An Untyped Opponent

- A malicious and cheating entity is modeled as an untyped opponent.
- We prove that after erasing type annotation from a well typed process P_O, it can interact with an untyped one P_U and preserve the secrets.
- The approach presented here is inspired by the work of Cardelli et al.[1].

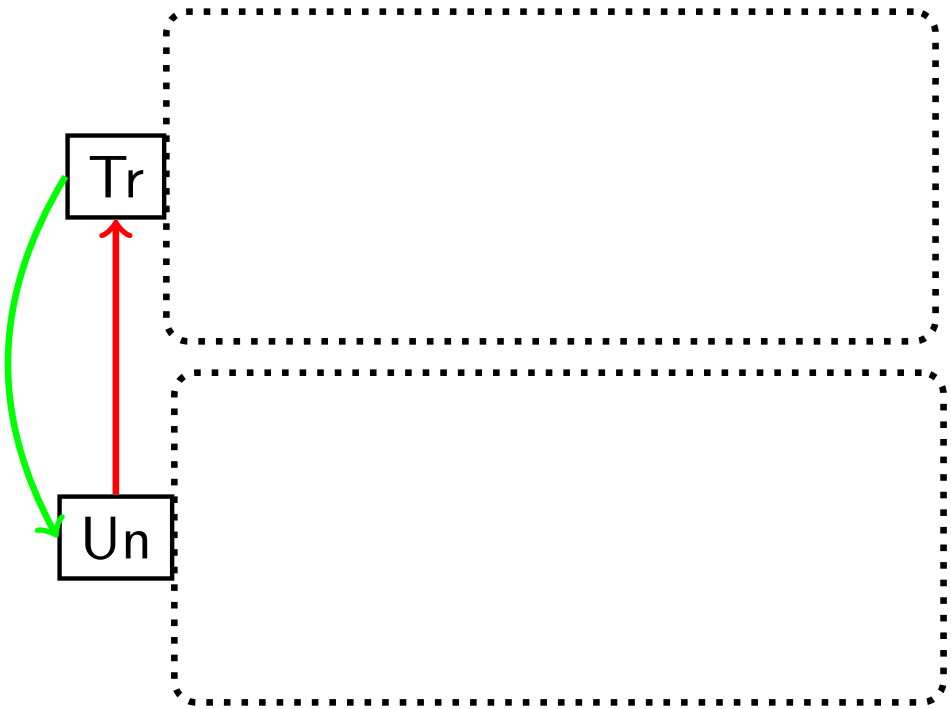
[1] Luca Cardelli, Giorgio Ghelli, and Andrew D. Gordon. Secrecy and group creation. *Inf. Comput.*, 196(2):127–155, 2005.

An Untyped Opponent

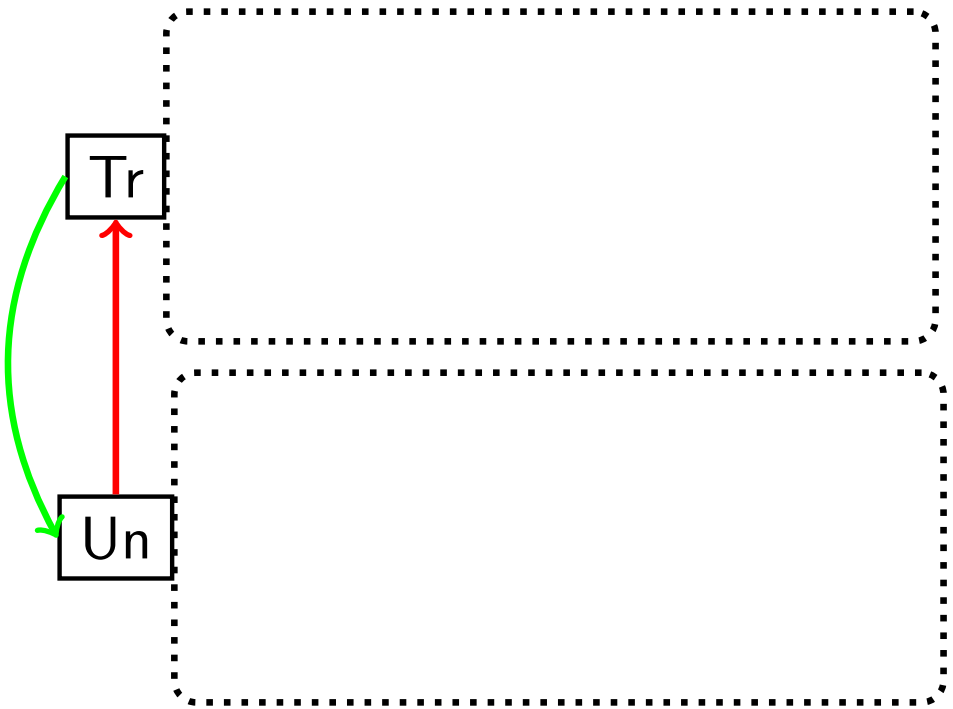
- A malicious and cheating entity is modeled as an untyped opponent.
- We prove that after erasing type annotation from a well typed process P_O, it can interact with an untyped one P_U and preserve the secrets.
- The approach presented here is inspired by the work of Cardelli et al.[1].
- Now i provide a quick sketch.

[1] Luca Cardelli, Giorgio Ghelli, and Andrew D. Gordon. Secrecy and group creation. *Inf. Comput.*, 196(2):127–155, 2005.

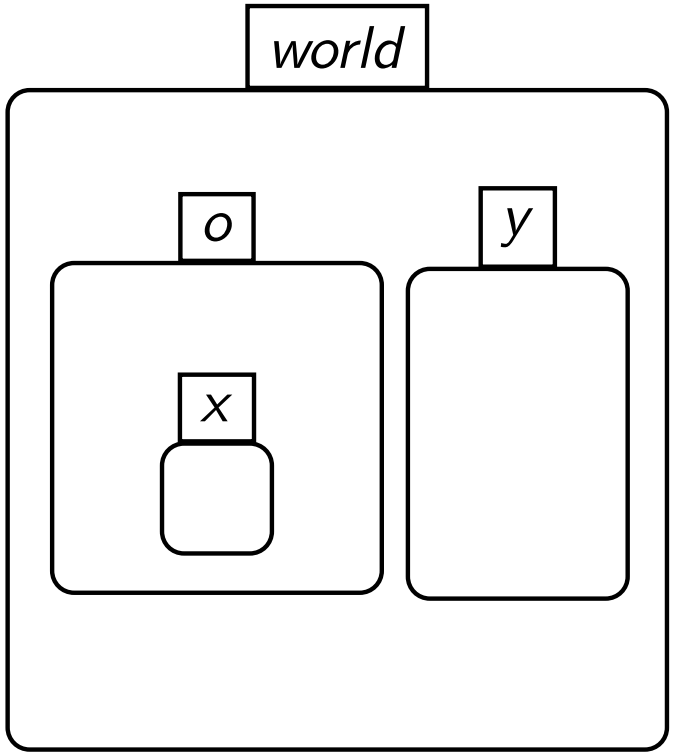
An Untyped Opponent



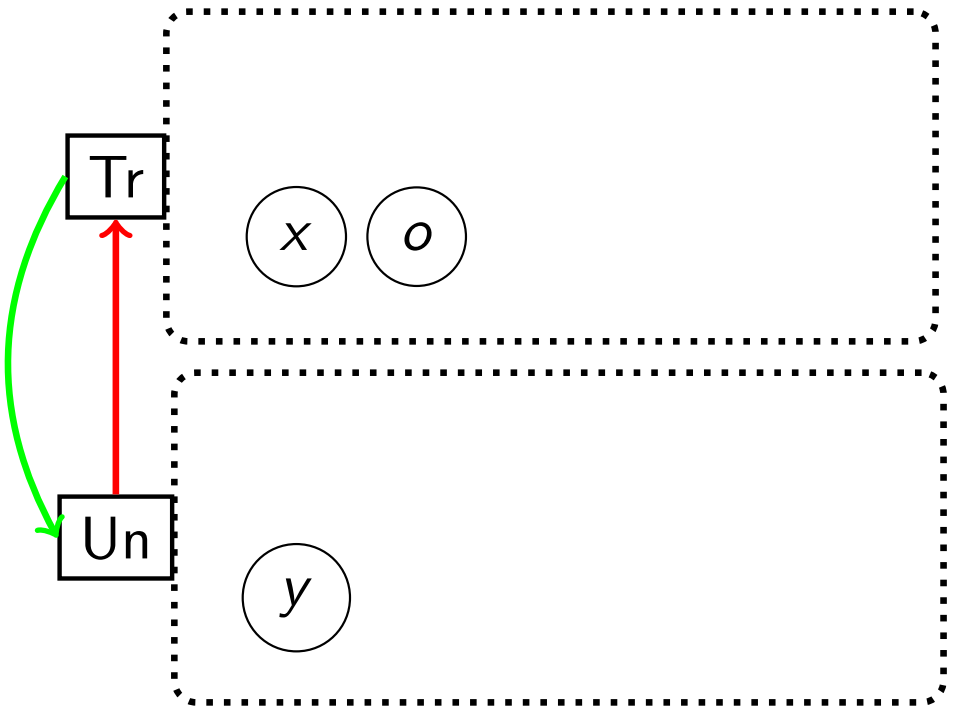
An Untyped Opponent

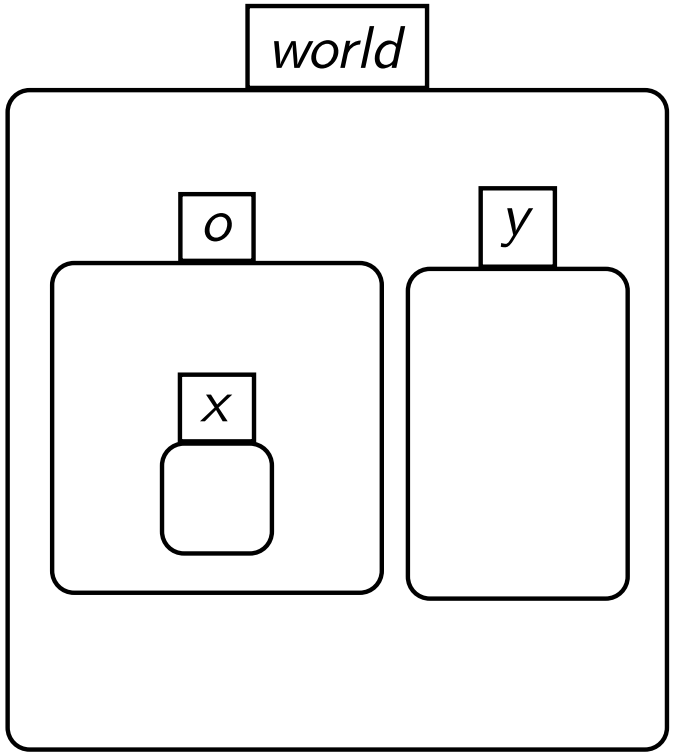


$\Gamma_o \vdash_o P_o$



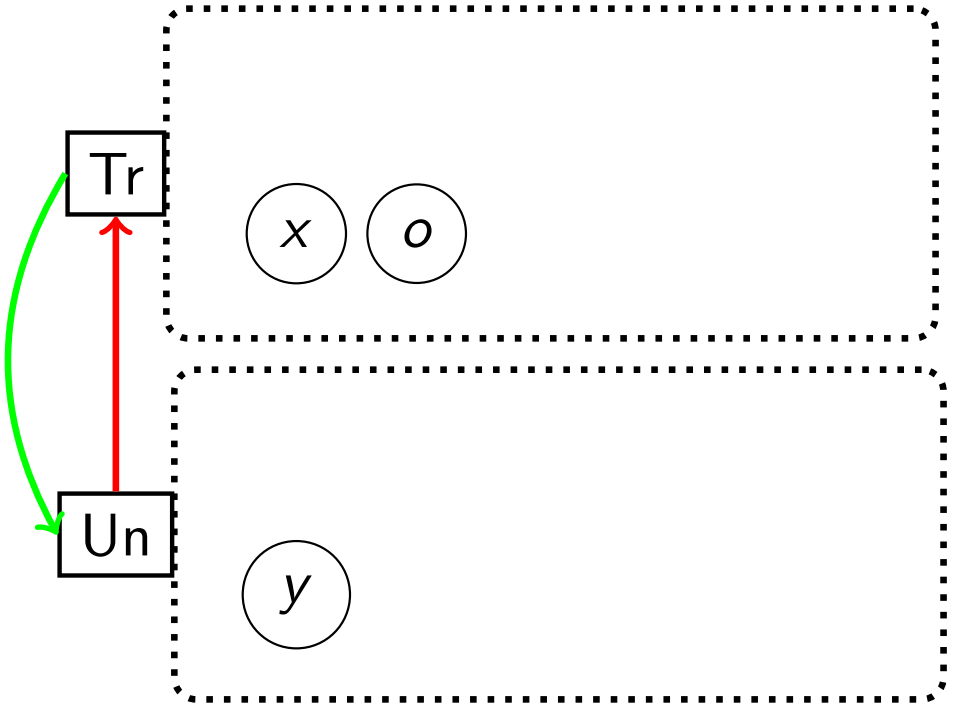
An Untyped Opponent

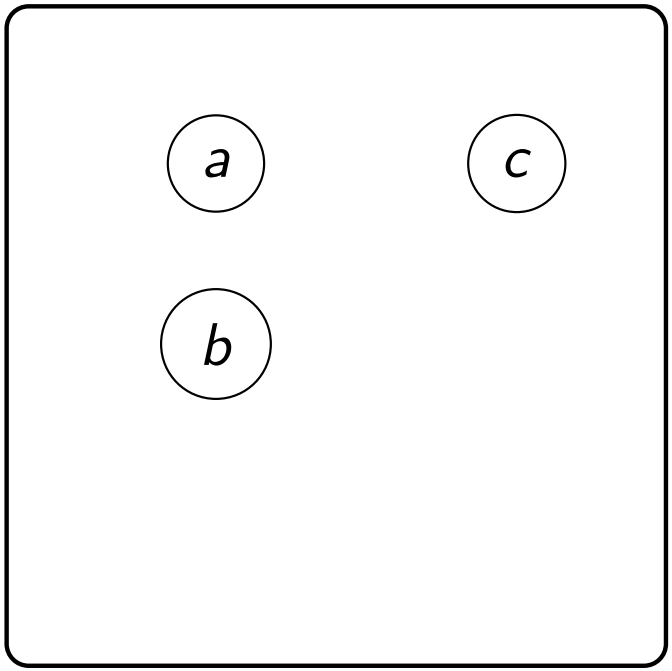




$\Gamma_O \vdash_O P_O \rightarrow \Gamma_{OA} \vdash_A P_{OA}$

An Untyped Opponent

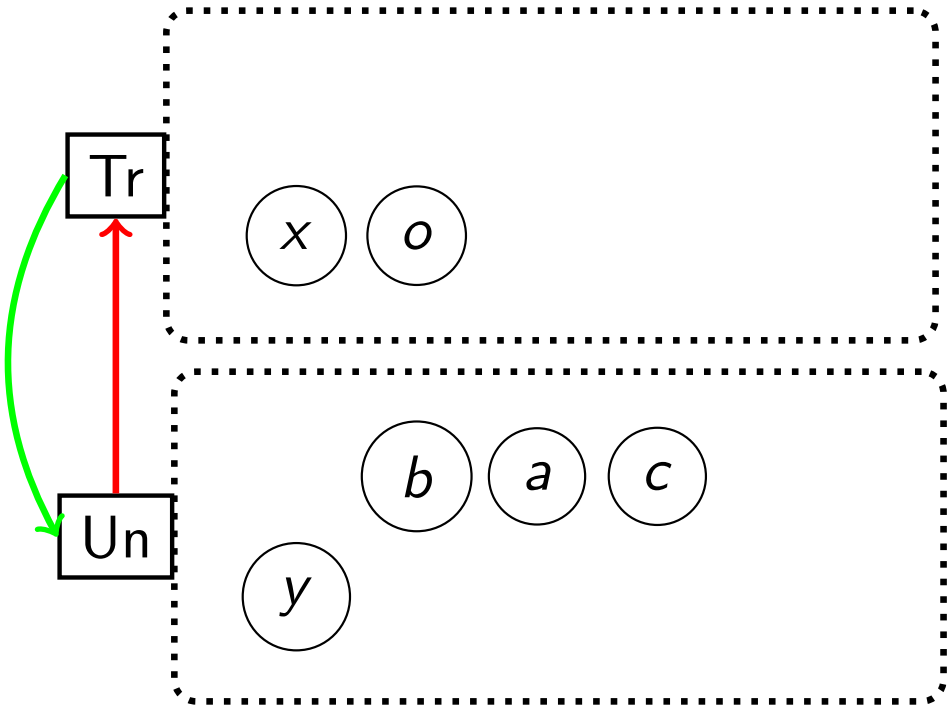


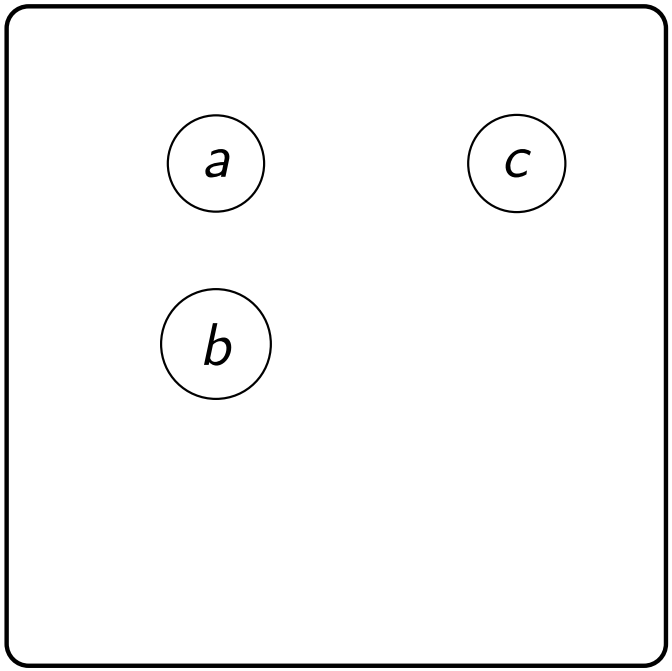


$\Gamma_O \vdash_O P_O \rightarrow \Gamma_{OA} \vdash_A P_{OA}$

$\Gamma_U \vdash_U P_U$

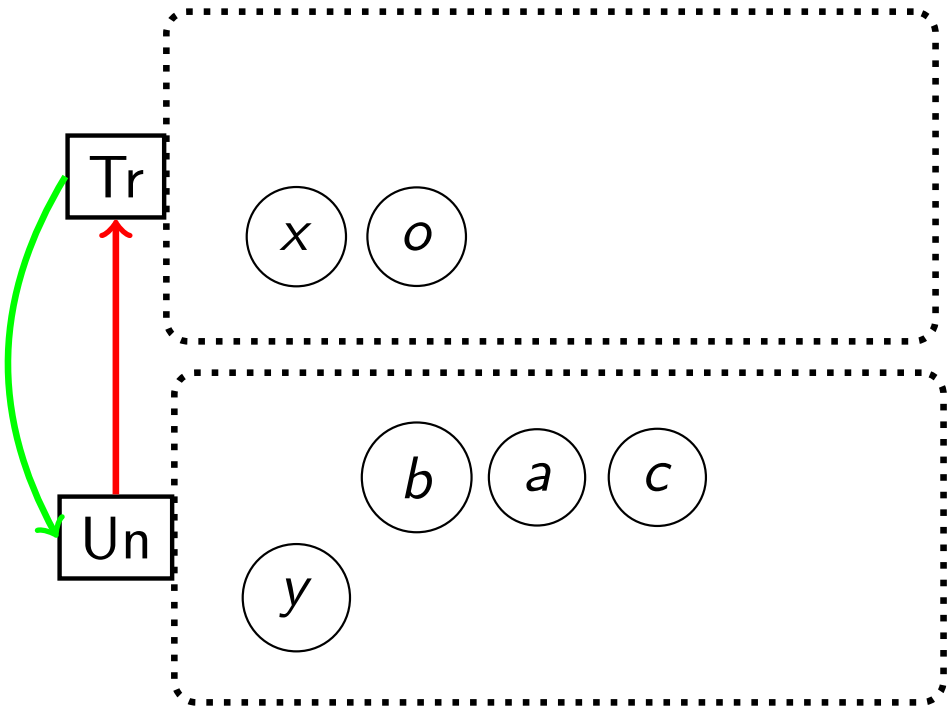
An Untyped Opponent





$$\Gamma_O \vdash_O P_O \rightarrow \Gamma_{OA} \vdash_A P_{OA}$$
$$\Gamma_U \vdash_U P_U \rightarrow \Gamma_{UA} \vdash_A P_{UA}$$

An Untyped Opponent



$$\left.\begin{array}{l}\Gamma_O \vdash_O P_O \rightarrow \Gamma_{OA} \vdash_A P_{OA} \\ \Gamma_U \vdash_U P_U \rightarrow \Gamma_{UA} \vdash_A P_{UA}\end{array}\right\} \Gamma_{OA}, \Gamma_{UA} \vdash_A P_{OA}, P_{UA}$$

Outline

Conclusion

- Typed processes do not leak secret channels to typed opponents.

Conclusion

- Typed processes do not leak secret channels to typed opponents.
- Typed processes do not leak secret channels to untyped opponents too!

Conclusion

- Typed processes do not leak secret channels to typed opponents.
- Typed processes do not leak secret channels to untyped opponents too!
- Sound type system that maps the idea of Ownership Types in the Join Calculus.