

LOG1000 - Ingénierie Logicielle

Travail Pratique # 3



Mélody Roy (1991902)
Pier-Luc Tanguay (1953707)

4 novembre 2019
Polytechnique Montréal

Partie 1

1. Révision technique de code source

1. Qu'est-ce que le logiciel Gerrit et à quoi sert-il ?

Gerrit est une application web de revue de code pour faciliter la collaboration entre les membres d'une équipe travaillant sur un projet. Il permet entre autres de pouvoir accepter et refuser des modifications qui ont été soumis par l'entremise de l'outil VCS Git.

2. Dans un contexte de contribution à un projet, qu'est-ce qu'une révision (patch) ?

Une révision (patch) est une partie de code d'un logiciel qui a été ajouté à un projet déjà conçu afin d'y apporter des améliorations ou modifications.

3. Ouvrez la page Gerrit de Wireshark (<https://code.wireshark.org/review>). Dans l'onglet All dans le coin gauche de la page, que signifie les sous-sections Open, Merged and Abandoned ?

Ces sections correspondent à trois différents états des révisions qui ont été soumis.

- a. Open : ce sont les révisions qui sont en cours de vérification dans le projet.
- b. Merged : les contributeurs ont accepté les changements et la révision de code a été "merged" avec la boîte distante.
- c. Abandoned : correspond aux révisions refusés par les collaborateurs du projet, et qui seront ignorés. Ceux-ci ne seront pas envoyé à la boîte distance.

Dans la barre de recherche, envoyez la requête **77f7721305ff04ac40bf442d196d0a9066bb6d58**, correspondant à l'identifiant d'une révision technique.

4. Qui est l'auteur de cette révision ?

Vadim Yanitskiy

5. Combien de fichier(s) a/ont subi de changement ? Le(s) quel(s) ?

1 fichier a été modifié.

Modifications:

epan/dissectors/packet-gsmtap.c (62 insertions)

6. Décrivez brièvement ce que fait cette révision.

Ajout de la fonction de dissection des messages PTCCH/D (soit le Packet Timing advance control channel de type Downlink).

7. Sur cette même révision, repérez la section Code-Review. Qu'est-il indiqué et qu'est-ce que cela signifie ?

Code-Review +2  Anders Broman
+1  Pau Espin Pedrol

Il est affiché:

Cela signifie que Pau Espin Pedrol a vérifié le code (+1) et que Anders Broman a approuvé le code (+2).

Répondez aux questions 8 à 11 inclusivement, mais cette fois avec la révision 49e259f04f86ccadb499be43f5b2b16668d0ad68.

8. Qui est l'auteur de cette révision ?

Pascal Quantin

9. Combien de fichier(s) a/ont subi de changement ? Le(s) quel(s) ?

3 fichiers ont été modifiés


Modifications:

epan/dissectors/asn1/hnbap/hnbap.cnf
epan/dissectors/asn1/hnbap/**packet-hnbap-template.c**
epan/dissectors/**packet-hnbap.c**

10. Décrivez brièvement ce que fait cette révision.

Selon, le Commit Message, decode IMSI IE.

11. Sur cette même révision, repérez la section Code-Review. Qu'est-il indiqué et qu'est-ce que cela signifie ?

Il est affiché: Code-Review +2  Pascal Quantin . Pascal Quantin a approuvé la révision (+2).

2. Gestion des bogues

Le projet Wireshark utilise Bugzilla pour la gestion des bogues. Accéder au dépôt Bugzilla du projet par <https://bugs.wireshark.org/bugzilla/>.

12. Trouvez le bogue #10476. Quel est le nom de la personne qui a trouvé le bogue? Est-ce que ce rapporteur est un être humain ?

Alex Kirk, il s'agit d'un être humain qui a trouvé le bogue

13. Quel est le niveau d'importance de ce bogue ?

Il s'agit d'un bogue légèrement majeur (low Major)

14. Comment est déterminé le niveau d'importance d'un bogue ?

Il est déterminé en fonction de sa priorité et de sa sévérité.

15. Est-ce que le bogue #10476 risque de compromettre le fonctionnement du logiciel ? Pourquoi ?

Oui, ce bogue fait échouer la compilation dans certaines conditions, ce qui compromet le bon fonctionnement du logiciel pour ces cas précis.

16. Combien de personnes ont commenté le bogue, excluant le commentaire initial.

Il y a eu 43 commentaires, excluant le commentaire initial, provenant de 9 sources différentes.

17. Est-ce que la communauté a confirmé la validité du bogue ? Si oui, comment ?

Oui, on peut voir en analysant les commentaires que plusieurs autres membres ont le même problème. Ils ont pu reproduire le crash dans certaines conditions (tel que vu au commentaire 43).

18. Est-ce que le bogue a été résolu ? Qu'est-ce qui vous l'indique ?

Ce bogue a été résolu et corrigé tel qu'on peut le voir dans la section Status.

3. Intégration continue

L'intégration continue est une pratique de développement logiciel qui consiste à intégrer le code pour produire un exécutable régulièrement et automatiquement. L'objectif est de savoir en continu si les changements ("commits") effectués produisent toujours un logiciel fonctionnel qui passent les cas de tests définis. Wireshark utilise un outil d'intégration continue appelé Buildbot.

19. Expliquez l'utilité de Buildbot dans un contexte de projet open source.

Buildbot est un outil de développement logiciel qui permet de valider des révisions par un processus automatique de compilations et de tests.

20. Identifiez les différents builders disponibles pour ce projet en cliquant sur Builders au haut de la page
(<https://buildbot.wireshark.org/wireshark-master/>).

Builders:

Clang Code Analysis	#5139 failed test-tsan fuzz-menagerie	building 14 pending
Ubuntu 18.04 x64	#3402 build successful	idle
Visual Studio Code Analysis	#25122 build successful	idle
Windows Server 2019 x64	#372 build successful	idle
Windows Server 2019 x86	#447 build successful	idle
macOS 10.14 x64	#1958 build successful	idle

Figure 1: capture d'écran des builders

21. Visitez la page Waterfall. Que signifient les couleurs vert, rouge et orange ?

Le build a réussi la compilation et tests

En processus de vérification

Avertissement suite à la vérification

La révision n'a pas passé les test

22. Identifiez un build qui a échoué ou bien qui possède des avertissements et collez une capture d'écran.

compile failed stdio
ran CMake stdio
created CMake build directory stdio
set release information stdio
update stdio
Build 1956

Figure 2: capture d'écran d'un build qui échoue

On voit à la figure 2 qu'il y a une erreur à la compilation.

23. Expliquez la cause des problèmes en accédant aux messages du build.

Erreur de compilation. Le fichier objet “filter_dialog.cpp.o” n’a pas été en mesure d’être produit. Il semble y avoir des erreurs aux lignes 20 de “filter_dialog.cpp” et 15 de “filter_dialog.h”.

```
[2177/2484] Building CXX object ui/qt/CMakeFiles/qtui.dir/extcap_argument_multiselect.o
[2178/2484] Building CXX object ui/qt/CMakeFiles/qtui.dir/file_set_dialog.cpp.o
[2179/2484] Building CXX object ui/qt/CMakeFiles/qtui.dir/filter_dialog.cpp.o
FAILED: ui/qt/CMakeFiles/qtui.dir/filter_dialog.cpp.o
/Applications/Xcode.app/Contents/Developer/Toolchains/XcodeDefault.xctoolchain/usr/bin/
DQT_MACEXTRAS_LIB -DQT_MULTIMEDIA_LIB -DQT_NETWORK_LIB -DQT_PRINTSUPPORT_LIB -DQT_SVG_
I../ui/qt -isystem /usr/local/include/glib-2.0 -isystem /usr/local/lib/glib-2.0/includ
/Users/buildslave/Qt5.12.5/5.12.5/clang_64/lib/QtCore.framework/Headers -isystem /User
/Users/buildslave/Qt5.12.5/5.12.5/clang_64/lib/QtMultimedia.framework/Headers -isystem
/Users/buildslave/Qt5.12.5/5.12.5/clang_64/lib/QtGui.framework/Headers -isystem
/Applications/Xcode.app/Contents/Developer/Platforms/MacOSX.platform/Developer/SDKs/Ma
/Users/buildslave/Qt5.12.5/5.12.5/clang_64/lib/QtPrintSupport.framework/Headers -isyst
/Users/buildslave/Qt5.12.5/5.12.5/clang_64/lib/QtSvg.framework/Headers -isystem /Users
-Wextra -Wendif-labels -Wpointer-arith -Wformat-security -fwrapv -fno-strict-overflow
strings -Wno-long-long -Wheader-guard -Wcomma -Wshorten-64-to-32 -Wframe-larger-than=3
/Applications/Xcode.app/Contents/Developer/Platforms/MacOSX.platform/Developer/SDKs/Ma
ui/qt/CMakeFiles/qtui.dir/filter_dialog.cpp.o -MF ui/qt/CMakeFiles/qtui.dir/filter_dia
In file included from ../ui/qt/filter_dialog.cpp:20:
In file included from ../ui/qt/filter_dialog.h:15:
../ui/qt/models/filter_list_model.h:51:18: error: 'setData' overrides a member function
virtual bool setData(const QModelIndex &index, const QVariant &value, int role);
```

Figure 3: capture d'écran d'erreur de l'échec de la question #22

24. Quel est le numéro de la révision technique de ce build échoué et qui en est l'auteur ?

Le numéro de la révision technique est “34879”.

Ronald Knall est l'auteur de la révision.

Change **34879** - Merged

Qt: Speed up filtering in VoIP dialog

Speed up the filtering, by only taking rows once


Change-Id: [I6123b2974fd58480463e4a0bd02524f6b5c0a353](#)

Reviewed-on: <https://code.wireshark.org/review/34879>


Petri-Dish: Roland Knall <rknall@gmail.com>

Tested-by: Petri Dish Buildbot

Reviewed-by: Roland Knall <rknall@gmail.com>

Owner  Roland Knall

Assignee


Reviewers Petri Dish Buildbot 


Project [wireshark](#)

Branch [master](#)


Topic [qt_enabled_protocols](#)


Updated 22 hours ago

Code-Review +2 

Petri-Dish +1 

Verified +1 Petri Dish Buildbot

Author  [Roland Knall <rknall@gmail.com>](mailto:rknall@gmail.com)

Committer  [Roland Knall <rknall@gmail.com>](mailto:rknall@gmail.com)


Commit 8d62cc3aea1d01cd23387feaeaf510ef51694003

Parent(s) 541afedfbc64503df0e7301d7614a30c9ba8ad49

Change-Id I6123b2974fd58480463e4a0bd02524f6b5c0a353

Oct 28, 2019 12:07 PM

Oct 28, 2019 1:31 PM






Figure 4: Révision du build qui a échoué

Partie 2

Dans cette deuxième partie, le processus de contribution à un projet open source publié sur Gitlab sera exploré. L'objectif étant de s'initier au processus de contribution à un projet d'équipe.

1. Prérequis

Ouvrez la page de Gitlab (<https://gitlab.com/>) et créez un compte de type *Libre*.

2. Développement

25. Après avoir créé un compte Libre sur Gitlab, Allez sur la page Gitlab : https://gitlab.com/Sergiubest18/log1000_a19. Dans ce dernier URL, à quoi correspond log1000_A19 (du point de vue de Gitlab) ?

Il s'agit d'un projet personnel de l'utilisateur Sergiu Mihaltan.

26. À quoi l'option fork sert-elle ? Sur la page des détails du répertoire, faites-un fork du projet pour votre compte Gitlab.

Il s'agit d'une copie du projet qui nous permet d'effectuer des modifications sans toucher au projet original.

27. Sur cette même page, cliquez sur le bouton possédant le symbole d'addition et puis créer une nouvelle branche en lui donnant comme nom, le numéro de groupe de votre équipe (equipeX).

Créer nouvelle branche:

```
git branch equipe05
```

28. Clonez le répertoire localement sur votre machine à l'aide de Git et accéder à votre branche dans le terminal. Quelle commande avez-vous utilisée ? Le but des modifications de cette partie est d'ajouter, une commande correspondante à créer l'objet « moteur.o » dans le makefile. Exécuter make pour générer les objets et l'exécutable du projet

```
git clone https://gitlab.com/Meleau/log1000\_a19.git
```


29. Ajoutez vos modifications au staging local de votre répertoire Git. Quelle commande avez-vous utilisé ? Faites un commit et un push de vos changements vers le serveur. Retournez sur votre compte Gitlab, Projects -> Your projects et puis le répertoire correspondant du TP. Sur la barre de navigation de gauche, accédez à la section Merge Requests

Pour modifier au staging local:

`git add .`

30. Qu'est-ce qu'un Merge Request?

C'est une demande que nous envoyons à l'auteur original du code pour y ajouter nos modifications.

31. Quel est l'utilité d'un Merge Request? Cliquez sur New merge request, afin de créer un nouveau Merge Request.

Si nous effectuons des modifications sur un projet quelconque pas notre compte GitLab, les modifications ne seront jamais "merge" avec l'auteur qui a créé le code. Nous n'avons qu'une copie que nous avons ensuite modifiée. Le merge request nous permettra d'envoyer une demande à l'auteur pour lui soumettre notre proposition de modification, qu'il pourra ensuite valider.

32. Quelles sont les branches source et destination correspondantes à votre Merge Request ? Continuez la création du Merge Request en y ajoutant un titre (supprimez WIP du titre) et une description. Indiquer dans votre titre le numéro de groupe de votre équipe. Laissez les deux options décochées, et faites la soumission du Merge Request

La branche source est *equipe05* et la branche de destination est *master*.

3. Révision de code

Chaque équipe doit faire la révision de code d'une autre équipe correspondant à un Merge Request.

33. Choisir une équipe que vous allez réviser (N'oubliez pas de mentionner l'équipe que vous avez choisie dans votre rapport). Il faut choisir une équipe à réviser qui a eu au maximum 2 révisions. Une équipe pourra avoir au maximum 3 révisions.

Nous allons réviser l'équipe 30.

Faites une révision technique du Merge Request que vous avez choisie.

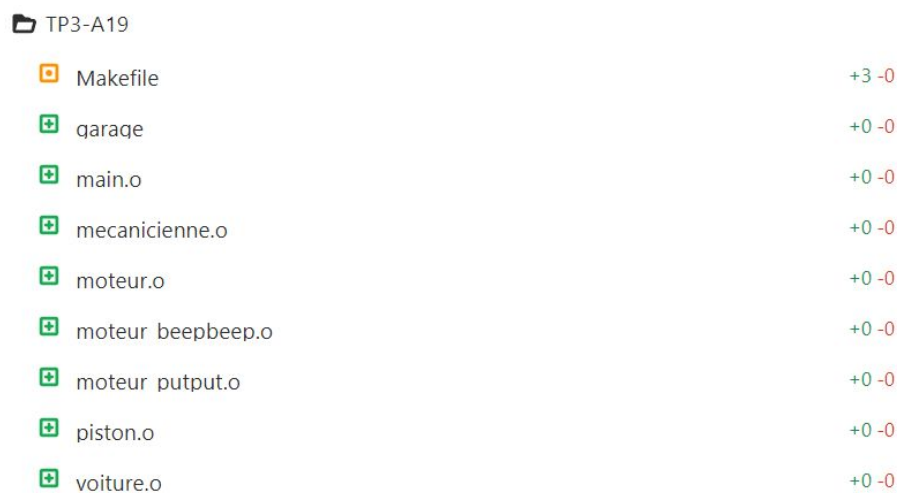
34. Quel(s) est/sont les commits associées ?

Un commit a été effectué nommée "modification du makefile".

commit shaw: 25e7a7b0ea9803c3e2fe83cf4b3b2f8c5120a7f8

35. Quel(s) est/sont les fichiers qui ont été modifiés ?

Il y a eu 3 insertions dans le Makefile.



TP3-A19	
Makefile	+3 -0
garage	+0 -0
main.o	+0 -0
mecanicienne.o	+0 -0
moteur.o	+0 -0
moteur beepbeep.o	+0 -0
moteur putput.o	+0 -0
piston.o	+0 -0
voiture.o	+0 -0

Figure 5: capture d'écran des fichiers différents de l'original

36. Est-ce que les modifications demandées sont correctes ?

Oui, on peut voir que le makefile a bien été modifié pour créer le fichier moteur.o.

8	8	main.o: main.cpp moteur.h voiture.h mecanicienne.h
9	9	moteur_putput.h moteur_beepbeep.h
10	10	g++ -o \$@ -c \$<
	11	+ moteur.o: moteur.cpp moteur.h piston.h
	12	+ g++ -o \$@ -c \$<
	13	+
11	14	piston.o: piston.cpp piston.h
12	15	g++ -o \$@ -c \$<
13	16	

Figure 6: modification du Makefile

37. Est-ce que les objets et l'exécutable du projet ont été créés ?

Oui, on peut voir dans la figure 5 que les fichiers garage, main.o, mecanicienne.o, moteur.o, moteur beepbeep.o, moteur putput.o, piston.o et voiture.o ont été créés.

38. Attribuez un score aux modifications et justifiez votre choix.

(+1) Les modifications apportées au makefile permettent de créer correctement le fichier moteur.o. De plus, tous les fichiers .o et l'exécutable ont été créés avec succès.

39. Ajoutez des commentaires, incluant le score et votre justification et publiez-les dans la discussion du Merge Request.



Mélody Roy @Meleau · just now

😊
🗨️
✎️
⋮

(+1) Les modifications apportées au makefile permettent de créer correctement le fichier moteur.o. De plus, tous les fichiers .o et l'exécutable ont été créés avec succès.

Figure 7: ajout d'un commentaire