Types for information access control in X$d\pi$ calculus

Role-based access control for dynamic web data

Mariangiola Dezani-Ciancaglini [1] Silvia Ghilezan [2]
Svetlana Jakšić [2] Jovanka Pantović[2]

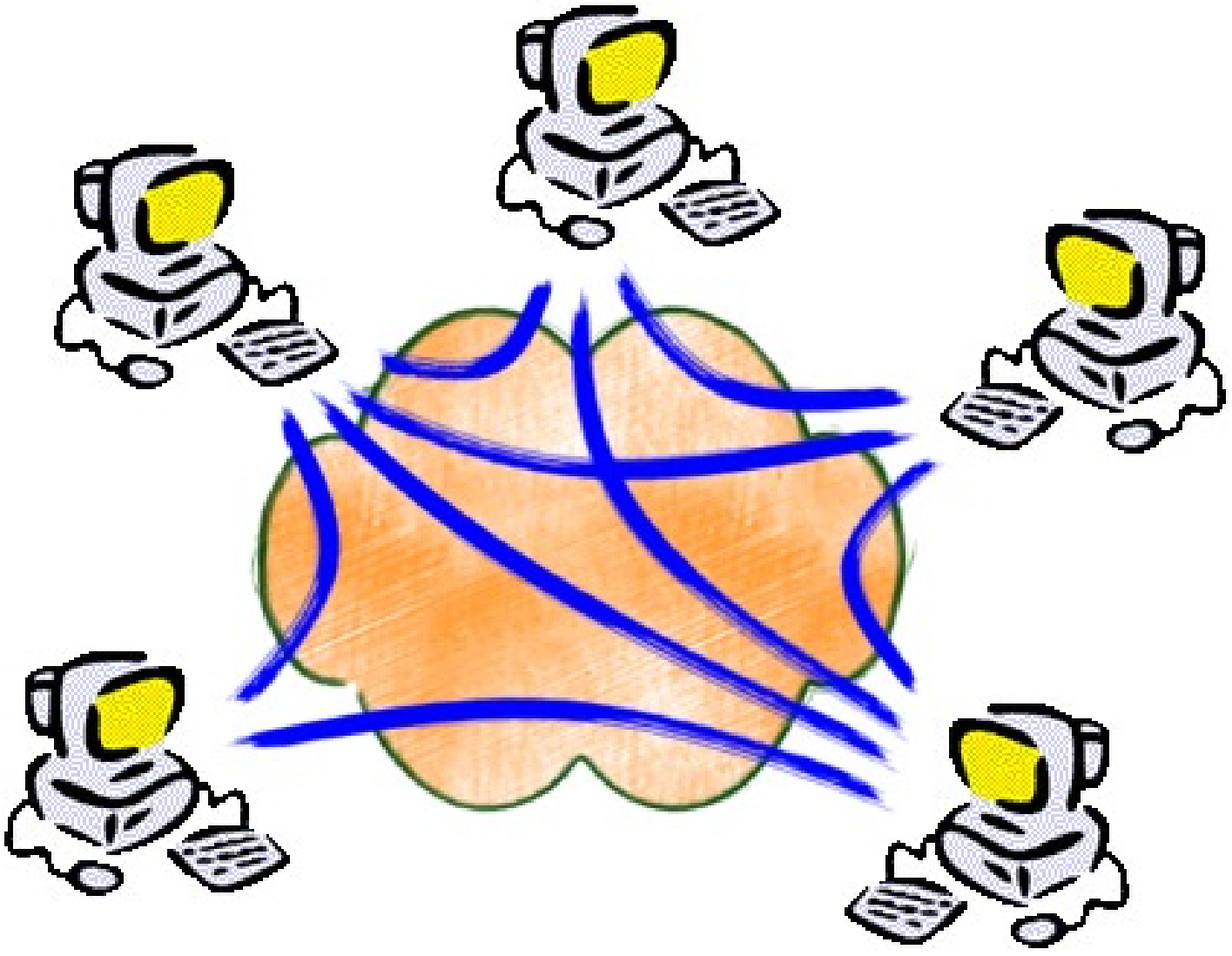
[1]Università di Torino

[2]University of Novi Sad

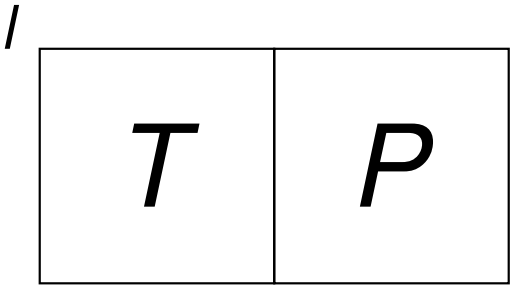
CALCO-jnr, 2009

Outline

Peer-to-Peer Network



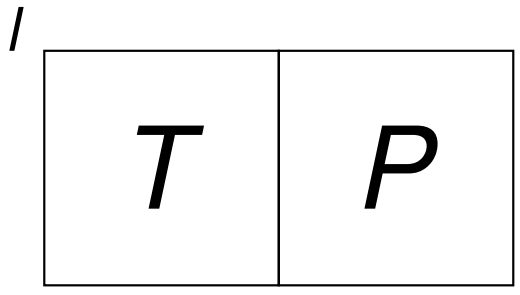
Peer-Location

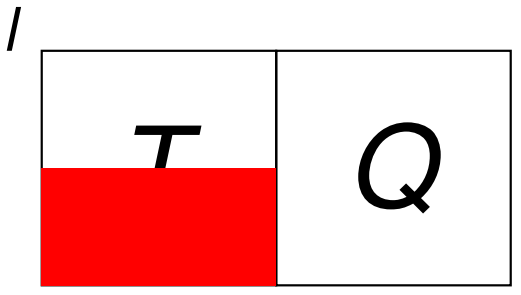


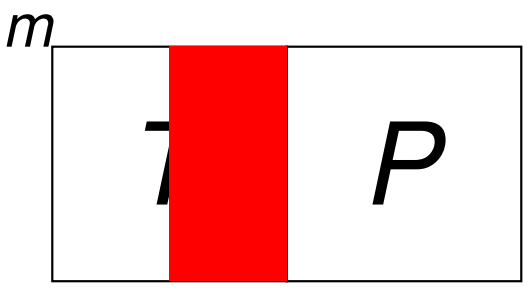
Location

In our model one peer or location is represented with data and a process.

Access Rights



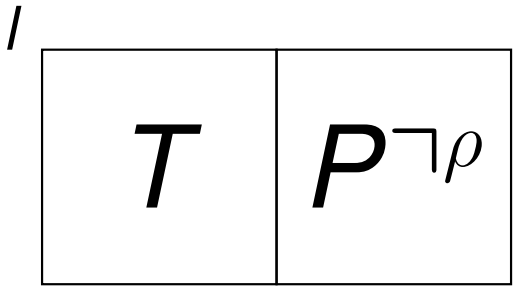




Security condition

Not all data is visible to all processes.

Roles



ρ - set of roles

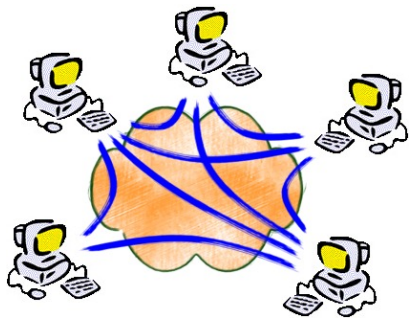
Role

- a collection of processes that have something in common (task, capabilities, responsibilities, membership,...)
- a collection of permissions

Modelling Dynamic Web Data
Philippa Gardner, Sergio Maffeis, 2005
X$d\pi$ calculus

Interactions in the System

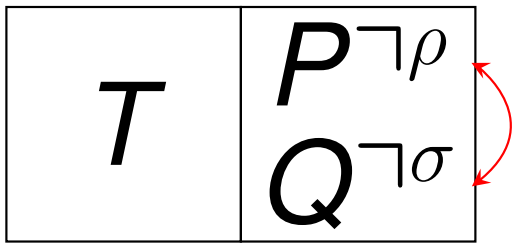
- Communication
- Movement
- Role association
- Interaction with local data

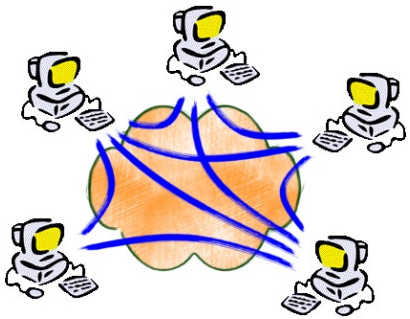


Interactions in the System

- Communication

I

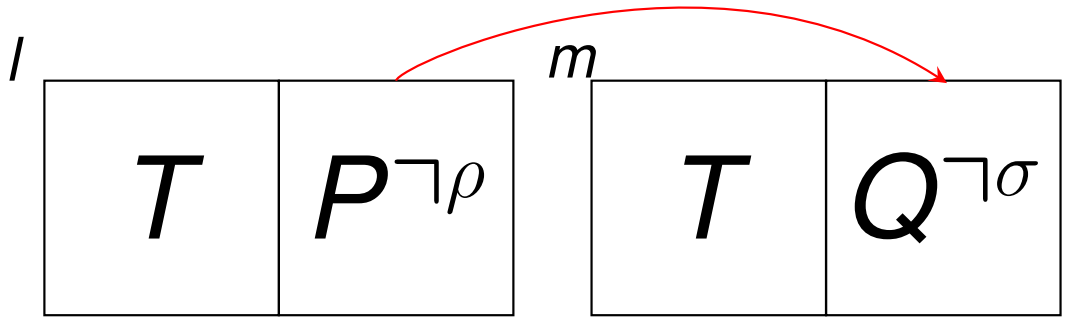


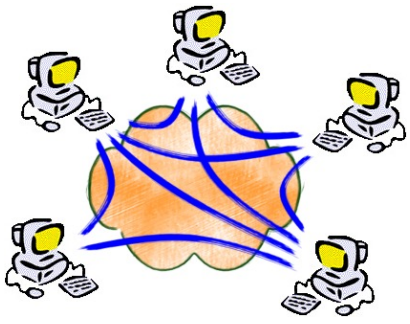


- Movement
- Role association
- Interaction with local data

Interactions in the System

- Communication
- Movement





- Role association
- Interaction with local data

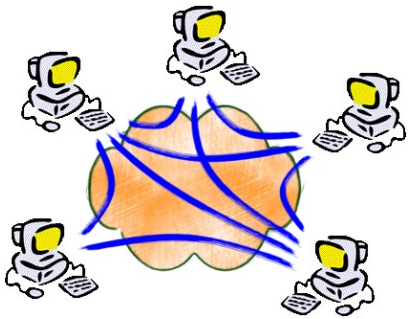
Interactions in the System

- Communication
- Movement
- Role association

I

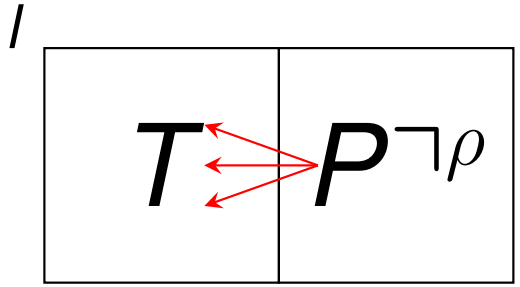
T	$P^{\neg\rho}$

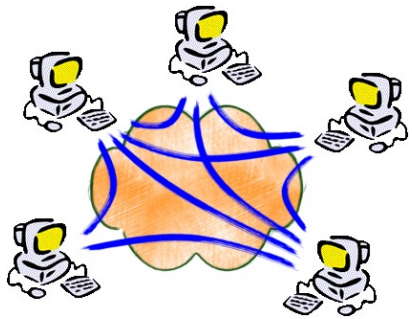
- Interaction with local data



Interactions in the System

- Communication
- Movement
- Role association
- Interaction with local data





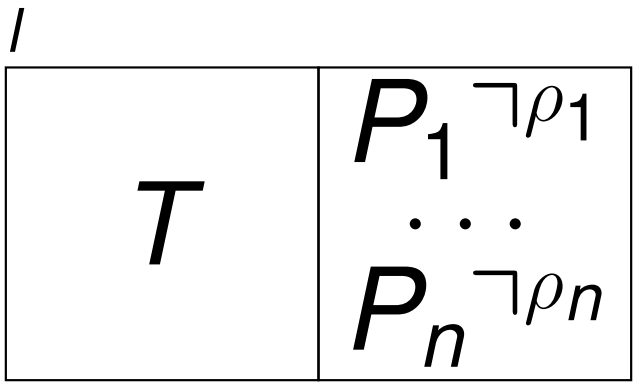
How does it work?

- Location has a policy
- Data has a type, saying how does the data look like
- Process has a type, saying how does the process behaves
- Location is well typed if types of the tree and the process ”agree” with location policy.
- Interactions in the system are described by reduction rules
- Well typed location reduces to a well typed location.

- Theorem (Subject reduction)

  Well typed network reduces to a well typed network.

Location



$$l[T \parallel P_1{}^{\neg\rho_1} | \ldots | P_n{}^{\neg\rho_n}]$$

Location l contains data tree T and processes with roles $P_i{}^{\neg\rho_i}$. At one location we can have one or more processes with different sets of roles running in parallel.

Processes

π-calculus

$P^{\urcorner\rho}, Q^{\urcorner\rho} ::=$	$0^{\urcorner\rho}$	the nil process
	$\mid (P^{\urcorner\rho} \mid Q^{\urcorner\rho})^{\urcorner\rho}$	composition of processes
	$\mid (\nu c)P^{\urcorner\rho}$	declare new channel name c
	$\mid \bar{\gamma}\langle v \rangle^{\urcorner\rho}$	output value v on channel γ
	$\mid \gamma(x).P^{\urcorner\rho}$	input parametrised by a variable x
	$\mid !\gamma(x).P^{\urcorner\rho}$	replication of an input process

Processes

$d\pi$-calculus

$$P^{\urcorner\rho}, Q^{\urcorner\rho} ::= \quad | \ \texttt{go}\ \lambda.P^{\urcorner\rho} \quad \text{migrate to location } \lambda$$
$$| \ \texttt{go}\ \circlearrowleft.P^{\urcorner\rho} \quad \text{migrate to home location}$$

X$d\pi$-calculus

$$P^{\urcorner\rho}, Q^{\urcorner\rho} ::= \quad | \ \texttt{run}_p{}^{\urcorner\rho} \quad \text{run command } p$$
$$| \ \texttt{read}_p(\chi).P^{\urcorner\rho} \quad \text{read command}$$
$$| \ \texttt{change}_p(\chi, V).P^{\urcorner\rho} \quad \text{change command}$$

Processes

Roles

$$
\begin{array}{lll}
P^{\urcorner\rho}, Q^{\urcorner\rho} ::= & \mid \texttt{activate}(r).P^{\urcorner\rho} & \text{activates role } r \text{ in the set of roles } \rho \\
& \mid \texttt{deactivate}(r).P^{\urcorner\rho} & \text{deactivates role } r \text{ in the set of roles } \rho \\
& \mid \texttt{enable}(r)_p.P^{\urcorner\rho} & \text{gives permission to the role } r \text{ to see data on the path } p \\
& \mid \texttt{disable}(r)_p.P^{\urcorner\rho} & \text{removes the role } r \text{ from roles that are allowed to see data on the path } p
\end{array}
$$

Roles on Processes

- *Role-based Access Control for Boxed Ambients*
  Adriana Compagnoni, Elsa L. Gunter, Philippe Bidinger, 2007
- *Role-Based Access Control for a Distributed Calculus*
  Chiara Braghin, Daniele Gorla, Vladimiro Sassone, 2006

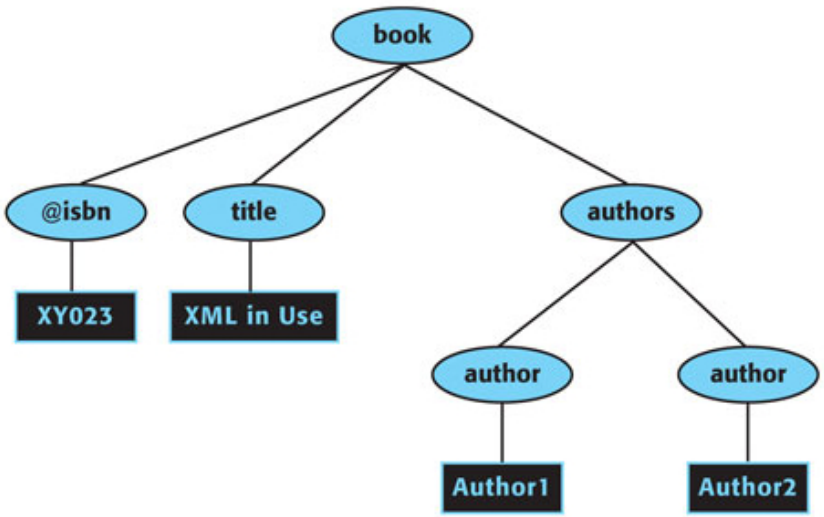
Our processes have roles. By putting roles on data we control access rights.

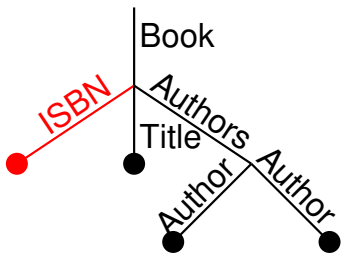
Syntax

Data

XML document

```
<book isbn='XY023'>
  <title>XML in Use</title>
  <authors>
    <author>Author1</author>
    <author>Author2</author>
  </authors>
</book>
```



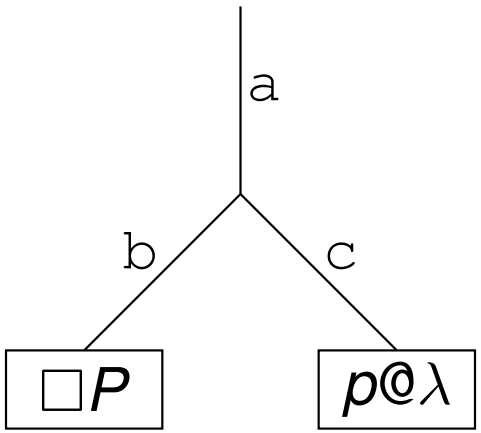


Data Tree

XML document is represented as a tree.

Syntax

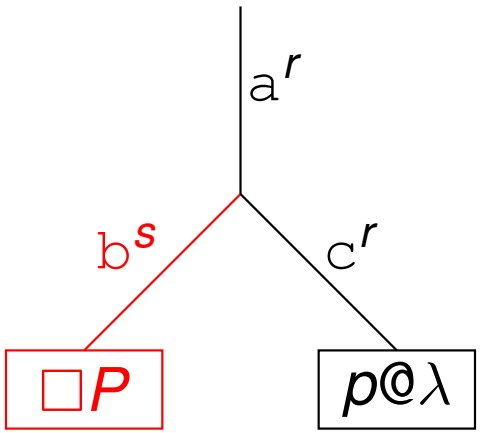
Data
Access Rights



- $\square P$-scripted process
- p-path
- $p@\lambda$-pointer

Syntax

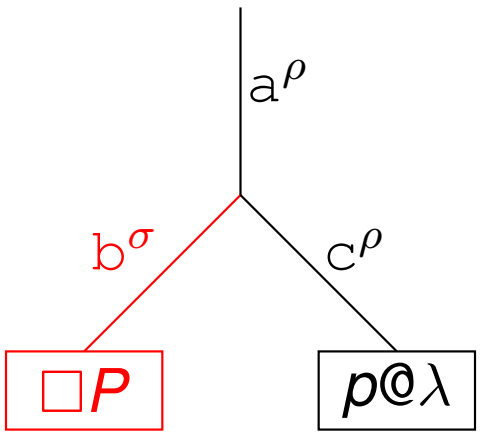
Data
Access Rights



- role *s* is higher then role *r*
- ”view” of role *r* is path a/c and pointer
- ”view” of role *s* is the whole tree
- on branches we put minimal roles

Syntax

Data
Access Rights



$$
\begin{array}{lcl}
T & ::= & \emptyset \mid x \mid T|T \mid a^\rho[T] \mid a^\rho[\Box\Pi] \mid a^\rho[p@\lambda] \\
\rho & ::= & \emptyset \mid \{r\} \mid \rho \cup \rho
\end{array}
$$

Roles

- $(\Re, \sqsubseteq)$ is a CPO and $\bot \in \Re$ is its least element
- $r, s, t, \ldots$ roles range and $\rho, \sigma, \rho_1, \ldots$ sets of roles.
- Depending on roles, a process has access to different branches of tree at some location (view). A higher (bigger) role has to have access to all branches, scripts and pointers that are visible (accessible) to lower (smaller) roles.
- $\rho \leqq \sigma$ if $(\forall s \in \sigma)(\exists r \in \rho)\ r \sqsubseteq s$

  $\rho \leq \sigma$ if $(\exists s \in \sigma)(\exists r \in \rho)\ r \sqsubseteq s$
- Two roles r, s are *compatible* if they have a join and it is denoted by $r \asymp s$. Two roles are incompatible otherwise.
- A set of roles is *coherent* if it does not contain incompatible roles.

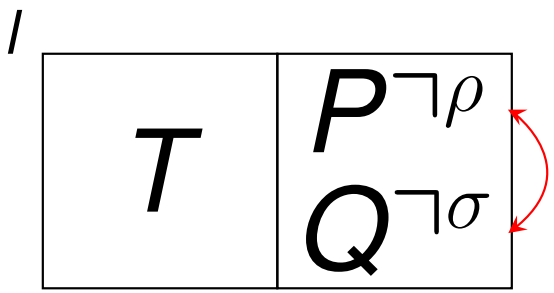
Reduction rules

Reduction rules formally describe interactions in the system.

- Communication
- Movement
- Role association
- Interaction with local data

Reduction rules

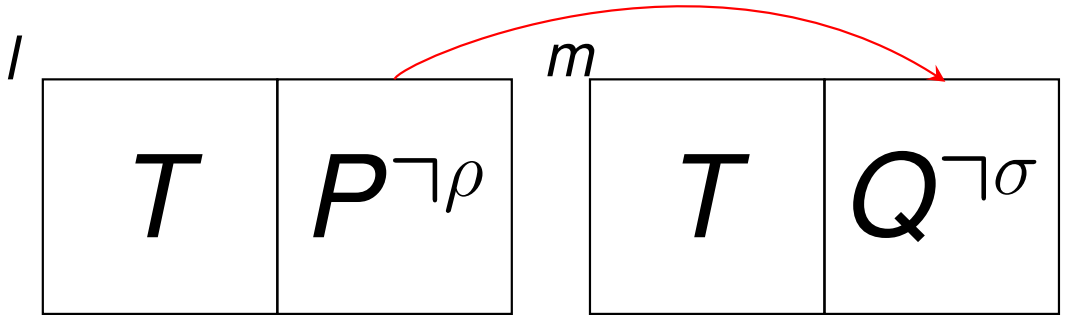
- Communication



$$I[T \parallel \bar{c}\langle v\rangle^{\neg\rho} | c(z).P^{\neg\sigma} | R] \rightarrow I[T \parallel P\{v/z\}^{\neg\sigma} | R]$$

Reduction rules

- Movement



$$l[T_1 \parallel \mathrm{go}\ m.P^{\neg\rho}|R_1] \mid m[T_2 \parallel R_2] \rightarrow l[T_1 \parallel R_1] \mid m[T_2 \parallel P^{\neg\rho}|R_2]$$

Reduction rules

- Role association

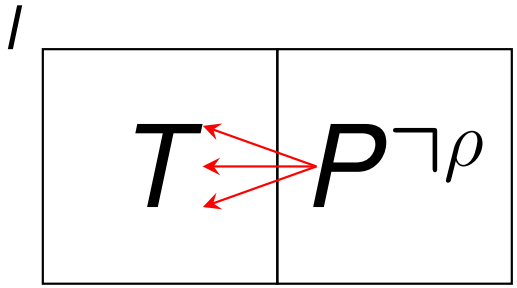
l

T	$P^{\neg\rho}$

$$l[T \parallel \texttt{activate}(s).P^{\neg\rho} | R] \longrightarrow l[T \parallel P^{\neg\rho \cup \{s\}} | R]$$

Reduction rules

- Interaction with local data



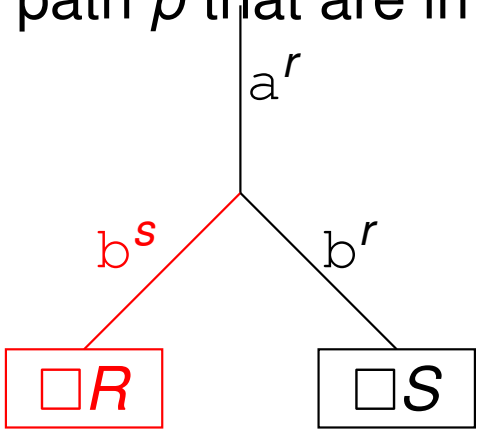
  - $\texttt{run}_p{}^{\neg\rho}$
  - $\texttt{read}_p(\chi).P^{\neg\rho}$ and $\texttt{change}_p(\chi, V).P^{\neg\rho}$
  - roles can be granted new permissions and permissions can be revoked from roles as needed. A process of high role is allowed to change access rights of the processes with lower roles by administrating the data. For this purpose we use actions $\texttt{enable}(r)_p$ and $\texttt{disable}(r)_p$.

Interaction with local data

$\mathtt{run}_p$

$$\frac{\mathtt{sub}_{\rho,p,\Box x} = \{\{\Box R_1/\Box x\},\ldots,\{\Box R_n/\Box x\}\}}{I[T \parallel \mathtt{run}_p{}^{\ulcorner\rho} | R] \longrightarrow I[T \parallel R_1 | \ldots | \ldots | R_n | R]} \quad \text{(run)}$$

The rule (run) describes how a process $\mathtt{run}_p{}^{\ulcorner\rho}$ activates a parallel execution of all scripts in the local tree identified by the path p that are in the view of ρ.



$$\frac{\mathtt{sub}_{\rho,\mathtt{a/b},\Box x} = \{\{\Box S/\Box x\}\}}{I[T \parallel \mathtt{run}_{\mathtt{a/b}}{}^{\ulcorner\rho}] \longrightarrow I[T \parallel S]}$$

Semantics

Interaction with local data

$\mathtt{read}_p(\chi)$ and $\mathtt{change}_p(\chi, V)$

$$\chi ::= \Box x \mid y@x \mid x$$

$$\frac{\mathtt{sub}_{\rho,p,\chi} = \{\mathtt{s}_1, \ldots, \mathtt{s}_n\}}{I[T \parallel \mathtt{read}_p(\chi).P^{\neg\rho} | R] \to I[T \parallel P\mathtt{s}_1{}^{\neg\rho} | \ldots | P\mathtt{s}_n{}^{\neg\rho} | R]} \quad \text{(read)}$$

$$\frac{\mathtt{ch}_{\rho,p,\chi,V}\, T = T_1,}{I[T \parallel \mathtt{change}_p(\chi, V).P^{\neg\rho} | R] \to I[T_1 \parallel P^{\neg\rho} | R]} \quad \text{(change)}$$

The rule (read) describes how a process $\mathtt{read}_p(\chi).P^{\neg\rho}$ finds all data terms V_k given by the path p in the view of ρ, matches them with the pattern χ, obtains substitutions $\mathtt{s}_k$ and activates a parallel execution of $P\mathtt{s}_k{}^{\neg\rho}$.

Semantics

Interaction with local data

$\texttt{read}_p(\chi)$ and $\texttt{change}_p(\chi, V)$

$$\chi ::= \Box x \mid y@x \mid x$$

$$\frac{\texttt{sub}_{\rho,p,\chi} = \{\texttt{s}_1, \ldots, \texttt{s}_n\}}{I[T \parallel \texttt{read}_p(\chi).P^{\neg\rho} | R] \to I[T \parallel P\texttt{s}_1{}^{\neg\rho} | \ldots | P\texttt{s}_n{}^{\neg\rho} | R]} \quad \text{(read)}$$

$$\frac{\texttt{ch}_{\rho,p,\chi,V}\, T = T_1,}{I[T \parallel \texttt{change}_p(\chi, V).P^{\neg\rho} | R] \to I[T_1 \parallel P^{\neg\rho} | R]} \quad \text{(change)}$$

The rule (change) describes how a process $\texttt{change}_p(\chi, V).P^{\neg\rho}$ finds all data terms given by the path p in the view of ρ, matches them with the pattern χ and replaces them with V.

Semantics

Interaction with local data

$\texttt{enable}(r)_p$ and $\texttt{disable}(r)_p$

$$\frac{\texttt{en}_{\rho,p,r}\, T = T_1,}{I[T \parallel \texttt{enable}(r)_p.P^{\urcorner\rho} | R] \longrightarrow I[T_1 \parallel P^{\urcorner\rho} | R]} \quad \text{(enable)}$$

$$\frac{\texttt{di}_{\rho,p,r}\, VT = T_1,}{I[T \parallel \texttt{disable}(r)_p.P^{\urcorner\rho} | R] \longrightarrow I[T_1 \parallel P^{\urcorner\rho} | R]} \quad \text{(disable)}$$

The rule (enable) describes how a process $\texttt{enable}(p)_r.P^{\urcorner\rho}$ activates in the tree the role r in ρ at the path p. Similarly, the rule (disable) describes how a process $\texttt{disable}(p)_r.P^{\urcorner\rho}$ deactivates in the tree the role r in ρ at the path p.

Reduction rules

X$d\pi$

$$l[T \parallel \bar{c}\langle v\rangle | c(z).P^{\neg\rho} | R] \longrightarrow l[T \parallel P\{v/z\}^{\neg\rho} | R] \quad \text{(com)}$$

$$l[T \parallel \bar{c}\langle v\rangle | !c(z).P^{\neg\rho} | R] \longrightarrow l[T \parallel !c(z).P^{\neg\rho} \mid P\{v/z\}^{\neg\rho} | R] \quad \text{(com!)}$$

$$l[T_1 \parallel \mathtt{go}\ m.P^{\neg\rho} | R_1] \mid m[T_2 \parallel R_2] \longrightarrow l[T_1 \parallel R_1] \mid m[T_2 \parallel P^{\neg\rho} | R_2] \quad \text{(go)}$$

$$l[T \parallel \mathtt{go}\ l.P^{\rho} | R] \longrightarrow l[T \parallel P^{\neg\rho} | R] \quad \text{(stay)}$$

$$\frac{\mathtt{sub}_{\rho,p,\Box x} = \{\{\Box R_1/\Box x\}, \ldots, \{\Box R_n/\Box x\}\}}{l[T \parallel \mathtt{run}_p{}^{\neg\rho} | R] \longrightarrow l[T \parallel R_1 | \ldots | \ldots | R_n | R]} \quad \text{(run)}$$

$$\frac{\mathtt{sub}_{\rho,p,\chi} = \{\mathtt{s}_1, \ldots, \mathtt{s}_n\}}{l[T \parallel \mathtt{read}_p(\chi).P^{\neg\rho} | R] \longrightarrow l[T \parallel P\mathtt{s}_1{}^{\neg\rho} | \ldots | P\mathtt{s}_n{}^{\neg\rho} | R]} \quad \text{(read)}$$

$$\frac{\mathtt{ch}_{\rho,p,\chi,V}\, T = T_1,}{l[T \parallel \mathtt{change}_p(\chi, V).P^{\neg\rho} | R] \longrightarrow l[T_1 \parallel P^{\neg\rho} | R]} \quad \text{(change)}$$

Reduction rules

X$d\pi$

(Empty tree) $\emptyset \rightsquigarrow_\theta \emptyset, \emptyset$

(Script) $\square P \rightsquigarrow_\theta \square P, \emptyset$

(Pointer) $p@ \rightsquigarrow_\theta p@l, \emptyset$

(Node) $$\frac{U \rightsquigarrow_\theta V, \Theta}{\mathtt{a}[U] \rightsquigarrow_\theta \mathtt{a}[V], \Theta}$$

(Par) $$\frac{U_1 \rightsquigarrow_\theta T_1, \Theta_1 \qquad U_2 \rightsquigarrow_\theta T_2, \Theta_2}{U_1 | U_2 \rightsquigarrow_\theta T_1 | T_2, \Theta_1 \cup \Theta_2}$$

(Id) $$\frac{\mathtt{match}(U, \chi) \text{ undefined} \qquad U \rightsquigarrow_\theta V, \Theta}{\mathtt{a}[\underline{U}] \rightsquigarrow_\theta \mathtt{a}[V], \Theta}$$

(Up) $$\frac{\mathtt{match}(U, \chi) = \mathtt{s} \qquad V \rightsquigarrow_\theta V', \Theta \qquad \theta = p, l, \chi, V}{\mathtt{a}[\underline{U}] \rightsquigarrow_\theta \mathtt{a}[V'], \{\mathtt{s}\{l/\circlearrowleft, p/\cdot\}\} \cup \Theta}$$

Semantics

Reduction rules

X$d\pi$

$$\mathtt{sub}_{r,p,\chi}\emptyset = \emptyset$$

$$\mathtt{sub}_{r,\mathtt{a}/p,\chi}\mathtt{b}^{\rho}[U] = \begin{cases} \mathtt{sub}_{r,p,\chi}U, & \mathtt{b} = \mathtt{a} \quad r \ltimes \rho \\ \mathtt{sub}_{r,\mathtt{a}/p,\chi}T, & \text{otherwise} \end{cases}$$

$$\mathtt{sub}_{r,\mathtt{a}/p,\chi}T_1|T_2 = \mathtt{sub}_{r,\mathtt{a}/p,\chi}T_1 \cup \mathtt{sub}_{r,\mathtt{a}/p,\chi}T_2$$

$$\mathtt{sub}_{r,\epsilon,\chi}U = \begin{cases} \{U{\downarrow_r}/\chi\}, & \mathtt{match}(U{\downarrow_r},\chi) = \mathit{True} \\ \emptyset, & \text{otherwise} \end{cases}$$

Semantics

Reduction rules

new

$$l[T \parallel \texttt{activate}(s).P^{\neg\rho}|R] \to l[T \parallel P^{\neg\rho\cup\{s\}}|R] \qquad \text{(activate)}$$

$$l[T \parallel \texttt{deactivate}(s).P^{\neg\rho}|R] \to l[T \parallel P^{\neg\rho\setminus\{s\}}|R] \qquad \text{(deactivate)}$$

$$\frac{\texttt{en}_{\rho,p,r}\, T = T_1,}{l[T \parallel \texttt{enable}(r)_p.P^{\neg\rho}|R] \to l[T_1 \parallel P^{\neg\rho}|R]} \qquad \text{(enable)}$$

$$\frac{\texttt{di}_{\rho,p,r}\, VT = T_1,}{l[T \parallel \texttt{disable}(r)_p.P^{\neg\rho}|R] \to l[T_1 \parallel P^{\neg\rho}|R]} \qquad \text{(disable)}$$

Location Policy

The function $\mathcal{T}$ associates to location its policy.

$$\frac{\mathcal{T}(l) = \mathit{Loc}(\rho, \mathcal{E}, \mathcal{D})}{\Gamma \vdash l : \mathit{Loc}(\rho, \mathcal{E}, \mathcal{D})} \text{(loc)}$$

Location only allow processes and trees that are compatible with its policy:

- trees that have only roles in ρ or higher
- processes that will enable some roles in $\mathcal{E}$ and disable roles in $\mathcal{D}$

Tree Type

$$\frac{}{\Gamma \vdash \emptyset : \mathit{ETree}(\emptyset, \bot, \emptyset, \emptyset)} \qquad \frac{}{\Gamma \vdash \mathtt{a}^{\rho}[\emptyset] : \mathit{Tree}(\rho, \rho, \emptyset, \emptyset)}$$

$$\frac{\Gamma \vdash p@\lambda : \mathit{Pointer}}{\Gamma \vdash \mathtt{a}^{\rho}[p@\lambda] : \mathit{Tree}(\rho, \rho, \emptyset, \emptyset)}$$

$$\frac{\Gamma \vdash \Box P^{\neg\rho} : \mathit{Script}(\mathcal{E}, \mathcal{D})}{\Gamma \vdash \mathtt{a}^{\rho}[\Box P^{\neg\sigma}] : \mathit{Tree}(\rho, \rho, \mathcal{E}, \mathcal{D})}$$

$$\frac{\Gamma \vdash T : \mathit{Tree}(\rho, \gamma, \mathcal{E}, \mathcal{D}) \quad \sigma \leqq \rho}{\Gamma \vdash \mathtt{a}^{\sigma}[T] : \mathit{Tree}(\sigma, \gamma, \mathcal{E}, \mathcal{D})}$$

$$\frac{\Gamma \vdash T_1 : \mathit{Tree}(\rho_1, \gamma_1, \mathcal{E}_1, \mathcal{D}_1) \quad \Gamma \vdash T_2 : \mathit{Tree}(\rho_2, \gamma_2, \mathcal{E}_2, \mathcal{D}_2)}{\Gamma \vdash T_1 | T_2 : \mathit{Tree}(\rho_1 \cup \rho_2, \gamma_1 \cap \gamma_2, \mathcal{E}_1 \cup \mathcal{E}_2, \mathcal{D}_1 \cup \mathcal{D}_2)}$$

Process Type

$$\frac{}{\Gamma \vdash 0^{\neg\rho} : Proc(\emptyset, \emptyset)} \text{ (Nil)}$$

$$\frac{\Gamma \vdash P_1{}^{\neg\rho_1} : Proc(\mathcal{E}_1, \mathcal{D}_1) \quad \Gamma \vdash P_2{}^{\neg\rho_2} : Proc(\mathcal{E}_2, \mathcal{D}_2)}{\Gamma \vdash P_1{}^{\neg\rho_1} | P_2{}^{\neg\rho_2} : Proc(\mathcal{E}_1 \cup \mathcal{E}_2, \mathcal{D}_1 \cup \mathcal{D}_2)} \text{ (Par)}$$

$$\frac{}{\Gamma \vdash \bar{\gamma}\langle v \rangle^{\neg\rho} : Proc(\emptyset, \emptyset)} \text{ (Out)}$$

$$\frac{\Gamma \vdash P^{\neg\rho} : Proc(\mathcal{E}, \mathcal{D})}{\Gamma \vdash \gamma(x).P^{\neg\rho} : Proc(\mathcal{E}, \mathcal{D})} \text{ (In)}$$

$$\frac{\Gamma \vdash P^{\neg\rho} : Proc(\mathcal{E}, \mathcal{D})}{\Gamma \vdash !\gamma(x).P^{\neg\rho} : Proc(\mathcal{E}, \mathcal{D})} \text{ (Rep)}$$

Process Type

(Go)

$$\frac{\Gamma \vdash P^{\urcorner\rho} : \mathit{Proc}(\mathcal{E}, \mathcal{D}) \quad \Gamma \vdash l : \mathit{Loc}(\sigma, \mathcal{E}', \mathcal{D}') \quad \mathcal{E} \subseteq \mathcal{E}' \; \mathcal{D} \subseteq \mathcal{D}'}{\Gamma \vdash \mathtt{go}\ l.P^{\urcorner\rho} : \mathit{Proc}(\emptyset, \emptyset)}$$

(Go_x)

$$\frac{\Gamma \vdash P^{\urcorner\rho} : \mathit{Proc}(\emptyset, \emptyset) \quad \Gamma \vdash x : \mathit{Loc}(\sigma, \mathcal{E}, \mathcal{D})}{\Gamma \vdash \mathtt{go}\ x.P^{\urcorner\rho} : \mathit{Proc}(\emptyset, \emptyset)}$$

$$\frac{\Gamma \vdash p : \mathit{Path}}{\Gamma \vdash \mathtt{run}_p{}^{\urcorner\rho} : \mathit{Proc}(\emptyset, \emptyset)} \text{ (Run)}$$

$$\frac{\Gamma \vdash p : \mathit{Path} \quad \Gamma \cup \Gamma_\chi \vdash P^{\urcorner\rho} : \mathit{Proc}(\mathcal{E}, \mathcal{D}) \quad \sigma \leq \rho}{\Gamma \vdash \mathtt{read}_p(\chi).P^{\urcorner\rho} : \mathit{Proc}(\mathcal{E}, \mathcal{D})} \text{ (Read)}$$

Process Type

$$\frac{\Gamma \cup \Gamma_\chi \vdash P^{\urcorner\rho} : \mathit{Proc}(\mathcal{E}, \mathcal{D}) \quad \Gamma \vdash p : \mathit{Path} \quad \Gamma \vdash V : \mathit{Pointer}}{\Gamma \vdash \mathtt{change}_p(\chi, V).P^{\urcorner\rho} : \mathit{Proc}(\mathcal{E}, \mathcal{D})}$$

$$\frac{\Gamma \cup \Gamma_\chi \vdash P^{\urcorner\rho} : \mathit{Proc}(\mathcal{E}, \mathcal{D}) \quad \Gamma \vdash p : \mathit{Path} \quad \Gamma \vdash V : \mathit{Script}(\mathcal{E}, \mathcal{D})}{\Gamma \vdash \mathtt{change}_p(\chi, V).P^{\urcorner\rho} : \mathit{Proc}(\mathcal{E} \cup \mathcal{E}', \mathcal{D} \cup \mathcal{D}')}$$

$$\frac{\Gamma \cup \Gamma_\chi \vdash P^{\urcorner\varrho} : \mathit{Proc}(\mathcal{E}, \mathcal{D}) \quad \Gamma \vdash p : \mathit{Path} \quad \Gamma \vdash V : \mathit{Tree}(\rho, \sigma, \mathcal{E}', \mathcal{D}')}{\Gamma \vdash \mathtt{change}_p(\chi, V).P^{\urcorner\varrho} : \mathit{Proc}(\mathcal{E} \cup \mathcal{E}', \mathcal{D} \cup \mathcal{D}')}$$

Process Type

$$\frac{\Gamma \vdash P : \mathit{Proc}(\mathcal{E}, \mathcal{D})}{\Gamma \vdash \texttt{enable}(r)_p.P : \mathit{Proc}(\mathcal{E} \cup \{(\rho, r)\}, \mathcal{D})} \text{ (Role enable)}$$

$$\frac{\Gamma \vdash P : \mathit{Proc}(\mathcal{E}, \mathcal{D})}{\Gamma \vdash \texttt{disable}(r)_p.P : \mathit{Proc}(\mathcal{E}, \mathcal{D} \cup \{(\rho, r)\})} \text{ (Role disable)}$$

Subject Reduction

- Location policy $l : Loc(\rho, \mathcal{E}, \mathcal{D})$
- Tree type $T : T : Tree(\rho, \sigma, \mathcal{E}, \mathcal{D}))$
- Process type $P : Proc(\mathcal{E}, \mathcal{D})$
- Location $l[T \parallel P]$ is well typed if types of the tree T and the process P "agree" with location policy.

$$\frac{\mathcal{T}(l) = (\sigma, \mathcal{E}, \mathcal{D}) \quad \vdash T : Tree(\rho, \gamma, \mathcal{E}', \mathcal{D}') \quad \vdash P^{\neg\rho} : Proc(\mathcal{E}'', \mathcal{D}'') \quad \rho \leq \sigma \quad \mathcal{E}' \cup \mathcal{E}'' \subset \mathcal{E} \quad \mathcal{D}' \cup \mathcal{D}'' \subset \mathcal{D}}{\vdash l[T \parallel P^{\neg\rho}] : Net}$$

- Well typed location reduces to a well typed location.

- **Theorem (Subject reduction)**

  Let $\vdash \mathbf{N} : Net$ *and* $\mathbf{N} \rightarrow \mathbf{N}'$, *then* $\vdash \mathbf{N}' : Net$. *Well typed network reduces to a well typed network.*

Future and Related Work

- *Security Types for Dynamic Web Data*
  Mariangiola Dezani-Ciancaglini, Silvia Ghilezan, Jovanka Pantović, Daniele Varacca, 2008
- *Role-based Access Control for Boxed Ambients*
  Adriana Compagnoni, Elsa L. Gunter, Philippe Bidinger, 2007
- *Role-Based Access Control for a Distributed Calculus*
  Chiara Braghin, Daniele Gorla, Vladimiro Sassone, 2006
- *Security policies as membranes in systems for global computing*
  Daniele Gorla, Matthew Hannessy, Vladimiro Sassone
- syntax of path
- roles as values - more dynamic system
- new properties