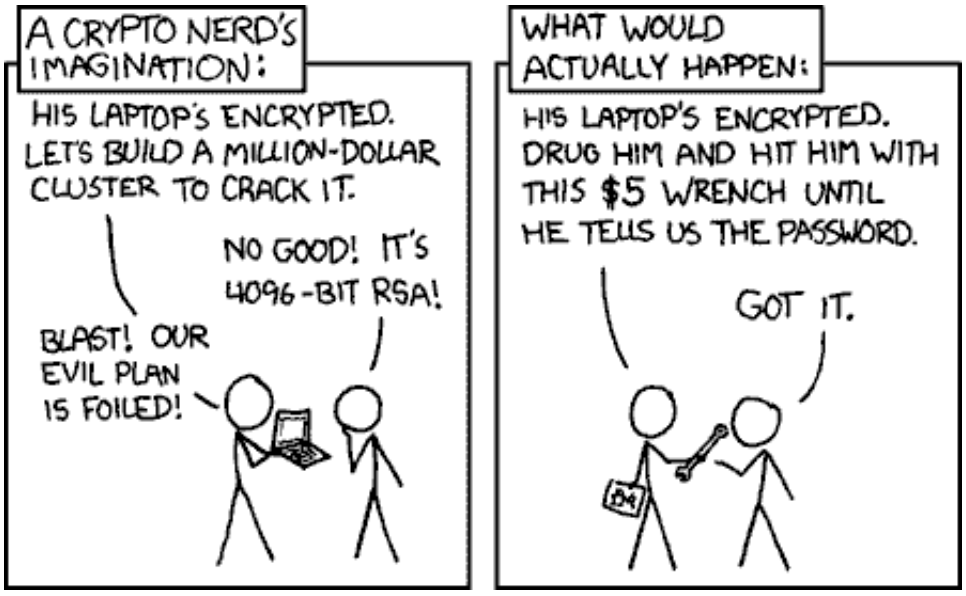
Programmation concurrente
Password Cracker

Claudio Sousa - David Gonzalez

22 octobre 2016



1 Introduction

1.1 Division des tâches

Chaque membre du group a fait seul le programme dans son entièreté. Le travail rendu est la mise en commun du code des differents programmes.

1.2 Machine cible

lscpu:

```
Architecture:          x86_64
CPU op-mode(s):        32-bit, 64-bit
Byte Order:            Little Endian
CPU(s):                8
On-line CPU(s) list:   0-7
Thread(s) per core:    2
Core(s) per socket:    4
Socket(s):             1
NUMA node(s):          1
Vendor ID:             GenuineIntel
CPU family:            6
Model:                 60
Stepping:              3
CPU MHz:               800.000
BogoMIPS:              7183.36
Virtualization:        VT-x
L1d cache:             32K
L1i cache:             32K
L2 cache:              256K
L3 cache:              8192K
NUMA node0 CPU(s):     0-7
```

2 Thread performance comparison

2.1 Méthodologie

Afin de produire les données statistiques de performance, le cracker fut modifié afin de s'éxécuter pendant 10s et imprimer le nombre de mot de passes encryptées par seconde. Le cracker fut executé avec differents nombres de threads ($2^{\{0..8\}}$) et, pour chaque configuration, 20 executions ont été faites.

2.2 Données

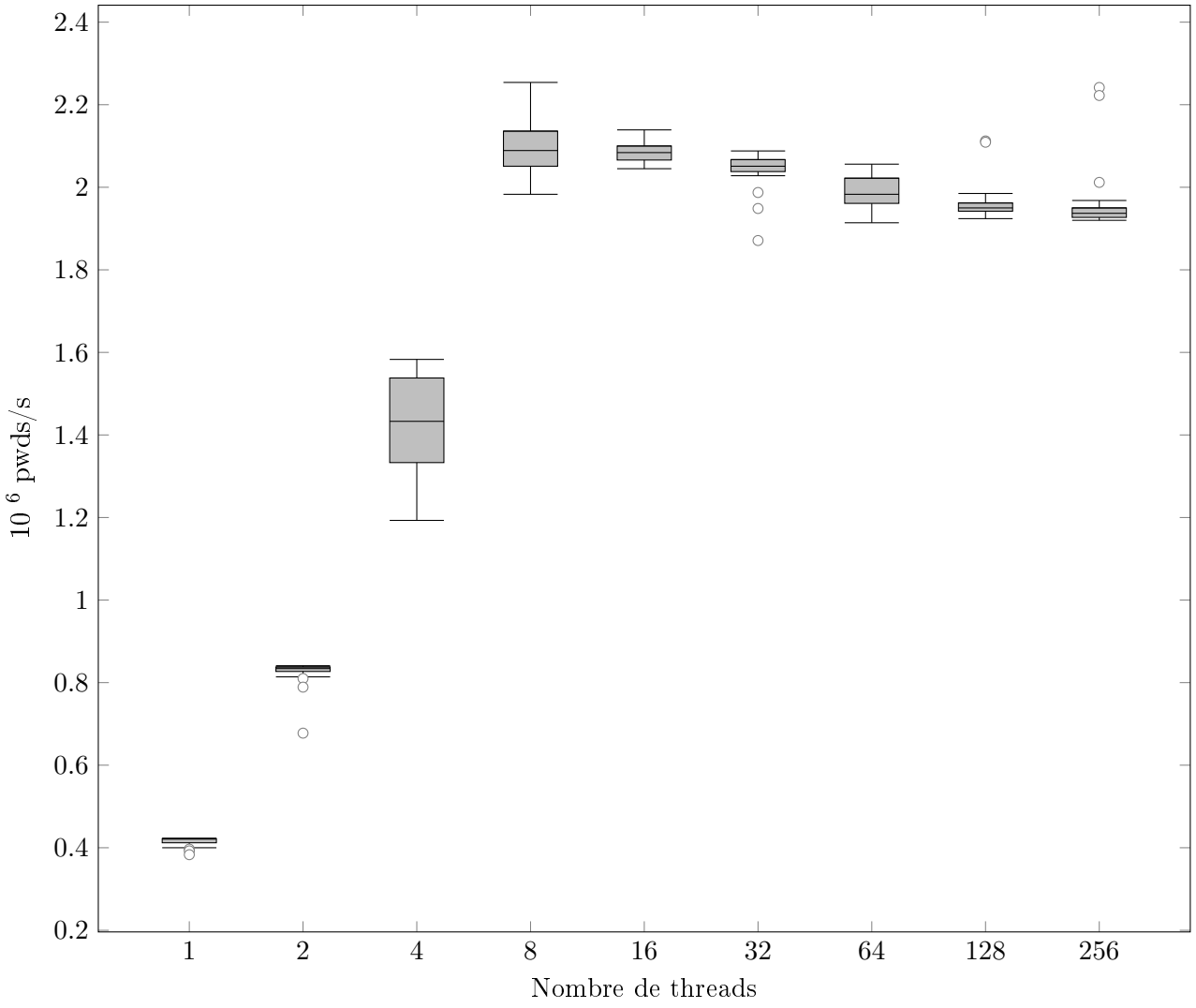


Figure 1: Comparaison des performances avec un nombre différent de threads

Nombre de threads	Median	Q3	Q1	Q4	Q0
1	0.421	0.422	0.412	0.423	0.4
2	0.834	0.838	0.827	0.841	0.814
4	1.433	1.538	1.333	1.583	1.193
8	2.089	2.136	2.051	2.254	1.983
16	2.084	2.1	2.066	2.139	2.045
32	2.051	2.067	2.038	2.088	2.028
64	1.983	2.022	1.961	2.056	1.914
128	1.95	1.962	1.942	1.985	1.924
256	1.937	1.95	1.927	1.968	1.92

Table 1: Comparaison des performances avec un nombre différent de threads