

# Python et l'analyse forensique

Récupérer et analyser les données produites par les ordinateurs

## Introduction : Python et l'informatique forensique

<b>1. Pourquoi un livre sur le sujet ?</b>	<b>15</b>
1.1 La mutation de l'informatique forensique	15
1.1.1 Des usages toujours plus nombreux et variés	15
1.1.2 Un cadre légal qui évolue à l'échelle planétaire	16
1.2 Une pratique qui se démocratise	17
1.2.1 Des attentes en sécurité et en fiabilité toujours plus grandes	17
1.2.2 De la nécessité de maîtriser ses outils	18
<b>2. Présentation du langage</b>	<b>19</b>
2.1 Un langage adapté	19
2.1.1 Python, un langage de script	19
2.1.2 Une interface efficace avec C	19
2.1.3 Une communauté active	19
2.2 Interprétation de Python	20
2.2.1 CPython	20
2.2.2 Autres implémentations	20
2.2.3 URL	21
<b>3. Le choix des logiciels</b>	<b>21</b>

# Python et l'analyse forensique

Récupérer et analyser les données produites par les ordinateurs

3.1 Le choix du système d'exploitation	21
3.1.1 GNU/Linux	21
3.1.2 Le choix d'une distribution	22
3.1.3 Les procédures d'installation dans ce livre	22

## Premiers pas en Python

<b>1. Installation de CPython</b>	<b>23</b>
1.1 Avec le gestionnaire de paquets de la distribution GNU/Linux	23
1.1.1 Sur GNU/Linux Debian 9	23
1.1.2 Autres gestionnaires de paquets	24
1.2 Autres méthodes	26
1.2.1 Depuis les sources	26
1.2.2 Avec les binaires fournis sur le site python.org	28
1.3 Exécuter du Python	28
1.3.1 Depuis l'interpréteur interactif	28
1.3.2 Depuis un fichier	29
1.3.3 Hyperliens	30
<b>2. Éléments de syntaxe fondamentaux</b>	<b>30</b>
2.1 Les commentaires	30
2.1.1 Commentaire monoligne	

# Python et l'analyse forensique

Récupérer et analyser les données produites par les ordinateurs

2.1.2 Les commentaires multilignes	30
2.2 Variables	31
2.2.1 Nommage	31
2.2.2 Affectation	31
2.3 Les fonctions	34
2.3.1 Appeler une fonction	34
2.3.2 Définir une fonction	35
2.3.3 Fonction lambda	36
2.4 Les objets et leurs instances	37
2.4.1 Rapide rappel sur la programmation orientée objet	37
2.4.2 Créer une instance	40
2.4.3 Accéder aux attributs, appeler une méthode	40
2.4.4 Définir une classe en Python	41
2.4.5 Les méthodes « magiques »	43
2.4.6 Le « modèle objet » de Python	46
2.4.7 Les attributs et les méthodes « de classe »	46
2.4.8 Hyperliens	49
<b>3. Les types de données essentiels</b>	<b>49</b>
3.1 Les nombres	49
3.1.1 Entiers et flottants	

# Python et l'analyse forensique

Récupérer et analyser les données produites par les ordinateurs

3.1.2 Opérations sur les nombres	49
3.1.3 Opérateurs de comparaisons	53
3.1.4 Opérations bits à bits	55
3.1.5 Nombres complexes	56
3.2 Autres types	56
3.2.1 None	57
3.2.2 Booléens	57
3.2.3 Opérateurs booléens	57
3.2.4 Opérateurs d'identité	58
3.3 Itérables	61
3.3.1 Introduction	63
3.3.2 Les tableaux : list	63
3.3.3 Le dépaquetage de séquences	63
3.3.4 Les listes immuables : tuple	68
3.3.5 Les tableaux associatifs : dict	69
3.3.6 Les ensembles : set	71
3.3.7 Les itérateurs	72
3.3.8 Les fonctions de génération	73
3.3.9 Regrouper des arguments de fonctions dans un itérable	76
3.4 Les chaînes de caractères	77
3.4.1 Présentation	80

3.4.2 Fonctions et méthodes des chaînes de caractères	80
3.4.3 La méthode « format »	81
3.4.4 Formater une chaîne avec la syntaxe « printf »	84
3.4.5 Les chaînes d'octets	86
3.4.6 Encode et decode	87
3.4.7 URL	89
<b>4. Les structures de contrôles</b>	<b>90</b>
4.1 Conditionnels et boucles	91
4.1.1 Les branchements conditionnels	91
4.1.2 Les expressions conditionnelles	93
4.1.3 La répétition « tant que » avec while	94
4.1.4 L'itération avec for	95
4.1.5 Les instructions de contrôle de l'itération	98
4.1.6 Le else de boucle	99
4.2 La compréhension de liste	99
4.2.1 L'expression d'une collection	99
4.2.2 L'introduction des conditions	100
4.3 Gestion des exceptions	101
4.3.1 Introduction	101
4.3.2 Les classes d'exceptions de Python	101

# Python et l'analyse forensique

Récupérer et analyser les données produites par les ordinateurs

4.3.3 Try, Except, Raise	103
4.3.4 else/finally	104
<b>5. Mécanismes d'import</b>	<b>109</b>
5.1 Définitions	109
5.1.1 Les modules de Python	109
5.1.2 Les packages en Python	109
5.2 Syntaxe de l'import	110
5.2.1 Le mot-clé « import »	110
5.2.2 Les chemins d'import	114
5.2.3 Le mot-clé « from »	115
<b>6. Environnement, dépendances et communauté</b>	<b>116</b>
6.1 Déploiement/installation	116
6.1.1 Présentation	116
6.1.2 Le script setup.py	116
6.1.3 Egg et Wheel	117
6.1.4 Pip	118
6.1.5 Gestionnaire de paquet d'une distribution	119
6.1.6 Hyperliens	119
6.2 Les environnements virtuels avec virtualenv	120

# Python et l'analyse forensique

Récupérer et analyser les données produites par les ordinateurs

6.2.1 Présentation et mise en garde	120
6.2.2 Installation et utilisation	121
6.3 Qualité du code	122
6.3.1 Les conventions de codage PEP8	122
6.3.2 Utilitaires	123
6.3.3 Hyperliens	124

## Bibliothèque standard

<b>1. Bibliothèques utilitaires</b>	<b>125</b>
1.1 Fonctions utilitaires	125
1.1.1 Afficher avec print	125
1.1.2 Interroger le type d'une variable	127
1.1.3 Obtenir de l'aide avec help	128
1.2 Modules utilitaires de Python	129
1.2.1 Garder une trace de l'exécution des programmes avec logging	129
1.2.2 Analyser les arguments de la ligne de commande avec argparse	131
1.2.3 Parallélisation facile avec multiprocessing.Pool	135
1.2.4 Hyperliens	137
<b>2. Opérations de base sur un système de fichiers</b>	<b>137</b>
2.1 Se déplacer et explorer l'arborescence des dossiers	

# Python et l'analyse forensique

Récupérer et analyser les données produites par les ordinateurs

2.1.1 Les chemins d'accès : les paths	137
2.1.2 Le dossier courant cwd et le déplacement dans l'arborescence	137
2.1.3 Les fonctions utilitaires sur les chemins de os.path	138
2.1.4 Exploration d'une arborescence avec le module os	139
2.2 Opérations sur les fichiers	142
2.2.1 Ouverture et fermeture d'un fichier	144
2.2.2 stdin, stdout & stderr	144
2.2.3 Lecture et écriture dans un fichier	145
2.2.4 Exemple : réaliser une image disque	147
2.3 Utilitaires	151
2.3.1 Manipulation des fichiers avec le module os	157
2.3.2 Aller plus loin dans la manipulation des fichiers avec shutil	157
2.3.3 Gérer les fichiers temporaires avec tempfile	160
<b>3. Opérations de base sur le réseau</b>	<b>164</b>
3.1 TCP/UDP	164
3.1.1 Python et les sockets	164
3.1.2 Serveur	167
3.1.3 Client	170
3.1.4 Recevoir et envoyer des données sur un socket	170
3.1.5 Un faux serveur chargen	170



3.2 Le module urllib	173
3.2.1 Introduction	177
3.2.2 Analyse d'URL avec urllib.parse	177
3.2.3 Forger une URL avec urllib.parse	179
3.2.4 Hyperliens	181
	182

## Premiers pas dans l'analyse d'un fichier

<b>1. Le fichier dans son ensemble</b>	<b>183</b>
1.1 Trouver le type d'un fichier avec les nombres magiques	183
1.1.1 Présentation	183
1.1.2 Installation de python-magic	184
1.1.3 Exemple d'utilisation	185
1.1.4 Hyperliens	187
1.2 Les fonctions de hachage	187
1.2.1 Présentation du concept	187
1.2.2 Le module hashlib de la bibliothèque standard	188
1.2.3 Contrôler l'intégrité des fichiers	189
1.3 Génération de "diff" avec difflib	194
1.3.1 Présentation de diff	194
1.3.2 Le programme GNU diff	194

# Python et l'analyse forensique

Récupérer et analyser les données produites par les ordinateurs

1.3.3 Générer le diff de deux fichiers avec Python	194
1.3.4 Générer des diff de fichiers binaires	196
	199
<b>2. Utiliser Python pour lire des métadonnées</b>	<b>206</b>
2.1 De l'importance des métadonnées	206
2.1.1 Définition	206
2.1.2 Quels intérêts pour les métadonnées ?	207
2.2 Lire les informations fournies par un système de fichiers	208
2.2.1 L'espace utilisé	208
2.2.2 Les métadonnées temporelles avec os.path	209
2.2.3 Le système de fichiers vu par le noyau	212
2.2.4 Identifier les liens matériels à l'aide des inodes	215
2.2.5 Hyperliens	218
2.3 Métadonnées de fichiers multimédias avec XMP	218
2.3.1 Le choix du format de métadonnée XMP	218
2.3.2 Installation de python-xmp-toolkit	219
2.3.3 Utilisation de la bibliothèque	220
2.3.4 Présentation rapide de XML	222
2.3.5 Un petit explorateur de métadonnées	223
2.3.6 Hyperliens	232

## Analyser un historique de navigation web

<b>1. Bibliothèques HTML et HTTP haut niveau</b>	<b>233</b>
1.1 La bibliothèque requests	233
1.1.1 Présentation	233
1.1.2 Installation	235
1.1.3 Envoyer une requête avec une commande spécifique	236
1.1.4 Passage de paramètres dans l'URL	237
1.1.5 Passage de paramètres pour une requête POST	238
1.1.6 Hyperliens	239
1.2 La bibliothèque BeautifulSoup	239
1.2.1 Présentation	239
1.2.2 Installation	240
1.2.3 Charger du HTML	240
1.2.4 Extraire du contenu	241
1.2.5 Hyperliens	242
<b>2. Les bases de données SQLite3</b>	<b>242</b>
2.1 SGBD et SQL	242
2.1.1 Les Systèmes de Gestion de Base de Données	242
2.1.2 Structured Query Langage (Langage de requête structurée)	243

# Python et l'analyse forensique

Récupérer et analyser les données produites par les ordinateurs

2.1.3 Un ORM pour les interfacier tous	244
2.2 SQLite3 un SGBD léger	244
2.2.1 Présentation	244
2.2.2 Les types de données SQLite3	245
2.2.3 Hyperlien	245
2.3 Le module SQLite3	246
2.3.1 Connexion à une base de données	246
2.3.2 Exécuter une requête SQL	247
2.3.3 Les curseurs	248
2.3.4 Hyperliens	250
<b>3. Accéder à l'historique d'un navigateur</b>	<b>251</b>
3.1 L'historique des navigateurs les plus utilisés	251
3.1.1 Introduction	251
3.1.2 Mozilla Firefox	251
3.1.3 Safari®	253
3.1.4 Navigateur Android®	254
3.1.5 Navigateur Chrome™/Chromium™	255
3.1.6 Hyperliens	255
3.2 Écriture d'un module d'accès	256
3.2.1 Organisation générale du code	256

# Python et l'analyse forensique

Récupérer et analyser les données produites par les ordinateurs

3.2.2 Exécuter la requête selon le navigateur	261
3.2.3 Exemple d'utilisation	266
3.2.4 Récupérer un document HTML	266
3.2.5 Mise en place d'un cache	267
3.2.6 Accélérer l'exécution avec multiprocessing	270
3.2.7 Hyperliens	273
3.3 Utilisations	274
3.3.1 Lister les noms de domaines visités	274
3.3.2 Établir des statistiques sur les en-têtes des documents	275

## Partitionnement automatique de données

<b>1. L'apprentissage automatique ou apprentissage statistique</b>	<b>277</b>
1.1 Présentation	277
1.1.1 Définition	277
1.1.2 « Supervision » des algorithmes	279
1.2 Le partitionnement non supervisé	281
1.2.1 Présentation de la famille d'algorithmes	281
1.2.2 Principaux algorithmes	281
<b>2. Bibliothèques Python</b>	<b>282</b>
2.1 Calcul matriciel optimisé avec NumPy	

# Python et l'analyse forensique

Récupérer et analyser les données produites par les ordinateurs

2.1.1	Présentation	282
2.1.2	Installation	282
2.1.3	Le type numpy.array	283
2.1.4	Hyperliens	284
2.2	L'apprentissage automatique avec Scikit-Learn	288
2.2.1	Présentation	288
2.2.2	Installation	288
2.2.3	Le partitionnement de données avec scikit-learn	288
2.2.4	Hyperliens	289
2.3	Afficher des données avec Matplotlib	289
2.3.1	Présentation	289
2.3.2	Installation	289
2.3.3	Afficher un ensemble de données dans un graphique	289
2.3.4	Hyperliens	290
<b>3.</b>	<b>Une première application : regrouper les fichiers par date de création</b>	<b>292</b>
3.1	Représentation des données	293
3.1.1	Description de la problématique	293
3.1.2	Stockage et identification des temps et des chemins	294
3.1.3	Écrire un script pour créer des instances de FilesTime	300
3.1.4	Normalisation des données	

# Python et l'analyse forensique

Récupérer et analyser les données produites par les ordinateurs

3.2 Création du modèle et construction des sous-groupes	301
3.2.1 Préambule	305
3.2.2 Création du modèle avec MiniBatchKmeans	305
3.2.3 Création du modèle avec BIRCH	306
3.2.4 Création du modèle avec DBSCAN	307
3.2.5 Création des groupes à partir des modèles	308
3.3 Écrire une interface textuelle	309
3.3.1 Présentation de l'interface	309
3.3.2 Les fonctions utilitaires d'une interface textuelle	309
3.3.3 Une représentation des modèles pour l'interface	312
3.3.4 Le module ui	316
3.3.5 Afficher les données dans un graphique	320
3.3.6 Le menu principal et la boucle de traitement	324
3.3.7 Utilisation	327

## Extraire les sujets d'un ensemble de textes

<b>1. Le traitement automatique des langues naturelles</b>	<b>329</b>
1.1 Introduction	329
1.1.1 Présentation	329
1.1.2 La modélisation de sujets	329

# Python et l'analyse forensique

Récupérer et analyser les données produites par les ordinateurs

1.2 Les langues naturelles en Python avec NLTK	331
1.2.1 Présentation	332
1.2.2 Installation	332
1.2.3 Hyperliens	333
1.3 La modélisation de sujets avec gensim	333
1.3.1 Présentation	333
1.3.2 Installation	334
1.3.3 Hyperliens	335
1.4 Les autres bibliothèques utilisées dans ce chapitre	335
1.4.1 Les caractères accentués avec unidecode	335
1.4.2 La progression avec tqdm	336
1.4.3 Hyperliens	336
<b>2. Modélisation de sujet avec LDA et LSI</b>	<b>337</b>
2.1 Extraire une liste de mots d'un texte	337
2.1.1 Analyse lexicale ou tokenization	337
2.1.2 Suppression des caractères accentués et de la ponctuation	338
2.1.3 Suppression des mots vides	338
2.1.4 Une première version du module « corpus »	339
2.2 Sac de mots et TF-IDF	340
2.2.1 Introduction	



# Python et l'analyse forensique

Récupérer et analyser les données produites par les ordinateurs

2.2.2 Génération d'un dictionnaire de mots	340
2.2.3 Création de "sacs de mots"	341
2.2.4 Le modèle TF-IDF de gensim	341
2.2.5 Hyperliens	342
2.3 Les modèles gensim pour LDA et LSI	343
2.3.1 Génération du modèle	344
2.3.2 Regrouper les documents par sujets	344
2.3.3 Récupération des sujets	346
<b>3. Exemples d'application</b>	<b>347</b>
3.1 Depuis l'historique d'un navigateur web	349
3.1.1 Réutilisation de l'analyse de l'historique	349
3.1.2 Adaptation de l'interface d'extraction	349
3.1.3 Écrire un script doté d'une interface	351
3.2 Depuis des fichiers textes	355
3.2.1 Utilisation de LibreOffice comme convertisseur	355
3.2.2 TF-IDF depuis un système de fichier	356
3.2.3 Un script et une interface	358
3.2.4 Hyperliens	360

## Inspection des processus du noyau Linux

<b>1. Introduction</b>	<b>361</b>
1.1 Ptrace et le noyau Linux	361
1.1.1 Exécutables et processus	361
1.1.2 La mémoire vue par un processus	362
1.1.3 Présentation de ptrace	363
1.1.4 Hyperliens	365
<b>2. Inspection d'un processus avec Python</b>	<b>366</b>
2.1 La bibliothèque python-pttrace	366
2.1.1 Introduction	366
2.1.2 Installation de python-pttrace	367
2.1.3 Hyperliens	367
2.2 « Tracer » un processus	367
2.2.1 Démarrer un nouveau processus	367
2.2.2 Attacher un processus existant	372
2.2.3 Mise en pause et reprise de l'exécution	374
2.2.4 Récupérer l'état des registres	380
2.2.5 Reprise d'exécution conditionnelle	383
2.2.6 Hyperliens	387
2.3 La mémoire d'un processus	388

# Python et l'analyse forensique

Récupérer et analyser les données produites par les ordinateurs

2.3.1 Généralités sur l'organisation de la mémoire pour un processus	388
2.3.2 Les maps mémoire	390
2.3.3 L'interface d'accès à la mémoire de python-pttrace	392
2.3.4 Hyperliens	396
<b>3. Exemple d'utilisation</b>	<b>396</b>
3.1 Extraire le texte	396
3.1.1 Objectif du programme	396
3.1.2 La fonction d'extraction des chaînes de caractères	397
3.1.3 Récupérer les maps à partir d'un PID	399
3.1.4 Initialisation et appel	400
3.2 Tricher à un jeu vidéo	402
3.2.1 Présentation de l'approche	402
3.2.2 La classe MemState	403
3.2.3 Un script et une interface pour MemState	411
3.2.4 Exemple d'utilisation	413
3.2.5 Hyperliens	418
Index	419