Projet de fin d'étude

Déchiffrement de disque LUKS sous FreeBSD

Romain CHERRÉ, Pierre KOEBELIN

31 janvier 2019

Plan

Introduction

État de l'art

Développement

Utilisation

Difficultés rencontrées

Démonstration

Conclusion

Introduction

Chiffrement de disque

- **Linux** : géré par *dm-crypt*
- **FreeBSD** : géré par *GELI*

  $\rightarrow$ modules de noyau

Chiffrement de disque

- **Linux** : géré par *dm-crypt*
- **FreeBSD** : géré par *GELI*

  $\rightarrow$ modules de noyau

Création d'un module noyau

- Module pour **FreeBSD**
- Lire et écrire disques chiffrés *LUKS*

État de l'art

Différents types de chiffrement

Chiffrement à différents niveaux

- Chiffrement matériel
- **Chiffrement logiciel de disque**
- Système de fichiers chiffré
- Chiffrement de fichier

Chiffrement et organisation dans le système

Chiffrement logiciel

- Géré par des modules noyau
- Possible avec différents algorithmes
- Réalisé au moment de la lecture ou l'écriture
- Transparent pour l'utilisateur

Chiffrement et organisation dans le système

Sur **Linux**

- Outil *cryptsetup*
- Module *dm-crypt*
- Standard *LUKS* : *Linux Unified Key Setup*
- Empilement avec d'autres transformations *device-mapper*

Sur **FreeBSD**

- Outil et module *GELI*
- Paramètres possibles codés en dur
- Empilement avec d'autres transformations *GEOM*

Chiffrement et organisation dans le système - **Linux**

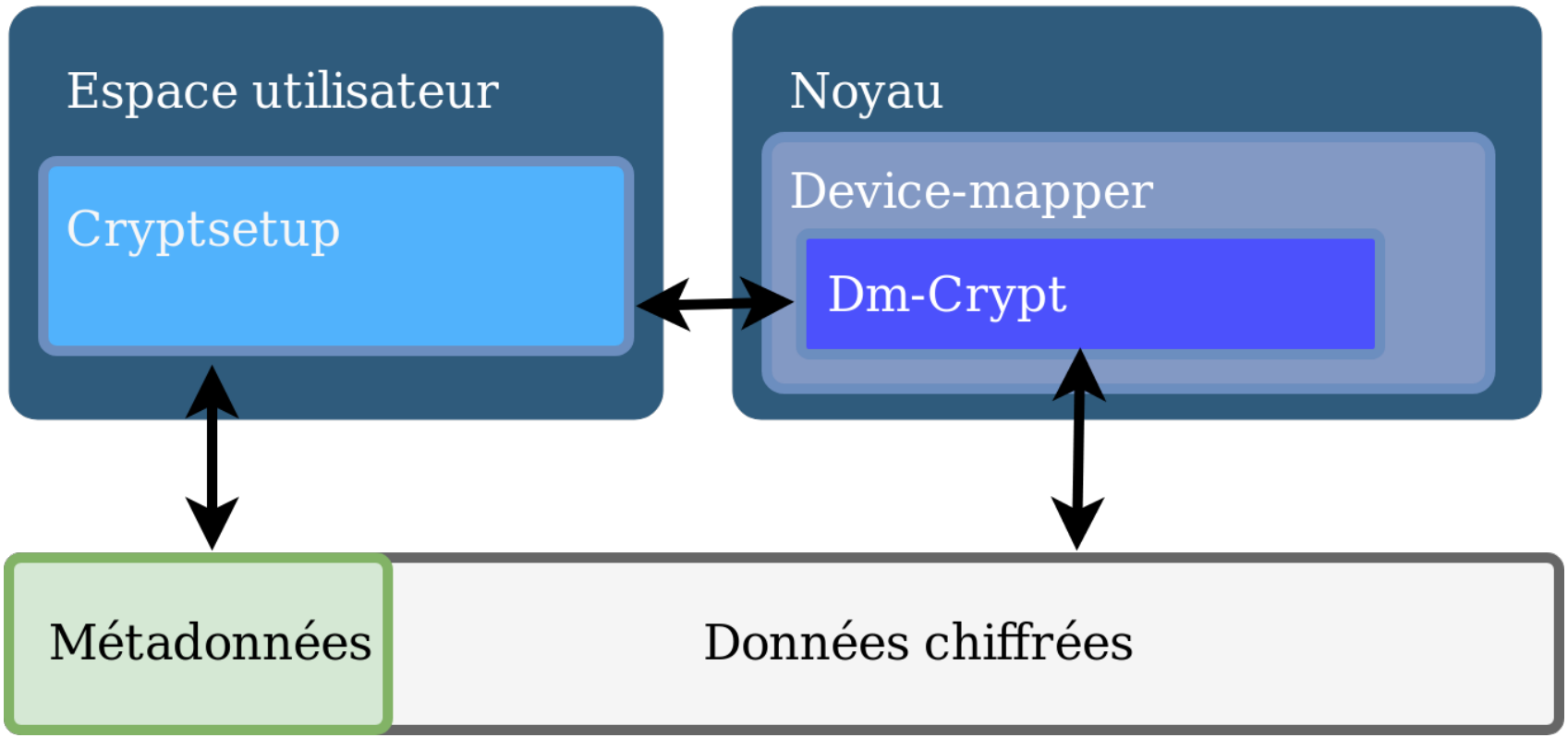

FIGURE – Organisation du code dans Linux

Chiffrement et organisation dans le système - **FreeBSD**

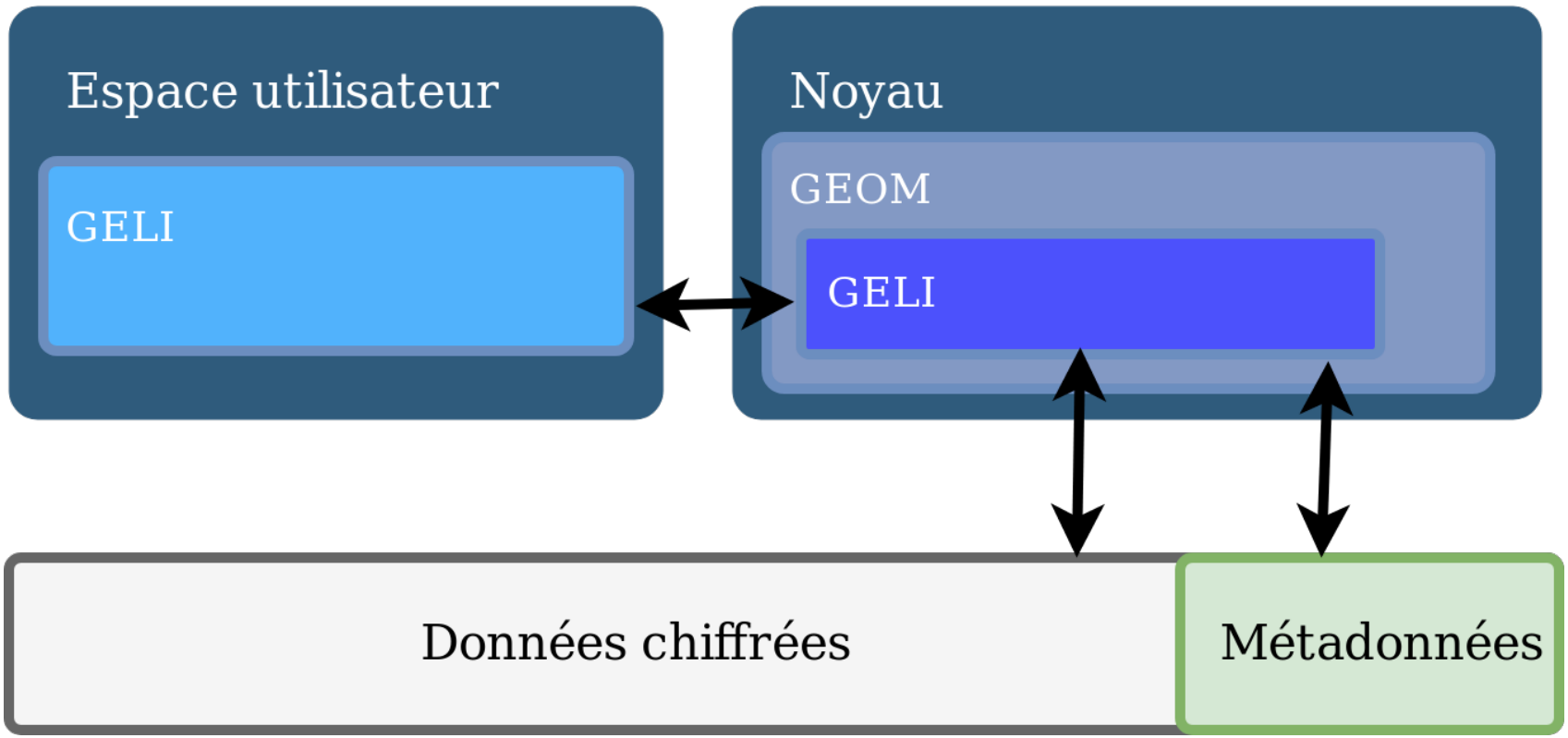

FIGURE – Organisation du code dans FreeBSD

Format sur le disque

Format = organisation des données et métadonnées sur le disque

Format *LUKS*

- *LUKS header* → métadonnées
- *Keyslots* → 8 phrases secrètes
- Données chiffrées

Format *GELI*

- Données chiffrées
- *GELI header* → métadonnées

Format sur le disque

FIGURE – Format sur disque *LUKS*

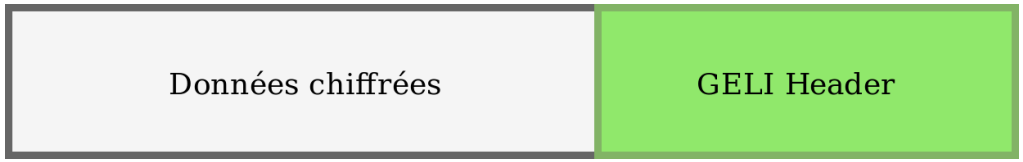

FIGURE – Format sur disque *GELI*

Algorithmes

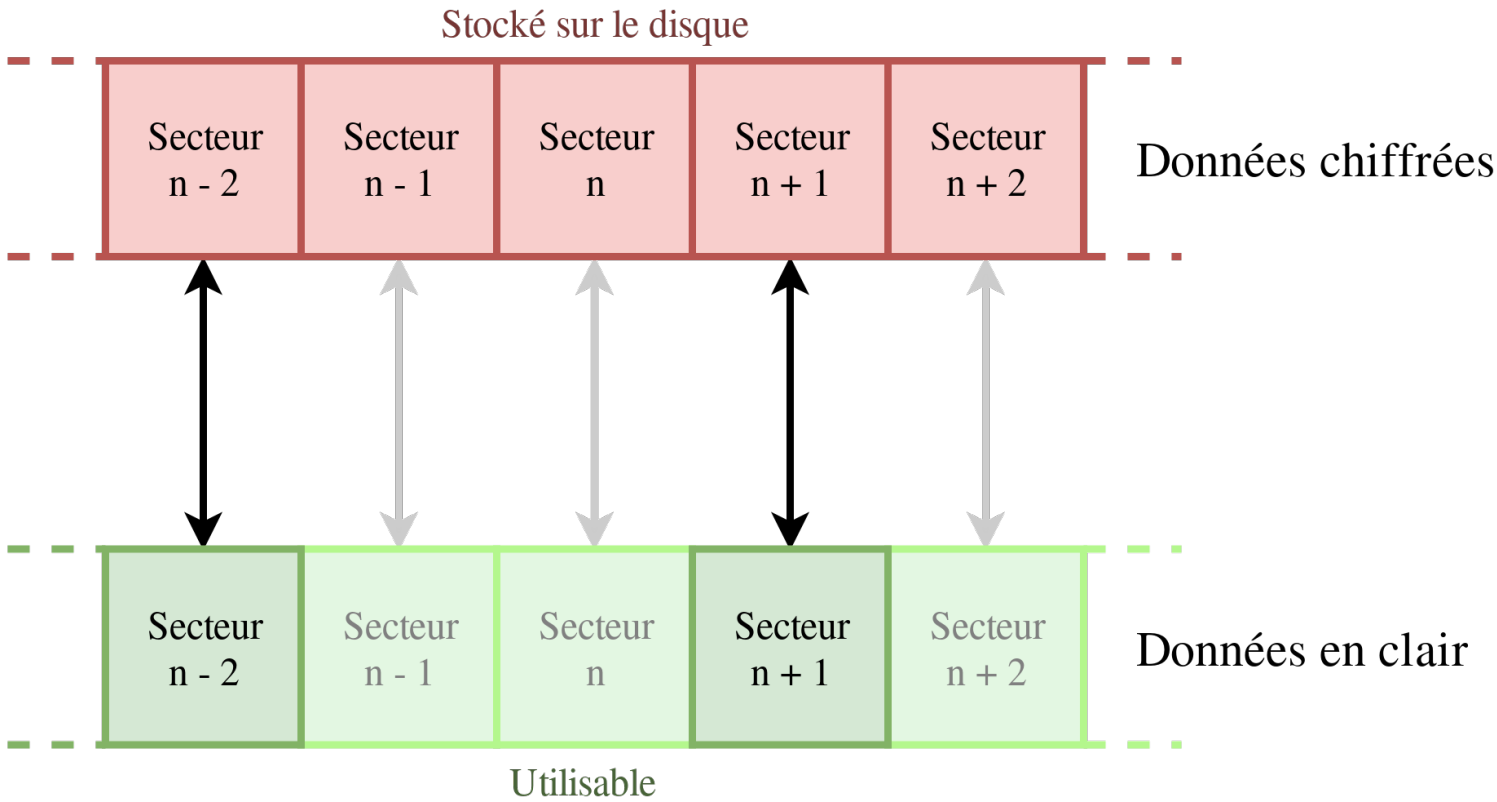

FIGURE – Chiffrement par secteur

Algorithmes

Chiffrement

- Même algorithme pour tout le disque

Vecteur d'initialisation

- Unique à chaque secteur
- Secteurs de 512 ou 4096 octets

Algorithmes

	Chiffrement		
Génération d'IV	`AES-XTS`	`AES-CBC`	`CAST5-CBC`
`plain64`	**X**		
`plain`		**X**	**X**
`essiv:sha256`		**X**	

Développement

Les métadonnées

Des métadonnées en communs

- Le *MAGIC*
- Version
- Algorithme de chiffrement et de hashage
- Sel
- Nombre d'itérations pour PKCS5v2
- Clé maître chiffrée

Les métadonnées - **FreeBSD**

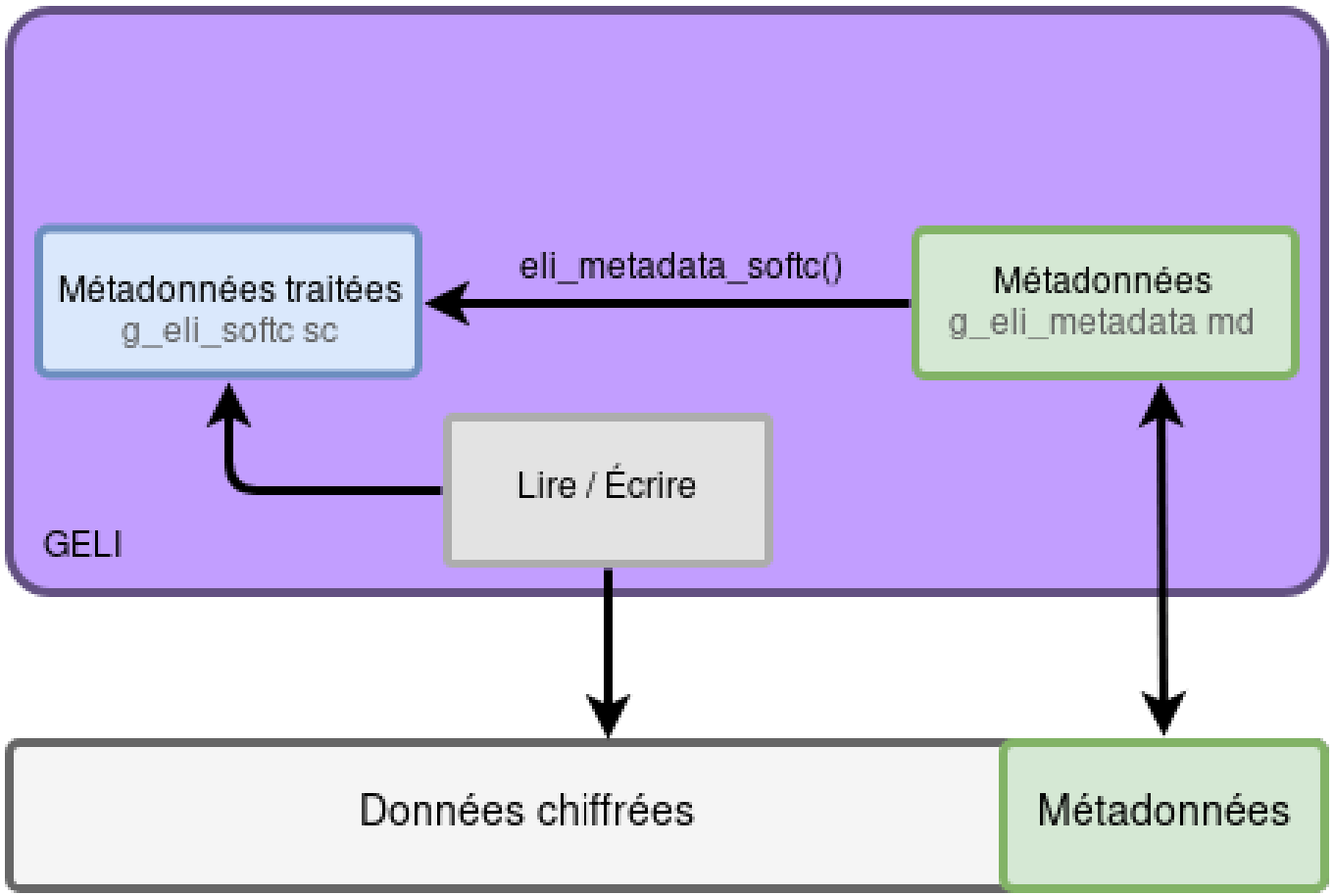

FIGURE – Utilisation des métadonnées dans FreeBSD

Les métadonnées - Utilisation dans le code

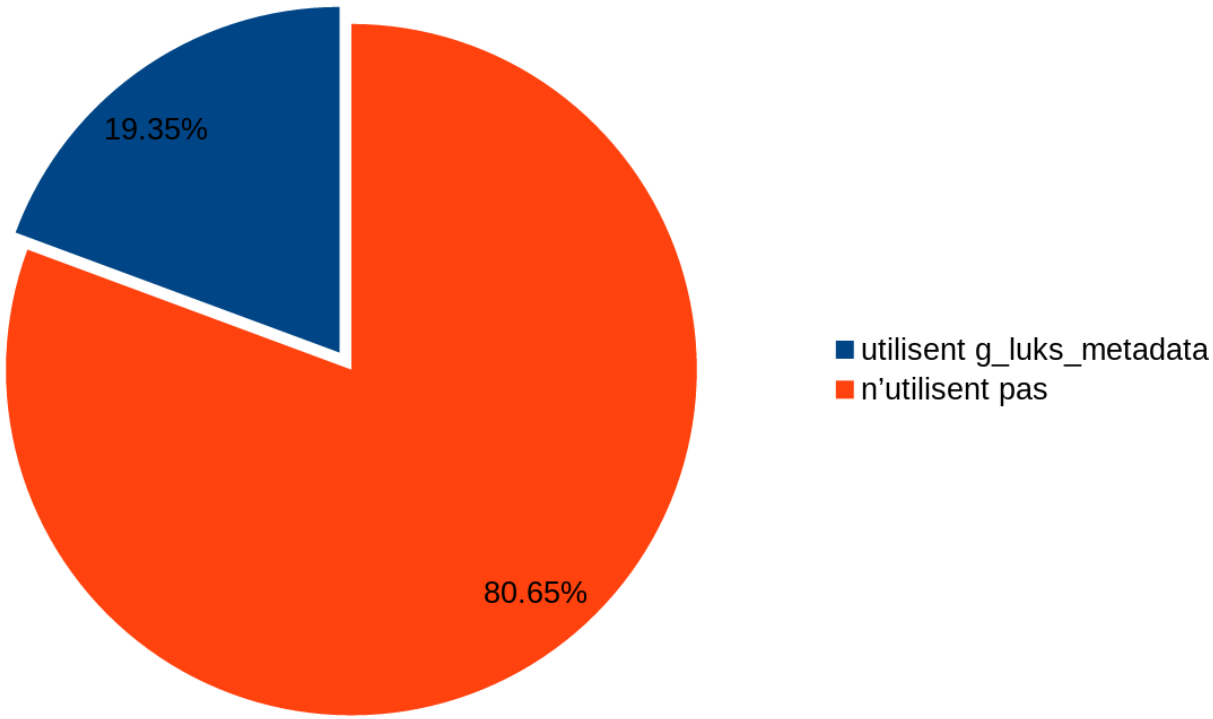

FIGURE – Utilisation de la structure g_luks_metadata

Les métadonnées - Utilisation dans le code

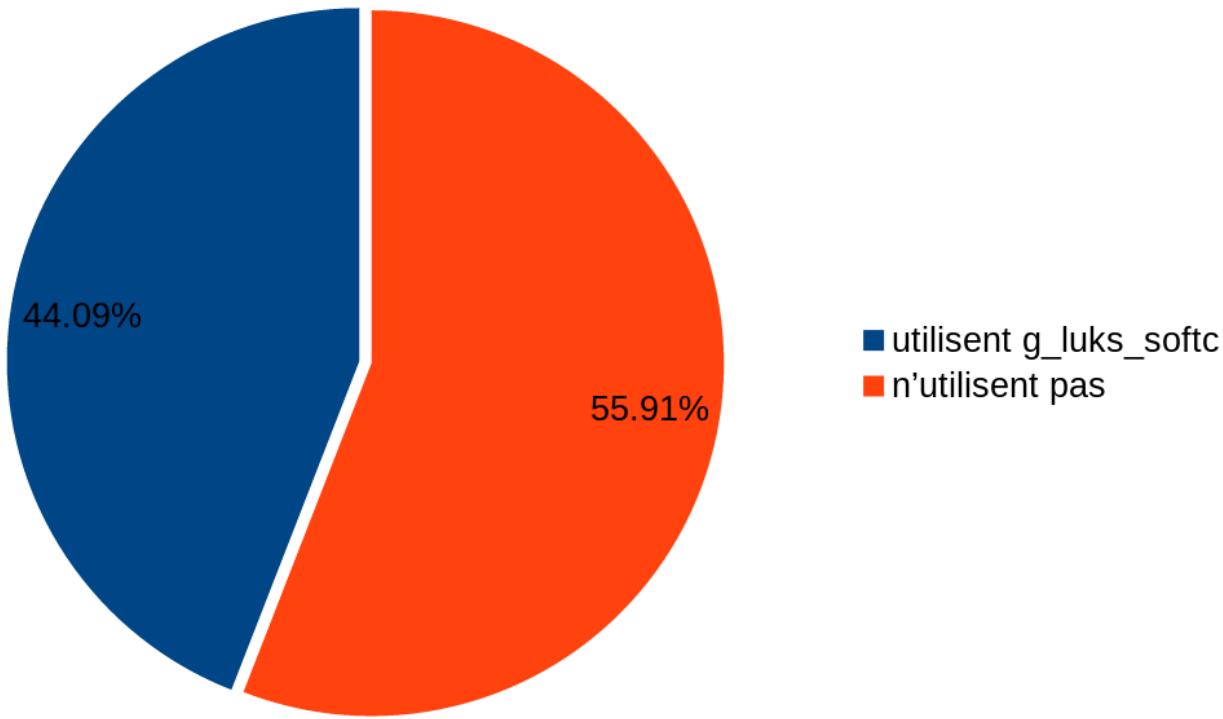

FIGURE – Utilisation de la structure g_luks_softc

Les métadonnées - Transformation des métadonnées

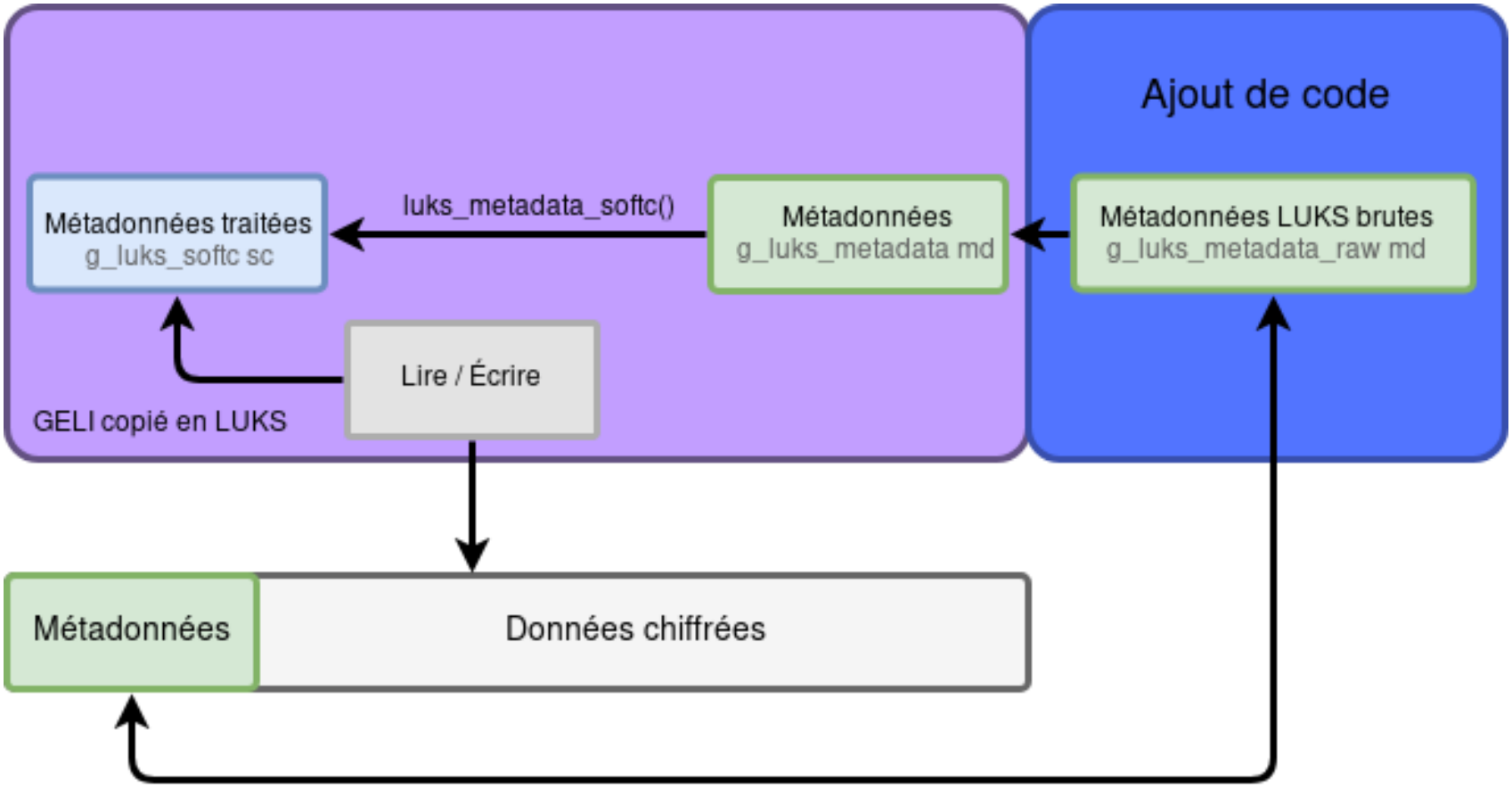

FIGURE – Introduction d'un structure intermédiaire

Déchiffrement de la clé

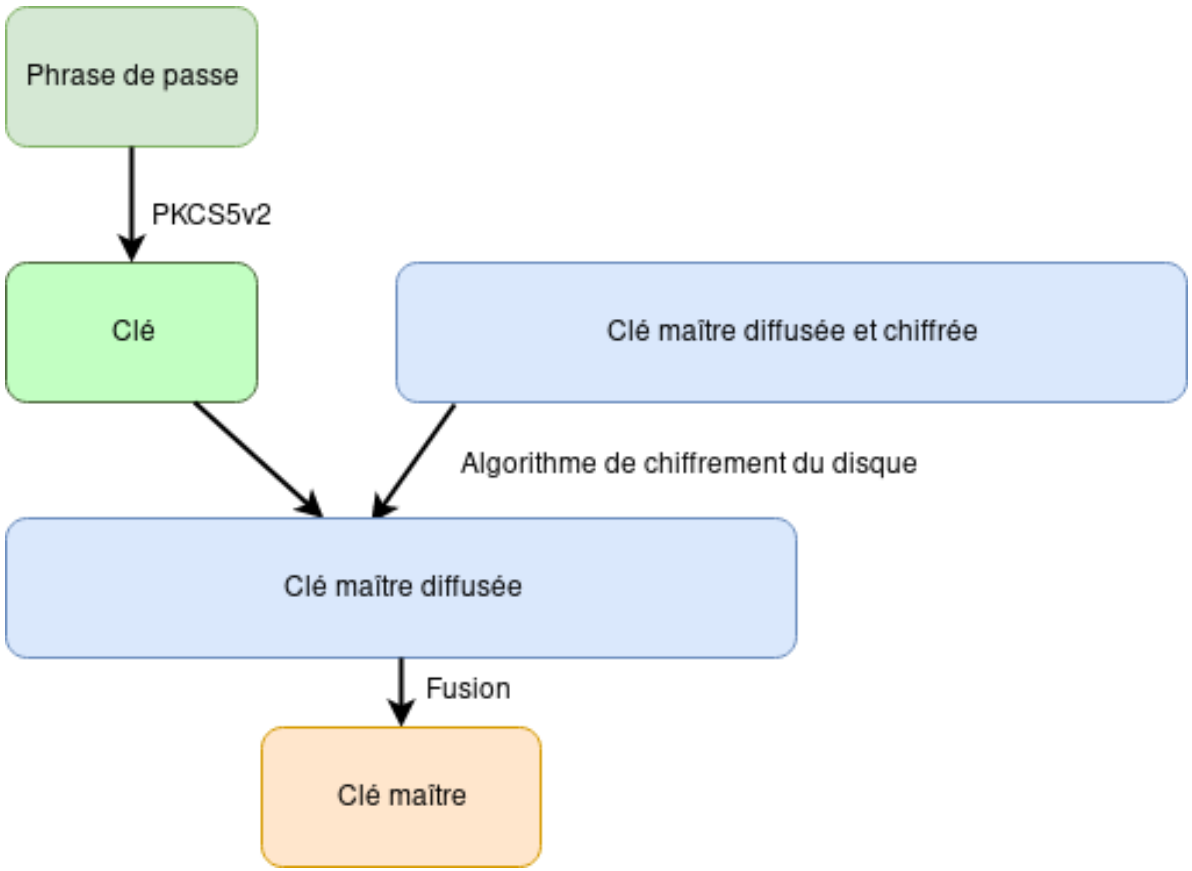

FIGURE – Déchiffrement de la clé sous LUKS

Utilisation de drapeaux *GELI*

Activer/désactiver options de chiffrement de disques

- `G_ELI_FLAG_AUTH`
  $\rightarrow$ activer l'auhentification
- `G_ELI_FLAG_SINGLE_KEY`
  $\rightarrow$ utiliser la même clé maître pour tous les secteurs

Création du disque déchiffré

FIGURE – Déchiffrement de la clé sous LUKS

Quelques changements par rapport à *GELI*

- Taille du disque

Création du disque déchiffré

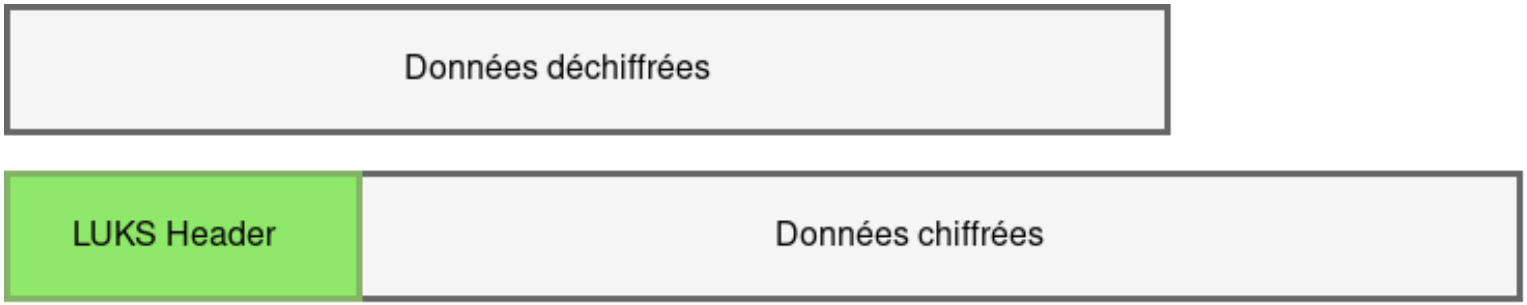

FIGURE – Déchiffrement de la clé sous LUKS

Quelques changements par rapport à *GELI*

- Taille du disque
- Décalage

Création du disque déchiffré

FIGURE – Déchiffrement de la clé sous LUKS

Quelques changements par rapport à *GELI*

- Taille du disque
- Décalage
- Lecture de métadonnées au début du disque

Passage de l'espace utilisateur au noyau

API GEOM

- Structure `g_ctl_req`
- Instruction `g_ctl_issue`
- Structure `g_command`

Passage de l'espace utilisateur au noyau

Code noyau ou espace utilisateur ?

- API noyau Opencrypto - Openssl
- Malloc
- HMAC et PKCS5v2
- Dérivation de la phrase de passe

Passage de l'espace utilisateur au noyau - Implémentation

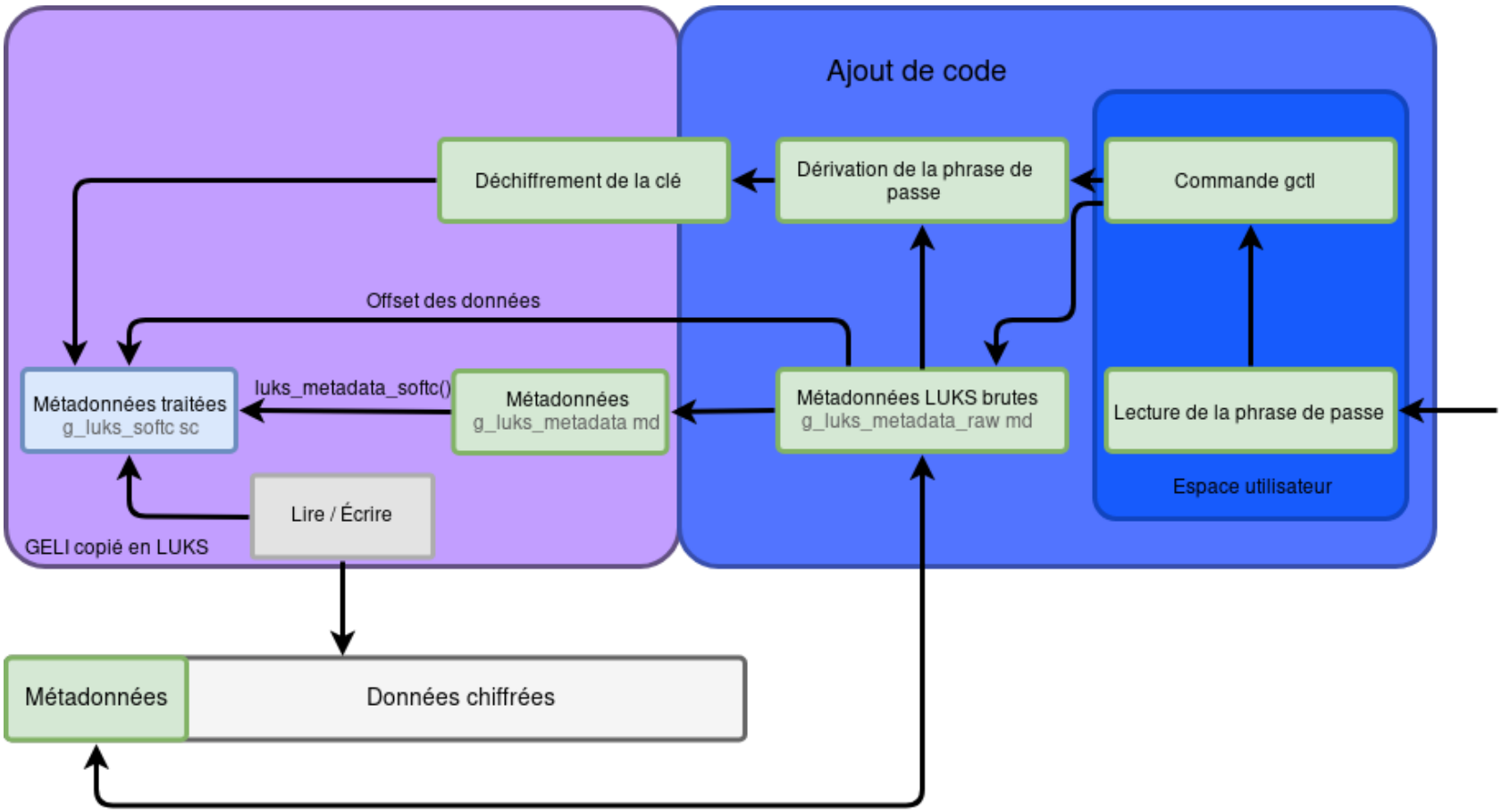

FIGURE – Fonctionnement de GEOM_LUKS

Utilisation

Utilisation

Le système de fichiers

- EXT2, EXT3
- EXT4 : lecture seule sur **FreeBSD**
- ZFS
- FAT, FAT32
- NTFS

Utilisation

Partage de fichiers et haute disponibilité

- Dual boot
- Partage réseau : iSCSI, NFS, ...
- Supports amovibles

Difficultés rencontrées

Difficultés rencontrées

Noyau et userspace

- différentes bibliothèques
- algorithme de hashage : différents types
- allocation dynamique

Difficultés rencontrées

Débogage : dmesg et kgdb

- kern.geom.luks.debug
- printf
- kgdb et vmcore

Démonstration

Conclusion

Conclusion

Module actuel : support de

- lecture / écriture
- SHA256
- AES-XTS, AES-CBC
- plain et plain64

Conclusion

Module actuel : support de

- lecture / écriture
- SHA256
- AES-XTS, AES-CBC
- plain et plain64

Suite du projet

- amélioration du code
- scripts de tests
- revue par les développeurs FreeBSD
- ajout de fonctionnalités

Questions ?

Démarrage

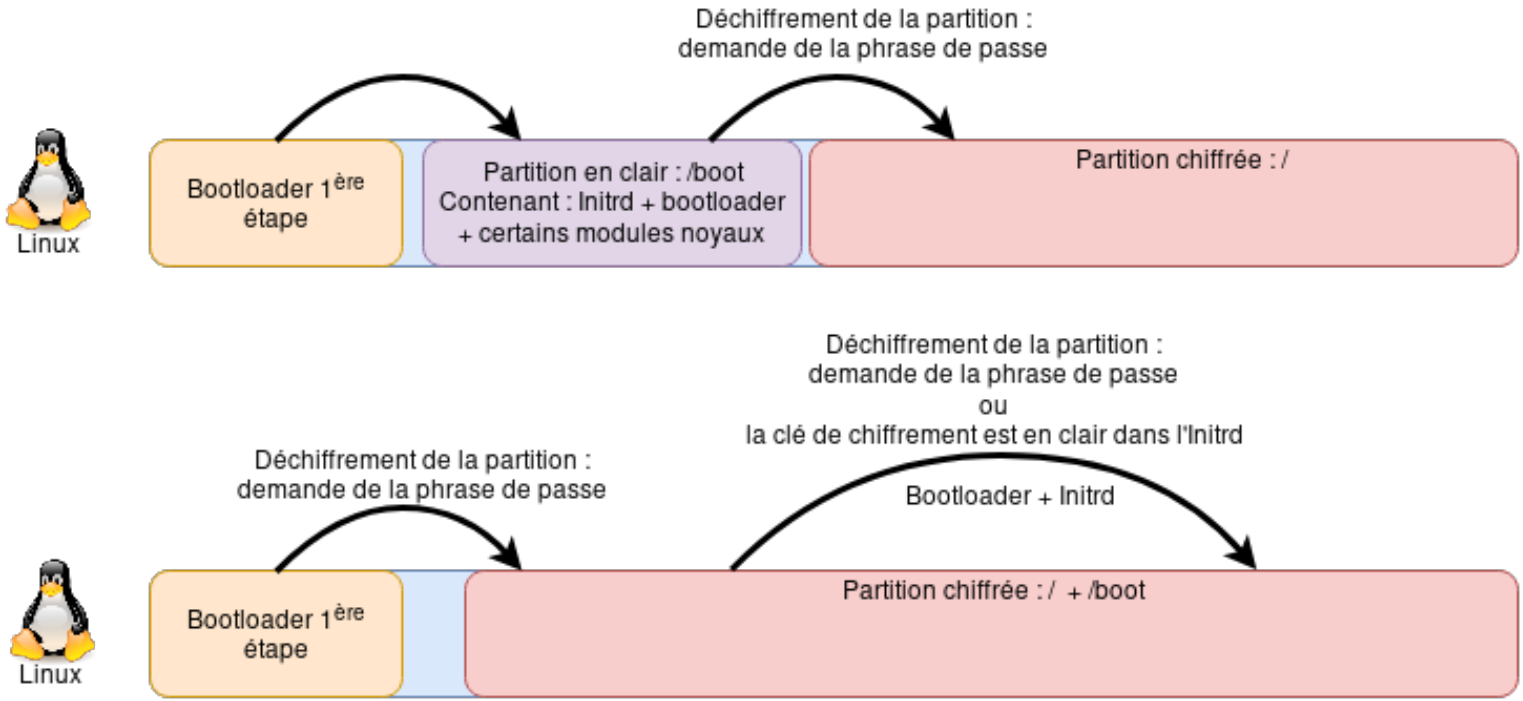

FIGURE – Démarrage sous Linux

Démarrage

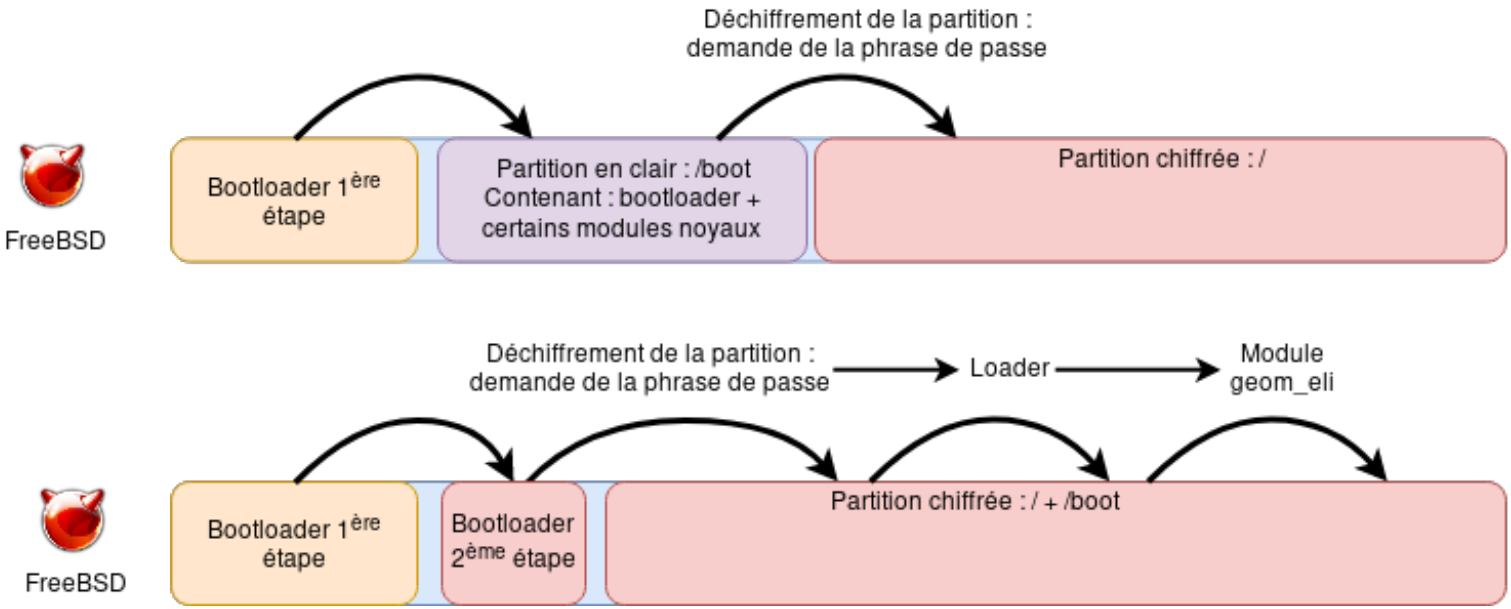

FIGURE – Démarrage sous FreeBSD