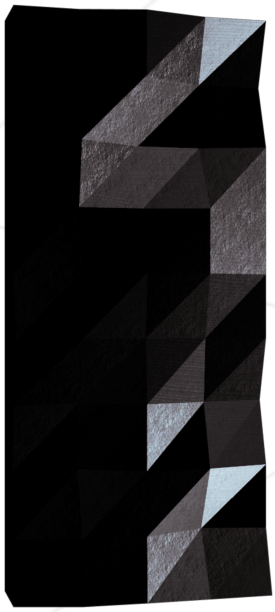

Project UNIX

Matt_daemon

42 Staff pedago@staff.42.fr

Résumé: Ce projet consiste à coder un daemon.

Table des matières

Chapitre I

Préambule

Chapitre II

Introduction

Un daemon, qui ne se traduit pas par démon, désigne un type de programme informatique, un processus ou un ensemble de processus qui s'exécute en arrière-plan plutôt que sous le contrôle direct d'un utilisateur.

Le terme daemon semble avoir été introduit, en 1963, par les concepteurs de CTSS du MIT, en réponse au « dragon », terme employé par les concepteurs d'ITS. Le rétro-acronyme Disk And Execution MONitor (moniteur de disque et d'exécution) a été inventé pour justifier le terme daemon après sa démocratisation.

Les daemons sont souvent démarrés lors du chargement du système d'exploitation, et servent en général à répondre à des requêtes du réseau, à l'activité du matériel ou à d'autres programmes en exécutant certaines tâches. Sous Microsoft Windows, ces fonctions sont exécutées par des programmes appelés « services ».

Chapitre III

Objectifs

Ce projet constitue une introduction au concept de daemon.

Vous allez donc devoir coder un petit programme -un deamon- ayant pour fonction de traiter et stocker les messages reçus sur un port spécifique, ouvert par ce daemon justement.

Chapitre IV

Partie obligatoire

Usage :

- L'exécutable devra se nommer `Matt_daemon`.
- Le programme va se lancer uniquement avec les droits root.
- Votre programme devra s'exécuter en tâche de fond à la façon d'un vrai daemon.
- Le daemon va devoir écouter sur le port 4242.
- Pour la journalisation de votre daemon, il sera impératif de créer une classe appelée `Tintin_reporter` (celle-ci pourra vous re-servir dans de futurs projets).
- Tout ce que le daemon fait doit être visible dans un fichier de log `matt_daemon.log` avec timestamp (sous la forme [DD / MM / YYYY - HH : MM : SS]) situé dans le dossier /var/log/matt_daemon/.
- Vous avez la possibilité de créer plusieurs fichiers de logs si vous le souhaitez.
- Une seule instance du daemon doit pouvoir être lancée.
- Lors de la tentative de lancement d'un second daemon alors qu'une instance de celui-ci est déjà en cours, un message d'erreur indiquant une tentative de création/ouverture de fichier sur `matt_daemon.lock` doit être visible.
- Un fichier `matt_daemon.lock` doit être créé dans /var/lock/ au lancement du daemon.
- À la fermeture du daemon le fichier matt_daemon.lock doit être effacé.
- La fermeture du programme doit se faire par l'envoi d'une simple chaîne de caractère "quit" sur le socket ouvert.
- Toute autre chaîne de caractère doit être inscrite dans le fichier de log.

- Seuls 3 clients peuvent se connecter en simultané sur le daemon.
- Lorsque le daemon reçoit un signal, il doit l'intercepter et l'inscrire dans le fichier `matt_daemon.log` avec un message explicite, puis quitter proprement.

`Les signaux ne pouvant pas être reçus ne sont pas à gérer bien entendu. Dans ce cas une explication sera nécessaire durant votre soutenance.`

- Voici un exemple de log possible :

```
# ./Matt_daemon
# ls -ali /var/lock/ | grep matt
34957 -rw-r--r-- 1 root root  0 Jan 11 14:34 matt_daemon.lock
# ps aux | grep Matt
root     6498 99.7 0.0  15172  2164 ?      Rs   14:34 43:44 ./Matt_daemon
# kill -15 6498
# ls -ali /var/lock/ | grep matt
# tail -n 7 /var/log/matt_daemon/matt_daemon.log
[11/01/2016-14:34:58] [ INFO ] - Matt_daemon: Started.
[11/01/2016-14:34:58] [ INFO ] - Matt_daemon: Creating server.
[11/01/2016-14:34:58] [ INFO ] - Matt_daemon: Server created.
[11/01/2016-14:34:58] [ INFO ] - Matt_daemon: Entering Daemon mode.
[11/01/2016-14:34:58] [ INFO ] - Matt_daemon: started. PID: 6498.
[11/01/2016-14:35:24] [ INFO ] - Matt_daemon: Signal handler.
[11/01/2016-14:35:24] [ INFO ] - Matt_daemon: Quitting.
# ./Matt_daemon
# ./Matt_daemon
Can't open :/var/lock/matt_daemon.lock
# ps aux | grep Matt
root     8062 98.7 0.0  15172  2364 ?      Rs   15:00  0:46 ./Matt_daemon
# tail -n 8 /var/log/matt_daemon/matt_daemon.log
[11/01/2016-15:00:25] [ INFO ] - Matt_daemon: Started.
[11/01/2016-15:00:25] [ INFO ] - Matt_daemon: Creating server.
[11/01/2016-15:00:25] [ INFO ] - Matt_daemon: Server created.
[11/01/2016-15:00:25] [ INFO ] - Matt_daemon: Entering Daemon mode.
[11/01/2016-15:00:25] [ INFO ] - Matt_daemon: started. PID: 8062.
[11/01/2016-15:00:26] [ INFO ] - Matt_daemon: Started.
[11/01/2016-15:00:26] [ ERROR ] - Matt_daemon: Error file locked.
[11/01/2016-15:00:26] [ INFO ] - Matt_daemon: Quitting.
```

- Voici un exemple d'utilisation possible :

```
# ./Matt_daemon
# ps aux | grep Matt
root     3328 98.1 0.0  15168   208 ?      Rs   16:36  0:06 ./Matt_daemon
# nc localhost 4242
Salut
xd
quit
# tail -n 9 /var/log/matt_daemon/matt_daemon.log
[17/01/2016-16:36:07] [ INFO ] - Matt_daemon: Started.
[17/01/2016-16:36:07] [ INFO ] - Matt_daemon: Creating server.
[17/01/2016-16:36:07] [ INFO ] - Matt_daemon: Server created.
[17/01/2016-16:36:07] [ INFO ] - Matt_daemon: Entering Daemon mode.
[17/01/2016-16:36:07] [ INFO ] - Matt_daemon: started. PID: 3328.
[17/01/2016-16:36:43] [ LOG ] - Matt_daemon: User input: Salut
[17/01/2016-16:36:44] [ LOG ] - Matt_daemon: User input: xd
[17/01/2016-16:36:47] [ INFO ] - Matt_daemon: Request quit.
[17/01/2016-16:36:47] [ INFO ] - Matt_daemon: Quitting.
```

Chapitre V

Partie bonus

Les bonus ne seront comptabilisés que si votre partie obligatoire est PARFAITE. Par PARFAITE, on entend bien évidemment qu'elle est entièrement réalisée, et qu'il n'est pas possible de mettre son comportement en défaut, même en cas d'erreur aussi vicieuse soit-elle, de mauvaise utilisation, etc ... Concrètement, cela signifie que si votre partie obligatoire n'est pas validée, vos bonus seront intégralement IGNORÉS.

Des idées de bonus :

- Créer un client graphique pour interagir avec le daemon .
- Ajouter des fonctions utilitaires à votre daemon (création d'un remote shell par exemple !).
- Chiffrer l'envoi et la réception des données (implique un client, logiquement).
- Archivage avancé des logs.
- Envoi de mail suivant des règles de filtrages choisis.
- Créer un système d'authentification pour se connecter au daemon (via client graphique/remote shell).
- Utilisation de systèmes de chiffrement avancé (clé publique/privée).

Dans le cas de la création d'un client, le nom du binaire créé devra impérativement être Ben_AFK.

Chapitre VI

Rendu et peer-évaluation

- Ce projet ne sera corrigé que par des humains. Vous êtes donc libres d'organiser et nommer vos fichiers comme vous le désirez, en respectant néanmoins les contraintes listées ici.
- Vous devez coder en C++ (toutes versions confondues) et rendre un Makefile (respectant les règles habituelles).
- Vos classes doivent respecter la forme de Coplien.
- Vous devez gérer les erreurs de façon raisonnée. En aucun cas votre programme ne doit quitter de façon inattendue (Segmentation fault, etc).
- Rendez-votre travail sur votre dépot `GiT` comme d'habitude. Seul le travail présent sur votre dépot sera évalué en soutenance.
- Vous devez être sous une VM avec un noyau Linux > 3.14. Pour info, le barême a été fait avec une Debian 7.0 stable.
- Vous être libre d'utiliser ce dont vous avez besoin, dans la limite des bibliothèques faisant le travail pour vous, ce qui correspondrait à un cas de triche.
- Vous pouvez poser vos questions sur le forum, sur jabber, IRC, slack...

`Bien entendu la fonction daemon est interdite et l'utilisation de fork, chdir, flock (pas de LOCK_SH), ainsi que signal sera vérifiée durant la correction.`