Concept de Thread et de Processus Multi-Threads

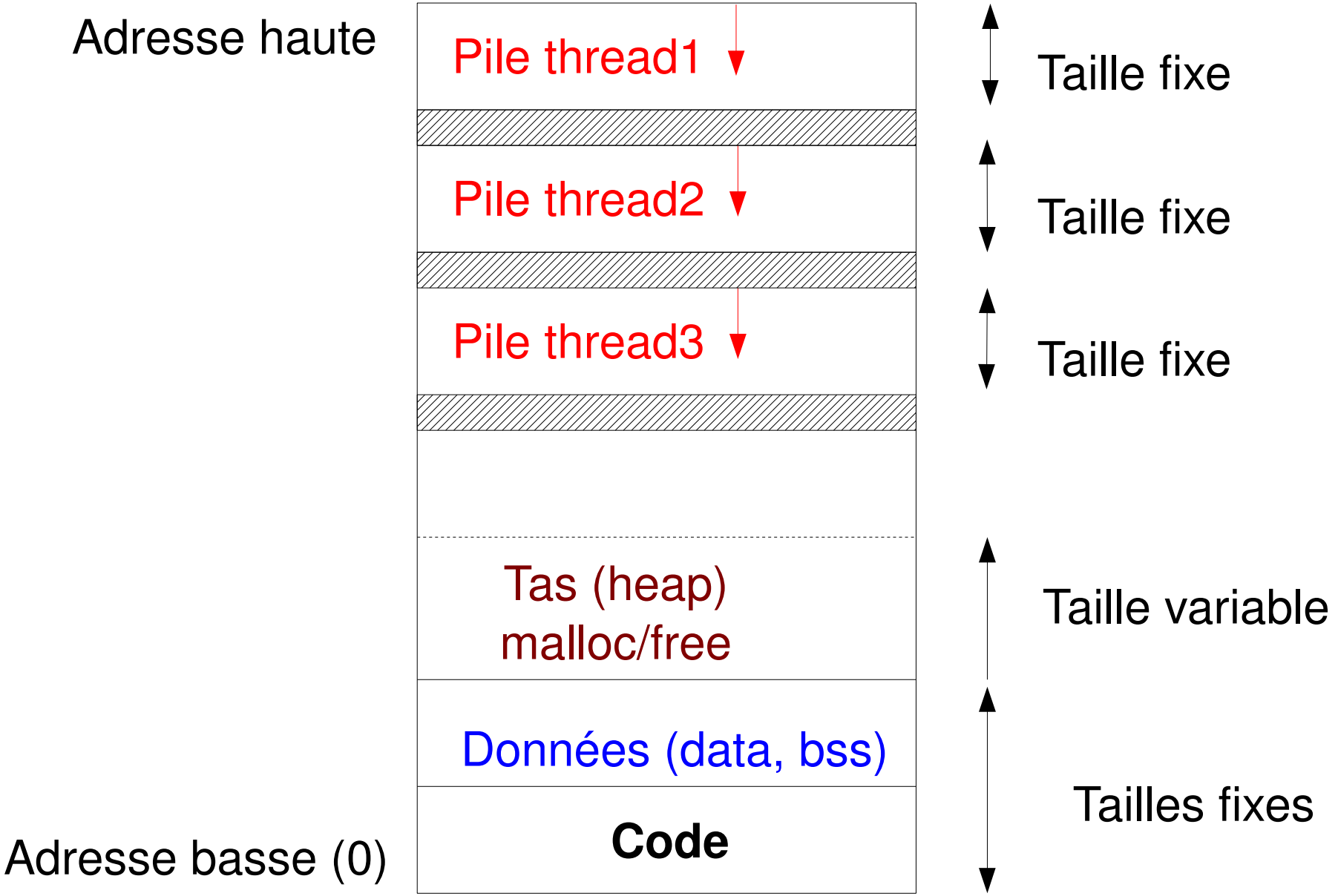
Concept de Thread

- Notion combinant avantages :
  - Exécution parallèle
  - Partage code et données applicatifs communs
- Facile à mettre en oeuvre
  - Programmable
  - Contrôlable
  - Configurable
- Efficace et "scalable"

Processus ≠ Thread

- Processus ne convient pas
  - Contexte lourd, coûteux à créer et à détruire
  - Pas efficace (changement de contexte lent)
  - Contexte mémoire important
- Partage de données "pas naturel"
  - Pas intégré dans langage de programmation
  - Offerts par services OS complexes, pas souples
- Programmation difficile, non contrôlable

Processus Multi-Threads



Thread – Objet Programmable

- Dérivée de notion de processus
  - Unité d'exécution concurrente dans un même programme
- Threads partagent espace d'adressage virtuel du processus "englobant"
  - Code
  - Données globales
  - Tas (malloc/free)
- Partagent tout ou partie ressources OS

Thread – Objet Programmable

- Thread principale créée par système
- Autres threads créées et détruites par application
- Exécutent même code ou des fonctions dédiées
- Contexte de travail privé alloué par application

Serveur Multi-threads

```
static endpoint_t srv_endpoint;

service_thread() {
    struct request_t rqst;
    struct reply_t reply;
    for (;;) {
        wait_client_request(&srv_endpoint, &rqst);
        process_request(&rqst, &reply);
        send_reply_to_client(&srv_endpoint, &reply);
    }
}

main(argc, argv) {
    create_endpoint(SERVICE_ID, &srv_endpoint);
    for (i = 0; i < nb_serv_th; i++)
        thread_create(service_thread, THREAD_PRIO);
}
```

Thread – Objet OS

- Unité d'exécution indépendante
  - Pile(s) d'exécution(s)
    - Mode utilisateur du CPU (pile applicative)
    - Mode superviseur du CPU (pile système)
  - Contexte CPU (registres)
  - Etat courant et attributs d'ordonnancement
  - Identificateur
- Entité d'attribution d'un CPU par ordonnanceur du système