

# Projet 2A

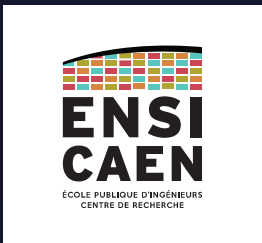
Recherche de mots-clés dans  
les noms et contenus de fichiers

---

PORTE-MITATRE Cydony  
DE PERMENTIER Thibault  
DUBOIS Timothei  
JANCI Fabien  
MAHIER Awen

# Sommaire

1. Cadre et objectifs du projet
2. Organisation interne
3. Travail réalisé
4. Bilan : Objectifs atteints / Voies d'amélioration
5. REX



# Cadre du projet

---

## Contextualisation

---

- Clients : GREYC et Gendarmerie
- EWF
- Go
- Objectifs
  - o Faciliter l'exploration d'une archive EWF
  - o Recherche mot clé dans le nom d'un fichier/dossier et dans son contenu

## Contraintes

---

- Utilisation unique du système de fichier EXT4
- Choix limité des extensions de fichiers
- Séparation du code en sous-modules

# Organisation interne

---

*Répartition des tâches*

## Répartition des tâches

---

<b>Fabien</b>	→	<b>Lecture ZIP, contenu fichiers texte et noms</b>
<b>Cydony</b>	→	<b>Lecture PDF et CLI</b>
<b>Thibault</b>	→	<b>Lecture fichiers compressés, Lien EGOWF</b>
<b>Timothei</b>	→	<b>Lecture EWF et EXT4</b>
<b>Awen</b>	→	<b>Lecture EWF et EXT4, Lien entre les modules</b>

# Organisation interne

---

*Méthodologie utilisée*

## Méthode d'organisation

Réunion tous les matins

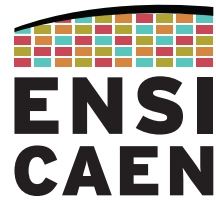


Communication et  
rapport en fin de journée



Partage des différentes  
tâches





**ENSI  
CAEN**

ÉCOLE PUBLIQUE D'INGÉNIEURS  
CENTRE DE RECHERCHE

# Travail réalisé

---

*Architecture micro-modules*

*Lecture de l'archive EWF*

## Architecture modulaire

Pré-requis client

Nombreux avantages:

- Réutilisabilité
- Indépendance des modules
- Agilité
- Résilience

Similaire à l'architecture Micro-Services

## Lecture EWF / EXT4

Format EWF :

- Header
  - Section
- => Section 'sector'

Format du disque EXT4

- > Group Descriptor
- > Inode

Bibliothèques GO-EWF et GO-EXT4

# Travail réalisé

---

*Recherche de mots clés par types de fichiers*

# Algorithme de recherche de mots clés

- Dans le nom (parcours de l'arborescence archive EWF)
- Dans le contenu (archives compressées ou fichiers non-compressés)
- Structure commune des fonctions de recherche :

- Paramètres :

```
(words []string, data []byte, verbose bool) map[string][]string
```

- Liste des mots clés
  - Arborescence / Contenu du fichier
  - Booléen verbose
- Valeur de retour :
    - Dictionnaire[mot clé][liste d'emplacements]

## Fichiers compressés

- Office ou équivalent : majorité sont des ZIP camouflés
- Lecture du fichier XML de contenu
- Archives ZIP :
  - Ouverture et lecture récursive
  - Parcours de l'arborescence en récupérant :
    - Les noms des dossiers et fichiers
    - Les contenus des fichiers
  - Appel de la fonction de recherche correspondant à l'extension de chaque nom de fichier sur son contenu
  - Fusion de tous les dictionnaires obtenus en un seul output

## HTML / XML / TXT

- Récupération du contenu d'un fichier texte quelle que soit son extension :
  - Parcours de chaque mot clé dans la liste en paramètre
  - Comparaison entre ce mot-clé et le contenu du fichier ligne par ligne
  - Normalise la ligne étudiée et le mot clé recherché avant de les comparer en retirant les accents
  - Ajoute au dictionnaire renvoyé, pour le mot clé correspondant :
    - Nombre d'occurrences dans le fichier si la verbose est désactivée
    - Les numéros de ligne du fichier où se situe au moins une occurrence, et le nombre d'occurrences pour cette ligne si la verbose est activée

# PDF

---

- Bibliothèque : [pdf](#)
- ReaderPDF
- Découpage en page / lignes
- Pour chaque mot clé :
  - Récupération texte par ligne
  - Normalisation
  - Enregistrement (format verbose ou non)

The screenshot shows the GitHub page for the 'go pdf' package. The page header includes the Go logo, the package name 'pdf', and options for 'package' and 'module'. It also displays the version 'v0.0.2 Latest', the publication date 'Jan 24, 2024', and the license 'BSD-3-Clause'. The main content area is titled 'README' and features a 'PDF Reader' section. The description states it is a simple Go library for reading PDF files. The 'Features' section lists two capabilities: getting plain text content without formatting and getting content including all font and formatting information. The 'Install' section provides the command: `go get -u github.com/dslipak/pdf`. The 'Read plain text' section is also visible at the bottom.

# Travail réalisé

---

*Command Line Interface*

# searchEWF : commandes et flags

```
./searchEWF search <archive_EWF>
```

--word : mots clés

--type : extension fichiers

--verbose : format réponse

```
./searchEWF infoEWF <archive_EWF>
```

- *Nombre de secteurs par bloc*
- *Methode de compression*
- *Le Média (disque dur) ...*

=> SAUVEGARDE RESULTATS : *output.txt*

```
Start analysis
Analyzing ./segment/ewf_full.E01
EWF Information:
  Format : EnCase 6
  Sector Per Chunk : 64
  Chunk Size : 32768
  Error Granularity : 64
  Compression Method : unknown compression
  Compression Level : unknown compression
-----
Media Information:
  Media Size : 7340032 bytes
  Media Type : fixed disk
  Number Of Sectors : 14336
  Bytes per Sector : 512
-----
Digest hash information:
  Number of Hash Values : 1
  MD5 : 5be195f22760d04d4f91a6427f2e61d7
-----
```



```
>>./searchEWF search
Error: requires at least 1 arg(s), only received 0
Usage:
  searchEWF search <path_to_ewf_image> < -w word_to_search | -t file_extension> [--verbose] [flags]

Flags:
  -h, --help           help for search
  -t, --type string    File extension to search for (e.g., .txt, .docx)
  -v, --verbose        Display search results in the terminal
  -w, --word string    Word or pattern to search for
```

```
>>./searchEWF search segment/ewf_full.E01 -w "drogues compta"
EWF Image Path: segment/ewf_full.E01
Searching for word(s): [drogues compta]
Find in tous les crimes que j'ai commis.txt : compta -> 1 occurrence(s)
Find in 1-Fascination.txt : compta -> 14 occurrence(s)
Find in 1-Fascination.txt : drogues -> 1 occurrence(s)
Find in 1-Fascination.odt : drogues -> 1 occurrence(s)
Find in 1-Fascination.odt : compta -> 14 occurrence(s)
Find in 1-Fascination.pdf : drogues -> 1 occurrence(s)
Find in 1-Fascination.pdf : compta -> 14 occurrence(s)
Search completed: results saved in ./output.txt
>>■
```

# Bilan : Objectifs du projet

---

## Objectifs atteints

---

- Architecture modulaire
- Traitement des extensions choisies
- CLI claire
- Recherche avec différentes options
- Sauvegarde des résultats de recherche en .txt

## Pistes d'amélioration

---

- Raccordement avec projet partenaire
- Portabilité
- Traitement de nouveaux types de fichiers
  - o Cache navigateur
  - o Images
  - o Fichiers chiffrés

# Retour d'expérience

---

# Bilan

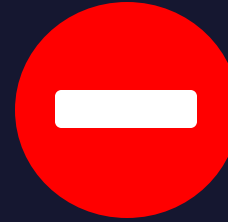
---



**Communication**

**Répartition des tâches**

**Recherche documentaire**



**Compréhension**



**ENSI  
CAEN**

ÉCOLE PUBLIQUE D'INGÉNIEURS  
CENTRE DE RECHERCHE

**MERCI**

pour votre écoute



L'École des Ingénieurs Scientifiques