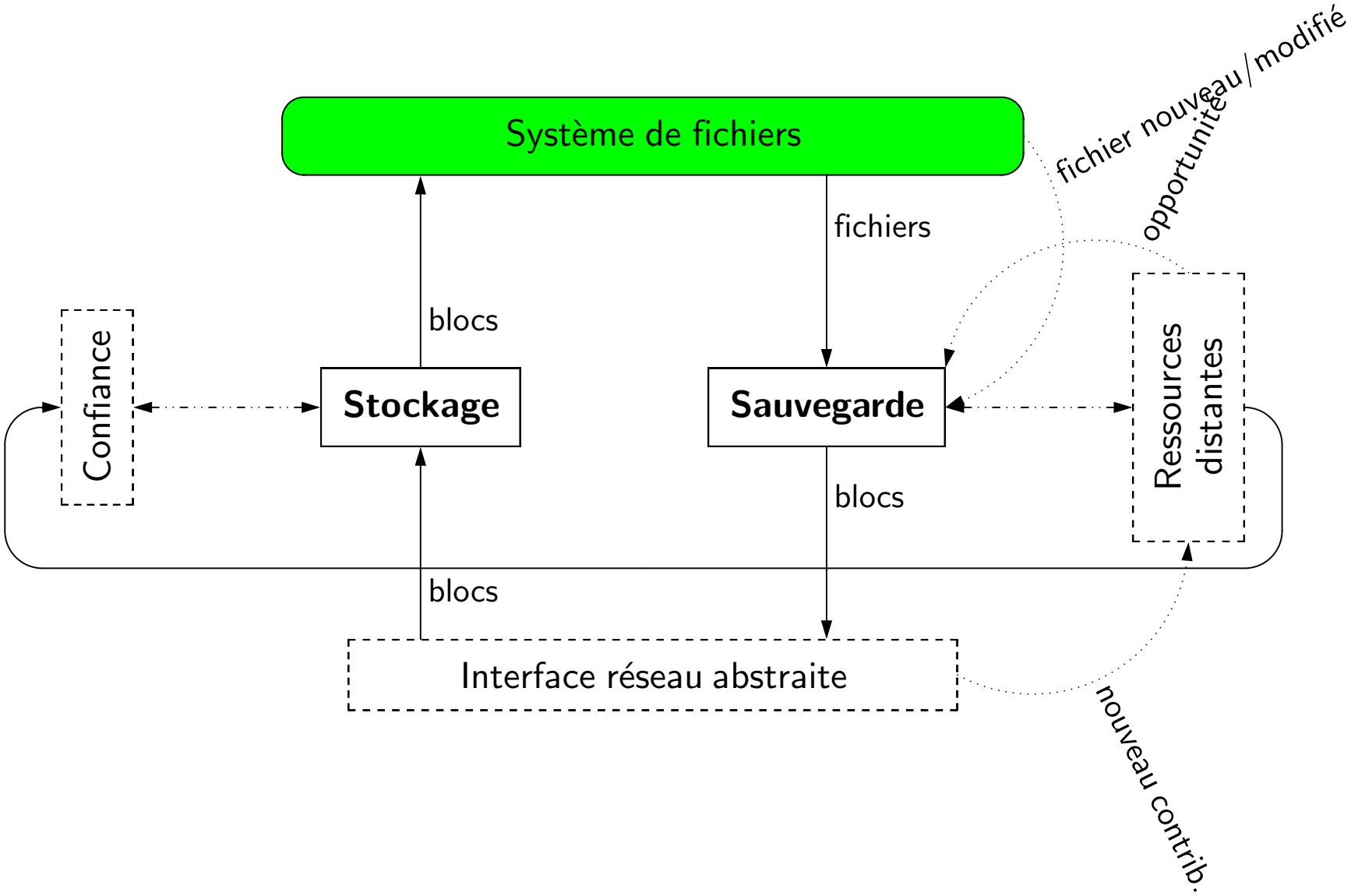
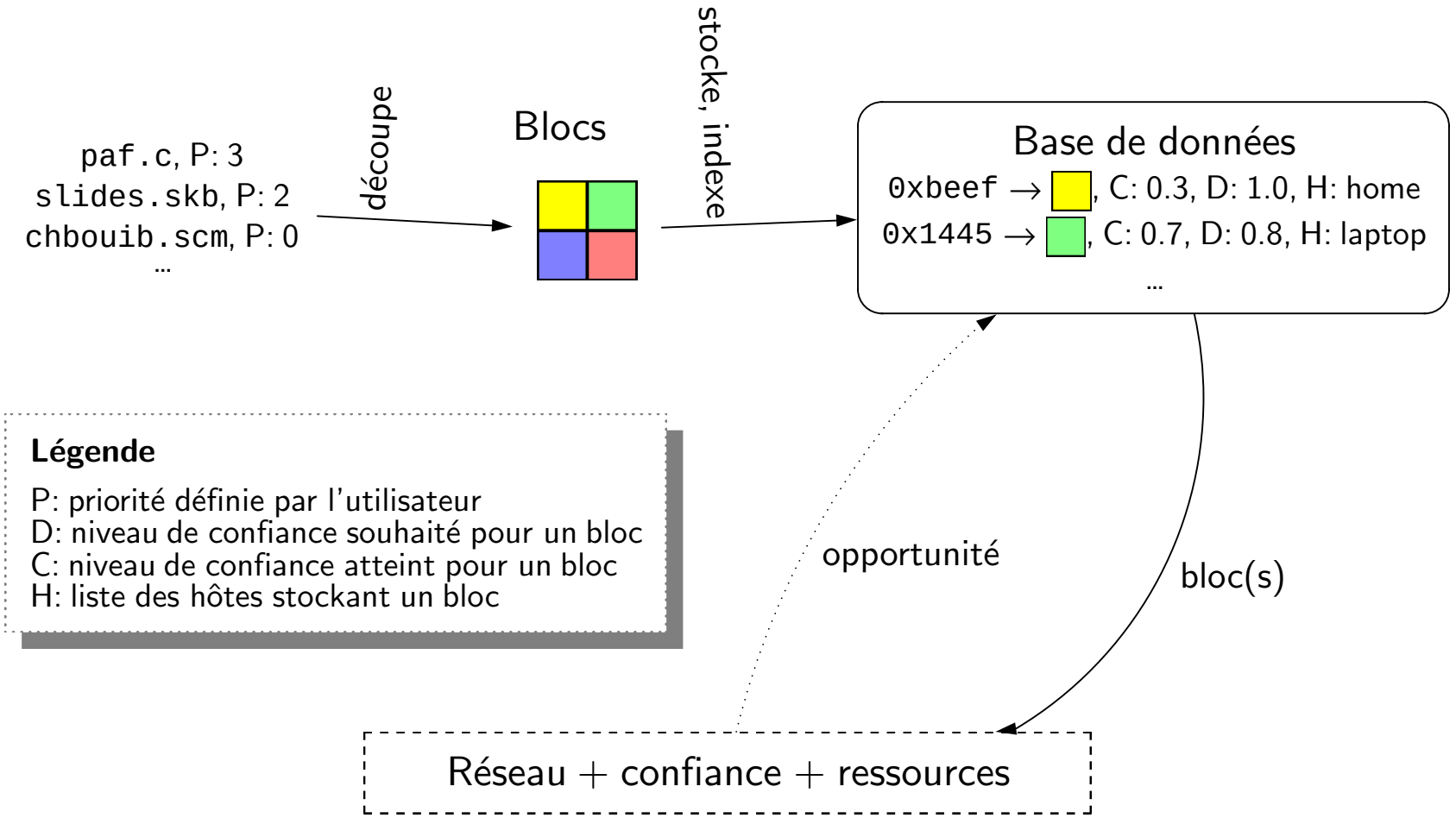
Un aperçu de l'architecture de sauvegarde dans MoSAIC

Ludovic Courtès, Yves Deswarte,
Marc-Olivier Killijian, David Powell, Matthieu Roy
LAAS-CNRS

Sauvegarde et contribution : architecture



Sauvegarde : vue détaillée



Sauvegarde : détails

Préparatifs

- **Découpage** des fichiers à sauvegarder
- **Assignation d'un identifiant** à chaque bloc
- Chaque fichier/bloc a un **niveau de confiance souhaité** défini par l'utilisateur

Fonctionnement

- Pour chaque bloc, maintien d'un **niveau de confiance atteint**
- Blocs **en stationnement dans une file d'attente** jusqu'à obtention du niveau de confiance souhaité
- Blocs en attente **envoyés à un pair** de manière opportuniste (cf. couche réseau)
- Blocs **sélectionnés en fonction de leur priorité**

Sauvegarde : gestion des ressources (distantes)

À chaque bloc est associé :

- son **niveau de confiance atteint**, mis à jour **en fonction des répliques crées**
- la **liste des contributeurs qui le stockent**
  - pour ne pas stocker toujours au même endroit
  - pour pouvoir **supprimer des répliques** ultérieurement

Sauvegarde : gestion des modifications

Que faire lors de la **notification d'une modification** d'un fichier à sauvegarder ?

L'approche « optimiste »

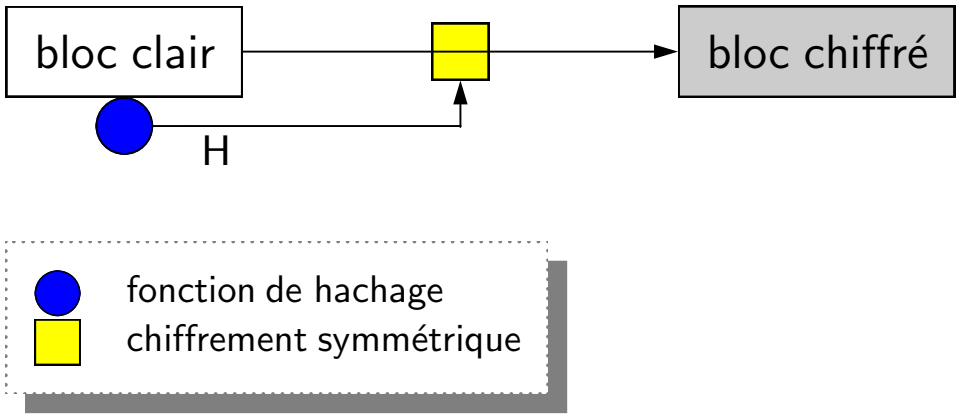
- **privilégier la fraîcheur** des données
- on arrête de sauvegarder l'ancienne version
- on ne sauvegarde **plus que la nouvelle version**

L'approche « pessimiste »

- **privilégier la disponibilité**
- on termine la sauvegarde de la version courante
- la nouvelle version est **mise en attente**

Autre possibilité : demander à l'utilisateur

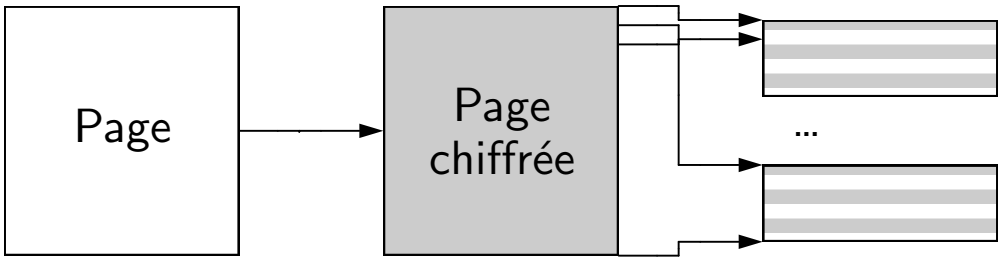
Fragmentation vs. chiffrement convergent (1)



Chiffrement convergent [Bolosky et al. 2000]

- chiffrement et vérification d'intégrité **incrémentaux** (par bloc)
- chiffrement symétrique des blocs avec une **clef fonction de leur contenu** (un condensé)
- pour déchiffrer un bloc, **il faut connaître (avoir connu) son contenu (son condensé)**
- permet de **repérer des blocs identiques**

Fragmentation vs. chiffrement convergent (2)



Fragmentation [Deswarte et al. 1991]

- fichiers découpés en **blocs de taille fixe** (pages)
- pages **chiffrées avec une clef propre au propriétaire**
- pages chiffrées **fragmentées** de manière non linéaire (entrelacement du contenu)
- fragments **dispersés**

Fragmentation vs. chiffrement convergent (3)

Confidentialité

- CC et F : « blocs » **dépourvus de sémantique** et indéchiffrables
- F : un fragment seul ne fournit **aucune information utile**
- CC : un contributeur **peut savoir si un propriétaire dispose de mêmes blocs que lui**
- CC : ... mais un propriétaire **peut choisir de chiffrer ses données** pour les rendre uniques

Efficacité en termes d'espace consommé

- F : chaque jeu de fragments est **unique à son propriétaire**
- CC : un bloc provenant d'origines différentes **peut n'être stocké qu'une fois** (indexation fonction du contenu)

$\Rightarrow$ Compromis entre confidentialité et efficacité

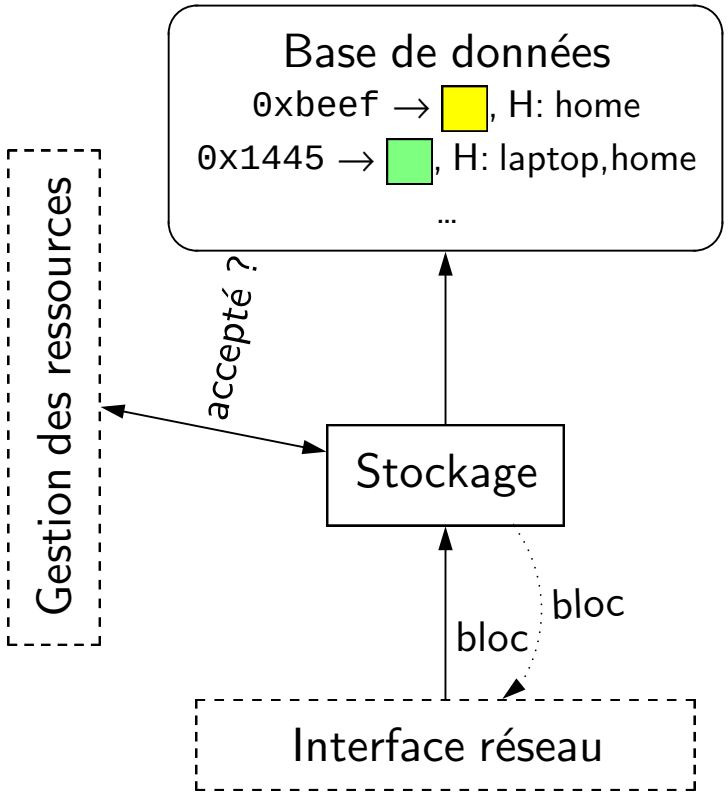
Fragmentation vs. chiffrement convergent (4)

Adaptation au scénario mobile

- Interactions **éphémères et imprévisibles**
- Possibilité de **collusion de contributeurs** (problème pour F)
- **Passage à l'échelle du mécanisme de ré-assemblage des fragments** (diffusion)
- Difficulté **d'anticipation du placement des fragments** (requis par F)
- Préférence au **placement de tous les blocs/fragments chez un contributeur** (?)

$\Rightarrow$ **Compromis entre disponibilité et tolérance aux intrusions**

Stockage : vue générale



- **Réception et sauvegarde de blocs** au hasard des rencontres
- **Gestion des ressources locales**
- Prise en compte des **relations de confiance et de la volonté de collaborer** (interaction avec « confiance »)
- **Maintien d'informations** sur l'origine de chaque bloc
- **« Déchargement » des blocs vers une infrastructure** de manière opportuniste
- **Migration de blocs ?** (→ gain/perte de confidentialité)

Stockage : gestion des ressources

- associations bloc $\rightarrow$ liste des propriétaires
- paramètres de la **politique d'acceptation de blocs** :
  - taille du bloc à stocker
  - **espace disponible** (excédentaire ? limité ?)
  - **fréquence** et **coût** d'accès à une infrastructure
  - énergie disponible, utilisation CPU, bande passante, etc.

Fin

Questions ?