

# PSIR

## Laboratorium 2

Piotr Niewiński

Szymon Wiśniewski

Bartosz Stachyra

### Zadanie

#### Programowanie systemów internetu rzeczy i aplikacji sieciowych (PSIR)

##### Ćwiczenie 2

Politechnika Warszawska, Instytut Telekomunikacji

Prowadzący: Aleksander Pruszkowski

##### Charakterystyka dokumentu:

- Temat numer 10

##### Przed przystąpieniem do realizacji ćwiczenia należy:

- Przeczytać ze zrozumieniem dokument `Psir_lab2.pdf`.
- Pobrać maszynę wirtualną, proszę pamiętać, że plik OVA ma wielkość 1,3GB z adresu:  
`https://secure.tele.pw.edu.pl/~apruszko/psir/psir23z_20230908_1052.ova`

**Zadanie:** Zakładając architekturę mieszaną gdzie mamy wiele serwerów i jednego klienta, napisać oprogramowanie dla tych elementów. Przyjmij następujące założenia

Serwer 1 ma działać na sprzętowym węźle Arduino UNO, do węzła tego jest podłączony pojedynczy włącznik chwilowy, podłączony do D2.

Serwer 2 ma działać na Arduino emulowanym przez EBSimUnoEthCurses, węzeł ten wyposażony jest w diodę LED podłączoną do D4.

Klient ma działać jako aplikacja POSIX pod systemem Linux, klient ten ma: a)czekać na zgłoszenie się serwerów 1 (na porcie: 15281 protokołu UDP), oraz serwera 2 (na porcie: 18510 protokołu UDP), b)cyklicznie co 1700 ms, odpytywać serwer 1 o stan jego zasobu, c)zgodnie ze stanem tego zasobu klient ma wydawać polecenia serwerowi 2 tj. włączyć element sygnalizacyjny na czas 4250 ms, d)cyklicznie sprawdzać czy serwer 1 jest aktywny, gdyby wykryto, że przez czas 15300 ms był on nie aktywny, klient ma zakończyć swoją pracę z odpowiednim komunikatem na konsoli w której został uruchomiony podając jak najwięcej szczegółów związanych z tą sytuacją.

Przyjmij także że: a)zarówno serwer 1 jak i 2 znają adres IP klienta - możesz wpisać tę informację w ich kod, b)oba serwery nie komunikują się ze sobą bezpośrednio, c)cała komunikacja odbywa się wyłącznie z wykorzystaniem datagramów UDP, zachowaj także numery portów na jakich nasłuchuje klient datagramów od obu serwerów czyli takie jakie podano przy opisie procedury ich zgłaszania się (tj. odpowiednio: 15281 i 18510).

Zadbaj o to aby przysyłane pakiety UDP były możliwe krótkie i przekazywały treści w sposób możliwie jak najbardziej zwięzły (zwięzłość i jednoznaczność wpływa bezpośrednio na ocenę z laboratorium).

Pamiętaj aby dla zadania 2 i wszystkich użytych emulatorów Arduino utworzyć pliki `infile.txt`, tak aby pokazać za ich pomocą zmiany stanu wejść tych emulatorów.

Dla wyjaśnienia przyjętych rozwiązań (implementacja, wewnętrzne protokoły, ...), opisz je w raporcie - wskazane jest aby opis ten był zwięzły i utworzony w formacie PDF. Pamiętaj, że żadne inne formaty dokumentów elektronicznych np.: DOC, DOCX, ... nie będą przyjmowane.

Fianlinie raport i wszelkie pliki źródłowe (C, CPP, INO oraz wszystkie pliki `infile.txt`) będące wynikiem prac nad tym laboratorium, proszę umieścić na przydzielonym Tobie indywidualnym repozytorium GIT - z tego miejsca prowadzący będzie pobierał te pliki do późniejszego ocenia i wystawienia oceny z tego laboratorium.

Dla przypomnienia - aby właściwe wersje plików znalazły się na przydzielonym Tobie zdalnym repozytrum GIT należy wykonać polecenia:

```
git add .
git commit -a -m "Rozwiazanie zadania dla Lab2"
git push
```

Piotr Niewiński

VLAN689

```

student@iot:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:13:93:ee brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.40/24 brd 192.168.0.255 scope global dynamic enp0s3
        valid_lft 3041sec preferred_lft 3041sec
    inet6 fe80::a00:27ff:fe13:93ee/64 scope link
        valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:2a:ab:8c brd ff:ff:ff:ff:ff:ff
    inet 192.168.56.101/24 brd 192.168.56.255 scope global dynamic enp0s8
        valid_lft 471sec preferred_lft 471sec
    inet6 fe80::a00:27ff:fe2a:ab8c/64 scope link
        valid_lft forever preferred_lft forever

```

Adres klienta 192.168.0.40

```

student@iot:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:13:93:ee brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.41/24 brd 192.168.0.255 scope global dynamic enp0s3
        valid_lft 3175sec preferred_lft 3175sec
    inet6 fe80::a00:27ff:fe13:93ee/64 scope link dadfailed tentative
        valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:39:e9:4e brd ff:ff:ff:ff:ff:ff
    inet 192.168.56.102/24 brd 192.168.56.255 scope global dynamic enp0s8
        valid_lft 370sec preferred_lft 370sec
    inet6 fe80::a00:27ff:fe39:e94e/64 scope link
        valid_lft forever preferred_lft forever

```

Adres serwera na emulatorze 192.168.0.41

```

student@iot:~$ ping -c 4 192.168.0.41
PING 192.168.0.41 (192.168.0.41) 56(84) bytes of data.
64 bytes from 192.168.0.41: icmp_seq=1 ttl=64 time=0.211 ms
64 bytes from 192.168.0.41: icmp_seq=2 ttl=64 time=0.299 ms
64 bytes from 192.168.0.41: icmp_seq=3 ttl=64 time=0.362 ms
64 bytes from 192.168.0.41: icmp_seq=4 ttl=64 time=0.244 ms

--- 192.168.0.41 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3064ms
rtt min/avg/max/mdev = 0.211/0.279/0.362/0.057 ms

```

Klient pinguje serwer

```
student@iot:~$ ping -c 4 192.168.0.40
PING 192.168.0.40 (192.168.0.40) 56(84) bytes of data.
64 bytes from 192.168.0.40: icmp_seq=1 ttl=64 time=0.304 ms
64 bytes from 192.168.0.40: icmp_seq=2 ttl=64 time=0.273 ms
64 bytes from 192.168.0.40: icmp_seq=3 ttl=64 time=0.320 ms
64 bytes from 192.168.0.40: icmp_seq=4 ttl=64 time=0.425 ms

--- 192.168.0.40 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3060ms
rtt min/avg/max/mdev = 0.273/0.330/0.425/0.057 ms
```

Serwer pinguje klienta

```
192.168.0.25
```

Adres IP serwera Arduino

```
student@iot:~$ ./a.out
Server 2 connected
Server 1 connected
^C
```

Zgłoszenie się serwerów

```
student@iot:~$ ./a.out
Server 1 connected
Server 2 connected
Server timed out
```

Poprawny timeout

```
GPIO
Z0:0x03ff Z1:0x03ff Z2:0x03ff Z3:0x03ff Z4:0x03ff Z5:0x03ff
D0:1 D1:1 D2:1 D3:1 D4:1 D5:1 D6:1 D7:1 D8:1 D9:1 D10:1 D11:1 D12:1 D13:1

UART
192.168.0.41
1
```

Po wciśnięciu przycisku serwer otrzymuje polecenie o zapaleniu diody i w reakcji wypisuje 1 (nie udało się doprowadzić do działania zapalania diody, dlatego w celu przetestowania komunikacji wyświetlamy 1)