mickael.hoerdt@hesge.ch 2017/2018

Mini-projet de conception de protocoles réseaux informatiques

Un client de sélection de données Syncthing en lecture seule

Syncthing[1] est un logiciel qui implémente un service de synchronisation de répertoires entre plusieurs hôtes distant via le protocole pair à pair Block Exchange Protocol 1.0 (BEP [2]). Chaque participant est à la fois client et serveur et modifie un répertoire local en y créant/modifiant/supprimant des fichiers. Ce répertoire local est ensuite synchronisé avec les autres participants du partage via le protocole BEP. Une fois la phase de synchronisation entre les pairs terminée, le contenu d'un répertoire partagé entre chaque participant est répliqué localement chez chaque participant. Le but de ce mini-projet sera d'implémenter en python un client Syncthing simplifié permettant au niveau applicatif (c'est à dire sans modifier le protocole BEP), de sélectionner un sous-ensemble des données à répliquer vers le répertoire partagé local à un pair.

Les simplifications de votre implémentation seront les suivantes :

- Le client sera en lecture seule. Il ne permet que de consulter le contenu de l'ensemble ou d'un sous-ensemble d'un répertoire partagé mais pas de le modifier.

- La synchronisation pourra se limiter à récupérer les données depuis un seul nœud Syncthing. Il n'est pas nécessaire d'implémenter une synchronisation tirant partie de l'existence de plusieurs participants à un répertoire partagé.

- On supposera que les répertoires à Synchroniser est vide à chaque fois qu'on relancera votre application, le client n'aura pas à gérer les mise à jour partielles d'un répertoire partagé.

- La découverte ainsi que l'inscription d'un participant à un partage Syncthing se fera manuellement. Votre implémentation n'aura **pas** à gérer la notification d'un nouveau participant ou la découvert de participants via le protocol Global Discovery ou Local Discovery spécifié dans la documentation de Syncthing.

Le travail sera divisé en deux étapes, décrites ci-après.

1. Modélisation et analyse du protocole BEP 1.0

L'ensemble de la dernière version du protocole BEP sera analysé en détail, c'est à dire ses messages, leur sémantique et l'ordre de leur l'échange entre les participants. A partir de l'analyse, une modélisation du fonctionnement devra être produite en utilisant les méthodes vues en cours : machines à états associées à la définition des primitives du protocole, ainsi que les conditions et les actions qui relient les états du protocole..

La modélisation devra spécifier le fonctionnement et le déroulement du service offert par le protocole du coté client en le limitant volontairement à un fonctionnement client c'est à dire sans possibilité d'envoyer des données. Une machine a état devra spécifier le fonctionnement du service

1 https://synthing.net
2 https://docs.syncthing.net/specs/bep-v1.html

et du protocole coté client,

Les différents cas d'erreurs pouvant se produire devront être déroulés sur des diagramme temps séquence et démontrer que le protocole récupère bien l'erreur dans les cas présentés, sur la base des machines à état proposées. Un diagramme temps séquence ne devra dérouler qu'un seul cas d'erreur.

La modélisation du protocole devra aussi contenir une première version du diagramme de classes représentant les différents objets qui serviront à implémenter le protocole, ses messages et son application.

2. Implémentation du protocole BEP 1.0. et de l'application cliente Syncthing simplifiée

L'implémentation devra utiliser le langage python 2 ou 3, les sockets SSL BSD[3] TCP. La sérialisation des données devra utiliser protobuf[4]. Les messages Syncthing compressés en LZ4 devront être supportés à partir d'une librairie python tierce.

L'interface utilisateur de votre client Syncthing deva se présenter sous la forme d'une ligne de commande dont la spécification est libre.

Modalité de rendu

L'ensemble du projet devra être réalisé en priorité en binôme. L'ensemble des sources du projet ainsi que ses sources devront être stockés sur https://githepia.hesge.ch. Vous devrez me communiquer l'adresse du dépôt git de votre projet avant le 1/11/2017.

L'évaluation de l'étape 1 compte pour 40 % de la note, l'étape 2 pour 60%.

Dans le cas où une personne se retrouverait seule, elle doit effectuer le projet en trinôme et dans ce cas votre client syncthing doit alors aussi être capable de synchroniser des données en écriture sur un hôte distant.

- L'étape 1 devra être rendu sous la forme d'un document au format pdf déposé sur le dépôt git de votre projet. Ce document devra contenir la modélisation et l'analyse telle que décrite au point 1. Elle devra être rendue avant le **1/12/2017 23h59.**

- L'étape 2 devra être rendu avant le **19/1/2018** sur le dépôt git de votre projet. Un fichier README.md présent sur votre dépôt git contiendra des captures de la sortie de votre programme ainsi que des exemples d'exécution et de test de votre implémentation.

3 https://en.wikipedia.org/wiki/Berkeley_sockets
4 https://developers.google.com/protocol-buffers/