# Laboratoria 10

Krzysztof Żywiecki

21 maja 2020

## 1 Zadanie 1

### 1.1 Co identyfikuje adres IP a co port?

Adres IP służy do identyfikacji komputera w sieci, natomiast port służy do określania który proces ma korzystać z danych.

### 1.2 Czym różni się deskryptor gniazda od deskryptora pliku?

Deskryptory stanowią pewną abstrakcję do korzystania z zasobów komputera, plików, urządzeń i gniazd. Deskryptory posiadają funkcje *read()* oraz *write()*, ale poza tym tworzenie i korzystanie z nich jest bardzo różne (dla przykładu, deskryptor pliku tworzymy przez funkcję *open()*, a deskryptor gniazda przez funkcję *socket()*).

### 1.3 Który argument funkcji służy do określenia typu gniazda?

Do tego służy drugi argument funkcji, czyli *int type*.

### 1.4 Jakie wartości przyjmuje ten argument dla gniazd połączeniowych a jakie dla bezpołączeniowych?

Do socketów połączeniowych użyć możemy SOCK_STREAM, SOCK_SEQPACKET, natomiast do bezpołączeniowych można użyć SOCK_DGRAM lub SOCK_RAW.

### 1.5 Jaki jest zakres liczbowy portów dostępnych do wykorzystania dla uzytkownika niebędącego administratorem?

Użytkownik może wykorzystać porty w zakresie 1024-49151.

### 1.6 Wyjeśnienie pojęć:

Big oraz Little endian mają w kontekście komunikacji sieciowej związek z kolejnością bajtów w liczbach zajmujących więcej niż jeden bajt. W Big-endian na

początku jest najbardziej znaczący bajt, a w Little endian na początku jest najmniej znaczący bajt.

Network Byte Order jest standardem dotyczącym kolejności bajtów liczb przesyłanych przez internet. Jest on równoznaczny z Big-endian.

### 1.7 Do czego służą funkcje: htonl, htons, ntohl, ntohs? Co oznaczają ich nazwy (od czego są to skróty)?

- htonl() - funkcja konwertuje liczbę dodatnią 32 bitową na porządek sieciowy
- htons() - jak wyżej tylko dla liczb 16 bitowych
- ntohl() - funkcja odwrotna do htonl()
- ntohs() - funkcja odwrotna do htons()

### 1.8 Liczbę w postaci szesnastkowej: cafe zapisano na dwóch bajtach w postaci: feca. Jaka to reprezentacja?

Jest to reprezentacja Little-endian, gdyż najmniej znaczący bajt znajduje się na początku.

## 2 Rozbudowa serwera

W punkcie pierwszym należy sprawić żeby po zakończeniu działania serwer nie kończył programu, ale dalej przyjmował żądania. Żeby to osiągnąć, wystarczy umieścić wewnątrz funkcji while kod odpowiedzialny za łączenie i komunikację.

Żeby program mógł obsłużyć więcej niż jednego klienta, po wykonaniu accept, program robi fork. Proces potomny obsługuje połączenie, a proces rodzica czeka na nowe połączenie.

Na potrzeby komunikatora serwer przerobiono tak żeby obsługiwał tylko jednego klienta. Serwer i klient w podobny sposób odczytują i wysyłają dane: program tworzy proces potomny do odbierania danych, podczas gdy proces rodzica wysyła komunikaty. Co prawda implementacja nie działa po stronie serwera, ale plan był dobry.

## 3 Przeglądarka

Program wysyła zapytanie GET na zadany URL, i pobiera zawartość strony. Kod odpowiedzialny za łączenie się z serwerem jest identyczny jak kod klienta w zadaniu drugim.