ENSI CAEN
ÉCOLE PUBLIQUE D'INGÉNIEURS
CENTRE DE RECHERCHE

Implémentation d'un "find forensic" en Go pour images EWF

Projet 2A – Informatique

Encadré par l'équipe SAFE du GREYC

ENCADRANTS: Tanguy GERNOT / Emmanuel GIGUET
ETUDIANTS: Lino LANDRY / Florestan TRILLOT / Rania MEDIANE / Chloë BARTOLONE / Isaac ABOYO / Jean BOUDINSKI

Sommaire

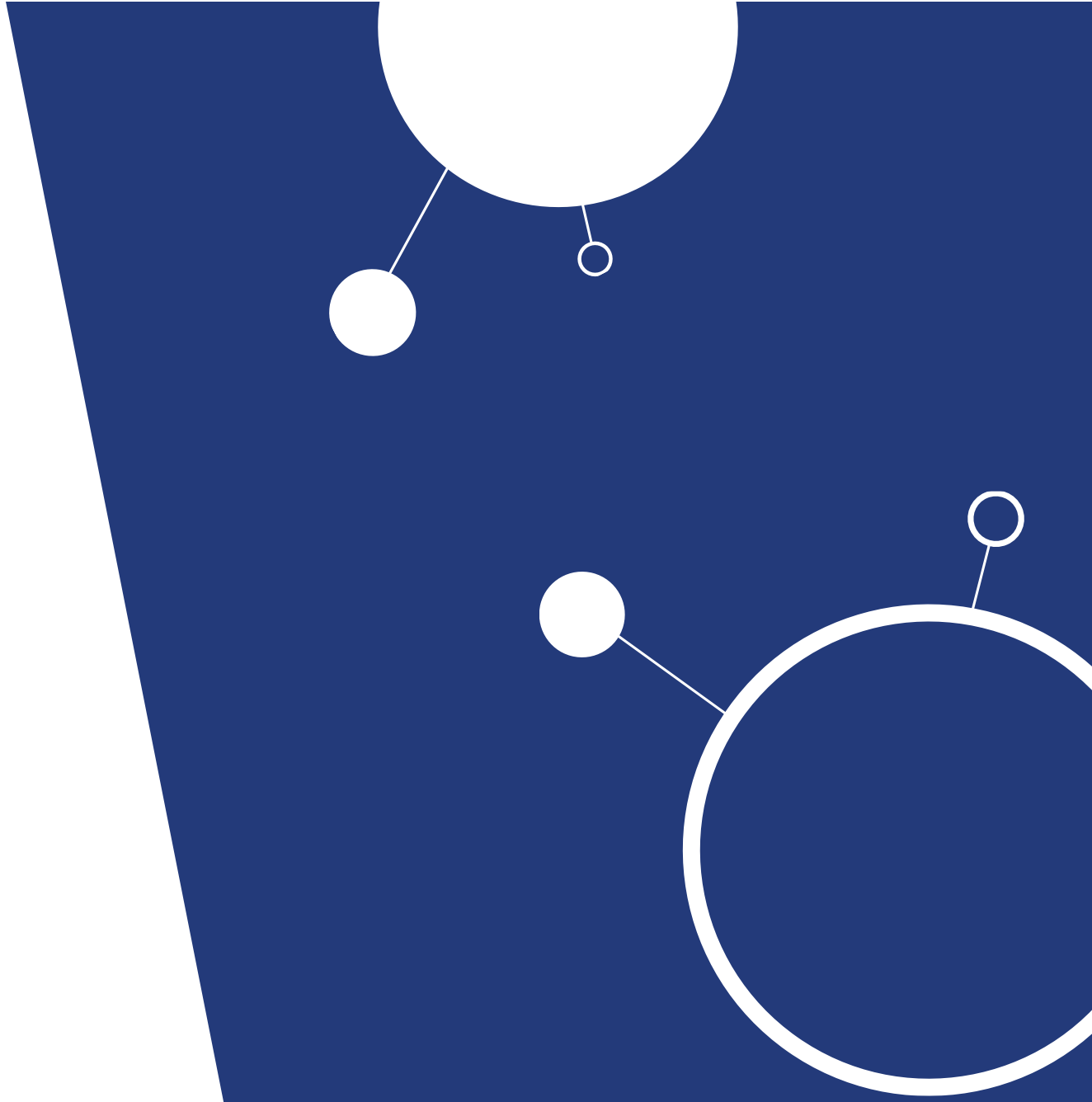
- Introduction et contexte
- Enjeux et solutions techniques
- Architecture globale
- Parcours de l'arborescence de fichiers
- Interface Utilisateur
- Bilan et suite

ENSI CAEN
ÉCOLE PUBLIQUE D'INGÉNIEURS
CENTRE DE RECHERCHE

Introduction

Contexte et besoins

ENSI CAEN
ÉCOLE PUBLIQUE D'INGÉNIEURS
CENTRE DE RECHERCHE



Introduction et contexte

Forensic = investigation numérique
-> concerne 85% des affaires criminelles

- Historique de navigation
- Documents suspects
- Communication
- Dossiers cachés

EWF - Expert Witness Format

Besoins

Aider les forces de l'ordre à trouver des documents sensibles, repérer des fichiers récents ou suspects

Garantir une exploitation rapide des données contenues dans la pièce à conviction pour trouver des preuves dans le temps d'une garde à vue

Outil **Multiplateforme** fonctionnel sur Windows et Linux

Notre solution

EWF
Expert Witness Format

Format standardisé et sécurisé utilisé pour figer les pièces à conviction numériques

Création d'un outil "find forensic" pour chercher des données sensibles directement depuis l'image EWF sans extraction

ÉTAT DE L'ART

Commande "find" classique

- **Points forts :** commande système simple d'utilisation.
- **Inconvénients :** Nécessite une extraction préalable de l'image. Incapable de lire nativement des formats d'archives forensiques comme l'EWF

Logiciels d'investigation type "Autopsy" / "EnCase"

- **Points forts :** Analyse exhaustive, interface complète, gestion de cas complexes.
- **Inconvénients :** Logiciels extrêmement lourds, payants (pour EnCase), nécessitant des ressources machines importantes pour une simple recherche de fichiers.

Enjeux et solutions techniques

ENSI CAEN
ÉCOLE PUBLIQUE D'INGÉNIEURS
CENTRE DE RECHERCHE

Méthodologie

Architecture Modulaire:

Module Find : Un package Go pur gérant toute la logique forensique (ouverture EWF, parcours de partitions, filtrage de fichiers).

Module UI : Gère uniquement l'affichage et l'interaction avec l'utilisateur via le framework Qt.

Apprentissage:

Langage Go et librairies spécialisées

Gestion de projet et Outils:

Versionnage avec GIT

Développement itératif : Priorisation des fonctionnalités critiques (lecture EWF, filtres de base) avant d'aborder l'interface graphique.

Enjeux techniques

01 L'accès bas niveau aux données

Lire un format propriétaire complexe (EWF) et détecter automatiquement le système de fichiers sous-jacent

02 Parcours efficace et Filtres combinés

Garantir un parcours efficace tout en appliquant de multiples filtres combinés (nom, taille, dates) sur des milliers de fichiers potentiels.

03 Interface Graphique + CLI

Développer une véritable IHM de type 'Desktop' avec Qt. Et une interface accessible depuis le terminal qui permet d'effectuer une recherche dans une archive EWF

Solutions techniques

L'accès bas niveau aux données

01

Utilisation de go-tsk Une seule librairie pour tous les systèmes de fichiers (ext4, ntfs) interfacée en EWF

L'appel à la fonction **tsk.OpenImageExtern** permet d'ouvrir directement les fichiers EWF (ex: .e01) de manière logicielle, sans nécessiter de point de montage sur le système d'exploitation hôte

Implémentation de la fonction **dir.Walk** de go-tsk avec une fonction de filtrage personnalisée en Go Utilisation du principe de ***"early return"*** : le fichier est testé séquentiellement contre chaque filtre actif (nom, taille, date). Dès qu'un critère n'est pas respecté, la fonction s'arrête.

02

Parcours efficace et Filtres combinés

Conception d'une IHM ergonomique : Développement d'une interface graphique native via le Framework **Qt**, permettant une saisie structurée des filtres de recherche (dates, tailles, noms).
Interopérabilité Go/C++ : Utilisation des **bindings** therecipe/qt pour l'interfaçage entre la logique métier en Go et les composants graphiques C++ de Qt.

Interface Graphique + CLI

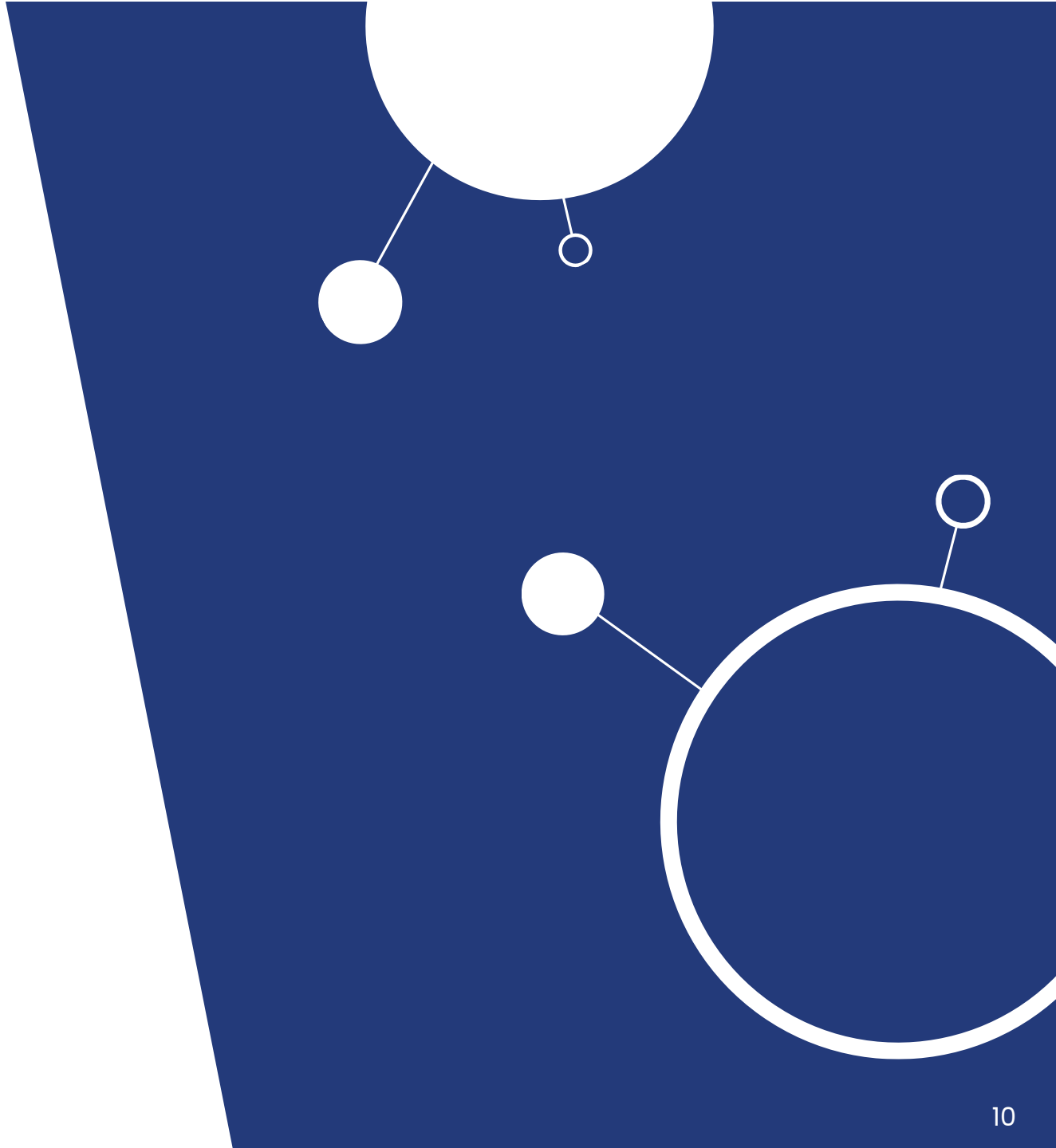
03

La mise en place de

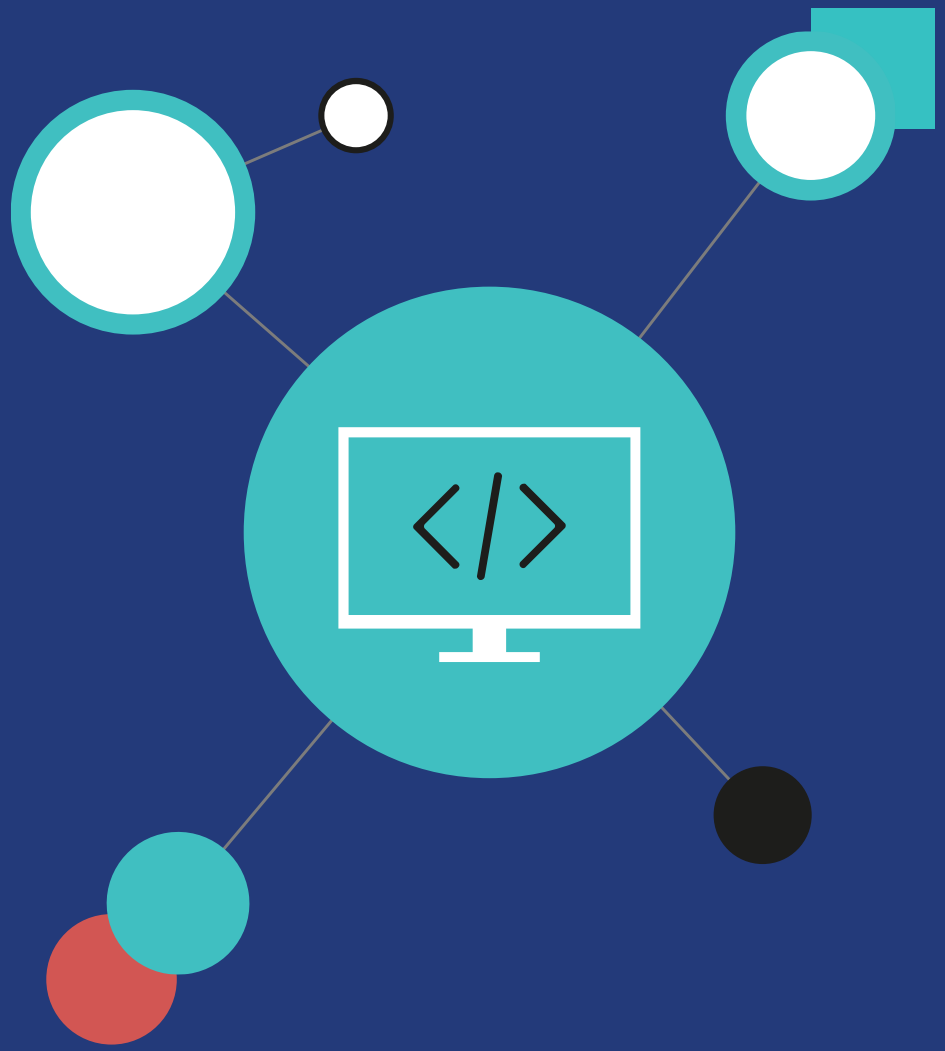
L'architecture

Problèmes & Solutions

ENSI CAEN
ÉCOLE PUBLIQUE D'INGÉNIEURS
CENTRE DE RECHERCHE



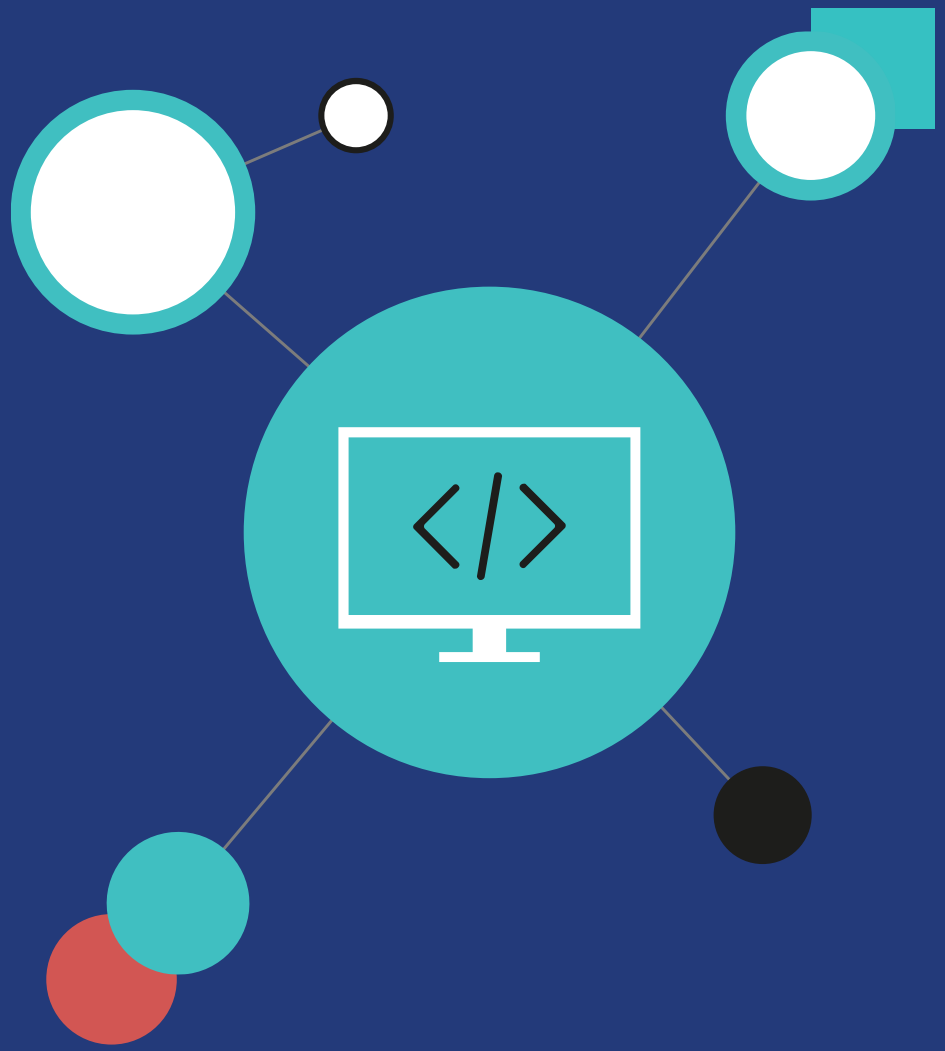
Architecture



Interface TSK – EWF

- Librairie EWF : ouverture et lecture des archives
- Librairie TSK : ouverture et lecture des disques
- Réécriture des fonctions read et seek appelées par TSK pour exploiter EWF
- Point d’entrée en une unique fonction qui ouvre l’archive EWF et l’image disque

Architecture



Exposition des librairies en Go

- Fournit une unique fonction ouvrant l'archive et l'image disque contenue
- Accès aux métadonnées des fichiers (date, taille)
- Fonction de parcours (fonctions d'ordre supérieur)

Les étapes du

Parcours de l'arborescence

Processus & Données

ENSI CAEN
ÉCOLE PUBLIQUE D'INGÉNIEURS
CENTRE DE RECHERCHE

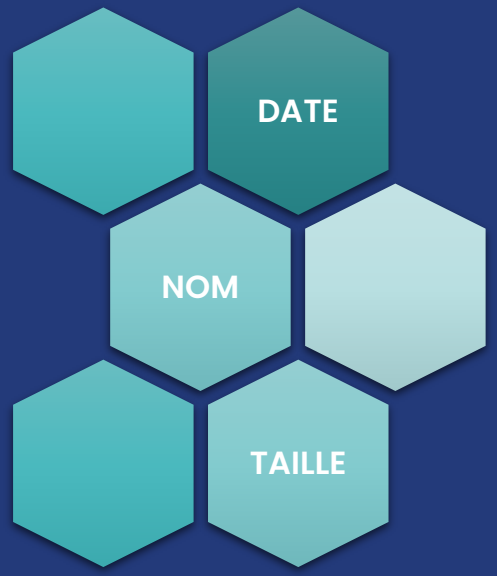
Parcours de l’arborescence de fichiers

EXTRACTION DE MÉTADONNÉES

MOTEUR DE FILTRAGE

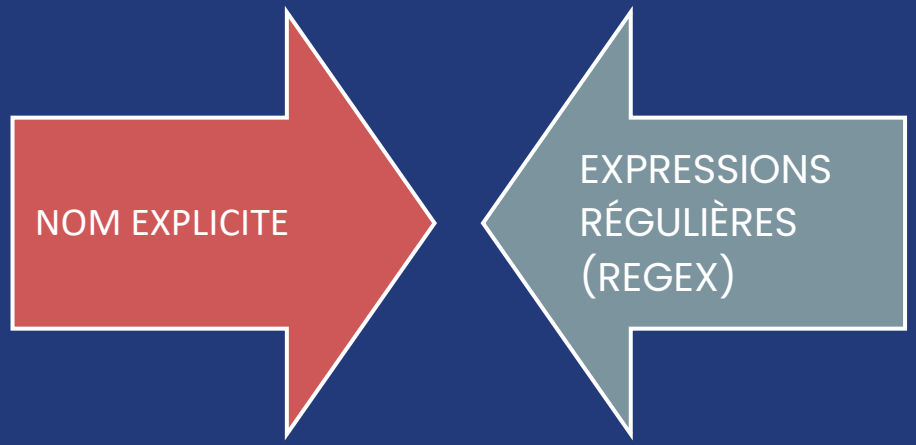
ELIMINATION DE DOSSIERS

Extraction des métadonnées



- **Métadonnées C** : La structure C offre un accès aux noms, tailles, etc.
- **Limitation de Go-TSK** : Ces données n'étant pas mappées en Go, l'utilisation directe du module était insuffisante.
- **Extension par interfaçage** : Module étendu en interfaçant les structures C existantes pour les rendre exploitables en Go.

Conception du moteur de filtrage



Exemple :

(?i)\.(png|jpe?g|gif|bmp|tiff|webp)$

- (?i) : insensible à la casse
- \. : Cherche le caractère "point" qui précède l'extension.
- (png|jpe?g|gif|bmp|tiff|webp) : toutes extensions
- jpe?g : "e" optionnel (trouve .jpg et .jpeg).
- $: Force la correspondance à la fin du nom du fichier.

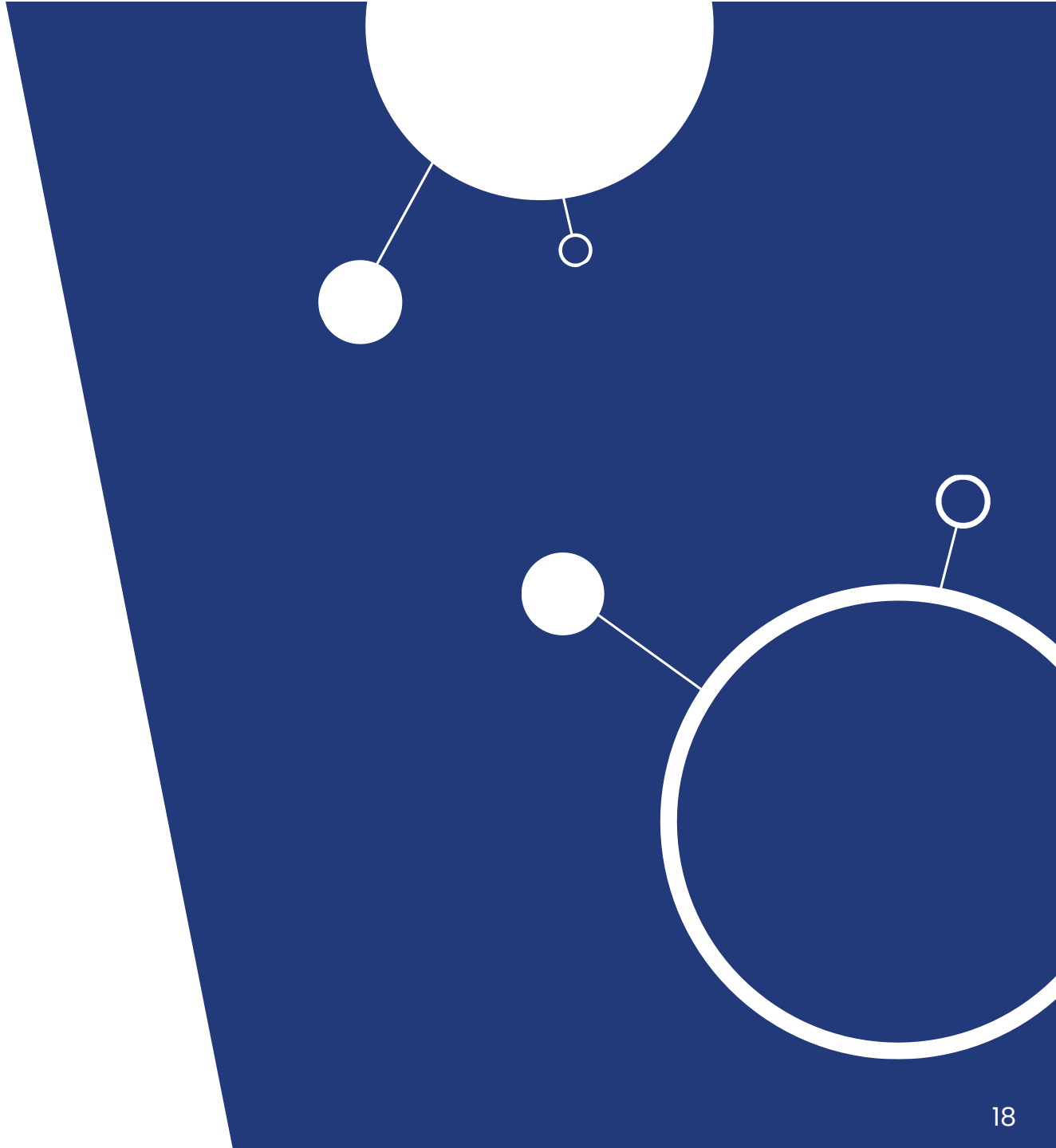
Elimination des dossiers inutiles

VOLUME TOTAL DE L'IMAGE EWF	Données Brutes
EXCLUSION DES DOSSIERS SUPERFLUS	Exemple : Lost & Found
PREUVES POTENTIELLES	Gain de temps et efficacité

Les différentes

Interfaces utilisateurs

Conception & implémentation



ENSI CAEN
ÉCOLE PUBLIQUE D'INGÉNIEURS
CENTRE DE RECHERCHE

Finder

- Point d'entrée pour les deux interfaces (graphique ou cli)
- Implémente les filtres de recherche
- Contient la fonction de recherche avec la logique : Find

Patron de conception

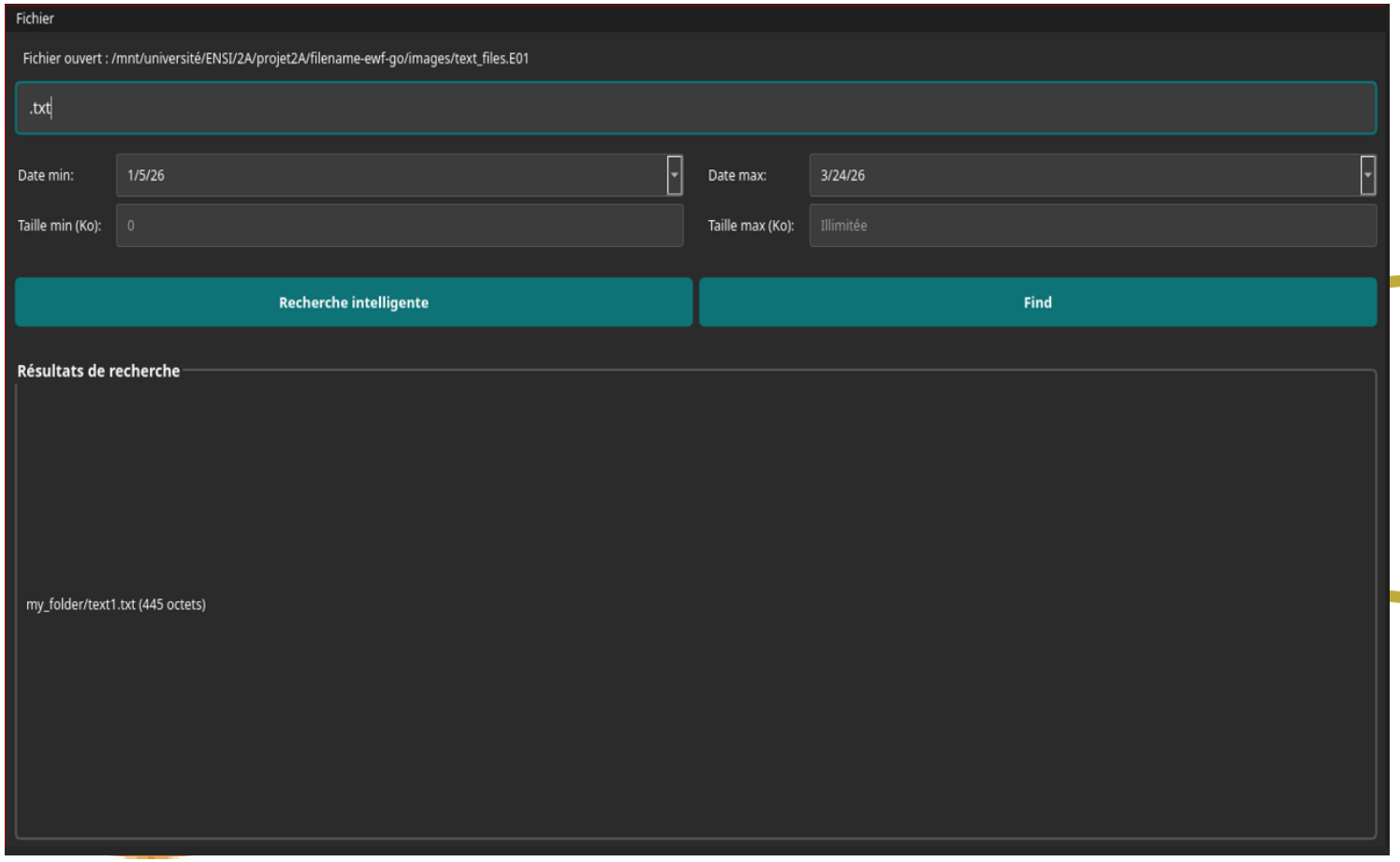
Monteur (Builder) :

- Configuration progressive
- Séparation construction/utilisation
- possède plusieurs paramètres optionnels

Interface graphique

- Permet d'ouvrir une image
- Permet le paramétrage des différents filtres mis en place
- Gère l'affichage des résultats

Visuel du GUI

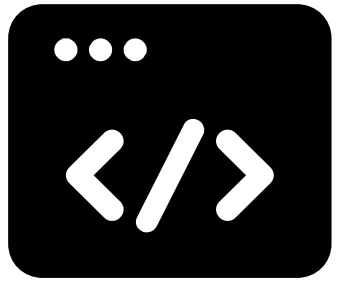


Interface CLI

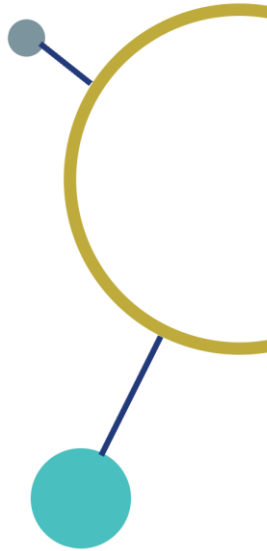
```
$ ./cli_tool ../../images/text_files.E01 "tex"
my_folder/text1.txt
my_folder/text2.jpg
my_folder/text3

$
```

Enjeux

- Fonctionnelle
- Simple
- Modulaire
- Utilisable par d'autres outils



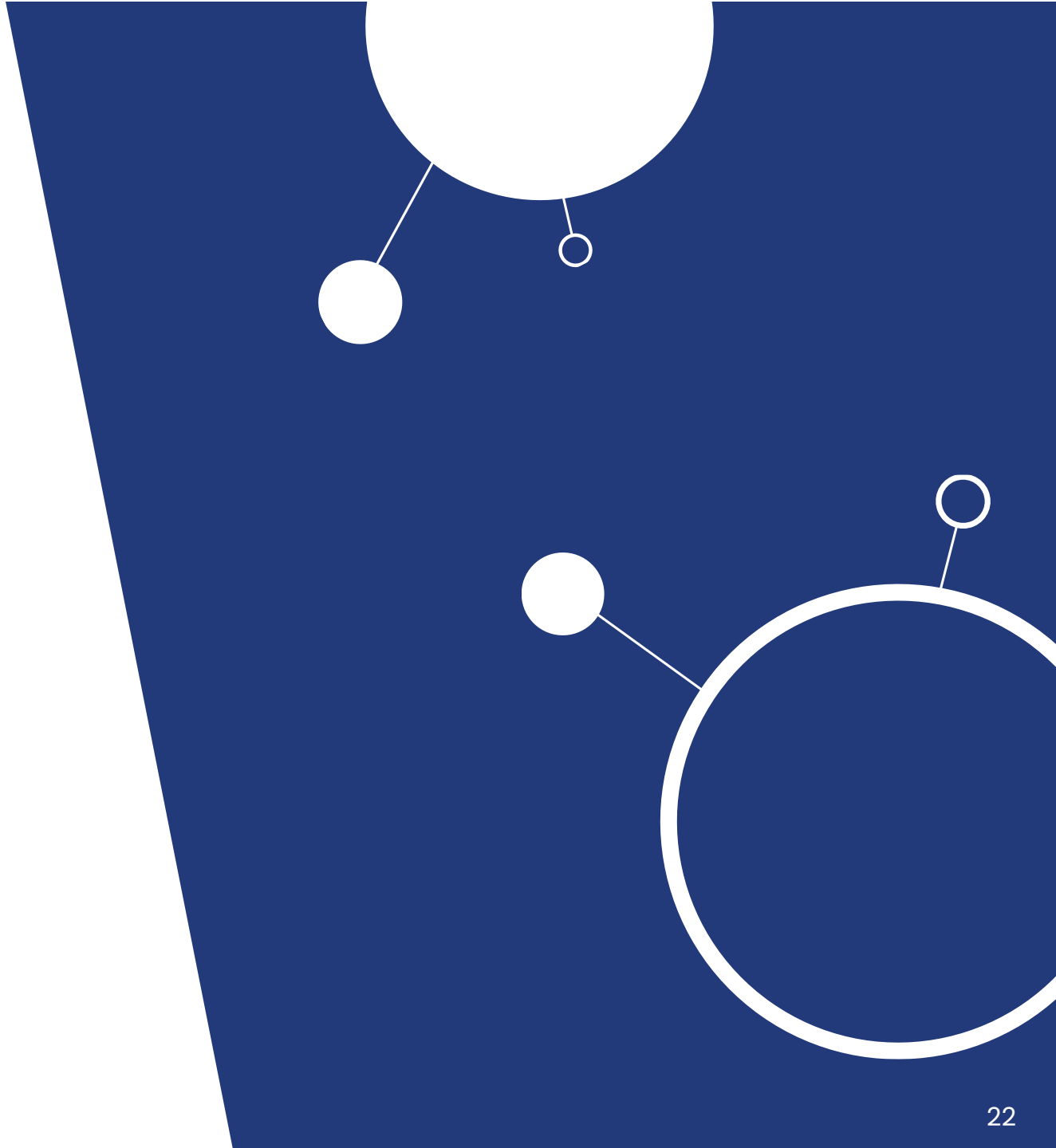
```
$ ./cli_tool --help
Utilisation :     [image EWF] [nom de fichier]
Utilisation : -d  [image EWF] [nom de fichier] [dateMin] [dateMax]
Utilisation : -s  [image EWF] [nom de fichier] [sizeMin] [sizeMax]
Utilisation : -ds [image EWF] [nom de fichier] [dateMin] [dateMax] [sizeMin] [sizeMax]
Utilisation : -sd [image EWF] [nom de fichier] [sizeMin] [sizeMax] [dateMin] [dateMax]
Format [dateMin] [dateMax] : jj/mm/aaaa
Unité [sizeMin] [sizeMax] Ko
$
```



La

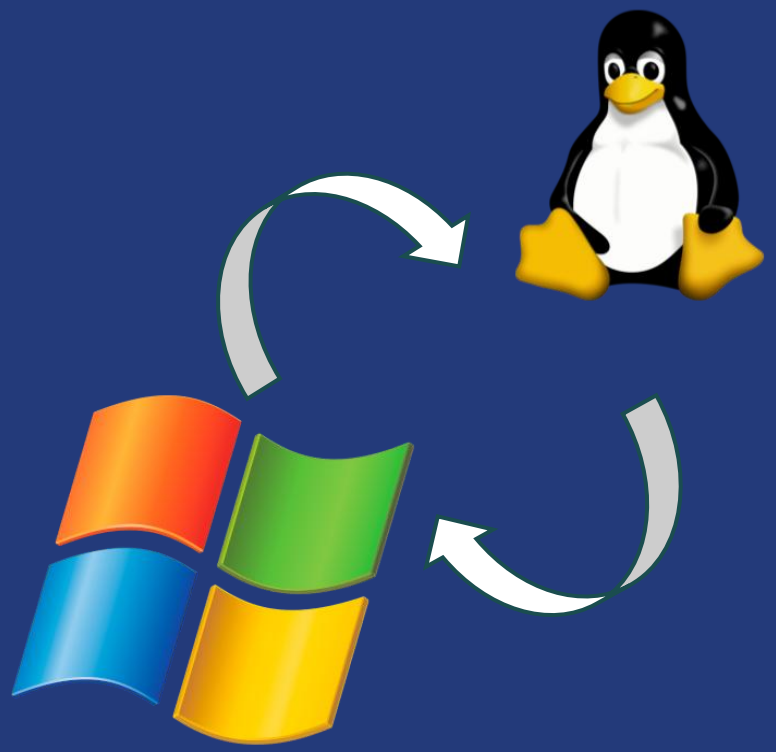
Cross-compilation

Enjeux & Défis techniques



ENSI CAEN
ÉCOLE PUBLIQUE D'INGÉNIEURS
CENTRE DE RECHERCHE

Cross compilation



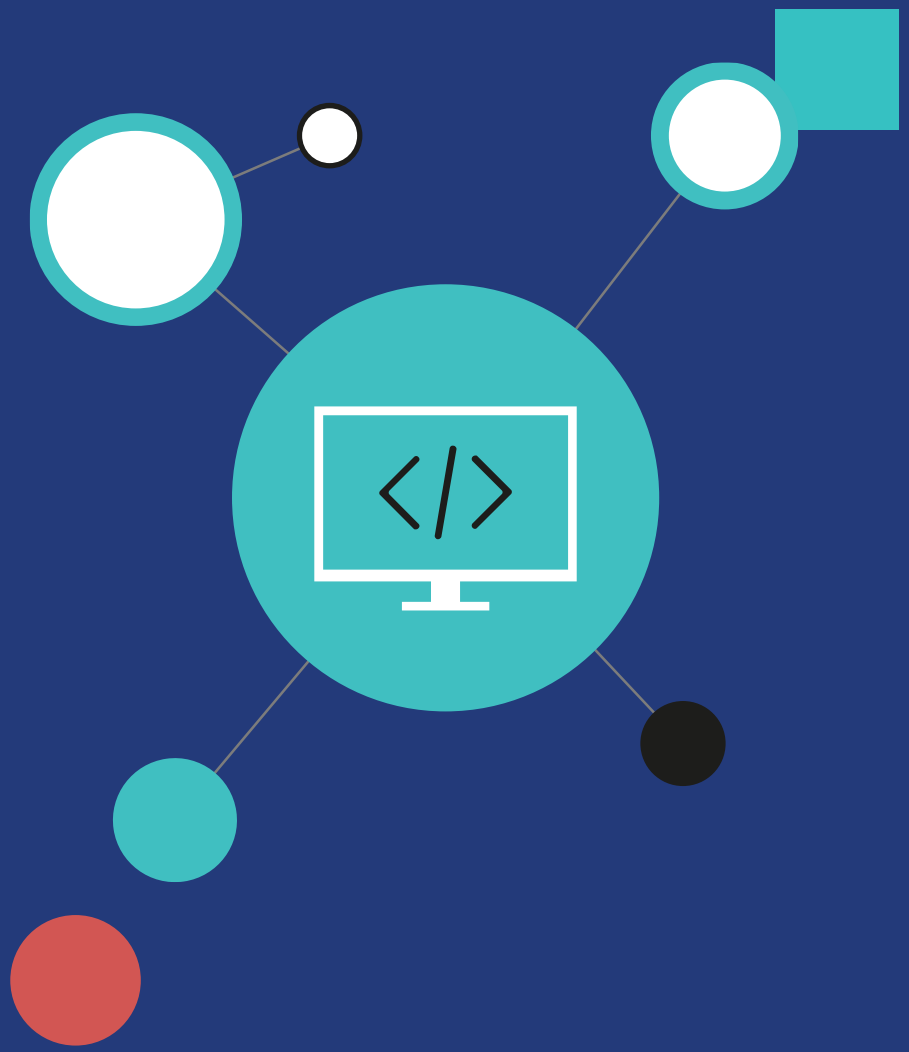
Enjeux

- Importance de Windows dans le domaine de la forensique

Défis techniques

- Dépendances
- Conservation de la propriété cross compilable de LibEWF
- Cross compilation de LibTSK

Conclusion



Découverte des outils en forensique : EWF, images disques, partitions

Extension possible sur le contenu des fichiers : export des fichiers en tableau d'octets

Possibilité d'extraction des fichiers identifiés

Création de nos propres pièces à conviction

ENSI CAEN
ÉCOLE PUBLIQUE D'INGÉNIEURS
CENTRE DE RECHERCHE

Merci

Pour votre écoute

L'École des Ingénieurs Scientifiques