

Contexte

- ▶ Réseaux de capteurs
- ▶ Sécurité
- ▶ Agrégation des données

Contexte

- ▶ Réseaux de capteurs
- ▶ Sécurité
- ▶ Agrégation des données

Contribution

- ▶ 3 nouvelles techniques pour l'agrégation
 - ▶ Agrégation Multi-chemins Secrète (AMS)
 - ▶ Agrégation Multi-chemins Dispersée (AMD)
 - ▶ Agrégation Multi-chemins Dispersée Authentifiée (AMD-A)
- ▶ Analyse et implémentation

Introduction

État de l'art

Techniques proposées

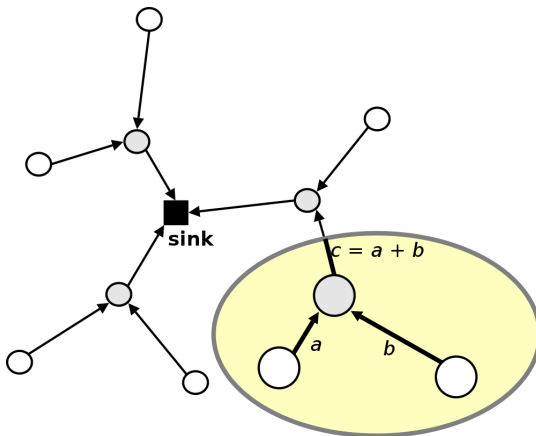
Agrégation Multi-chemins Secrète (AMS)

Agrégation Multi-chemins Dispersée (AMD)

Agrégation Multi-chemins Dispersée Authentifiée (AMD-A)

Analyse et implémentation

Introduction: réseaux de capteurs et agrégation



- ▶ Dédiés à la surveillance d'évènements physiques
- ▶ **Agrégation des messages** pour réduire les communications

Capture des noeuds d'agrégation

- ▶ Contraintes sur les noeuds \Rightarrow captures de noeuds intéressantes
- ▶ Les agrégateurs sont plus intéressant à compromettre

Capture des noeuds d'agrégation

- ▶ Contraintes sur les noeuds \Rightarrow captures de noeuds intéressantes
- ▶ Les agrégateurs sont plus intéressant à compromettre

Attaques considérées

- ▶ Écoutes
- ▶ Falsifications
 - ▶ Modification
 - ▶ Rejeu
 - ▶ Envoi de bruit
- ▶ Dénis de service

Attaques externes

- ▶ Mécanismes de niveau lien et réseau

Attaques externes

- ▶ Mécanismes de niveau lien et réseau

Attaques internes (captures de noeuds)

- ▶ Écoutes
 - ▶ Cryptographie homomorphique

Attaques externes

- ▶ Mécanismes de niveau lien et réseau

Attaques internes (captures de noeuds)

- ▶ Écoutes
 - ▶ Cryptographie homomorphique
- ▶ Falsifications
 - ▶ Preuves interactives
 - ▶ Authentification et agrégation par des noeuds distincts

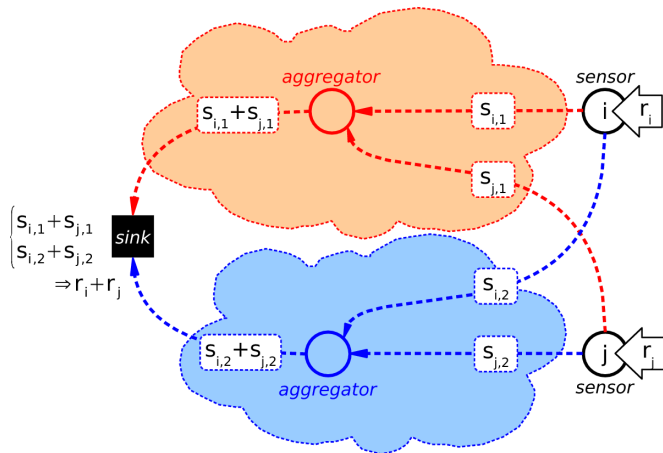
Attaques externes

- ▶ Mécanismes de niveau lien et réseau

Attaques internes (captures de noeuds)

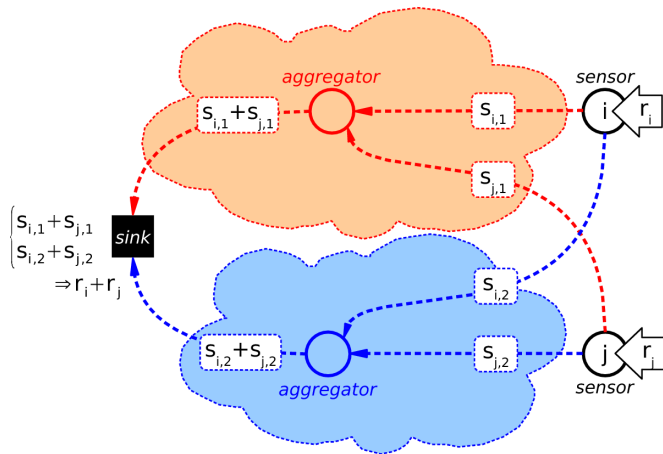
- ▶ Écoutes
 - ▶ Cryptographie homomorphique
- ▶ Falsifications
 - ▶ Preuves interactives
 - ▶ Authentification et agrégation par des noeuds distincts
- ▶ **Solutions intéressantes mais incomplètes**

Techniques proposées : principes de base



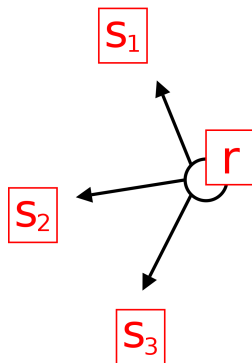
- ▶ Répartir les mesures en plusieurs parts
- ▶ Envoyer chaque part sur différents chemins

Techniques proposées : principes de base



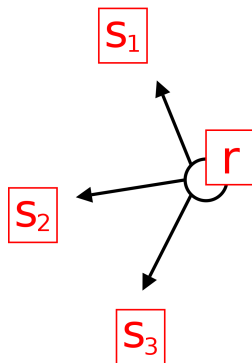
- ▶ Répartir les mesures en plusieurs parts
- ▶ Envoyer chaque part sur différents chemins
- ▶ Agréger les parts

Agrégation Multi-chemins Secrète (AMS)



- ▶ AMS utilise la **cryptographie à seuil**
- ▶ 1 mesure $\Rightarrow p$ parts
- ▶ t parts nécessaires pour la reconstruction

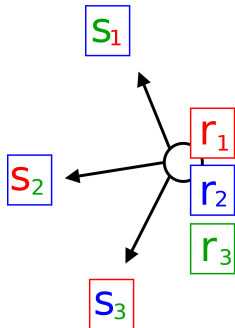
Agrégation Multi-chemins Secrète (AMS)



- ▶ AMS utilise la **cryptographie à seuil**
- ▶ 1 mesure $\Rightarrow p$ parts
- ▶ t parts nécessaires pour la reconstruction
- ▶ **Défauts:** pas d'authentification, nécessite beaucoup de messages
- ▶ **Avantages:** confidentialité très forte

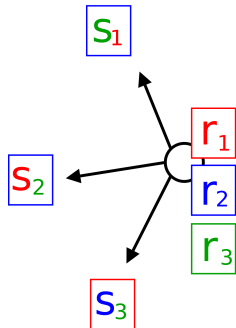
Surcoût: $p - 1$ messages supplémentaires par mesure

Agrégation Multi-chemins Dispersée (AMD)



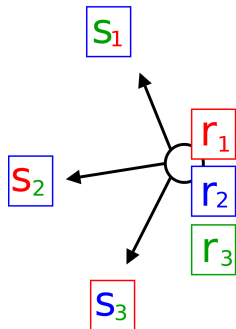
- ▶ AMD utilise la **dissémination d'information**
- ▶ t mesures $\Rightarrow p$ parts
- ▶ t parts nécessaires pour la reconstruction

Agrégation Multi-chemins Dispersée (AMD)



- ▶ AMD utilise la **dissémination d'information**
- ▶ t mesures $\Rightarrow p$ parts
- ▶ t parts nécessaires pour la reconstruction
- ▶ AMD assure une confidentialité, mais plus faible que AMS

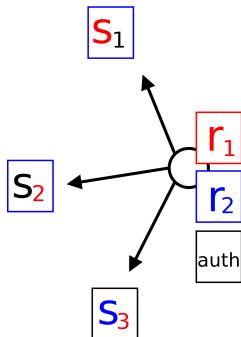
Agrégation Multi-chemins Dispersée (AMD)



- ▶ AMD utilise la **dissémination d'information**
- ▶ t mesures $\Rightarrow p$ parts
- ▶ t parts nécessaires pour la reconstruction
- ▶ AMD assure une confidentialité, mais plus faible que AMS
- ▶ **Défauts:** pas d'authentification, introduit des délais
- ▶ **Avantages:** surcoût de communication optimal

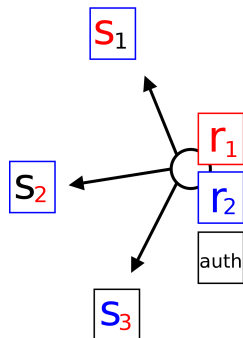
Surcoût: $\frac{p-t}{p}$ messages supplémentaires par mesure

Agrégation Multi-chemins Dispersée Authentifiée (AMD-A)



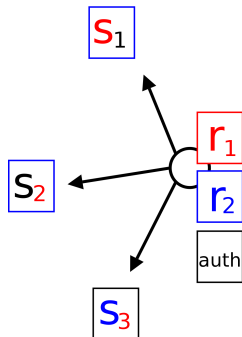
- ▶ AMD-A remplace la dernière mesure d'un bloc de AMD avec un **jeton d'authentification**
- ▶ Si une part est falsifiée, le jeton qui sera reconstruit sera invalide

Agrégation Multi-chemins Dispersée Authentifiée (AMD-A)



- ▶ AMD-A remplace la dernière mesure d'un bloc de AMD avec un **jeton d'authentification**
- ▶ Si une part est falsifiée, le jeton qui sera reconstruit sera invalide
- ▶ AMD-A coûte légèrement plus que AMD, et offre la même confidentialité

Agrégation Multi-chemins Dispersée Authentifiée (AMD-A)



- ▶ AMD-A remplace la dernière mesure d'un bloc de AMD avec un **jeton d'authentification**
- ▶ Si une part est falsifiée, le jeton qui sera reconstruit sera invalide
- ▶ AMD-A coûte légèrement plus que AMD, et offre la même confidentialité
- ▶ **Défauts:** introduit des délais
- ▶ **Avantages:** Offre une protection contre toutes les attaques considérées

Surcoût: $\frac{p-t+1}{p-1}$ messages supplémentaires par mesure

Écoute

- ▶ Pire cas: t noeuds compromis permettent de reconstruire des messages

Écoute

- ▶ Pire cas: t noeuds compromis permettent de reconstruire des messages

Falsification

- ▶ Pire cas: t noeuds compromis permettent de contrôler la valeur des agrégats

Écoute

- ▶ Pire cas: t noeuds compromis permettent de reconstruire des messages

Falsification

- ▶ Pire cas: t noeuds compromis permettent de contrôler la valeur des agrégats

Dénis de service

- ▶ Pire cas: $p - t + 1$ noeuds compromis permettent de bloquer les reconstructions

En pratique \Rightarrow la complexité du réseau **augmente ces seuils**

- ▶ Toutes les techniques sont implémentés dans des capteurs MICAz
- ▶ \approx 4000 lignes de nesC pour TinyOS
- ▶ 2 à 3 kB en RAM, 47 à 50 kB de code compilé
- ▶ À la fois des **expérimentations en vraie grandeur** et des **simulations**

- ▶ Toutes les techniques sont implémentés dans des capteurs MICAz
- ▶ \approx 4000 lignes de nesC pour TinyOS
- ▶ 2 à 3 kB en RAM, 47 à 50 kB de code compilé
- ▶ À la fois des **expérimentations en vraie grandeur** et des **simulations**

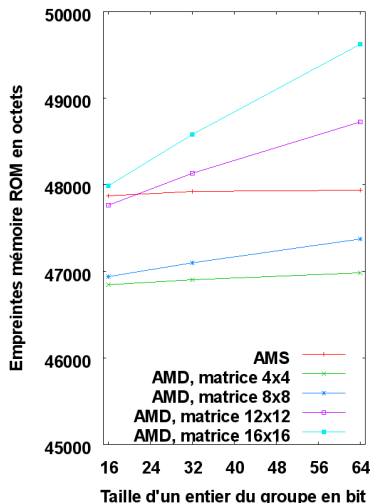
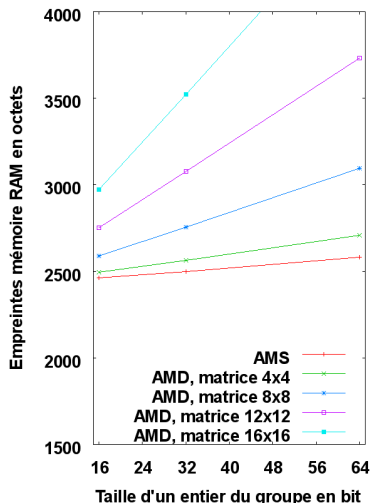
- ▶ Plutôt une **preuve de concept** qu'une solution clef en mains complète
- ▶ Montre que ces méthodes sont réalistes

- ▶ Toutes les techniques sont implémentés dans des capteurs MICAz
- ▶ \approx 4000 lignes de nesC pour TinyOS
- ▶ 2 à 3 kB en RAM, 47 à 50 kB de code compilé
- ▶ À la fois des **expérimentations en vraie grandeur** et des **simulations**

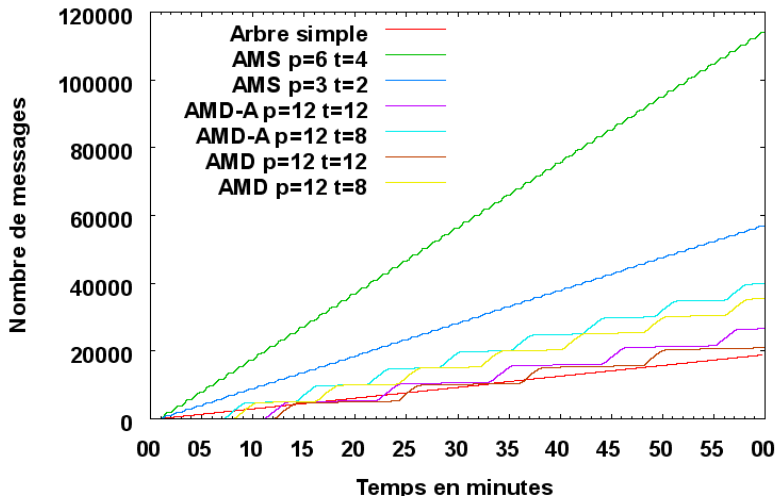
- ▶ Plutôt une **preuve de concept** qu'une solution clef en mains complète
- ▶ Montre que ces méthodes sont réalistes

- ▶ Les résultats confirment la théorie

Consommation de mémoire



Surcoût des communications



Contributions

- ▶ Une nouvelle approche de la sécurité de l'agrégation
- ▶ 3 techniques de sécurité pour l'agrégation de données

Travaux futurs / discussions ouvertes

- ▶ Analyse de la sécurité plus détaillée
 - ▶ Comportement statistique en présence de captures aléatoires
 - ▶ Quantifier le plus possible
- ▶ Généraliser les techniques pour d'autres contextes que les réseaux de capteurs