

# Implantation d'un modèle de contrôle d'accès discrétionnaire avec Focal

Florian Brecht

UPMC

25 Septembre 2007

# Formalisation

Formalisation dans un cadre sémantique développé dans l'équipe SPI (M. Jaume, C. Morisset).

Cadre instancié pour décrire un modèle de contrôle d'accès à base de rôles (L. Habib).

Instanciation du cadre pour décrire un modèle de contrôle d'accès discrétionnaire.

# Politiques discrétionnaires vs politiques obligatoires

- Discrétionnaire :

- ▶ les droits d'accès sont laissés "à la discrétion" des utilisateurs
- ▶ les utilisateurs peuvent déléguer leurs droits

- Obligatoire :

- ▶ les droits d'accès sont prédéfinis
- ▶ les utilisateurs ne peuvent pas gérer leurs droits d'accès

# ACL et Capabilities

- ACL :
  - ▶ information rattachée aux objets
  - ▶ liste (sujet, mode d'accès)
- Capabilities :
  - ▶ information rattachée aux sujets
  - ▶ liste (objet, mode d'accès)

Objets : { *ssurf*, *focal\_pas\_a\_pas*, *photos\_de\_vacances* }

Sujets : { *Diane*, *Florian* }

Modes d'accès : { *lecture*, *écriture* }

objet/sujet	diane	florian
ssurf	rw	-
focal_pas_a_pas	-	-
photos_de_vacances	w	-

# ACL et Capabilities

ACL :

objet	liste
<i>ssurf</i>	$[(Diane, \textit{ecriture}); (Diane, \textit{lecture})]$
<i>focal_pas_a_pas</i>	$[\ ]$
<i>photos_de_vacances</i>	$[(Diane, \textit{ecriture})]$

Capabilities :

sujet	liste
<i>Diane</i>	$[(\textit{ssurf}, \textit{lecture}); (\textit{ssurf}, \textit{ecriture}); (\textit{photos\_de\_vacances}, \textit{ecriture})]$
<i>Florian</i>	$[\ ]$

# Entités, Accès, Paramètre de sécurité

Cadre	Modèle discrétionnaire
$\mathcal{S}$ : Sujets	utilisateurs
$\mathcal{O}$ : Objets	fichiers
$\mathcal{A}$ : Modes d'accès	read, write
$\mathbb{A} = \mathcal{S} \times \mathcal{O} \times \mathcal{A}$ : Accès	
$\rho$ : Paramètre de sécurité	vide

# Etats

## Cadre

Les politiques de contrôle d'accès s'appliquent sur des systèmes vus comme des machines abstraites à états.

- $\Sigma$  : ensemble des états
  - ▶  $\Lambda : \Sigma \rightarrow \wp(\mathbb{A})$  : ensemble des accès courants d'un état
  - ▶  $\Upsilon : \Sigma \rightarrow \text{SF}$  : fonctions de sécurité d'un état

## Modèle discrétionnaire

$\Sigma_d$  : ensemble des états d'un modèle discrétionnaire.

$\sigma = (m, f_d) :$

- $\Lambda(\sigma) = m$  : ensemble des accès courants de  $\sigma$
- $\Upsilon(\sigma) = f_d : \mathbb{A} \rightarrow \text{bool}$  : indique si un accès est autorisé ou non

# Prédicat de Sécurité

## Cadre

- $\Omega$  : prédicat caractérisant les états sûrs du système

## Modèle discrétionnaire

- $\Omega_d$  : prédicat caractérisant la politique de sécurité du modèle discrétionnaire :

## Propriété

$$\forall a \in \mathbb{A} \quad a \in \Lambda(\sigma) \Rightarrow f_d(a)$$

# Requêtes (Syntaxe)

## Cadre

$\mathcal{R}$  : ensemble des requêtes. En général, deux types de requêtes sont considérées :

- $\langle +, s, o, x \rangle, \langle -, s, o, x \rangle$  avec  $s \in \mathcal{S}$ ,  $o \in \mathcal{O}$  et  $x \in \mathcal{A}$

## Modèle discrétionnaire

Le même ensemble de requêtes est considéré.

# Requêtes (Sémantique)

## Cadre

- $\llbracket \mathcal{R} \rrbracket_{\Sigma} \subseteq \mathcal{R} \times \Sigma$  : sémantique des requêtes

$$\langle \langle +, \mathbf{s}, \mathbf{o}, \mathbf{x} \rangle, \sigma \rangle \in \llbracket \mathcal{R} \rrbracket_{\Sigma} \Leftrightarrow (\mathbf{s}, \mathbf{o}, \mathbf{x}) \in \Lambda(\sigma)$$

$$\langle \langle -, \mathbf{s}, \mathbf{o}, \mathbf{x} \rangle, \sigma \rangle \in \llbracket \mathcal{R} \rrbracket_{\Sigma} \Leftrightarrow (\mathbf{s}, \mathbf{o}, \mathbf{x}) \notin \Lambda(\sigma)$$

- $\mathcal{R} = \mathcal{R}^{\ominus} \cup \mathcal{R}^{\ominus} \cup \mathcal{R}^{\oplus}$  : partitionnement des requêtes

## Modèle discrétionnaire

Sémantique des requêtes  $\llbracket \mathcal{R} \rrbracket_{\Sigma} \subseteq \mathcal{R} \times \Sigma$  sur le même modèle et partitionnement suivant :

$$\mathcal{R} = \underbrace{\{ \langle +, \mathbf{s}, \mathbf{o}, \mathbf{x} \rangle \}}_{\mathcal{R}^{\ominus}} \cup \underbrace{\{ \langle -, \mathbf{s}, \mathbf{o}, \mathbf{x} \rangle \}}_{\mathcal{R}^{\ominus}} \cup \underbrace{\emptyset}_{\mathcal{R}^{\oplus}}$$

# Définitions

## Cadre

### Définition (Politique de contrôle d'accès)

$$\mathbb{P}[\rho] = (\mathcal{S}, \mathcal{O}, \mathcal{A}, \Sigma, \Omega)$$

### Définition (Modèle de contrôle d'accès)

$$\mathbb{M}[\rho] = (\mathbb{P}[\rho], [\mathcal{R}]_{\Sigma})$$

### Définition (Implantation)

$$\begin{aligned} &(\tau, \Sigma_I) \text{ avec } \Sigma_I \subseteq \Sigma_{|\Omega} \\ &\text{et } \tau : \mathcal{R} \times \Sigma \rightarrow \mathcal{D} \times \Sigma \\ &\text{et } \mathcal{D} = \{\text{yes, no}\} \end{aligned}$$

# Fonction de transition

## Modèle discrétionnaire

$$\tau_d(R, (m, f_d)) = \begin{cases} (\text{yes}, (m \cup \{(s, o, x)\}, f_d)) & \text{si } R = \langle +, s, o, x \rangle \\ & \wedge f_d((s, o, x)) \\ (\text{yes}, (m \setminus \{(s, o, x)\}, f_d)) & \text{si } R = \langle -, s, o, x \rangle \\ (\text{no}, (m, f_d)) & \text{sinon} \end{cases}$$

# Correction de la fonction de transition (1)

## Modèle discrétionnaire

Démonstration de sa correction vis-à-vis :

- du prédicat de sécurité  $\Omega_d$
- de la sémantique des requêtes  $\llbracket \mathcal{R} \rrbracket_\Sigma$
- du partitionnement de l'ensemble des requêtes

$$\mathcal{R} = \mathcal{R}^\ominus \cup \mathcal{R}^\ominus \cup \mathcal{R}^\oplus$$

## Proposition

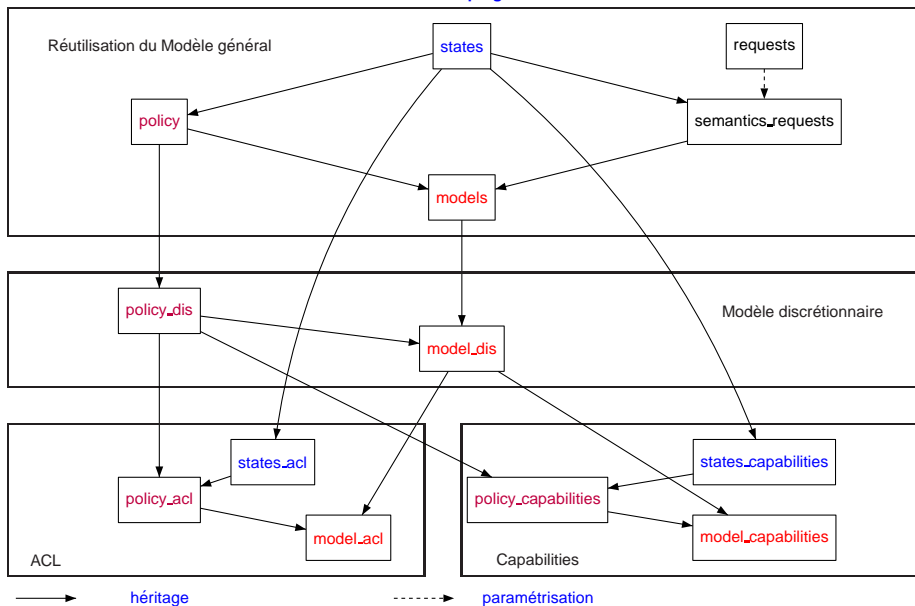
$$\mathbb{M}_d[\ ] \vdash (\tau_d, \Sigma_I)$$

# Implantation dans Focal

- Réutilisation du cadre générique (C. Morisset)
- Implantation du modèle discrétionnaire
- Réalisation des preuves de correction de la fonction de transition avec Zenon (démonstrateur automatique dans Focal)
- Développement d'un exemple

1500 lignes de code dont 400 lignes de preuve Zenon

## Structure du programme



# Implantation de la fonction de transition

```
let tau(r1 in r, st in self) =
  if r!is_get(r1)
  then
    if !fd(a!create(r!get_s(r1),
                    r!get_o(r1),
                    r!get_m(r1)))
    then (d!yes, !add(st, a!create(r!get_s(r1),
                                   r!get_o(r1),
                                   r!get_m(r1))))
    else (d!no, st)
  else (* r!is_rel(r1) *)
    (d!yes, !del(st, a!create(r!get_s(r1),
                              r!get_o(r1),
                              r!get_m(r1))));
```

## Correction de la fonction de transition (2)

Théorème de correction de la fonction de transition par rapport au prédicat de sécurité :

$$\forall \sigma, \sigma' \in \Sigma \forall R \in \mathcal{R} \forall d \in \mathcal{D} \quad (\Omega(\sigma) \wedge \tau(R, \sigma) = (d, \sigma')) \Rightarrow \Omega(\sigma')$$

```
theorem tau_secure :  
  all st1 st2 in self ,  
  all r1 in requests ,  
  all d1 in decisions ,  
    !tau(r1 , st1) = (d1 , st2)  
      -> !omega(st1)  
        -> !omega(st2)  
proof : .....
```

# Preuve Zenon

Architecture de la preuve :

```
<1>1 assume r1 in r
      st1 st2 in self
      d1 in d
      H1: !tau(r1, st1) = (d1, st2)
      H2: !omega(st1)
      prove !omega(st2)
<2>2 assume H3: r!is_get (r1)
      prove !omega(st2)
      .....
<2>3 assume H4: r!is_rel (r1)
      prove !omega(st2)
      .....
<2>f qed by r!get_or_rel, <2>2, <2>3
<1>f qed;
```

## Correction de la fonction de transition (3)

Preuve de deux autres propriétés relatives à la fonction de transition dans l'espèce **model\_dis** :

```
theorem tau_r_correct_yes :  
  all st1 st2 in self ,  
  all r1 in requests ,  
  all d1 in decisions ,  
    !tau(r1 , st1) = (d1, st2)  
      -> d!equal(d1, d!yes)  
        -> !sem_req(r1, st2);
```

```
theorem tau_r_correct_no :  
  all st1 st2 in self ,  
  all r1 in requests ,  
  all d1 in decisions ,  
    !tau(r1 , st1) = (d1, st2)  
      -> d!equal(d1, d!no)  
        -> !equal(st1, st2);
```

# Exemple d'exécution

Code Focal :

```
(* Creation de l'etat vide initial *)  
let st1 = c_model_acl!create(c_set_of_access!vide);;  
  
(* 1ere requete *)  
let requete = c_requests_gal!create(c_get_release!get,  
                                   c_subjects!diane,  
                                   c_objects!ssurf,  
                                   c_access_mode!write);;  
  
let reponse = c_model_acl!tau(#requete,#st1);;  
let d1 = #first(#e1);;  
let st2 = #scnd(#e1);;
```

Trace d'exécution :

exemple implantation ACL

acces courants: []

requete: < + , diane , ssurf , write >

decision: yes

acces courants: [( diane , ssurf , write );]

# Conclusion

- Arbre de preuves Zenon
  - ▶ permet d'expliciter l'implicite
  - ▶ réutilisation de preuves
- Réduire la taille des preuves Zenon
- Extension du modèle discrétionnaire