Projet PGP

Etude et implantation d'un outil graphique de gestion de clé PGP

Cahier de Recette 0.6

15 mai 2015

Auteur(s): Pierre Balmelle, Lucas Barbay
Relecteur(s): Olivier Thibault

Version	Date	Changelog
0.6	07/03/2015	Ajout des tests de non-régression.
0.5	15/12/2014	Respect de la STB v0.4 et ajout des données de test.
0.4	05/12/2014	Ajout de toutes les procédures liées aux commandes GPG.
0.3	24/11/2014	Respect de la STB v0.3.
0.2	15/11/2014	Ajout des parties Introduction, Documents applicables et de référence, Environnement de test, Responsabilités, Stratégie de tests, Gestion des anomalies.
0.1	31/10/2014	Version initiale.

Table des matières

1 Introduction

Ce document est le Cahier de Recette (CDR) de la réalisation d'une interface graphique pour le logiciel GnuPG. Cette interface permettra d'utiliser complètement l'outil OpenPGP de façon intuitive et pédagogique de façon à être accessible même par des personnes ayant des connaissances limitées dans le domaine de l'informatique.

1.1 Fonctionnalités du logiciel

Le logiciel est une interface graphique pour le GnuPG (implémentation GNU du standart OpenPGP). Les différents cas d'utilisations :

— Exécuter une action GPG.
— Affichage des commandes et des erreurs.
— Choix du fichier de configuration.
— Attaque sur la seconde pré-image.

1.2 Liste des objets à tester

Nous avons dégagé 2 objets à tester :

— L'interface graphique des fonctions GnuPG.
— L'implémentation de l'attaque sur les secondes pré-images.

L'interface graphique permettra d'utiliser chaque commande et option de GPG et ce de façon le plus intuitif possible. Nous devrons tester le résultat des commandes ainsi que le temps d'exécution pour satisfaire le client en terme de fluidité de l'interface.

Enfin, il nous faudra tester l'implémentation de l'attaque.

2 Documents applicables et de référence

Différents documents de référence :

— Définitions du standard OpenPGP RFC 4880 et RFC 2440.
— Le logiciel GnuPG (GNU Privacy Guard) implantation Open Source de OpenPGP.
— Editeurs graphiques existant (KGpg, GPA, Seahorse).

3 Terminologie et sigles utilisés

Le glossaire est dans un fichier à part, il est commun aux autres documents de gestion de projet.

4 Environnement de test

4.1 Sites de réalisation des tests

Les test seront réalisés soit sur le site de l'université soit au domicile des deux testeurs sur leur machine personnelle.

4.2 Configurations matérielles utilisées

Voici la configuration des machines personnelles des deux testeurs :

— PC n°1 : Un ordinateur portable équipé de Xubuntu 64 bits avec un processeur Intel i5-3210M @ 2.50GHz et 6Go de RAM.
— PC n°2 : Un ordinateur portable équipé de Xubuntu 64 bits avec un processeur Intel i7-2630QM @ 2.00 GHz et 6Go de RAM.

Seront également utilisées des machines virtuelles pour répondre aux besoins du client sur la compatibilité de l'interface (KDE et GNOME).

4.3 Outils de tests mis en oeuvre

Nous avons décidé d'utiliser le framework Qt pour réaliser ce projet. C'est donc le module Qttest de Qt que l'on a choisi car il permet de réaliser simplement les tests unitaires et benchmark.

4.4 Jeux de données/Bases de données

Nous aurons comme jeu de données les différentes clés contenues dans le trousseau de test.

4.5 Contraintes à prendre en compte

Chaque procédure de test devra être réalisée sous KDE et sous GNOME. Les tests unitaires et de non-régression seront réalisés à l'aide de l'api Qt (automatiquement), et les tests d'intégration à l'aide de l'interface (manuellement).

5 Responsabilités

La création et la gestion des procédures de test sera effectuée par les responsables qualité de l'équipe. Chaque équipe travaillant au développement d'un composant est chargée de tester celui-ci continuellement et de publier les résultats des tests.

Afin d'assurer la compréhension générale du code source, toute l'équipe peut et est encouragée à signaler une anomalie détectée au cours du développement : soit elle est résolue par la même personne/groupe, soit cette personne/ce groupe transmet les résultats des tests à une autre personne/groupe afin que celle-ci travaille à la réparation de l'anomalie.

À tout moment une équipe peut reporter des anomalies dans une procédure de test. Dans ce cas, les responsables techniques concernés travailleront à améliorer la procédure incriminée et diffuseront les mises à jour à l'ensemble de l'équipe.

Dans tous les cas, les instructions incluses dans la partie *Gestion des anomalies* sont à respecter.

6 Stratégies de tests

6.1 Description de l'approche et des phases de test

Dès qu'un test pour un composant sera réalisable, il sera fait. Ces tests devront être appliqués à chaque nouvelle version du composant.

6.2 Campagne de test

Les différentes campagnes de tests correspondront aux livrables envoyés au client.

6.3 Ordre d'exécution des tests

L'ordre d'exécution des tests suivra l'ordre du plan de développement.

6.4 Critères d'arrêt des tests

Un test est considéré comme échoué dès qu'une de ses étapes échoue, produit un résultat non attendu ou est impossible à mener à bout.

7 Gestion des anomalies

Quand des anomalies seront détectées, la procédure suivante devra être respectée :
— Création d'un mémo précisant l'anomalie rencontrée ainsi que les conditions de reproduction de l'anomalie si disponibles et l'état du système lors de l'apparition de l'anomalie. Un identifiant unique lui sera attribué.
— Ajout d'une entrée au journal de test précisant la date du test, la référence du mémo, le nom ainsi que la référence de l'exigence de qualité non respectée.
— Diffusion de la note à l'équipe de développement.
— Une personne est désignée pour corriger l'anomalie.
— Cette personne vérifie la gravité de l'anomalie, puis la corrige.
— Une deuxième personne s'assure que l'anomalie est bien corrigée.
— Une fois la vérification effectuée, la personne ayant corrigé l'anomalie établit un contre-mémo portant le même identifiant et précisant les raisons supposées de l'anomalie ainsi que les corrections mises en place.

Dans la pratique, la gestions des anomalies sera faite à l'aide de l'outil *issue tracker* de la plate-forme d'hébergement Gitlab. Cet outil permet la création de files de discussion associées à un bug de l'application. Il laisse aussi la possibilité aux collaborateurs du projet de prendre en charge une *issue* particulière. L'ensemble des *issues* ouvertes est conservé par Gitlab ainsi que les files de discussion associés permettant de parcourir l'historique des bugs du projet.

8 Procédures de test

8.1 Types de tests

Les différents types de tests utilisés sont :
— Les tests unitaires, préfixés par u,
— Les tests de non-régression, préfixés par nr,
— Les tests d'intégration, préfixés par i.

8.2 Tests unitaires

Objet testé : GuiPG			Version : 0.3	
Objectif de test : Tester les classes liées à la gestion de profil				
Procédure n° U3			Pré-requis :	
N°	Actions	Résultats attendus	Exigence	OK/KO
1	Demander la liste des profils	On récupère une liste de profils		OK
2	Créer un profil	Le profil est créé		OK
3	Supprimer ce profil	Le profil est supprimé		OK
4	Demander la liste des profils	La même liste est renvoyée qu'au début		OK
5				
6				
7				
8				
9				
10				

Objet testé : Classe Profile			Version : 0.5	
Objectif de test : Test de la classe				
Procédure n° U5			Pré-requis :	
N°	Actions	Résultats attendus	Exigence	OK/KO
1	Créer une instance de la classe			OK
2	Utiliser la fonction getId	La fonction renvoie l'id donnée au constructeur		OK
3	Utiliser la fonction setConfigurationPath			OK
4	Utiliser la fonction getConfigurationPath	La fonction renvoie le path donné à l'étape précédente		OK
5	Utiliser la fonction setGPGExecutable			OK
6	Utiliser la fonction getGPGExecutable	La fonction renvoie le path donné à l'étape précédente		OK
7	Utiliser la fonction getName	La fonction renvoie le nom donné au constructeur		OK
8				
9				
10				

Objet testé : Classe Launcher			Version : 0.5	
Objectif de test : Test de la classe				
Procédure n° U6			Pré-requis :	
N°	Actions	Résultats attendus	Exigence	OK/KO
1	Créer une instance de la classe			OK
2	Utiliser la fonction alreadyRun	La fonction renvoie false		OK
3	Utiliser la fonction run	Une exception IllegalArgumentException est levée		OK
4	Utiliser la fonction addMainWindow avec un argument à NULL	Une exception IllegalArgumentException est levée		OK
5	Utiliser la fonction addMainWindow			OK
6	Utiliser la fonction run			OK
7	Utiliser la fonction alreadyRun	La fonction renvoie true		OK
8	Utiliser la fonction profileIsLoad avec le même Profile que pour addMainWindow	La fonction ne renvoie pas NULL		OK
9	Utiliser la fonction unloadProfileWithWindow avec le même Profile que pour addMainWindow			OK
10	Utiliser la fonction stop			OK
11	Utiliser la fonction alreadyRun	La fonction renvoie false		OK

Objet testé : Classe Action			Version : 0.5	
Objectif de test : Test de la classe				
Procédure n° U7			Pré-requis :	
N°	Actions	Résultats attendus	Exigence	OK/KO
1	Créer une instance de la classe			OK
2	Utiliser la fonction getCmd	La fonction renvoie la commande donnée au constructeur		OK
3	Utiliser la fonction getOptions	La fonction renvoie les options données au constructeur		OK
4	Utiliser la fonction getInteractions	La fonction renvoie les interactions donnés au constructeur		OK
5	Utiliser la fonction getArgs	La fonction renvoie les arguments donnés au constructeur		OK
6	Utiliser la fonction hasInteractions	La fonction renvoie true si des interactions ont été données au constructeur, false sinon		OK
7	Utiliser la fonction nextInteraction autant de fois que le nombre d'interactions données au constructeur plus une	La fonction renvoie un QString contenant une interaction à chaque appel à la fonction, et NULL au dernier		OK
8				
9				
10				

Objet testé : Classe GPGManager			Version : 0.5	
Objectif de test : Test de la classe				
Procédure n° U8			Pré-requis :	
N°	Actions	Résultats attendus	Exigence	OK/KO
1	Créer une instance de la classe			OK
2	Utiliser la fonction setAction			OK
3	Utiliser la fonction execute			OK
4	Attendre le signal finished	Le signal finished est reçu		OK
5	Utiliser la fonction getOutput	La fonction renvoie le texte renvoyé par gpg quand gpg exécute l'action donnée en argument à setAction		OK
6				
7				
8				
9				
10				

Objet testé : Classe MainWindowModel			Version : 0.5	
Objectif de test : Test de la classe				
Procédure n° U10			Pré-requis :	
N°	Actions	Résultats attendus	Exigence	OK/KO
1	Créer une instance de la classe			OK
2	Utiliser la fonction getLauncher	La fonction renvoie le Launcher donné au constructeur		OK
3	Utiliser la fonction getGuiPGApp	La fonction renvoie le GuiPGApp donné au constructeur		OK
4	Utiliser la fonction getConf	La fonction renvoie la Configuration donnée au constructeur		OK
5	Utiliser la fonction getProfile	La fonction renvoie le Profile donné au constructeur		OK
6	Utiliser la fonction loadProfile			OK
7	Utiliser la fonction initKeyManager			OK
8	Utiliser la fonction getKeyManager	La fonction renvoie un KeyManager		OK
9				
10				

Objet testé : Classe KeyExport			Version : 0.5	
Objectif de test : Test de la classe				
Procédure n° U11			Pré-requis :	
N°	Actions	Résultats attendus	Exigence	OK/KO
1	Avoir un trousseau de clés non vide			OK
2	Créer une instance de la classe			OK
3	Utiliser la fonction exportKey sur /tmp	Le fichier correspondant au path a été créé.		OK
4				
5				
6				
7				
8				
9				
10				

8.3 Tests d'intégration

Objet testé : Interface graphique			Version : 0.3	
Objectif de test : Tester la sélection du profil				
Procédure n° I1			Pré-requis :	
N°	Actions	Résultats attendus	Exigence	OK/KO
1	Lancer l'interface graphique en ligne de commande avec l'option -P	On peut choisir son profil, et en créer un si besoin		OK
2	Choisir un profil	L'interface graphique s'affiche		OK
3	Cliquer sur le bouton "Changer de profil"	On peut changer, créer, modifier ou supprimer un profil		OK
4				
5				
6				
7				
8				
9				
10				

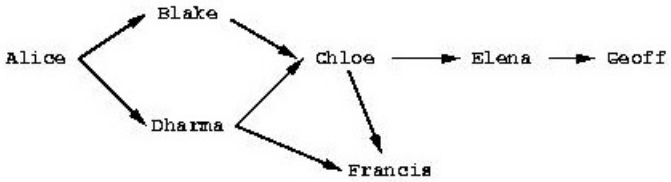
Objet testé : Interface graphique			Version : 0.3	
Objectif de test : Tester le mécanisme multi-instance				
Procédure n° I2			Pré-requis :	
N°	Actions	Résultats attendus	Exigence	OK/KO
1	Lancer GuiPG	GuiPG se lance		OK
2	Lancer un autre GuiPG	La fenêtre de sélection de profil apparait		OK
3	Sélectionner un profil non utilisé	Il y a deux fenêtres GuiPG ouvertes, chacune avec un profil différent		OK
4	Lancer un autre GuiPG	La fenêtre de sélection de profil apparait	OK	
5	Sélectionner un profil utilisé	La fenêtre GuiPG utilisant ce profil apparaît en premier plan	OK	
6				
7				
8				
9				
10				

Objet testé : Interface Graphique			Version : 0.1	
Objectif de test : Création de clé				
Procédure n° I3			Pré-requis :	
N°	Actions	Résultats attendus	Exigence	OK/KO
1	Créer une clé	La clé a été créée		NOK
2				
3				
4				
5				
6				
7				
8				
9				
10				

Objet testé : Classe KeyManager			Version : 0.5	
Objectif de test : Test d'un bug rencontré lors du changement rapide de profils				
Procédure n° NR2			Pré-requis :	
N°	Actions	Résultats attendus	Exigence	OK/KO
1	Lancer GuiPG avec un profil	GuiPG se lance		OK
2	Créer un nouveau profil	Le nouveau profil a été créé.		OK
3	Changer 20 fois entre le nouveau profil et l'ancien	GuiPG ne crashe pas		OK
4				
5				
6				
7				
8				
9				
10				

9 Jeux de données de test

Les clés qui seront utilisées pour les tests se trouvent dans PrivateKeys.gpg et PublicKeys.gpg, joints dans le même dossier que ce fichier. Elles suivent le schéma de signature décrit ci-dessous :



trust		validity	
marginal	**full**	**marginal**	**full**
	Dharma		Blake, Chloe, Dharma, Francis
Blake, Dharma		Francis	Blake, Chloe, Dharma
Chloe, Dharma		Chloe, Francis	Blake, Dharma
Blake, Chloe, Dharma		Elena	Blake, Chloe, Dharma, Francis
	Blake, Chloe, Elena		Blake, Chloe, Elena, Francis

10 Couverture de test

Id Exigence STB	Méthode de vérification	Procédure(s) utilisée(s)	Commentaire
EF_01	Tests unitaires et d'intégration	U3, U5, I1	
EF_02	Tests unitaires et d'intégration	U3, U5, I1	
EF_03	Tests unitaires et d'intégration	U3, U5, I1	
EF_04	Tests unitaires, d'intégration et de non-régression	U3, U5, I1, NR2	
EF_05	Test d'intégration	I1	
EF_06	Test d'intégration	I2	
EF_07	Test d'intégration	I6	
EF_08	Test d'intégration	I3	
EF_09	Test unitaire	U12	
EF_10	Test unitaire et de non-régression	NR1, U11	
EF_11	Non implémenté - Pas de test		
EF_12	Non implémenté - Pas de test		
EF_13	Non implémenté - Pas de test		
EF_14	Non implémenté - Pas de test		
EF_15	Non implémenté - Pas de test		
EO_06	Non implémenté - Pas de test		
EO_07	Test d'intégration	I2	
EO_11	Non implémenté - Pas de test		
EQ_01	Test d'intégration	I6	
EQ_04	Test d'intégration	I8	

Rappel des codes utilisés dans la STB :

Id	Intitulé	Acteur(s)	Priorité
EF_1	Créer un profil utilisateur	Utilisateur	Important
EF_2	Éditer un profil utilisateur	Utilisateur	Important
EF_3	Supprimer un profil utilisateur	Utilisateur	Important
EF_4	Charger un profil utilisateur	Utilisateur	Important
EF_5	Choisir un profil par défaut	Utilisateur	Important
EF_6	Gérer simultanément plusieurs trousseaux de clefs	Utilisateur	Indispensable
EF_7	Lister le contenu d'un trousseau de clefs	Utilisateur	Indispensable
EF_8	Générer une paire de clefs	Utilisateur	Indispensable
EF_9	Importer une ou plusieurs clefs depuis un fichier	Utilisateur	Indispensable
EF_10	Exporter une ou toutes les clefs vers un fichier	Utilisateur	Indispensable
EF_11	Signer un ou plusieurs identifiants utilisateur	Utilisateur	Indispensable
EF_12	Modifier la confiance d'une clef principale	Utilisateur	Indispensable
EF_13	Ajouter une sous clef à une clef principale	Utilisateur	Important
EF_14	Ajouter un uid à une clef principale	Utilisateur	Important
EF_15	Afficher les commandes exécutées, les retours et les erreurs de GPG.	Utilisateur	Indispensable

Id	Intitulé	Priorité
EO_1	Compatibilité GnuPG 2.0.*	Indispensable
EO_2	Compatibilité GnuPG 1.4.*	Important
EO_3	Compatibilité GnuPG 2.1.0	Secondaire
EO_4	Fonctionnement sous KDE 4.* et 5	Indispensable
EO_5	Fonctionnement sous Gnome 3.*	Important
EO_6	Visualisation de la Toile de confiance (non sous forme de graphe)	Indispensable
EO_7	Ouverture de deux interfaces graphiques avec des profils différents simultanément	Important
EO_8	Connexion SSH : Authentification via une clef signée par GPG	Secondaire
EO_9	Recherche des clefs en local	Important
EO_10	Choix du logiciel comme logiciel par défaut pour PGP	Secondaire
EO_11	Choix des couleurs pour les niveaux de confiance et de validité	Secondaire

Id	Intitulé	Priorité
EQ_1	Représentation des niveaux de confiance et de validité par couleurs	Indispensable
EQ_2	Temps d'exécution pour l'interface : $<$ temps d'exécution GPG $+$ 100 ms	Important
EQ_3	Interface livrée sous forme de paquet Ubuntu	Important
EQ_4	Les pass phrases ne sont jamais affichées en clair	Indispensable

11 Suivi des bugs

Date de détection	Date de correction	Description	Test NR associé
02/03/2015	28/03/2015	La fonction d'exportation ne faisait rien	NR1
02/04/2015	09/04/2015	Changer de profil rapidement faisait crasher GuiPG	NR2