AmarOk

CC

Geste de savoir

Oh ! Ma push notification s'est perdue

21 janvier 2019

Table des matières

1. Introduction 1
2. Cycle de vie simplifiée d'une notification 2
3. Problèmes 2
   - 3.1. Problème 1 : Une nuit trop sombre 2
   - 3.2. Problème 2 : Une nuit encore plus sombre 2
   - 3.3. Problème 3 : Une mer dechainée 3
   - 3.4. Problème 4 : La notification meurt en mer 3
   - 3.5. Problème 5 : Le capitaine ignore la notification 3
   - 3.6. Problème 6 : La mer est inconnue et dangereuse 3
4. Conclusion 3
5. Morale de l'histoire 3
   - 5.1. Quelques liens 3

% OH ! MA PUSH NOTIFICATION S&#39 ;EST PERDUE % AmarOk % 07 juillet 2018

1. Introduction

Je ne suis pas développeur Android et je ne connais pas bien cet environnement. Cependant, dans le cadre d'OpenDHT , j'ai dû m'intéresser un peu aux push notifications.

Pour faire court, cette librairie sert à stocker une table de hashage sur de multiples noeuds (peut-être que je ferais un article plus detaillé, ou un post sur le forum). Chaque noeud du réseau peut écrire (**PUT**) des valeurs (chiffrées pour un autre noeud ou en clair), lire les valeurs contenues à un hash précis (**GET**) ou lire en continu (**LISTEN**) les valeurs d'un hash. Le problème, c'est que le maintien d'un réseau distribué demande de la synchronisation entre ces noeuds (donc un envoi assez conséquent de données) ainsi qu'une connexion à ce réseau. Ces deux points ne sont pas assurés sur des appareils mobiles, qui demandent géneralement de limiter les datas consommées (sauf en Europe...) et qui pousse l'économie de batterie en coupant la connectivité ou même les applications (donc, quand le téléphone passe en veille, le noeud se trouve inactif). Pour le premier point, un noeud peut demander à un autre noeud du réseau de lui servir de proxy et d'écouter le réseau à sa place (ce qui évite de se synchroniser, tout en étant le seul à pouvoir déchiffrer le contenu qui lui est destiné). Pour la coupure de la connectivité, la solution la plus fiable et simple pour l'utilisateur est l'utilisation de push notifications. Ainsi, le proxy peut envoyer une *wake up* notification pour donner un peu de temps au noeud de récuperer via le proxy les valeurs à déchiffrer.

Bref, *seems simple...* et après quelques semaines de dev, tout marche relativement bien. Vient la partie debug, qui, au final, va nous faire découvrir toute une épopée pour certaines de ces notifications.

2. Cycle de vie simplifiée d'une notification

Tout d'abord, voici l'aventure normale d'une push notification telle que racontée dans les contes pour enfants :

1. Mon noeud sur le bateau du robot (Android) ou de la pomme (Apple) demande via la radio à un autre noeud sur la terre d'écouter le monde à sa place.
2. La nuit tombe sur le bateau, après quelques minutes la salle de la radio est fermée et le noeud sur le bateau ne peut pas communiquer avec la terre.
3. Le noeud sur terre reçoit un message urgent à livrer au noeud du bateau, mais ne peut plus lui communiquer par radio. Il demande donc à un service du robot ou de la pomme d'envoyer une super push notification d'aller réveiller le noeud sur le bateau.
4. La push notification voyage très vite jusqu'au bateau et réveille le capitaine.
5. Le capitaine va alors réveiller le noeud et lui donne accés a la radio pour quelques minutes
6. Le noeud du bateau peut demander le message pour lui au noeud sur la terre.

Sauf que dans la vraie vie, l'histoire peut se finir mal.

3. Problèmes

3.1. Problème 1 : Une nuit trop sombre

Sur certains téléphones Android, l'optimisation de la batterie est un peu trop forte. Certains constructeurs (ex, Oppo, Xiami, One plus) font en sorte que lorsque Android tue une application pour optimiser la batterie, le système tue aussi les taches de fonds reliées à ce processus, dont celles utilisées par Firebase pour que la push notification arrive. Dans ce cas... la push notification arrive sur le bateau mais le capitaine n'arrive pas à réveiller le noeud à temps, tout le monde meurt... Il est aussi nécessaire de noter que ce problème arrive relativement peu pour les applications connues. Mais une whitelist de certaines applications populaires est possible de la part des constructeurs.

Solution : sur certaines de ces ROMs, il existe une option pour pouvoir réveiller les taches de fonds, mais ça demande du travail manuel à l'utilisateur... Pas viable et souvent l'option est difficile à trouver.

3.2. Problème 2 : Une nuit encore plus sombre

Même s'il est possible de pouvoir désactiver l'optimisation batterie pour certaines applications depuis certaines ROM, parfois celle-ci est tellement forte que le téléphone lui-même n'a pas l'air de recevoir une seule notification (vu sur quelques Huawei ou Xiaomi). Je n'ai pas de solution viable à part laisser le téléphone branche. Ce scénario résulte dans le fait que la push notification n'arrive même pas sur le bateau mais s'explose dessus, tout le monde meurt.

3.3. Problème 3 : Une mer dechainée

Les téléphones doivent maintenir une socket avec le service du robot ou de la pomme. Pour maintenir cette connexion, elle est utilisée pour transmettre un petit message (*heartbeet*) toutes les X minutes (généralement entre 15 et 30 minutes). Si ce message n'est pas reçu (délais, téléphone sans connexion, etc.), alors le téléphone n'est plus connecté et ne va pas recevoir la notification. Dans ce scénario, le bateau est juste perdu en mer et la notification ne trouve jamais le bateau, tout le monde meurt.

3.4. Problème 4 : La notification meurt en mer

Dans certains cas, la notification peut mourrir à cause d'un TTL trop faible donne par FCM (Firebase, service d'Android). Le bateau n'est jamais prévenu, tout le monde meurt.

3.5. Problème 5 : Le capitaine ignore la notification

Parfois le téléphone s'enregistre auprès du service relié. Pendant cette période, le téléphone peut recevoir une notification sans être prêt et ne va pas réveiller l'application destinataire. Dans ce cas, le capitaine se reccouche, tout le monde meurt.

3.6. Problème 6 : La mer est inconnue et dangereuse

Ce problème regroupe tous les autres possibles, une règle de firewall, un drop de paquet, etc. Dans ce scénario, la push notification est enlevée par des pirates, revendue comme esclave et tout le monde remeurt.

4. Conclusion

5. Morale de l'histoire

Il fait pas bon de naviguer en mer. Le réseau, c'est parfois complexe et pas marrant à debugger.

5.1. Quelques liens

- https ://medium.freecodecamp.org/why-your-push-notifications-never-see-the-light-of-day-3fa297520793
- https ://www.moengage.com/blog/tagpush-notification-not-getting-delivered/ (interessant +)
- https ://productforums.google.com/forum/# !topic/nexus/fslYqYrULto%5B1-25%5D (interessant ++)
- https ://github.com/firebase/quickstart-android/issues/41
- https ://github.com/wix/react-native-notifications/issues/42