S. Lavirotte
Polytech'Nice – Sophia
SI4
2018-2019

TD n° 05
Optimisation d'un Système Embarqué

Le but de ce TD est de mettre en œuvre quelques techniques pour l'optimisation d'un système. Le système que vous devrez optimiser sera basé sur une machine de type Pentium avec 64Mo de RAM.
Vous ferez un compte-rendu des choix et des actions effectuées qui vous ont conduits à un système plus performant.

1 Optimisation de la configuration du noyau

Pour ce TD, nous allons réutiliser le contenu de l'image disque `sdb-linux-kernel` que vous monterez dans `/work/td05`. Cette image contient le noyau `2.6.32-68` que vous avez compilé ainsi que l'image d'une machine virtuelle que nous allons tenter d'optimiser.

1.1 Mettre en place les mesures sur le noyau

Pour pouvoir mesurer les gains que nous pourrons obtenir sur le temps de chargement du noyau, il est nécessaire d'instrumenter celui-ci. Vous pourrez partir de la configuration que vous avez réalisée lors du `TD02` ou bien repartir de la configuration donnée dans la correction du `TD02`.

Vous veillerez à bien faire le ménage complet des sources de votre noyau `2.6.32-68` si vous l'aviez déjà compilé. Pour utiliser cette configuration du noyau, vous copierez ce fichier dans les sources du noyau sous le nom `.config` et vous exécuterez la commande suivante afin de vérifier la validité du fichier de configuration :

```
make mrproper
cp config-2.6.32.68 .config
make oldconfig
```

1.2 Evaluation du temps de démarrage d'un noyau

Vous commencerez donc par configurer votre noyau `2.6.32-68` afin d'avoir accès aux mesures temporelles lors du chargement de celui-ci. Vous n'oublierez d'inclure ces fonctionnalités dans le noyau et pas en tant que module.

Après compilation et génération du noyau, vous démarrerez le système avec votre noyau à l'aide de `qemu`.

```
qemu-system-i386 -m 64 -kernel arch/x86/boot/bzImage -hda hda-bench.qcow2 -append
"root=/dev/hda1"          –device          ne2k_pci,netdev=mynet0          –netdev
user,id=mynet0,net=192.168.10.0/24,dhcpstart=192.168.10.10 –redir tcp:5555::22
```

Vous ferez une première sauvegarde des mesures réalisées du démarrage de ce système dans `output1.svg`. Pour cela, vous utiliserez le script `bootchart.pl` se trouvant dans `/work/scripts` de la machine testée (ce script se trouve normalement dans les sources du noyau). Vous veillerez à noter les conditions d'expérimentation, afin de les reproduire pour les futurs tests.

Vous utiliserez la commande `scp` depuis votre machine virtuelle de travail pour vous connecter par `ssh` à votre machine virtuelle `qemu` et récupérer le fichier `svg` via la commande:

```
scp -P 5555 root@localhost:/work/scripts/output1.svg .
```

Ce graphique vous donnera ainsi le temps de démarrage de référence (celui donc à optimiser) pour la configuration du noyau que nous avons. Pour visualisé celui-ci vous pourrez le copier sur votre machine physique via le partage de dossier VirtualBox (`/media/sf_...`).

2 Optimisation de la configuration du noyau

Après la visualisation du résultat obtenu dans `output1.svg`, vous devez constater qu'il est important d'optimiser certaines fonctionnalités très gourmandes en tant d'exécution lors de l'initialisation du noyau. Commencez donc par chercher comment optimiser celles-ci.

TD n° 05
Optimisation d'un Système Embarqué

2.1 Correction des problèmes les plus évidents

Vous pourrez consulter l'adresse http://elinux.org/Boot_Time pour y trouver des informations sur les actions à potentiellement entreprendre. Mais cette documentation ne correspond pas forcément avec votre version de noyau. Vous confronterez donc les informations obtenues sur ce site avec la documentation de votre version du noyau (`/work/td05/linux…/Documentation`) pour appliquer les bons correctifs.

Appliquez le correctif pour gagner sur le temps de boot du noyau. Vous vérifierez le temps gagné et stockerez votre résultat dans le fichier `output2.svg`.

2.2 Optimisations générales pour la vitesse

Dans un deuxième temps, vous tenterez d'obtenir un gain supplémentaire quant à la vitesse de démarrage de votre noyau. Vous veillerez tout particulièrement aux options à activer, à désactiver ou bien encore à régler (changement de valeurs) pour optimiser votre système en termes de vitesse de démarrage. Vous lirez la documentation des options afin de comprendre leur impact sur le temps de boot du noyau et son impact mémoire.

Voici plusieurs pistes pour l'amélioration de votre système :

- Activation de l'option `quiet` dans les paramètres de démarrage du noyau
- Activation de fonctionnalités dans le noyau : voir la section Processor type and feature

Vous comparerez ce nouveau noyau produit en termes de vitesse de boot toujours à l'aide du script `bootchart.pl`. Vous veillerez bien à ce que les conditions de benchmarking soient identiques d'un test à l'autre afin de ne pas influencer sur les mesures effectuées. Vous produirez un fichier `output3.svg` (pour l'option `quiet`) et un fichier `output4.svg` pour les fonctionnalités et les paramétrages activées dans le noyau.

2.3 Optimisations pour la taille du noyau

Il n'est pas tout d'avoir un noyau performant, il faut aussi avoir un noyau incluant toutes les fonctionnalités nécessaires et le plus compact possible. Nous allons travailler ce deuxième critère.

Pour améliorer la taille du noyau (le fichier), vous devez veillez à activer des options dans différentes catégories. Voici les principales. Ceci ne nous empêche pas de lire la documentation d'autres options et d'avoir des gains supplémentaires.

```
General Setup
     - Kernel Compression mode = LZMA
     - Kernel log buffer = 14
     - Optimize for size
     - Configure standard kernel features (for small systems)
Processor type and feature
     - High Memory Support (off)
Kernel Hacking:
 - Strip assembleur-generated symbols during link
```

Vous comparerez ainsi la taille du noyau avant et après l'activation de ces options.

3 Optimisation du temps de démarrage des services

A l'aide de l'application `bootchart` présente sur l'image `hda-bench.qcow2`, vous analyserez le temps nécessaire à l'initialisation des services sur votre système.

Pour réaliser ces mesures, vous ajouterez l'option suivante pour le démarrage de votre noyau : `-append "… init=/sbin/bootchartd"`. Pour visualiser les graphiques, il vous suffit de lancer `bootchart` juste après vous être

connectés au système. Ceci génèrera un fichier `svg` compressé incluant les graphiques sur les temps de démarrage des scripts de votre système.

Que pouvez-vous conclure à la visualisation des résultats obtenus par `bootchart` ? Est-ce que le système utilise déjà un lancement en parallèle des différents services ? Si c'est le cas, que pouvez proposer pour gagner du temps sur le démarrage du système ?

4 Optimisation de la consommation énergétique

Il est aussi possible de configurer votre système pour que celui-ci consomme moins d'énergie. Vous pouvez ainsi autoriser le changement de fréquence du cpu et aussi autoriser les modes de standby ou d'hivernation. Activez ces options dans le noyau.

Vous utiliserez `powertop` sur votre système afin de suivre ses recommandations en termes de configuration de votre noyau et de votre système pour le rendre moins énergivore possible. Pour l'utiliser, il faudra activer :

- Kernel Hacking
  - Kernel Debugging
    - Collect Kernel times statistics

Quel problème rencontrez-vous pour continuer à optimiser votre système ?