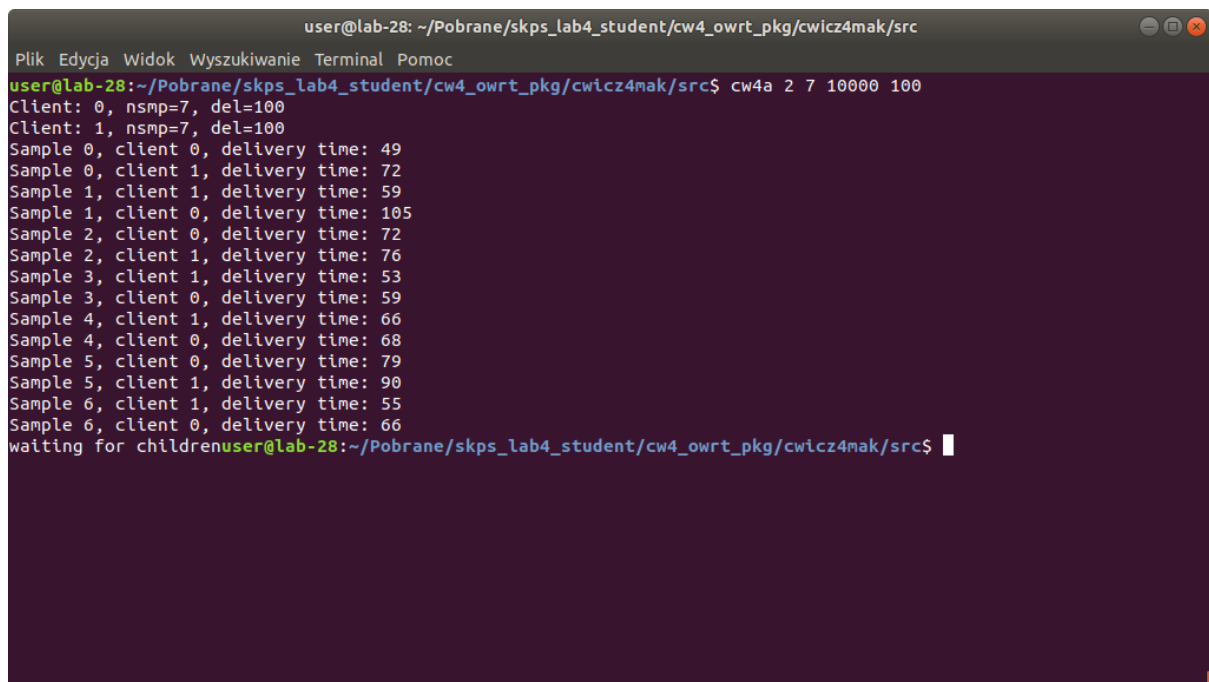


Sprawozdanie z laboratorium nr 4 z przedmiotu SKPS

Mateusz Okulus Mariusz Pakulski

Zadanie 1

```
export PATH="${PATH}:${(pwd)}"
```



```
user@lab-28: ~/Pobrane/skps_lab4_student/cw4_owrt_pkg/cwicz4mak/src
Plik Edycja Widok Wyszukiwanie Terminal Pomoc
user@lab-28:~/Pobrane/skps_lab4_student/cw4_owrt_pkg/cwicz4mak/src$ cw4a 2 7 10000 100
Client: 0, nsmp=7, del=100
Client: 1, nsmp=7, del=100
Sample 0, client 0, delivery time: 49
Sample 0, client 1, delivery time: 72
Sample 1, client 1, delivery time: 59
Sample 1, client 0, delivery time: 105
Sample 2, client 0, delivery time: 72
Sample 2, client 1, delivery time: 76
Sample 3, client 1, delivery time: 53
Sample 3, client 0, delivery time: 59
Sample 4, client 1, delivery time: 66
Sample 4, client 0, delivery time: 68
Sample 5, client 0, delivery time: 79
Sample 5, client 1, delivery time: 90
Sample 6, client 1, delivery time: 55
Sample 6, client 0, delivery time: 66
waiting for childrenuser@lab-28:~/Pobrane/skps_lab4_student/cw4_owrt_pkg/cwicz4mak/src$
```

Zadanie 2

W src wywołujemy make clean.

Do pliku feeds.conf.default dodajemy:
src-link lab4 /home/user/cw4_owrt_pkg

Wykonujemy:
./scripts/feeds update -a
./scripts/feeds install -p lab4 -a

W make menuconfig zaznaczamy pakiet w sekcji Examples.

make package/cwicz4mak/compile

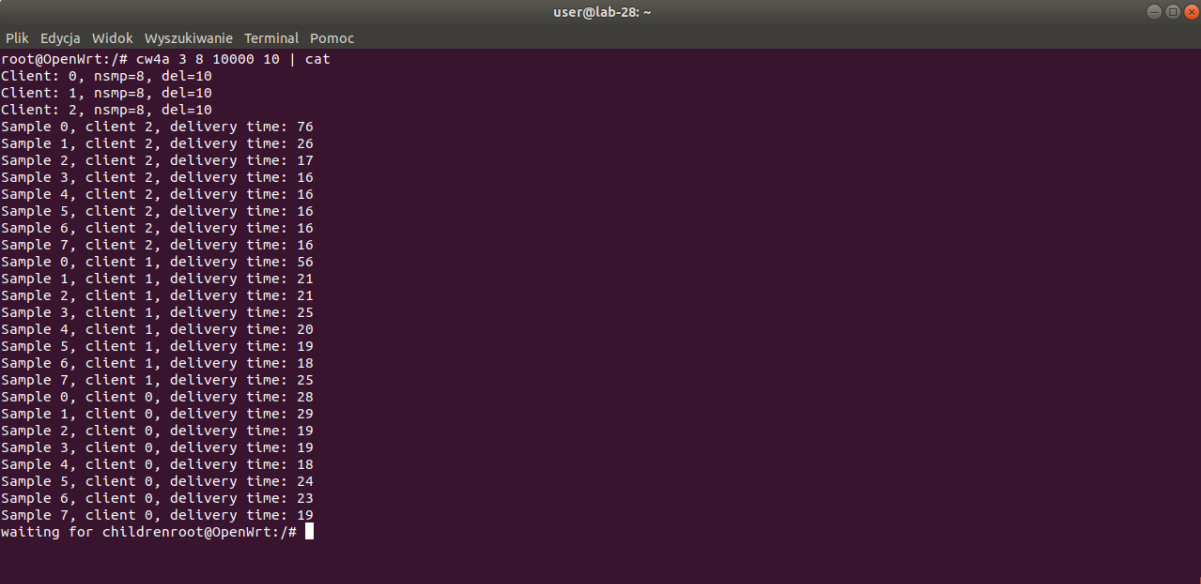
Kopiuujemy:

cd bin/packages/aarch64_cortex-a72/lab4/

python3 -m http.server

Na RPI: wget http://192.168.9.118:8000/cwicz4mak_1_aarch64_cortex-a72.ipk (adres IP z "ip a" na hoście)

opkg install cwicz4mak_1_aarch64_cortex-a72.ipk



```
user@lab-28: ~
Plik Edycja Widok Wyszukiwanie Terminal Pomoc
root@OpenWrt:/# cw4a 3 8 10000 10 | cat
Client: 0, nsmp=8, del=10
Client: 1, nsmp=8, del=10
Client: 2, nsmp=8, del=10
Sample 0, client 2, delivery time: 76
Sample 1, client 2, delivery time: 26
Sample 2, client 2, delivery time: 17
Sample 3, client 2, delivery time: 16
Sample 4, client 2, delivery time: 16
Sample 5, client 2, delivery time: 16
Sample 6, client 2, delivery time: 16
Sample 7, client 2, delivery time: 16
Sample 0, client 1, delivery time: 56
Sample 1, client 1, delivery time: 21
Sample 2, client 1, delivery time: 21
Sample 3, client 1, delivery time: 25
Sample 4, client 1, delivery time: 20
Sample 5, client 1, delivery time: 19
Sample 6, client 1, delivery time: 18
Sample 7, client 1, delivery time: 25
Sample 0, client 0, delivery time: 28
Sample 1, client 0, delivery time: 29
Sample 2, client 0, delivery time: 19
Sample 3, client 0, delivery time: 19
Sample 4, client 0, delivery time: 18
Sample 5, client 0, delivery time: 24
Sample 6, client 0, delivery time: 23
Sample 7, client 0, delivery time: 19
waiting for childrenroot@OpenWrt:/#
```

Zadanie 3

opkg update

opkg install htop

opkg install stress-ng

Do /bin/user/cmdline.txt dodajemy parametr maxcpus=1

Uruchamiamy screen. W jednym oknie uruchamiamy stress-ng --matrix 0 a w drugim cw4a.

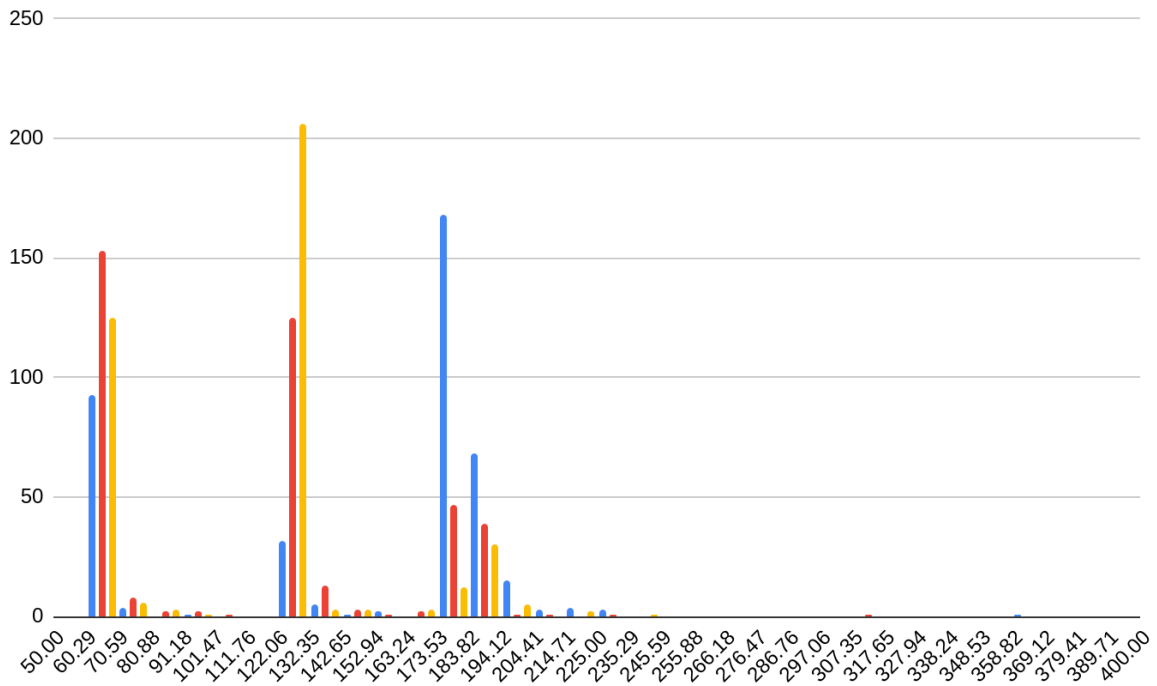
1. Pierwszy wariant: 3 klientów, 1 rdzeń, pełne obciążenie
cw4a 3 400 10000 310000
2. Drugi wariant: 3 klientów, 2 rdzenie, pełne obciążenie
cw4a 3 400 10000 490000
3. Trzeci wariant: 3 klientów, 2 rdzenie, bez obciążenia
cw4a 3 400 10000 600000
4. Czwarty wariant: 1 klient, 4 rdzenie, bez obciążenia
cw4a 1 400 10000 800000

Pliki z OpenWRT kopiujemy uruchamiając python3 -m http.server na RPI i pobierając pliki. Adres RPI uzyskujemy wywołując ifconfig.

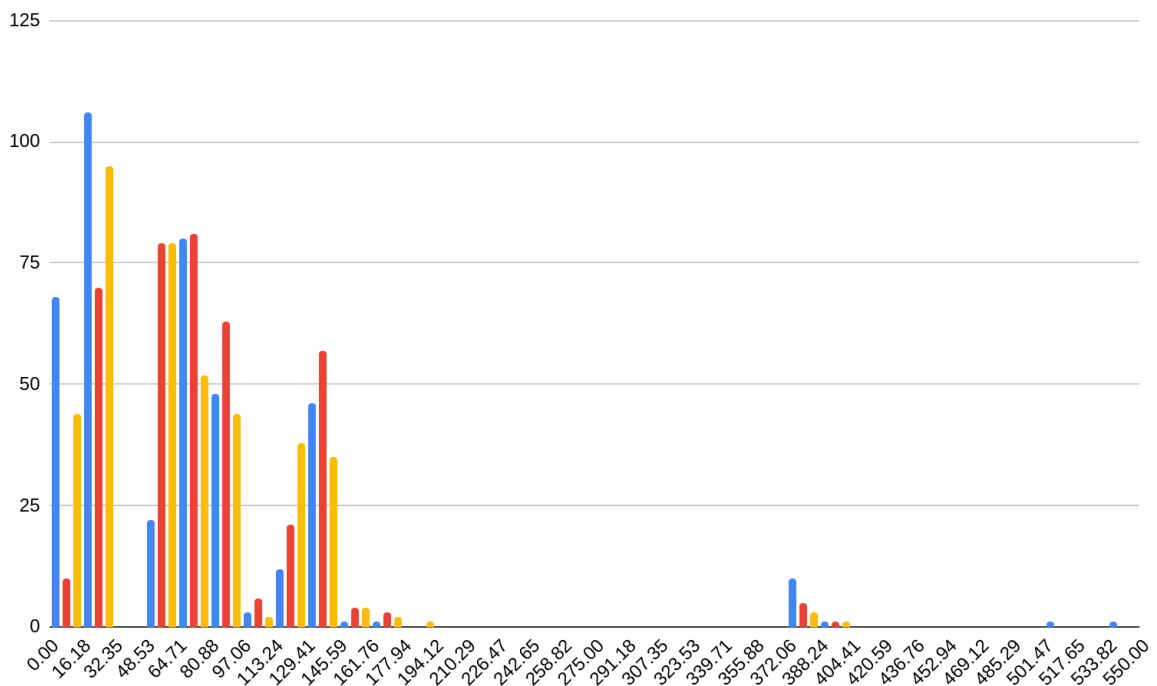
wget http://10.42.0.227:8000/server.txt

wget http://10.42.0.227:8000/client_{0,1,2}.txt

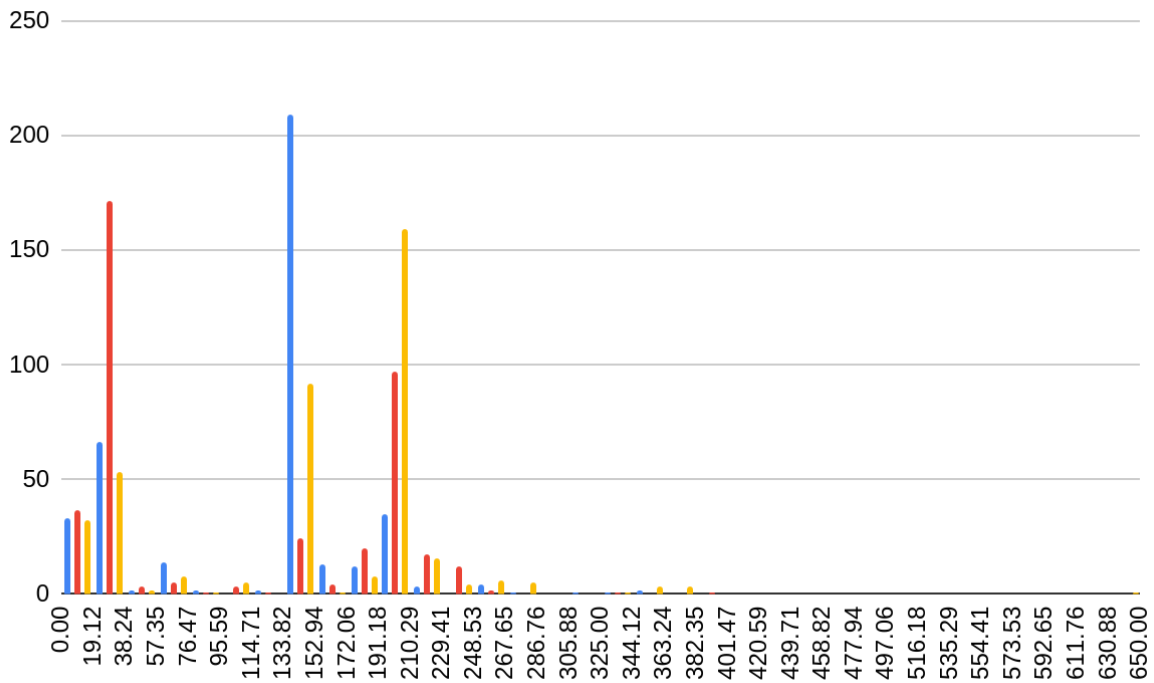
3.1 cw4a 3 400 10000 155000



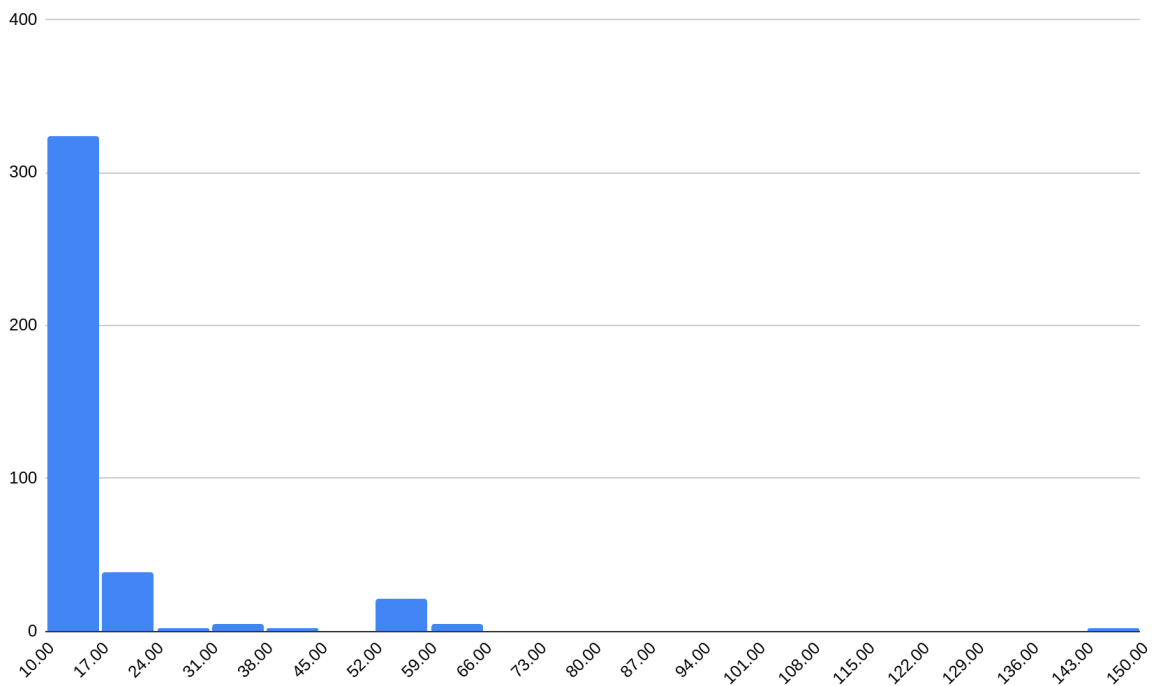
3.2 cw4a 3 400 10000 245000



3.3 cw4a 3 400 10000 300000



3.4 cw4a 1 400 10000 400000



Zadanie 5 (wszystkie wykresy są dla zmienionego wariantu z punktu 3.3 - 2 rdzenie, 3 klientów, bez obciążenia)

Zmieniamy linię:

```
pthread_cond_wait(&rbuf->cvar,&rbuf->cvar_lock);
```

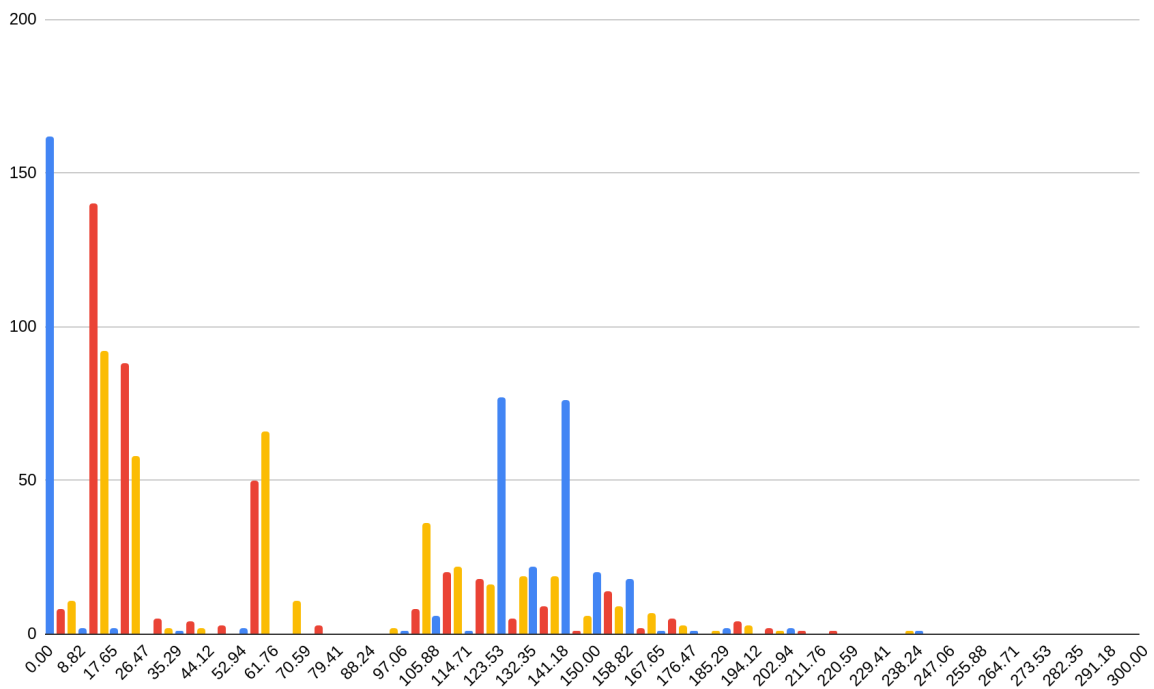
Na:

```
if (ncli != 0)
```

```
    pthread_cond_wait(&rbuf->cvar,&rbuf->cvar_lock);
```

Pakiet rekompilujemy i instalujemy poprzez:

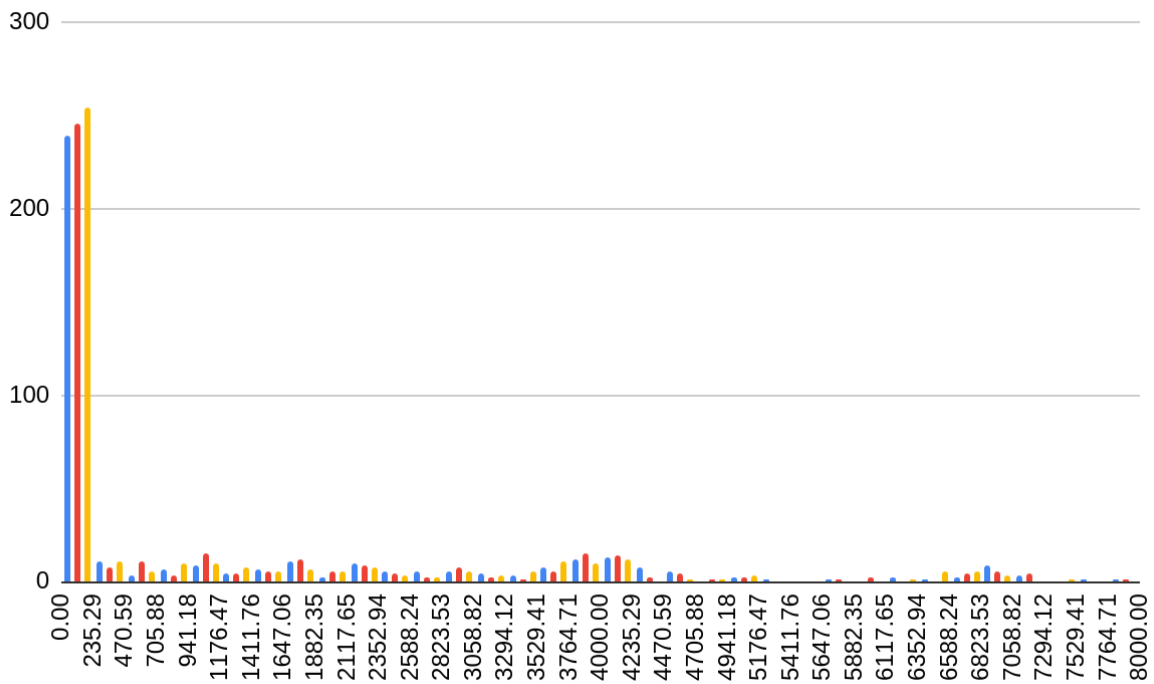
```
opkg install --force-reinstall cwicz4mak_1_aarch64_cortex-a72.ipk
```



Klient 0 (niebieski kolor) zwykle od razu wysyła próbkę. W oryginalnej wersji wariantu 3.3 występowało o wiele pakietów o opóźnieniu dostarczenia danych w okolicy zera.

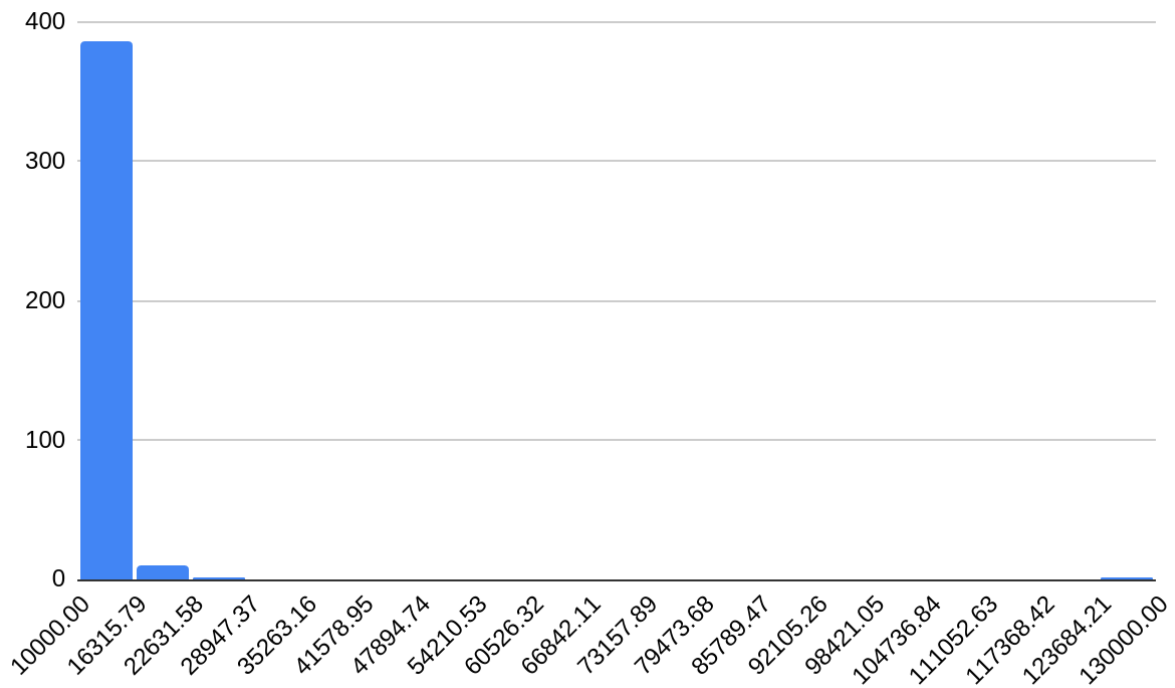
W celu ustawienia aktywnego oczekiwania dla wszystkich komentujemy oryginalną linię:

```
pthread_cond_wait(&rbuf->cvar,&rbuf->cvar_lock);
```



Zwykle czas oczekiwania jest bardzo mały, jednak czasem osiąga bardzo duże wartości wynikające z dużego obciążenia.

Zadanie 6



Taki histogram wynika z wartości dla zerowej próbki, która jest błędna.

0	1683821712405607	1683821712405607
1	1683821712423868	18261
2	1683821712438581	14713

Ponieważ odległość od poprzedniej próbki nie jest zdefiniowana dla pierwszej próbki zdecydowaliśmy nie wypisywać linii, gdy nie ma jeszcze ustawionego `prev_smptime`.

```
if (prev_smptime != 0) {  
    sprintf(line,"%d, %lu, %lu\n",i,smptime, smptime - prev_smptime);  
    assert(write(fout,line,strlen(line))>0);  
}  
prev_smptime = smptime;
```

Poprawiony wykres wygląda normalnie

