Text

Description automatically generated with medium confidence

LOG3430 - Méthodes de test validation du logiciel

Hiver 2022

Laboratoire 1

Tests unitaires

Soumis par :

1956611 – Bryan Junior Ngatshou

matricule – Dat Phi Tran

matricule – Tristan Dea

06 février 2022

1. Couverture des jeux de tests après modifciations des fichiers tests

Les images qui suivent représentent la couverture sur différentes classes affectées par la modification des fichiers tests :

1. Crud.py
2. Email\_analyser.py
3. Vocabulary\_creator.py
4. Représentation des tests de flots de données
5. Code de la fonction Trust

La fonction de confiance que nous avons nommé calculate\_trust\_level est représenté dans l’aimage qui suit

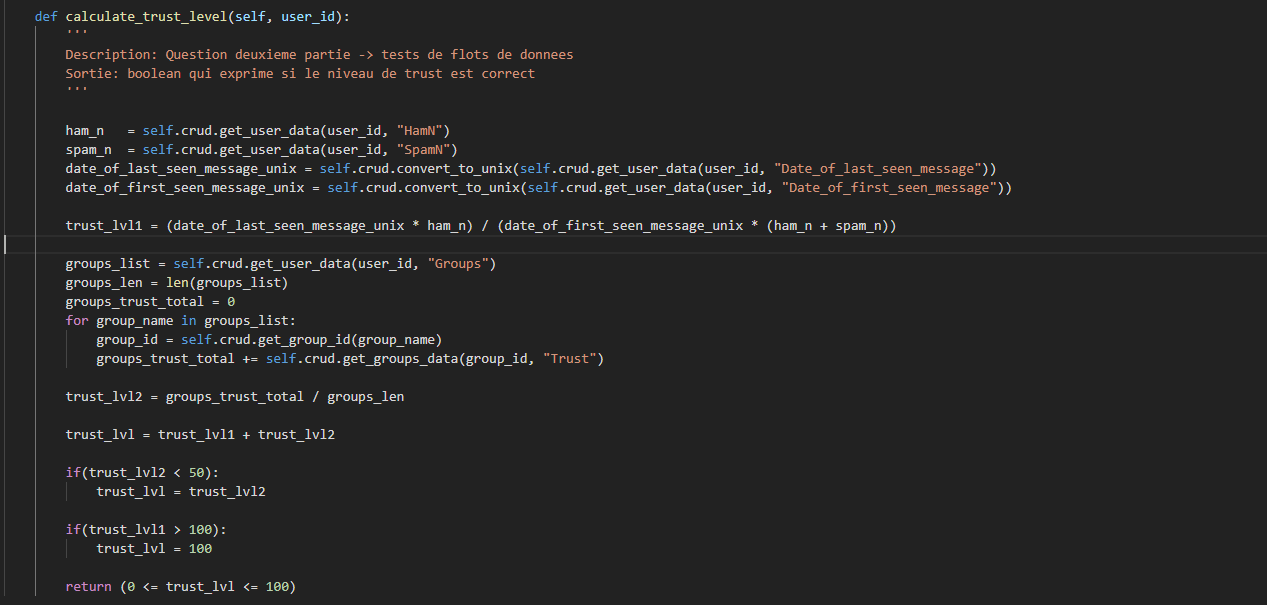


Figure 2.a : code de la fonction calculate\_trust\_level

1. **Graphe CFG de la fonction calculate\_trust\_level**

Nous avons divisé la fonction en 12 nœuds, représentés par l’image suivante.

Graphical user interface, application

Description automatically generated

Figure 2.b : représentation du graphe CFG de la fonction clacualte\_trust\_level

1. Tableaux de définitions et d’utilisation des variables de la fonction

Tout d’abord, nous avons identifié chaque variable dans chacun des nœuds et déterminé si il s’agissait d’une définition, d’une utilisation en tant que calcul ou d’une utilisation en tant que prédicat.

Tableau1 : identification des différentes variables

|  |  |  |  |
| --- | --- | --- | --- |
| Nœud n | DEF | C-USE | P-USE |
| 1 | user\_id, ham\_n, spam\_n, date\_of\_last\_seen\_message\_unix, date\_of\_first\_seen\_message\_unix | user\_id |  |
| 2 | trust\_lvl1 | date\_of\_last\_seen\_message\_unix, date\_of\_first\_seen\_message, ham\_n, spam\_n |  |
| 3 | groups\_list, groups\_len, groups\_trust\_total | user\_id, groups\_list |  |
| 4 | group\_name |  | groups\_list |
| 5 | group\_id, groups\_trust\_total | group\_name, group\_id |  |
| 6 | trust\_lvl2 | groups\_trust\_total, groups\_len |  |
| 7 | trust\_lvl | trust\_lvl1, trust\_lvl2 |  |
| 8 |  |  | trust\_lvl2 |
| 9 | trust\_lvl | trust\_lvl2 |  |
| 10 |  |  | trust\_lvl1 |
| 11 | trust\_lvl |  |  |
| 12 |  | trust\_lvl |  |

n : nœud

DEF : définition

C\_USE : utilisation pour calcul

P\_USE : utilisation de prédicats

Par la suite, nous avons identifié pour chacune des variables, le/les nœud(s) dans le(s)quel(s) elle était définie(ou redéfinie) et utilisée ( calcul ou prédicat).

Tableau 2 : identification des nœuds de définition et d’utilisation

|  |  |  |  |
| --- | --- | --- | --- |
| Variable | DEF | C-USE | P-USE |
| user\_id | 1 | 1 |  |
| ham\_n | 1 | 2 |  |
| spam\_n | 1 | 2 |  |
| date\_of\_last\_seen\_message\_unix | 1 | 2 |  |
| date\_of\_first\_seen\_message | 1 | 2 |  |
| trust\_lvl1 | 2 | 7 | 10 |
| groups\_list | 3 | 3 | 4 |
| groups\_len | 3 | 6 |  |
| groups\_trust\_total | 3, 5 | 6 |  |
| group\_name | 4 | 5 |  |
| group\_id | 5 | 5 |  |
| trust\_lvl2 | 6 | 7,9 | 8 |
| trust\_lvl | 7, 9, 11 | 12 |  |

DEF : définition

C\_USE : utilisation pour calcul

P\_USE : utilisation de prédicats

1. **ALL DEF**

Nous avons ensuite construis un tableau ALL DEFININITIONS ( ALL DEF) des différentes variables.

Tableau 3: DC-PATHS pour ALL DEF

|  |  |  |  |
| --- | --- | --- | --- |
| var | Src-dst | DC-PATH | #Chemin |
| user\_id | 1-1 | 1 | 1 |
| ham\_n | 1-2 | 1-2 | 1 |
| spam\_n | 1-2 | 1-2 | 1 |
| date\_of\_last\_seen\_message\_unix | 1-2 | 1-2 | 1 |
| date\_of\_first\_seen\_message\_unix | 1-2 | 1-2 | 1 |
| trust\_lvl1 | 2-7 | 2-3-4-6-7 | 1 |
| groups\_list | 3-3 | 3 | 1 |
| groups\_len | 3-6 | 3-4-6 | 1 |
| groups\_trust\_total | 3-6 | 3-4-6 | 1 |
| 5-6 | 5-4-6 | 2 |
| group\_name | 4-5 | 4-5 | 2 |
| group\_id | 5-5 | 5 | 2 |
| trust\_lvl2 | 6-7 | 6-7 | 1 |
| trust\_lvl | 7-12 | 7-8-10-11-12 | 2 |
| 9-12 | 9-10-12 | 1 |
| 11-12 | 11-12 | 2 |

Src-dst : nœud de première définition - nœud de dernière définition

DC-PATH : chemins possibles entre la première et la dernière définition

#Chemin: numéro de chemin

Pour des test de ce critères, il faudrait trouver des tests pour les chemins :

Path1 = {1, 2, 3, 4, 6, 7, 8, 9, 10, 12}

Path2 = {1, 2, 3, 4, 5, 4, 6, 7, 8, 10, 11, 12}

ou

Path1 = {1, 2, 3, 4, 6, 7, 8, 10,11, 12}

Path2 = {1, 2, 3, 4, 5, 4, 6, 7, 8, 9, 10, 12}

Et avoir des tests cas de tests :

D1 : <{user\_id= 0}> pour le chemin 1

D2: <{user\_id= 1}> pour le chemin 2

1. **ALL C-USE**

Le tableau qui suit représente le critère de toutes les utilisations par calcul

Tableau 4. DC-PATHS pour ALL C-USE/SOME P-USE

|  |  |  |  |
| --- | --- | --- | --- |
| var | Src-dst | DC-PATH | #Chemin |
| user\_id | 1-1 | 1 | 1 |
| ham\_n | 1-2 | 1-2 | 1 |
| spam\_n | 1-2 | 1-2 | 1 |
| date\_of\_last\_seen\_message\_unix | 1-2 | 1-2 | 1 |
| date\_of\_first\_seen\_message\_unix | 1-2 | 1-2 | 1 |
| trust\_lvl1 | 2-7 | 2-3-4-6-7 | 1 |
| groups\_list | 3-3 | 3 | 1 |
| groups\_len | 3-6 | 3-4-6 | 1 |
| groups\_trust\_total | 3-6 | 3-4-6 | 1 |
|  | 5-6 | 5-4-6 | 2 |
| group\_name | 4-5 | 4-5 | 2 |
| group\_id | 5-5 | 5 | 2 |
| trust\_lvl2 | 6-7 | 6-7 | 1 |
|  | 6-9 | 6-7-8-9 | 1 |
| trust\_lvl | 7-12 | 7-8-10-11-12 | 2 |
|  | 9-12 | 9-10-12 | 1 |
|  | 11-12 | 11-12 | 2 |

Src-dst : nœud de première définition - nœud de dernière définition

DC-PATH : chemins possibles entre la première et la dernière définition

#Chemin: numéro de chemin

Pour des test de ce critères, il faudrait trouver des tests pour les chemins :

Path1 = {1, 2, 3, 4, 6, 7, 8, 9, 10, 12}

Path2 = {1, 2, 3, 4, 5, 4, 6, 7, 8, 10, 11, 12}

ou

Path1 = {1, 2, 3, 4, 6, 7, 8, 10,11, 12}

Path2 = {1, 2, 3, 4, 5, 4, 6, 7, 8, 9, 10, 12}

Et avoir des tests cas de tests :

D1 : <{user\_id= 0}> pour le chemin 1

D2: <{user\_id= 1}> pour le chemin 2

1. **ALL P-USE**

Ce tableau décrit le critère d’utilisation par prédicat de nos variables.

Tableau 5 DC-PATHS pour ALL P-USE/SOME C-USE

|  |  |  |  |
| --- | --- | --- | --- |
| var | Src-dst | DC-PATH | #Chemin |
| user\_id | 1-1 | 1 | 1 |
| ham\_n | 1-2 | 1-2 | 1 |
| spam\_n | 1-2 | 1-2 | 1 |
| date\_of\_last\_seen\_message\_unix | 1-2 | 1-2 | 1 |
| date\_of\_first\_seen\_message\_unix | 1-2 | 1-2 | 1 |
| trust\_lvl1 | 2-10 | 2-3-4-6-7-8-9-10 | 1 |
| groups\_list | 3-4 | 3-4 | 1 |
| groups\_len | 3-6 | 3-4-6 | 1 |
| groups\_trust\_total | 3-6 | 3-4-6 | 1 |
|  | 5-6 | 5-4-6 | 2 |
| group\_name | 4-5 | 4-5 | 2 |
| group\_id | 5-5 | 5 | 2 |
| trust\_lvl2 | 6-8 | 6-7-8 | 1 |
| trust\_lvl | 7-12 | 7-8-10-11-12 | 2 |
|  | 9-12 | 9-10-12 | 1 |
|  | 11-12 | 11-12 | 2 |

Src-dst : nœud de première définition - nœud de dernière définition

DC-PATH : chemins possibles entre la première et la dernière définition

#Chemin: numéro de chemin

Pour des tests de ce critère, il faudrait trouver des tests pour les chemins :

Path1 = {1, 2, 3, 4, 6, 7, 8, 9, 10, 12}

Path2 = {1, 2, 3, 4, 5, 4, 6, 7, 8, 10, 11, 12}

ou

Path1 = {1, 2, 3, 4, 6, 7, 8, 10,11, 12}

Path2 = {1, 2, 3, 4, 5, 4, 6, 7, 8, 9, 10, 12}

Et avoir des jeux de tests

D1 : <{user\_id= 0}> pour le chemin 1

D2: <{user\_id= 1}> pour le chemin 2

1. **ALL USE**

Critère all-use pour la fonction calculate\_trust\_level

Tableau 6 : DC-PATHS pour ALL USE

|  |  |  |  |
| --- | --- | --- | --- |
| var | Src-dst | DC-PATH | #Chemin |
| user\_id | 1-1 | 1 | 1 |
| ham\_n | 1-2 | 1-2 | 1 |
| spam\_n | 1-2 | 1-2 | 1 |
| date\_of\_last\_seen\_message\_unix | 1-2 | 1-2 | 1 |
| date\_of\_first\_seen\_message\_unix | 1-2 | 1-2 | 1 |
| trust\_lvl1 | 2-7 | 2-3-4-6-7 | 1 |
|  | 2-10 | 2-3-4-6-7-8-10 | 1 |
| groups\_list | 3-3 | 3 | 1 |
|  | 3-4 | 3-4 | 1 |
| groups\_len | 3-6 | 3-4-6 | 1 |
| groups\_trust\_total | 3-6 | 3-4-6 | 1 |
|  | 5-6 | 5-4-6 | 2 |
| group\_name | 4-5 | 4-5 | 2 |
| group\_id | 5-5 | 5 | 2 |
| trust\_lvl2 | 6-7 | 6-7 | 1 |
|  | 6-8 | 6-7-8 | 1 |
|  | 6-9 | 6-7-8-9 | 1 |
| trust\_lvl | 7-12 | 7-8-10-11-12 | 2 |
|  | 9-12 | 9-10-12 | 1 |
|  | 11-12 | 11-12 | 2 |

Src-dst : nœud de première définition - nœud de dernière définition

DC-PATH : chemins possibles entre la première et la dernière définition

#Chemin: numéro de chemin

Pour des test de ce critères, il faudrait trouver des tests pour les chemins :

Path1 = {1, 2, 3, 4, 6, 7, 8, 9, 10, 12}

Path2 = {1, 2, 3, 4, 5, 4, 6, 7, 8, 10, 11, 12}

ou

Path1 = {1, 2, 3, 4, 6, 7, 8, 10,11, 12}

Path2 = {1, 2, 3, 4, 5, 4, 6, 7, 8, 9, 10, 12}

Et comme dans les cas précédant, avoir des jeux de tests

D1 : <{user\_id= 0}> pour le chemin 1

D2: <{user\_id= 1}> pour le chemin 2

Le critère le plus stricte serait le critère ALL\_USE car il vérifie en quelque sorte les 3 autres critères en même temps.