Institut Français de Géopolitique
M2 Cyberstratégie et terrain numérique

Approche technique de l'espace numérique

Chapitre 9

Modéliser le contrôle dans un système d'information

Pablo Rauzy <pr@up8.edu>
pablo.rauzy.name/teaching/aten

Modéliser le contrôle dans un système d'information

La privacy

- Crypto/infosec : protection d'information (pour l'utilisateurice).
- Privacy : protection du sens de l'information (nécessairement par l'utilisateurice).

→ D'où la notion de privacy par le contrôle.

Le contrôle des données personnelles

- La notion de privacy par le contrôle est prédominante dans la littérature.
- Cependant, elle n'a pas de définition formelle.
- Cela rend difficile de vérifier la conformité d'implémentations ou de spécifications, de comparer des options de conception, etc.

→ On souhaiterait disposer d'un cadre formel permettant de spécifier la notion de contrôle sur les données personnelles.

- Capturer formellement la notion de contrôle est notoirement difficile.
- Le contrôle se concentre sur la potentialité plutôt que la réalisation.
- Les travaux existants sur le contrôle (d'accès et d'usage) ne sont pas satisfaisants au regard de la notion intuitive de contrôle des données personnelles.

Trois dimensions de contrôle

- Dans leur papier* de 2015, Lazaro et Le Métayer identifient trois dimensions du contrôle des données personnelles.
- Ces trois dimensions correspondent aux capacités d'une personne :
  - de réaliser des actions sur ses données personnelles,
  - d'empêcher la réalisation d'actions sur ses données personnelles, et
  - d'être informé des actions réalisées sur ses données personnelles.

→ C'est sur cette base que nous avons construit Capacity.

* http://script-ed.org/?p=1927

Modéliser le contrôle avec Capacity

- Le but est de modéliser le contrôle des données personnelles aussi généralement que possible.
- L'abstraction et la minimalité ont été les principes moteurs de sa conception.
- Dans Capacity:
  - Des agents réalisent des opérations sur des ressources dans des contextes.
  - Le contrôle est modélisé par des exigences donnant des contraintes sur ces opérations.

→ Exemple fil rouge : un service basique de partage de photos.

Exemple fil rouge : Album

- Album est un service centralisé sur lequel les utilisateurices :
  - peuvent téléverser, supprimer, et voir des photos dans leur album ;
  - peuvent se connecter à d'autres utilisateurices pour devenir ami·es ;
  - peuvent voir les photos de leurs ami·es ;
  - peuvent taguer les photos qu'illes voient avec leur nom ou celui d'un·e ami·e ;
  - sont notifié·es quand illes sont tagué·es dans une photo par un·e ami·e.

Objets

- Il y a quatre types d'objets atomiques dans Capacity:
  - Agents :
    - les agents modélisent les utilisateurices et les services,
    - l'ensemble des agents est noté $\mathcal{A}$,
    - exemples : `Album` (le service) et ses utilisateurices (`Daniel`, `Pablo`, ...) ;
  - Ressources :
    - les ressources modélisent les données, et typiquement les données personnelles,
    - l'ensemble des ressources est noté $\mathcal{R}$,
    - exemples : les noms d'utilisateurices (`Pablo`), leur album ($album_{\texttt{Pablo}}$), et les photos (🖼).
  - Opérations :
    - les opérations modélisent ce qui peut être fait avec les ressources,
    - l'ensemble des opérations est noté $\mathcal{O}$,
    - exemples : $\mathrm{connect}$, upload, tag, access, delete ;
  - Contextes :
    - les contextes modélisent les facteurs externes pertinents à une opération,
    - l'ensemble des contextes est noté $\mathcal{C}$,
    - exemples : lieu, date, relation entre agents, but, exposition.

Objets

- Il y a quatre types d'objets atomiques dans Capacity:
  - Agents :
    - les agents modélisent les utilisateurices et les services,
    - l'ensemble des agents est noté $\mathcal{A}$,
    - exemples : `Album` (le service) et ses utilisateurices (`Daniel`, `Pablo`, ...) ;
  - Ressources :
    - les ressources modélisent les données, et typiquement les données personnelles,
    - l'ensemble des ressources est noté $\mathcal{R}$,
    - exemples : les noms d'utilisateurices (`Pablo`), leur album ($album_{\texttt{Pablo}}$), et les photos ( ).
  - Opérations :
    - les opérations modélisent ce qui peut être fait avec les ressources,
    - l'ensemble des opérations est noté $\mathcal{O}$,
    - exemples : $\mathrm{connect}, \mathrm{upload}, \mathrm{tag}, \mathrm{access}, \mathrm{delete}$;
  - Contextes :
    - les contextes modélisent les facteurs externes pertinents à une opération,
    - l'ensemble des contextes est noté $\mathcal{C}$,
    - exemples : lieu, date, relation entre agents, but, exposition.

Objets

- Il y a quatre types d'objets atomiques dans Capacity:
  - Agents :
    - les agents modélisent les utilisateurices et les services,
    - l'ensemble des agents est noté $\mathcal{A}$,
    - exemples : `Album` (le service) et ses utilisateurices (`Daniel`, `Pablo`, ...) ;
  - Ressources :
    - les ressources modélisent les données, et typiquement les données personnelles,
    - l'ensemble des ressources est noté $\mathcal{R}$,
    - exemples : les noms d'utilisateurices (`Pablo`), leur album ($album_{\texttt{Pablo}}$), et les photos (🖼).
  - Opérations :
    - les opérations modélisent ce qui peut être fait avec les ressources,
    - l'ensemble des opérations est noté $\mathcal{O}$,
    - exemples : $\mathrm{connect}, \mathrm{upload}, \mathrm{tag}, \mathrm{access}, \mathrm{delete}$;
  - Contextes :
    - les contextes modélisent les facteurs externes pertinents à une opération,
    - l'ensemble des contextes est noté $\mathcal{C}$,
    - exemples : lieu, date, relation entre agents, but, exposition.

Objets

- Il y a quatre types d'objets atomiques dans Capacity:
  - Agents :
    - les agents modélisent les utilisateurices et les services,
    - l'ensemble des agents est noté $\mathcal{A}$,
    - exemples : `Album` (le service) et ses utilisateurices (`Daniel`, `Pablo`, ...) ;
  - Ressources :
    - les ressources modélisent les données, et typiquement les données personnelles,
    - l'ensemble des ressources est noté $\mathcal{R}$,
    - exemples : les noms d'utilisateurices (`Pablo`), leur album ($album_{\texttt{Pablo}}$), et les photos ( ).
  - Opérations :
    - les opérations modélisent ce qui peut être fait avec les ressources,
    - l'ensemble des opérations est noté $\mathcal{O}$,
    - exemples : $\mathbf{connect}$, $\mathbf{upload}$, $\mathbf{tag}$, $\mathbf{access}$, $\mathbf{delete}$;
  - Contextes :
    - les contextes modélisent les facteurs externes pertinents à une opération,
    - l'ensemble des contextes est noté $\mathcal{C}$,
    - exemples : lieu, date, relation entre agents, but, exposition.

Objets

- Il y a quatre types d'objets atomiques dans Capacity:
  - Agents :
    - les agents modélisent les utilisateurices et les services,
    - l'ensemble des agents est noté $\mathcal{A}$,
    - exemples : `Album` (le service) et ses utilisateurices (`Daniel`, `Pablo`, ...) ;
  - Ressources :
    - les ressources modélisent les données, et typiquement les données personnelles,
    - l'ensemble des ressources est noté $\mathcal{R}$,
    - exemples : les noms d'utilisateurices (`Pablo`), leur album ($album_{\texttt{Pablo}}$), et les photos ( ).
  - Opérations :
    - les opérations modélisent ce qui peut être fait avec les ressources,
    - l'ensemble des opérations est noté $\mathcal{O}$,
    - exemples : $\mathrm{connect}, \mathrm{upload}, \mathrm{tag}, \mathrm{access}, \mathrm{delete}$;
  - Contextes :
    - les contextes modélisent les facteurs externes pertinents à une opération,
    - l'ensemble des contextes est noté $\mathcal{C}$,
    - exemples : lieu, date, relation entre agents, but, exposition.

Actions

- Les actions modélisent l'application d'une opération sur certaines ressources dans un certain contexte.
  - L'action $\mathrm{op}_c(r_1, \ldots, r_n)$ est l'application de l'opération op aux ressources $r_1, \ldots, r_n$ dans le contexte c.
- Exemples :
  - $\mathrm{connect}_c(\texttt{Daniel})$,
  - $\mathrm{upload}_c($  $, album_{\texttt{Pablo}})$,
  - $\mathrm{tag}_c($$, \texttt{Daniel})$.
- L'ensemble des actions est noté Δ.

- On définit trois relations sur les objets atomiques :
  - $Pers(r, a)$ exprime que la ressource r est une donnée personnelle de l'agent a,
  - $In(r, \alpha)$ exprime que l'action α manipule la ressource r,
  - $Trust(a, b)$ exprime que l'agent a à confiance en l'agent b.
- Exemples :
  - $Pers($  $, \texttt{Pablo})$,
  - $In($  $, \mathrm{tag}_c($  $, \texttt{Pablo}))$,
  - $Trust(\texttt{Pablo}, \texttt{Daniel})$.

Exigences

- Une exigence R est une relation $Can^R \subseteq \mathcal{A} \times \Delta \times \mathcal{P}(\mathcal{A}) \times \mathcal{P}(\mathcal{A})$.
- Intuitivement, $Can^R(a, \alpha, E, W)$ signifie que :
  - l'agent a peut réaliser l'action α
  - seulement si ça lui est permis par l'ensemble des agents dans E
  - alors que l'ensemble des agents dans W doivent en être informés.
- Exemples :
  - $Can^R(\texttt{Pablo}, \mathrm{upload}_c(\blacksquare, album_{\texttt{Pablo}}), \{\texttt{Album}\}, \{\texttt{Album}\})$,
  - $Can^R(\texttt{Daniel}, \mathrm{upload}_c(\blacksquare, album_{\texttt{Pablo}}), \{\bot\}, \{\bot\})$,
  - $Can^R(\texttt{Pablo}, \mathrm{tag}_c(\blacksquare, \texttt{Daniel}), \{\texttt{Daniel}, \texttt{Album}\}, \{\texttt{Daniel}, \texttt{Album}\})$.
- Cette seule relation permet d'exprimer les trois capacités de contrôle des données personnelles :
  - quand $x = a$ elle exprime la capacité de x de réaliser l'action α,
  - quand $x \in E$ elle exprime la capacité de x d'empêcher la réalisation de l'action α,
  - quand $x \in W$ elle exprime la capacité de x d'être informé de la réalisation de l'action α.

Exigences

- Une exigence R est une relation $Can^R \subseteq \mathcal{A} \times \Delta \times \mathcal{P}(\mathcal{A}) \times \mathcal{P}(\mathcal{A})$.
- Intuitivement, $Can^R(a, \alpha, E, W)$ signifie que :
  - l'agent a peut réaliser l'action α
  - seulement si ça lui est permis par l'ensemble des agents dans E
  - alors que l'ensemble des agents dans W doivent en être informés.
- Exemples :
  - $Can^R(\texttt{Pablo}, \mathrm{upload}_c(\blacksquare, album_{\texttt{Pablo}}), \{\texttt{Album}\}, \{\texttt{Album}\})$,
  - $Can^R(\texttt{Daniel}, \mathrm{upload}_c(\blacksquare, album_{\texttt{Pablo}}), \{\bot\}, \{\bot\})$,
  - $Can^R(\texttt{Pablo}, \mathrm{tag}_c(\blacksquare, \texttt{Daniel}), \{\texttt{Daniel}, \texttt{Album}\}, \{\texttt{Daniel}, \texttt{Album}\})$.
- Cette seule relation permet d'exprimer les trois capacités de contrôle des données personnelles :
  - quand $x = a$ elle exprime la capacité de x de réaliser l'action α,
  - quand $x \in E$ elle exprime la capacité de x d'empêcher la réalisation de l'action α,
  - quand $x \in W$ elle exprime la capacité de x d'être informé de la réalisation de l'action α.

Sémantique

- Les exigences sont dotées d'une sémantique de traces.
- Les traces d'exécution sont caractérisées par quatre propriétés abstraites :
  - $\theta \vdash \text{Requests}(a, \alpha)$:
    - dans la trace θ, l'agent a demande de réaliser l'action α,
    - exemple : $\theta \vdash \text{Requests}(\texttt{Pablo}, \text{tag}_c(\texttt{<image>}, \texttt{Daniel}))$;
  - $\theta \vdash \text{Enables}(a, b, \alpha)$:
    - dans la trace θ, l'agent a permet la réalisation de l'action α par l'agent b,
    - exemple : $\theta \vdash \text{Enables}(\texttt{Album}, \texttt{Pablo}, \text{tag}_c(\texttt{<image>}, \texttt{Daniel}))$,
      $\theta \vdash \text{Enables}(\texttt{Daniel}, \texttt{Pablo}, \text{tag}_c(\texttt{<image>}, \texttt{Daniel}))$;
  - $\theta \vdash \text{Does}(a, b, \alpha)$:
    - dans la trace θ, l'agent a réalise l'action α pour le compte de l'agent b,
    - exemple : $\theta \vdash \text{Does}(\texttt{Album}, \texttt{Pablo}, \text{tag}_c(\texttt{<image>}, \texttt{Daniel}))$;
  - $\theta \vdash \text{Notifies}(a, b, c, \alpha)$:
    - dans θ, l'agent a notifie l'agent b de la réalisation de l'action α pour le compte de l'agent c,
    - exemple : $\theta \vdash \text{Notifies}(\texttt{Album}, \texttt{Daniel}, \texttt{Pablo}, \text{tag}_c(\texttt{<image>}, \texttt{Daniel}))$.

Sémantique

- Les exigences sont dotées d'une sémantique de traces.
- Les traces d'exécution sont caractérisées par quatre propriétés abstraites :
  - $\theta \vdash \text{Requests}(a, \alpha)$:
    - dans la trace θ, l'agent a demande de réaliser l'action α,
    - exemple : $\theta \vdash \text{Requests}(\texttt{Pablo}, \text{tag}_c(\blacksquare, \texttt{Daniel}))$;
  - $\theta \vdash \text{Enables}(a, b, \alpha)$:
    - dans la trace θ, l'agent a permet la réalisation de l'action α par l'agent b,
    - exemple : $\theta \vdash \text{Enables}(\texttt{Album}, \texttt{Pablo}, \text{tag}_c(\blacksquare, \texttt{Daniel}))$,
      $\theta \vdash \text{Enables}(\texttt{Daniel}, \texttt{Pablo}, \text{tag}_c(\blacksquare, \texttt{Daniel}))$;
  - $\theta \vdash \text{Does}(a, b, \alpha)$:
    - dans la trace θ, l'agent a réalise l'action α pour le compte de l'agent b,
    - exemple : $\theta \vdash \text{Does}(\texttt{Album}, \texttt{Pablo}, \text{tag}_c(\blacksquare, \texttt{Daniel}))$;
  - $\theta \vdash \text{Notifies}(a, b, c, \alpha)$:
    - dans θ, l'agent a notifie l'agent b de la réalisation de l'action α pour le compte de l'agent c,
    - exemple : $\theta \vdash \text{Notifies}(\texttt{Album}, \texttt{Daniel}, \texttt{Pablo}, \text{tag}_c(\blacksquare, \texttt{Daniel}))$.

Sémantique

- Les exigences sont dotées d'une sémantique de traces.
- Les traces d'exécution sont caractérisées par quatre propriétés abstraites :
  - $\theta \vdash \text{Requests}(a, \alpha)$:
    - dans la trace θ, l'agent a demande de réaliser l'action α,
    - exemple : $\theta \vdash \text{Requests}(\texttt{Pablo}, \text{tag}_c(\blacksquare, \texttt{Daniel}))$;
  - $\theta \vdash \text{Enables}(a, b, \alpha)$:
    - dans la trace θ, l'agent a permet la réalisation de l'action α par l'agent b,
    - exemple : $\theta \vdash \text{Enables}(\texttt{Album}, \texttt{Pablo}, \text{tag}_c(\blacksquare, \texttt{Daniel}))$,
      $\theta \vdash \text{Enables}(\texttt{Daniel}, \texttt{Pablo}, \text{tag}_c(\blacksquare, \texttt{Daniel}))$;
  - $\theta \vdash \text{Does}(a, b, \alpha)$:
    - dans la trace θ, l'agent a réalise l'action α pour le compte de l'agent b,
    - exemple : $\theta \vdash \text{Does}(\texttt{Album}, \texttt{Pablo}, \text{tag}_c(\blacksquare, \texttt{Daniel}))$;
  - $\theta \vdash \text{Notifies}(a, b, c, \alpha)$:
    - dans θ, l'agent a notifie l'agent b de la réalisation de l'action α pour le compte de l'agent c,
    - exemple : $\theta \vdash \text{Notifies}(\texttt{Album}, \texttt{Daniel}, \texttt{Pablo}, \text{tag}_c(\blacksquare, \texttt{Daniel}))$.

Sémantique

- Les exigences sont dotées d'une sémantique de traces.
- Les traces d'exécution sont caractérisées par quatre propriétés abstraites :
  - $\theta \vdash \text{Requests}(a, \alpha)$:
    - dans la trace θ, l'agent a demande de réaliser l'action α,
    - exemple : $\theta \vdash \text{Requests}(\texttt{Pablo}, \text{tag}_c(\blacksquare, \texttt{Daniel}))$;
  - $\theta \vdash \text{Enables}(a, b, \alpha)$:
    - dans la trace θ, l'agent a permet la réalisation de l'action α par l'agent b,
    - exemple : $\theta \vdash \text{Enables}(\texttt{Album}, \texttt{Pablo}, \text{tag}_c(\blacksquare, \texttt{Daniel}))$,
      $\theta \vdash \text{Enables}(\texttt{Daniel}, \texttt{Pablo}, \text{tag}_c(\blacksquare, \texttt{Daniel}))$;
  - $\theta \vdash \text{Does}(a, b, \alpha)$:
    - dans la trace θ, l'agent a réalise l'action α pour le compte de l'agent b,
    - exemple : $\theta \vdash \text{Does}(\texttt{Album}, \texttt{Pablo}, \text{tag}_c(\blacksquare, \texttt{Daniel}))$;
  - $\theta \vdash \text{Notifies}(a, b, c, \alpha)$:
    - dans θ, l'agent a notifie l'agent b de la réalisation de l'action α pour le compte de l'agent c,
    - exemple : $\theta \vdash \text{Notifies}(\texttt{Album}, \texttt{Daniel}, \texttt{Pablo}, \text{tag}_c(\blacksquare, \texttt{Daniel}))$.

Sémantique

- Les exigences sont dotées d'une sémantique de traces.
- Les traces d'exécution sont caractérisées par quatre propriétés abstraites :
  - $\theta \vdash \text{Requests}(a, \alpha)$:
    - dans la trace θ, l'agent a demande de réaliser l'action α,
    - exemple : $\theta \vdash \text{Requests}(\texttt{Pablo}, \text{tag}_c(\texttt{Album}, \texttt{Daniel}))$;
  - $\theta \vdash \text{Enables}(a, b, \alpha)$:
    - dans la trace θ, l'agent a permet la réalisation de l'action α par l'agent b,
    - exemple : $\theta \vdash \text{Enables}(\texttt{Album}, \texttt{Pablo}, \text{tag}_c(\texttt{Album}, \texttt{Daniel}))$,
      $\theta \vdash \text{Enables}(\texttt{Daniel}, \texttt{Pablo}, \text{tag}_c(\texttt{Album}, \texttt{Daniel}))$;
  - $\theta \vdash \text{Does}(a, b, \alpha)$:
    - dans la trace θ, l'agent a réalise l'action α pour le compte de l'agent b,
    - exemple : $\theta \vdash \text{Does}(\texttt{Album}, \texttt{Pablo}, \text{tag}_c(\texttt{Album}, \texttt{Daniel}))$;
  - $\theta \vdash \text{Notifies}(a, b, c, \alpha)$:
    - dans θ, l'agent a notifie l'agent b de la réalisation de l'action α pour le compte de l'agent c,
    - exemple : $\theta \vdash \text{Notifies}(\texttt{Album}, \texttt{Daniel}, \texttt{Pablo}, \text{tag}_c(\texttt{Album}, \texttt{Daniel}))$.

Cohérence de traces

- Une trace θ est cohérente si :
  - $\theta \vdash \text{Does}(c, a, \alpha) \implies \theta \vdash \text{Requests}(a, \alpha)$,
  - $\theta \vdash \text{Notifies}(a, b, c, \alpha) \implies \exists d, \theta \vdash \text{Does}(d, c, \alpha)$.
- Intuitivement, une trace est incohérente si elle comprend :
  - une action réalisée pour le compte d'un agent qui ne l'a pas demandé, ou
  - la notification d'une action qui n'a pas été réalisée.

Complétude de traces

- Une trace θ est complète par rapport à une exigence R telle que $Can^R(a, \alpha, E, W)$ si :
  - $\theta \vdash \text{Requests}(a, \alpha) \wedge \forall b \in E, \theta \vdash \text{Enables}(b, a, \alpha) \implies \exists c \in \mathcal{A}, \theta \vdash \text{Does}(c, a, \alpha)$.
- Intuitivement, une trace est complète si une action est toujours réalisée quand
  - elle a été demandée, et
  - elle a été permise par tous les agents nécessaires.

Conformité de traces

- Une trace θ est conforme à une exigence R telle que $Can^R(a, \alpha, E, W)$ si :
  - $\forall d \in \mathcal{A}, \theta \vdash \text{Does}(d, a, \alpha) \implies \forall b \in E, \theta \vdash \text{Enables}(b, a, \alpha)$,
  - $\forall d \in \mathcal{A}, \theta \vdash \text{Does}(d, a, \alpha) \implies \forall b \in W, \exists c \in \mathcal{A}, \theta \vdash \text{Notifies}(c, b, a, \alpha)$.
- Intuitivement, une trace est conforme si toutes les contraintes exprimées par Can^R sont satisfaites :
  - aucune action n'est réalisée à moins d'être permise par tous les agents nécessaires, et
  - tous les gens qui doivent en être informés sont notifiés.
- La conformité d'une trace θ à une exigence R est notée $\theta \models R$.

Caractériser le contrôle avec Capacity

- On introduit quatre types de contrôle indépendants :
  - contrôle d'action,
  - contrôle d'observabilité,
  - contrôle d'autorisation,
  - contrôle de notification.
- Chaque type se décline sur trois niveaux de contrôle :
  - contrôle absolu,
  - contrôle relatif,
  - absence de contrôle.

- Le contrôle d'action décrit le contrôle d'un agent sur les actions qu'il initie.
- Par rapport à une exigence R, un agent a a :
  - un contrôle d'action absolu sur α si il ne dépend de personne pour la réaliser :
    - $AA_R(a, \alpha) \iff Can^R(a, \alpha, \emptyset, W)$;
  - un contrôle d'action relatif sur α si il ne dépend que d'agents de confiance pour la réaliser :
    - $RA_R(a, \alpha) \iff Can^R(a, \alpha, E, W) \wedge b \in E \implies Trust(a, b)$.
- Exemples :
  - $Trust(\texttt{Pablo}, \texttt{Album}) \implies RA_R(\texttt{Pablo}, \mathrm{upload}_c(\blacksquare, album_{\texttt{Pablo}}))$,
  - $Trust(\texttt{Pablo}, \texttt{Album}) \implies RA_R(\texttt{Pablo}, \mathrm{delete}_c(\blacksquare, album_{\texttt{Pablo}}))$.

- Le contrôle d'observabilité décrit la capacité d'un agent à réaliser des actions non observables par d'autres.
- Par rapport à une exigence R, un agent a a :
  - un contrôle d'observabilité absolu sur α si il peut réaliser α discrètement :
    - $AO_R(a, \alpha) \iff Can^R(a, \alpha, E, \emptyset)$;
  - un contrôle d'observabilité relatif sur α si seuls des agents de confiance sont informés de sa réalisation :
    - $RO_R(a, \alpha) \iff Can^R(a, \alpha, E, W) \wedge b \in W \implies Trust(a, b)$.
- Exemples :
  - $Trust(\texttt{Pablo}, \texttt{Album}) \implies RO_R(\texttt{Pablo}, \mathrm{upload}_c(\blacksquare, album_{\texttt{Pablo}}))$,
  - $Trust(\texttt{Pablo}, \texttt{Album}) \wedge Trust(\texttt{Pablo}, \texttt{Daniel}) \implies RO_R(\texttt{Pablo}, \mathrm{tag}_c(\blacksquare, \texttt{Daniel}))$.

- Le contrôle d'autorisation décrit le contrôle d'un agent sur les actions initiées par d'autres.
- Par rapport à une exigence R, un agent a a :
  - contrôle d'autorisation absolu sur α si il est le seul agent à pouvoir empêcher sa réalisation :
    - $AH_R(a, \alpha) \iff Can^R(b, \alpha, \{a\}, W)$;
  - un contrôle d'autorisation relatif sur α si il n'est pas le seul agent à avoir cette capacité :
    - $RH_R(a, \alpha) \iff Can^R(b, \alpha, E, W) \implies a \in E$.
- Exemples :
  - $AH_R(\texttt{Album}, \mathrm{upload}_c(\blacksquare, album_{\texttt{Pablo}}))$,
  - $RH_R(\texttt{Daniel}, \mathrm{tag}_c(\blacksquare, \texttt{Daniel}))$.

Contrôle de notification

- Le contrôle de notification décrit la capacité d'un agent à être informé des actions réalisées par d'autres.
- Par rapport à une exigence R, un agent a a :
  - un contrôle de notification absolu sur α si il est le seul agent à pouvoir être informé de sa réalisation :
    - $AN_R(a, \alpha) \iff Can^R(b, \alpha, E, \{a\})$;
  - un contrôle de notification relatif sur α si il n'est pas le seul agent à le pouvoir :
    - $RN_R(a, \alpha) \iff Can^R(b, \alpha, E, W) \implies a \in W$.
- Exemples :
  - $AN_R(\texttt{Album}, \text{upload}_c(\blacksquare, album_{\texttt{Pablo}}))$,
  - $RN_R(\texttt{Daniel}, \text{tag}_c(\blacksquare, \texttt{Daniel}))$.

Extensions

- Ces types de contrôle peuvent être étendus aux ressources et aux agents :
  - pour une ressource, en les généralisant à toutes les actions qui manipule cette ressource :
    - $AA_R(a, r) \iff \forall \alpha \in \Delta, In(r, \alpha) \implies AA_R(a, \alpha)$;
  - pour un agent, en les généralisant à toutes ses données personnelles :
    - $AA_R(a) \iff \forall r \in \mathcal{R}, Pers(r, a) \implies AA_R(a, r)$.
- Treillis de contrôle :
  - il est facile de vérifier que le contrôle absolu implique le contrôle relatif ;
  - en utilisant l'ordre défini par l'implication, on obtient un treillis de 3^4 formes de contrôle pour chaque action, ressource et agent.

Évaluer le contrôle avec Capacity

- On évalue un système en utilisant ses traces d'exécution concrètes.
- Les traces concrètes sont des séquences d'évènements concrets qui peuvent être clairement identifiés :
  - requêtes et réponses HTTP, requêtes SQL, manipulations de fichiers, etc.
- Modéliser un système concret dans Capacity nécessite :
  - d'identifier les ensembles d'agents, de ressources, d'actions, et de contextes ;
  - de définir les conditions dans lesquelles une trace concrète satisfait chaque propriété abstraite.
- Une fois ce modèle construit il est possible :
  - de calculer l'exigence qui correspond à ce système,
  - de vérifier si le système est conforme à une exigence particulière,
  - d'évaluer les types et niveaux de contrôle de chaque agent.

Un exemple avec Album

- Dans Album, les traces concrètes sont des suites des évènements suivants :
  - `U-registers(`u`)` : l'utilisateurice u s'inscrit à Album;
  - `U-uploads-pic(`u, p`)` : u téléverse une photo dans son album ;
  - `U-requests-album(`u_1, u_2`)` : u_1 demande l'album de u_2 ;
  - `U-submits-tag(`u_1, p, u_2`)` : u_1 tague u_2 dans la photo p ;
  - `U-deletes-pic(`u_1, p`)` : u_1 supprime la photo p de son album ;
  - `U-requests-con(`u_1, u_2`)` : u_1 demande à se connecter à u_2 ;
  - `U-accepts-con(`u_1, u_2`)` : u_1 accepte la connexion avec u_2 ;
  - `U-rejects-con(`u_1, u_2`)` : u_1 refuse la connexion avec u_2 ;
  - `U-disconnects(`u_1, u_2`)` : u_1 se déconnecte de u_2 ;
  - `A-creates-account(`u`)` : Album crée le compte de u ;
  - `A-publishes-pic(`p, u`)` : Album publie la photo p dans l'album de u ;
  - `A-serves-album(`u_1, u_2`)` : Album envoie l'album de u_2 à u_1 ;
  - `A-connects(`u_1, u_2`)` : Album connecte u_1 et u_2 ;
  - `A-disconnects(`u_1, u_2`)` : Album déconnecte u_1 et u_2 ;
  - `A-tags-pic(`u_1, p`)` : Album tague u_1 dans la photo p ;
  - `A-notifies-req(`u_1, u_2`)` : Album notifie u_2 de la demande de connexion de u_1 ;
  - `A-notifies-con(`u_1, u_2`)` : Album notifie u_1 et u_2 de leur connexion ;
  - `A-notifies-tag(`u_1, p, u_2`)` : Album notifie u_1 qu'ille est tagué·e dans la photo p par u_2.

Album : téléverser une photo

- Soit θ_n le $n$ième évènement dans la trace concrète θ.
- On définit les propriétés abstraites comme suit :
  - $\theta \vdash \text{Requests}(u, \text{upload}_n(p, album_u))$
    $\iff \exists m < n, \theta_m = \texttt{U-uploads-pic}(u, p)$.
  - $\theta \vdash \text{Enables}(\texttt{Album}, u, \text{upload}_n(p, album_u))$
    $\iff \exists m < n, \theta_m = \texttt{U-registers}(u)$.
  - $\theta \vdash \text{Does}(\texttt{Album}, u, \text{upload}_n(p, album_u))$
    $\iff \theta_n = \texttt{A-publishes-pic}(p, u)$.

Propriétés du contrôle

- Avec ces définitions on peut prouver que $\theta \vDash R$ telle que :
  - $Can^R(u, \text{upload}_n(p, album_u), \{\texttt{Album}\}, \{\texttt{Album}\})$.
- Ce qui signifie en terme de contrôle :
  - $RA_R(u, \text{upload}_n(p, album_u))$ si $Trust(u, \texttt{Album})$.
  - $RO_R(u, \text{upload}_n(p, album_u))$ si $Trust(u, \texttt{Album})$.
  - $AH_R(\texttt{Album}, \text{upload}_n(p, album_u))$.
  - $AN_R(\texttt{Album}, \text{upload}_n(p, album_u))$.

Album : taguer un·e ami·e

- Soit θ_n le $n$ième évènement dans la trace concrète θ.
- On définit les propriétés abstraites comme suit :
  - $\theta \vdash \text{Requests}(u_1, \text{tag}_n(p, u_2))$
    $\iff \exists m < n, \theta_m = \texttt{U-submits-tag}(u_1, p, u_2)$.
  - $\theta \vdash \text{Enables}(u_2, u_1, \text{tag}_n(p, u_2))$
    $\iff \exists m < n, \theta_m = \texttt{U-accepts-con}(u_2, u_1) \vee \theta_m = \texttt{U-accepts-con}(u_1, u_2)$
    $\wedge \ \nexists k, m < k < n, \theta_k = \texttt{U-disconnects}(u_2, u_1) \vee \theta_m = \texttt{U-disconnects}(u_1, u_2)$.
  - $\theta \vdash \text{Enables}(\texttt{Album}, u_1, \text{tag}_n(p, u_2))$
    $\iff \theta_n = \texttt{A-tags-pic}(u_2, p)$.
  - $\theta \vdash \text{Does}(\texttt{Album}, u_1, \text{tag}_n(p, u_2))$
    $\iff \theta_n = \texttt{A-tags-pic}(u_2, p)$.
  - $\theta \vdash \text{Notifies}(\texttt{Album}, u_2, u_1, \text{tag}_n(p, u_2))$
    $\iff \theta_{n+1} = \texttt{A-notifies-tag}(u_2, p, u_1)$.

Propriétés du contrôle

- Avec ces définitions on peut prouver que $\theta \vDash R$ telle que :
  - $Can^R(u_1, \mathrm{tag}_n(p, u_2), \{u_2, \texttt{Album}\}, \{u_2, \texttt{Album}\})$.
- Ce qui signifie en terme de contrôle :
  - $RA_R(u_1, \mathrm{tag}_n(p, u_2))$ si $Trust(u_1, \texttt{Album}) \wedge Trust(u_1, u_2)$.
  - $RO_R(u_1, \mathrm{tag}_n(p, u_2))$ si $Trust(u_1, \texttt{Album}) \wedge Trust(u_1, u_2)$.
  - $RH_R(u, \mathrm{tag}_n(p, u))$.
  - $RN_R(u, \mathrm{tag}_n(p, u))$.
  - $RH_R(\texttt{Album}, \mathrm{tag}_n(p, u))$.
  - $RN_R(\texttt{Album}, \mathrm{tag}_n(p, u))$.

Comparaison d'implémentations

- Les types et niveaux de contrôle permettent de comparer formellement la privacy offerte par différents systèmes.
- L'étude d'implémentations alternatives d'un même système lors de sa phase de conception permet une approche privacy by design.

Conclusions et perspectives

- Capacity fournit un cadre formel pour raisonner sur la privacy en termes de contrôle.
- Le but de ce travail est de servir de base à la construction de nouveaux outils techniques et théoriques pour la privacy.
- Travaux futurs :
  - trouver un meilleur moyen de capturer la notion d'exposition que le contexte ;
  - faire une interface conviviale de spécification d'exigence ;
  - modéliser les aspects contrôle des données personnelles du GDPR ;
  - développer des outils de vérification de modèle pour automatiser la preuve de conformité.