



POLYTECHNIQUE
MONTREAL

UNIVERSITÉ
D'INGÉNIERIE

Rapport TP5 LOG6302A

Groupe 3

SOMO JOEL RENE - 2182375

Ajouter

Analyser une plateforme web

1. Implémentation et tests

1.1 Description du procédé :

Pour l'implémentation de l'algorithme possibly tainted définitions nous avons définis les fonctions suivantes :

- `referencesExpression()` : fonction récursive qui prend en argument un CFG et l'id d'un nœud correspondant à la définition d'une expression ainsi qu'une liste vide et retourne la liste des ids des références correspondant à la partie droite de l'expression.
- `poss_tainted_defs()` : fonction qui prend en argument un CFG et le chemin du fichier tainted et exécute l'algorithme possibly tainted définitions. A noter que pour la construction du GEN nous avons suivi la procédure suivante : Pour chaque nœud dans le CFG si le nœud est une définition, on vérifie le côté droit de la définition, s'il s'agit d'une source, alors la définition est teintée, s'il correspond à un filtre ou à un "safe" alors rien ne se passe, sinon le côté droit est une expression, puis on vérifie toutes les références si l'une est teintée, la définition sera teintée aussi.
- `print_poss_tainted_defrefs()` : prend en argument un CFG et AST et chemin du tainted file et exécute la fonction `poss_tainted_defs()` puis affiche les paires définitions références possiblement teintées.

1.1 Tests et résultats :

Après exécution de l'algorithme `poss_tainted_definitions` sur les fichiers de la partie 1, on obtient les résultats suivants.

Fichier 1 :

```
# File 1
cfg_file1 = CFGReader().read_cfg('mnt/c/Users/Othman/Desktop/TP-LOG6302/log6302a_lab5/part_1/file_1.php.cfg.json')
ast_file1 = ASTReader().read_ast('mnt/c/Users/Othman/Desktop/TP-LOG6302/log6302a_lab5/part_1/file_1.php.ast.json')
taint_file1 = 'mnt/c/Users/Othman/Desktop/TP-LOG6302/log6302a_lab5/part_1/file_1.php.taint.json'

print_poss_tainted_defrefs(cfg_file1, ast_file1, taint_file1)
```

✓ 0.1s Python

La paire (definition : 'tainted' ligne 3, reference : 'tainted' ligne 6) est teintee

```
IN : {99: set(), 100: {109}, 101: set(), 102: set(), 103: set(), 104: set(), 105: set(), 106: set(), 107: set(), 108:
OUT : {99: set(), 100: {109}, 101: set(), 102: set(), 103: set(), 104: set(), 105: {109}, 106: set(), 107: set(), 108:
```

```
IN : {99: set(), 100: {109}, 101: set(), 102: set(), 103: set(), 104: set(),
105: set(), 106: set(), 107: set(), 108: set(), 109: set(), 110: {109}, 111:
{109}, 112: {109}, 113: {109}, 114: {109}, 115: {109}, 116: {109}, 117: {109},
118: {109}, 119: {109}, 120: {109}, 121: {109}, 122: {109}, 123: {109}, 124:
{109}, 125: {109}}
```

```
OUT : {99: set(), 100: {109}, 101: set(), 102: set(), 103: set(), 104: set(),
105: {109}, 106: set(), 107: set(), 108: set(), 109: set(), 110: {109}, 111:
{109}, 112: {109}, 113: {109}, 114: {109}, 115: {109}, 116: {109}, 117: {109},
118: {109}, 119: {109}, 120: {109}, 121: {109}, 122: {109}, 123: {109}, 124:
{109}, 125: {109}}
```

On remarque que la définition de la variable 'tainted' dans la ligne 3 est teintée (car sa partie droite correspond à une source) et cette définition est référencée dans la ligne 6 sans passer par une autre définition intermédiaire qui annule la teinte ou un filtre, donc la paire (definition ligne 3 , référence ligne 6) est bien teintée.

Fichier 2 :

```
# File 2
cfg_file2 = CFGReader().read_cfg('/mnt/c/Users/Othman/Desktop/TP-LOG6302/log6302a_lab5/part_1/file_2.php.cfg.json')
ast_file2 = ASTReader().read_ast('/mnt/c/Users/Othman/Desktop/TP-LOG6302/log6302a_lab5/part_1/file_2.php.ast.json')
taint_file2 = '/mnt/c/Users/Othman/Desktop/TP-LOG6302/log6302a_lab5/part_1/file_2.php.taint.json'

print_poss_tainted_defrefs(cfg_file2, ast_file2, taint_file2)
```

✓ 0.4s Python

La paire (definition : 'tainted2' ligne 5, reference : 'tainted2' ligne 9) est teinte

```
IN : {64: {40, 47}, 65: {40, 47}, 66: {40, 47}, 67: {40, 47}, 68: {40, 47}, 30: set(), 31: {40, 47}, 32: set(), 33:
OUT : {64: {40, 47}, 65: {40, 47}, 66: {40, 47}, 67: {40, 47}, 68: {40, 47}, 30: set(), 31: {40, 47}, 32: set(), 33:
```

```
IN : {64: {40, 47}, 65: {40, 47}, 66: {40, 47}, 67: {40, 47}, 68: {40, 47},
30: set(), 31: {40, 47}, 32: set(), 33: set(), 34: set(), 35: set(), 36:
set(), 37: set(), 38: set(), 39: set(), 40: set(), 41: {40}, 42: {40}, 43:
{40}, 44: {40}, 45: {40}, 46: {40}, 47: {40}, 48: {40, 47}, 49: {40, 47}, 50:
{40, 47}, 51: {40, 47}, 52: {40, 47}, 53: {40, 47}, 54: {40, 47}, 55: {40,
47}, 56: {40, 47}, 57: {40, 47}, 58: {40, 47}, 59: {40, 47}, 60: {40, 47}, 61:
{40, 47}, 62: {40, 47}, 63: {40, 47}}
```

```
OUT : {64: {40, 47}, 65: {40, 47}, 66: {40, 47}, 67: {40, 47}, 68: {40, 47},
30: set(), 31: {40, 47}, 32: set(), 33: set(), 34: set(), 35: set(), 36: {40},
37: set(), 38: set(), 39: set(), 40: set(), 41: {40, 47}, 42: {40}, 43: {40},
44: {40}, 45: {40}, 46: {40}, 47: {40}, 48: {40, 47}, 49: {40, 47}, 50: {40,
47}, 51: {40, 47}, 52: {40, 47}, 53: {40, 47}, 54: {40, 47}, 55: {40, 47}, 56:
{40, 47}, 57: {40, 47}, 58: {40, 47}, 59: {40, 47}, 60: {40, 47}, 61: {40,
47}, 62: {40, 47}, 63: {40, 47}}
```

On remarque que la définition de la variable 'tainted2' dans la ligne 5 est teintée car la partie droite de cette définition correspond à une expression

incluant une références pour la variable 'tainted' elle même teintée dans la ligne 3 par une source. Ensuite la définition de la variable 'tainted2' est référencée dans la ligne 9 sans passer par une autre définition intermédiaire qui annule la teinte ou un filtre, donc la paire (définition ligne 5 , référence ligne 9) est bien teintée.

Fichier 3 :

```
# File 3
cfg_file3 = CFGReader().read_cfg('/mnt/c/Users/Othman/Desktop/TP-LOG6302/log6302a_lab5/part_1/file_3.php.cfg.json')
ast_file3 = ASTReader().read_ast('/mnt/c/Users/Othman/Desktop/TP-LOG6302/log6302a_lab5/part_1/file_3.php.ast.json')
taint_file3 = '/mnt/c/Users/Othman/Desktop/TP-LOG6302/log6302a_lab5/part_1/file_3.php.taint.json'

print_poss_tainted_defrefs(cfg_file3, ast_file3, taint_file3)
```

✓ 0.4s Python

IN : {1: set(), 2: {11}, 3: set(), 4: set(), 5: set(), 6: set(), 7: set(), 8: set(), 9: set(), 10: set(), 11: set(),
OUT : {1: set(), 2: {11}, 3: set(), 4: set(), 5: set(), 6: set(), 7: {11}, 8: set(), 9: set(), 10: set(), 11: set(),

IN : {1: set(), 2: {11}, 3: set(), 4: set(), 5: set(), 6: set(), 7: set(), 8: set(), 9: set(), 10: set(), 11: set(), 12: {11}, 13: {11}, 14: {11}, 15: {11}, 16: {11}, 17: {11}, 18: {11}, 19: {11}, 20: {11}, 21: {11}, 22: {11}, 23: {11}, 24: {11}, 25: {11}, 26: {11}, 27: {11}, 28: {11}, 29: {11}}

OUT : {1: set(), 2: {11}, 3: set(), 4: set(), 5: set(), 6: set(), 7: {11}, 8: set(), 9: set(), 10: set(), 11: set(), 12: {11}, 13: {11}, 14: {11}, 15: {11}, 16: {11}, 17: {11}, 18: {11}, 19: {11}, 20: {11}, 21: {11}, 22: {11}, 23: {11}, 24: {11}, 25: {11}, 26: {11}, 27: {11}, 28: {11}, 29: {11}}

L'exécution de l'algorithme sur le fichier 3 ne renvoi aucune paire (definition,reference) teintée et cela représente un bon résultat car la variable 'tainted' est teintée dans la ligne 3 par une source mais filtrée ensuite dans la ligne 5 avant d'être passé en paramètre de la fonction sink dans la ligne 7.

Fichier 4 :

```
# File 4
cfg_file4 = CFGReader().read_cfg('/mnt/c/Users/Othman/Desktop/TP-LOG6302/log6302a_lab5/part_1/file_4.php.cfg.json')
ast_file4 = ASTReader().read_ast('/mnt/c/Users/Othman/Desktop/TP-LOG6302/log6302a_lab5/part_1/file_4.php.ast.json')
taint_file4 = '/mnt/c/Users/Othman/Desktop/TP-LOG6302/log6302a_lab5/part_1/file_4.php.taint.json'

print_poss_tainted_defrefs(cfg_file4, ast_file4, taint_file4)
```

✓ 0.5s Python

La paire (definition : 'tainted' ligne 3, reference : 'tainted' ligne 7) est teinte

IN : {69: set(), 70: {76}, 71: set(), 72: set(), 73: set(), 74: set(), 75: set(), 76: set(), 77: {76}, 78: {76}, 79:
OUT : {69: set(), 70: {76}, 71: set(), 72: {76}, 73: set(), 74: set(), 75: set(), 76: set(), 77: {76}, 78: {76}, 79:

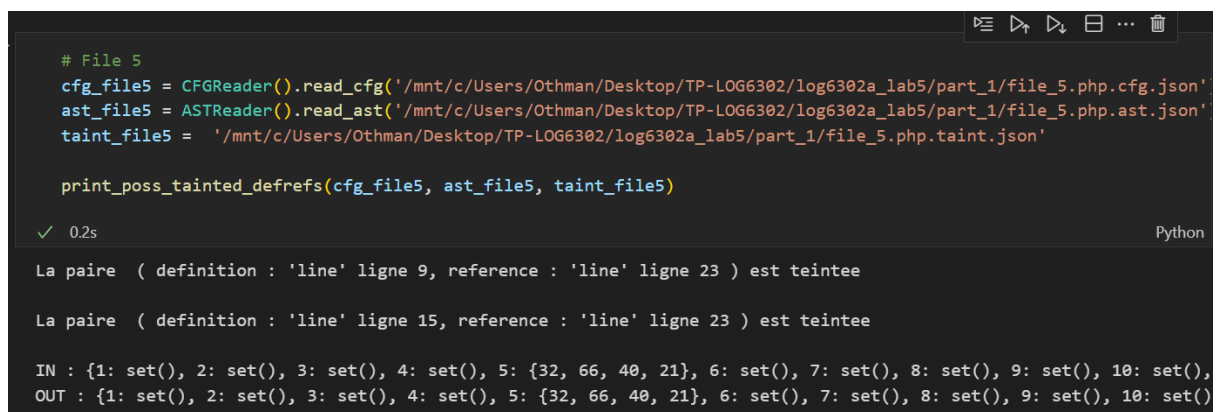
```

IN : {69: set(), 70: {76}, 71: set(), 72: set(), 73: set(), 74: set(), 75:
set(), 76: set(), 77: {76}, 78: {76}, 79: {76}, 80: {76}, 81: {76}, 82: {76},
83: {76}, 84: {76}, 85: {76}, 86: {76}, 87: {76}, 88: {76}, 89: {76}, 90:
{76}, 91: {76}, 92: {76}, 93: {76}, 94: {76}, 95: {76}, 96: {76}, 97: {76},
98: {76}}
OUT : {69: set(), 70: {76}, 71: set(), 72: {76}, 73: set(), 74: set(), 75:
set(), 76: set(), 77: {76}, 78: {76}, 79: {76}, 80: {76}, 81: {76}, 82: {76},
83: {76}, 84: {76}, 85: {76}, 86: {76}, 87: {76}, 88: {76}, 89: {76}, 90:
{76}, 91: {76}, 92: {76}, 93: {76}, 94: {76}, 95: {76}, 96: {76}, 97: {76},
98: {76}}

```

On remarque que la définition de la variable 'tainted' dans la ligne 3 est teintée par une source, ensuite cette variable est référencée dans la ligne 7 donc la paire (définition ligne 3 , référence ligne 7) est bien teintée. ceci est logique car même si la variable est filtrée dans le bloc if, la teinte pourrait atteindre le sink si la condition du if n'est pas vérifiée.

Fichier 5 :



```

# File 5
cfg_file5 = CFGReader().read_cfg('/mnt/c/Users/Othman/Desktop/TP-LOG6302/log6302a_lab5/part_1/file_5.php.cfg.json')
ast_file5 = ASTReader().read_ast('/mnt/c/Users/Othman/Desktop/TP-LOG6302/log6302a_lab5/part_1/file_5.php.ast.json')
taint_file5 = '/mnt/c/Users/Othman/Desktop/TP-LOG6302/log6302a_lab5/part_1/file_5.php.taint.json'

print_poss_tainted_defrefs(cfg_file5, ast_file5, taint_file5)

```

✓ 0.2s Python

La paire (definition : 'line' ligne 9, reference : 'line' ligne 23) est teintee

La paire (definition : 'line' ligne 15, reference : 'line' ligne 23) est teintee

IN : {1: set(), 2: set(), 3: set(), 4: set(), 5: {32, 66, 40, 21}, 6: set(), 7: set(), 8: set(), 9: set(), 10: set(),
OUT : {1: set(), 2: set(), 3: set(), 4: set(), 5: {32, 66, 40, 21}, 6: set(), 7: set(), 8: set(), 9: set(), 10: set() }

```

IN : {1: set(), 2: set(), 3: set(), 4: set(), 5: {32, 66, 40, 21}, 6: set(),
7: set(), 8: set(), 9: set(), 10: set(), 11: set(), 12: set(), 13: set(), 14:
set(), 15: set(), 16: set(), 17: set(), 18: set(), 19: set(), 20: set(), 21:
set(), 22: {32, 66, 40, 21}, 23: {32, 66, 40, 21}, 24: {32, 66, 40, 21}, 25:
{32, 66, 40, 21}, 26: {32, 66, 40, 21}, 27: {32, 66, 40, 21}, 28: {32, 66, 40,
21}, 29: {32, 66, 40, 21}, 30: {32, 66, 40, 21}, 31: {32, 66, 40, 21}, 32:
{32, 66, 40, 21}, 33: {32, 66, 40, 21}, 34: {32, 66, 40, 21}, 35: {32, 66, 40,
21}, 36: {32, 66, 40, 21}, 37: {32, 66, 40, 21}, 38: {32, 66, 40, 21}, 39:
{32, 66, 40, 21}, 40: {32, 66, 40, 21}, 41: {32, 66, 40, 21}, 42: {32, 66, 40,
21}, 43: {32, 66, 40, 21}, 44: {32, 66, 40, 21}, 45: {32, 66, 40, 21}, 46:
{32, 66, 40, 21}, 47: {32, 66, 40, 21}, 48: {32, 66, 40, 21}, 49: {32, 66, 40,
21}, 50: {32, 66, 40, 21}, 51: {32, 66, 40, 21}, 52: {32, 66, 40, 21}, 53:
{32, 66, 40, 21}, 54: {32, 66, 40, 21}, 55: {32, 66, 40, 21}, 56: {32, 66, 40,
21}, 57: {32, 66, 40, 21}, 58: {32, 66, 40, 21}, 59: {32, 66, 40, 21}, 60:
{32, 66, 40, 21}, 61: {32, 66, 40, 21}, 62: {32, 66, 40, 21}, 63: {32, 66, 40,
21}, 64: {32, 66, 40, 21}, 65: {32, 66, 40, 21}, 66: {32, 66, 40, 21}, 67:
{32, 66, 40, 21}, 68: {32, 66, 40, 21}, 69: {32, 66, 40, 21}, 70: {32, 66, 40,
21}, 71: {32, 66, 40, 21}, 72: {32, 66, 40, 21}, 73: {32, 66, 40, 21}, 74:

```

```
{32, 66, 40, 21}, 75: {32, 66, 40, 21}, 76: {32, 66, 40, 21}, 77: {32, 66, 40, 21}, 78: {32, 66, 40, 21}, 79: {32, 66, 40, 21}, 80: {32, 66, 40, 21}, 81: {32, 66, 40, 21}, 82: {32, 66, 40, 21}, 83: {32, 66, 40, 21}, 84: {32, 66, 40, 21}, 85: {32, 66, 40, 21}, 86: {32, 66, 40, 21}, 87: {32, 66, 40, 21}, 88: {32, 66, 40, 21}, 89: {32, 66, 40, 21}}
```

```
OUT : {1: set(), 2: set(), 3: set(), 4: set(), 5: {32, 66, 40, 21}, 6: set(), 7: set(), 8: set(), 9: set(), 10: set(), 11: set(), 12: set(), 13: set(), 14: {21}, 15: set(), 16: set(), 17: set(), 18: set(), 19: set(), 20: set(), 21: set(), 22: {32, 66, 40, 21}, 23: {32, 66, 40, 21}, 24: {32, 66, 40, 21}, 25: {32, 66, 40, 21}, 26: {32, 66, 40, 21}, 27: {32, 66, 40, 21}, 28: {32, 40, 21, 66}, 29: {32, 66, 40, 21}, 30: {32, 66, 40, 21}, 31: {32, 66, 40, 21}, 32: {32, 66, 40, 21}, 33: {40, 32, 21, 66}, 34: {32, 66, 40, 21}, 35: {32, 66, 40, 21}, 36: {32, 66, 40, 21}, 37: {32, 66, 40, 21}, 38: {32, 66, 40, 21}, 39: {32, 66, 40, 21}, 40: {32, 66, 40, 21}, 41: {32, 66, 40, 21}, 42: {32, 66, 40, 21}, 43: {32, 66, 40, 21}, 44: {32, 66, 40, 21}, 45: {32, 66, 40, 21}, 46: {32, 66, 40, 21}, 47: {32, 66, 40, 21}, 48: {32, 66, 40, 21}, 49: {32, 66, 40, 21}, 50: {32, 66, 40, 21}, 51: {32, 66, 40, 21}, 52: {32, 66, 40, 21}, 53: {32, 66, 40, 21}, 54: {32, 66, 40, 21}, 55: {32, 66, 40, 21}, 56: {32, 66, 40, 21}, 57: {32, 66, 40, 21}, 58: {32, 66, 40, 21}, 59: {32, 66, 40, 21}, 60: {32, 66, 40, 21}, 61: {32, 66, 40, 21}, 62: {32, 66, 40, 21}, 63: {32, 66, 40, 21}, 64: {32, 66, 40, 21}, 65: {32, 66, 40, 21}, 66: {32, 66, 40, 21}, 67: {32, 66, 40, 21}, 68: {32, 66, 40, 21}, 69: {32, 66, 40, 21}, 70: {32, 66, 40, 21}, 71: {32, 66, 40, 21}, 72: {32, 66, 40, 21}, 73: {32, 66, 40, 21}, 74: {32, 66, 40, 21}, 75: {32, 66, 40, 21}, 76: {32, 66, 40, 21}, 77: {32, 66, 40, 21}, 78: {32, 66, 40, 21}, 79: {32, 66, 40, 21}, 80: {32, 66, 40, 21}, 81: {32, 66, 40, 21}, 82: {32, 66, 40, 21}, 83: {32, 66, 40, 21}, 84: {32, 66, 40, 21}, 85: {32, 66, 40, 21}, 86: {32, 66, 40, 21}, 87: {32, 66, 40, 21}, 88: {32, 66, 40, 21}, 89: {32, 66, 40, 21}}
```

On remarque que la définition de la variable 'line' dans la ligne 9 est teintée par une source, ensuite cette variable est référencée dans la ligne 15 dans une redéfinition d'elle-même, ensuite la variable line est passée en paramètre du sink a la ligne 23 donc les paires (définition ligne 9 , référence ligne 23) et (définition ligne 15 , référence ligne 23) sont bien teintées. ceci est logique car la teinte pourrait atteindre le sink si l'exécution passe par le default du switch ou le case 2 dans les autres cas la variable line est soit redéfinie safe ou filtrée.

2. Implémentation et tests

2.1 Détection de la faille :

On exécute l'algorithme implémenté précédemment pour chacun des fichiers php de la plateforme web dans la partie 2 et on obtient une seule faille dans le fichier footer.php, la paire définition référence teinte correspondante à la faille est (définition ligne 33, référence ligne 36).

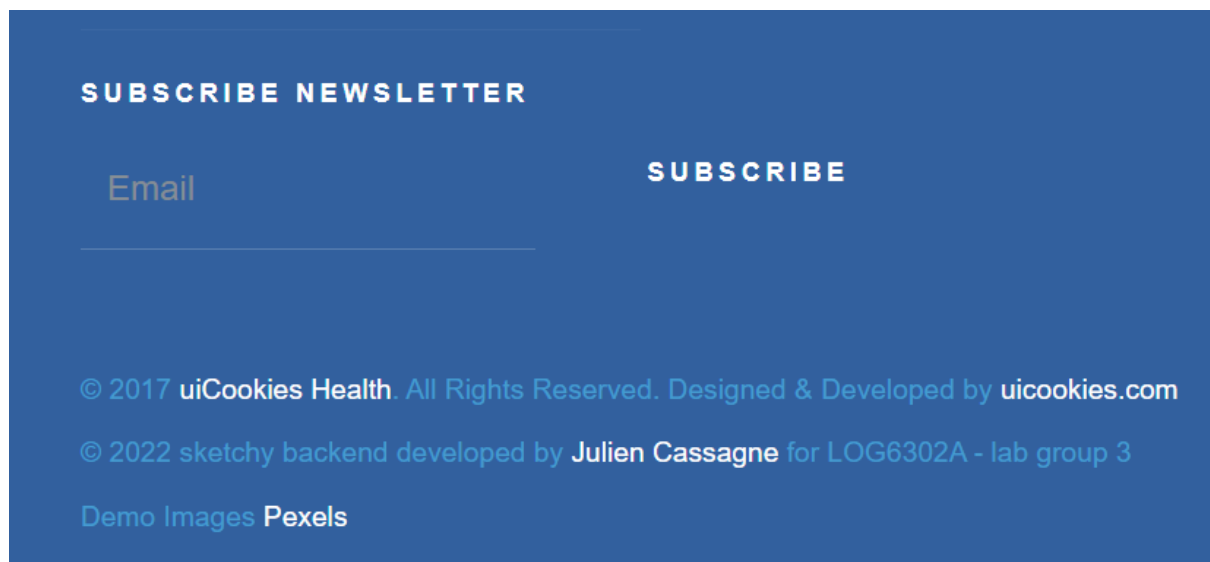
```
Les pairs def-refs possiblement teintés du fichier '/mnt/c/Users/Othman/Desktop/TP-LOG6302/log6302a_lab5/p

La paire ( définition : 'sql' ligne 33, référence : 'sql' ligne 36 ) est teintée

IN : {237: set(), 238: {261, 283, 348}, 239: set(), 240: set(), 241: set(), 242: set(), 243: set(), 244:
OUT : {237: set(), 238: {283, 348, 261}, 239: set(), 240: set(), 241: set(), 242: set(), 243: set(), 244:
```

en effet la variable \$email dans la ligne 30 est teintée, cette variable est ensuite filtrée dans la définition de \$mail (ligne 31), mais c'est la variable \$email qui est incluse dans la définition de \$sql (ligne 33) donc la teinte est propagée, et puisque \$sql est passée en paramètre de \$conn->query() (ligne 36) la teinte est alors propagée depuis la 33 jusqu'à la ligne 36.

Cette faille correspond au champs ci-dessous :



SUBSCRIBE NEWSLETTER

Email

SUBSCRIBE

© 2017 uiCookies Health. All Rights Reserved. Designed & Developed by uicookies.com

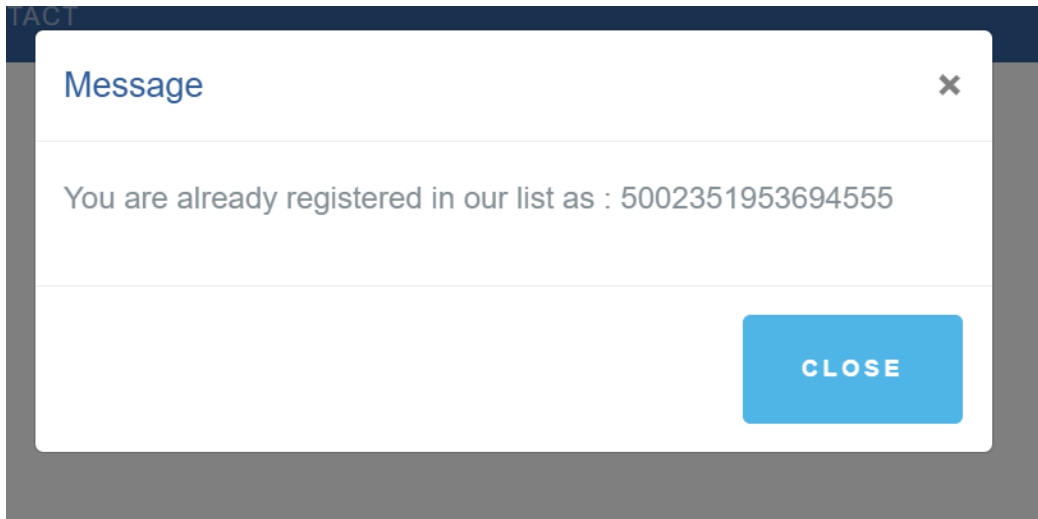
© 2022 sketchy backend developed by Julien Cassagne for LOG6302A - lab group 3

Demo Images Pexels

On peut exploiter cette faille pour extraire de la base de données des informations confidentielles relatives aux patients.

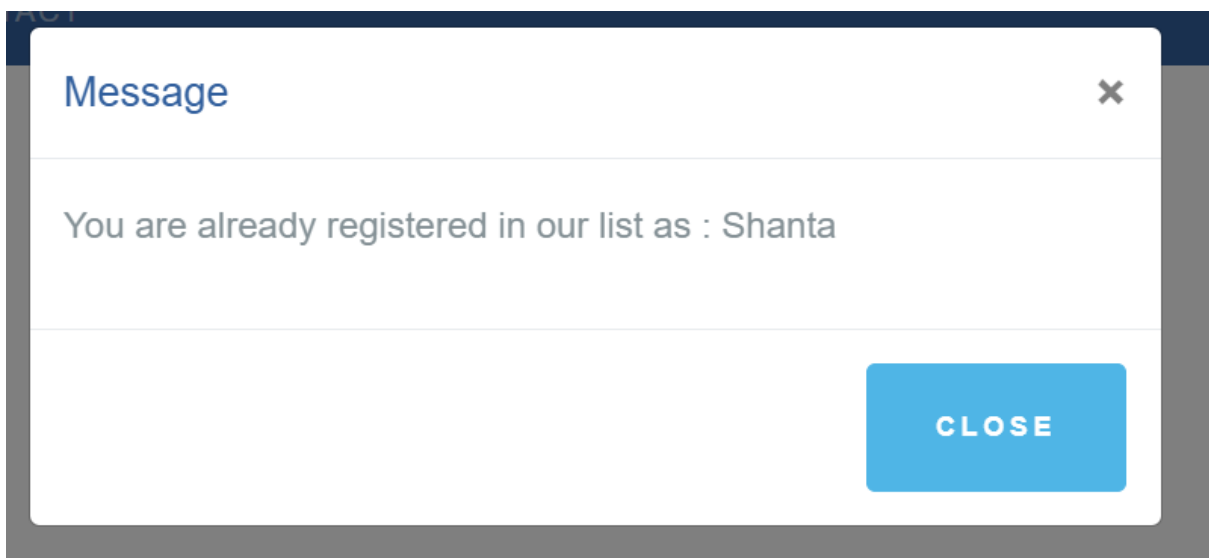
Par exemple remplir le champ par le contenu ci-dessous, permet d'extraire la carte de crédit de la base de données patients :

```
' UNION SELECT first_name,credit_card FROM patients WHERE last_name= '' OR '1'='1
```



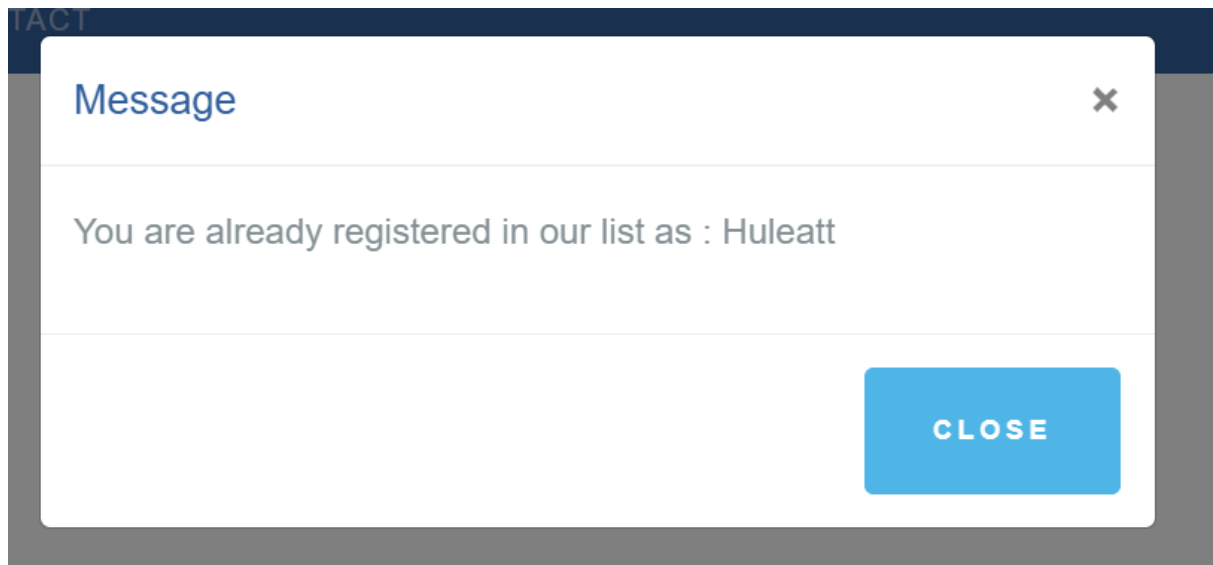
Pour obtenir first name :

```
' UNION SELECT first_name,first_name FROM patients WHERE last_name= '' OR '1'='1
```



Pour obtenir first name :

```
' UNION SELECT first_name,last_name FROM patients WHERE last_name= '' OR '1'='1
```

3. Limitations

Une limitation de l'approche suivi dans ce TP est que nous avons les fichiers taint préconçus et disponibles pour effectuer l'analyse, donc pour une approche complète il faudrait implémenter des algorithmes de détection des ensembles sources, safes, filters, sinks en phase préliminaire avant d'exécuter l'algorithme possibly tainted définitions.

4. Instructions pour exécuter le code

Le livrable se compose du présent rapport ainsi que du fichier `taint_annalysis.ipynb`. Pour une bonne exécution la seule chose à faire c'est d'adapter les chemins des fichiers dans le script.