Gniazda UDP

Bartłomiej Świercz

Katedra Mikroelektroniki i Technik Informatycznych

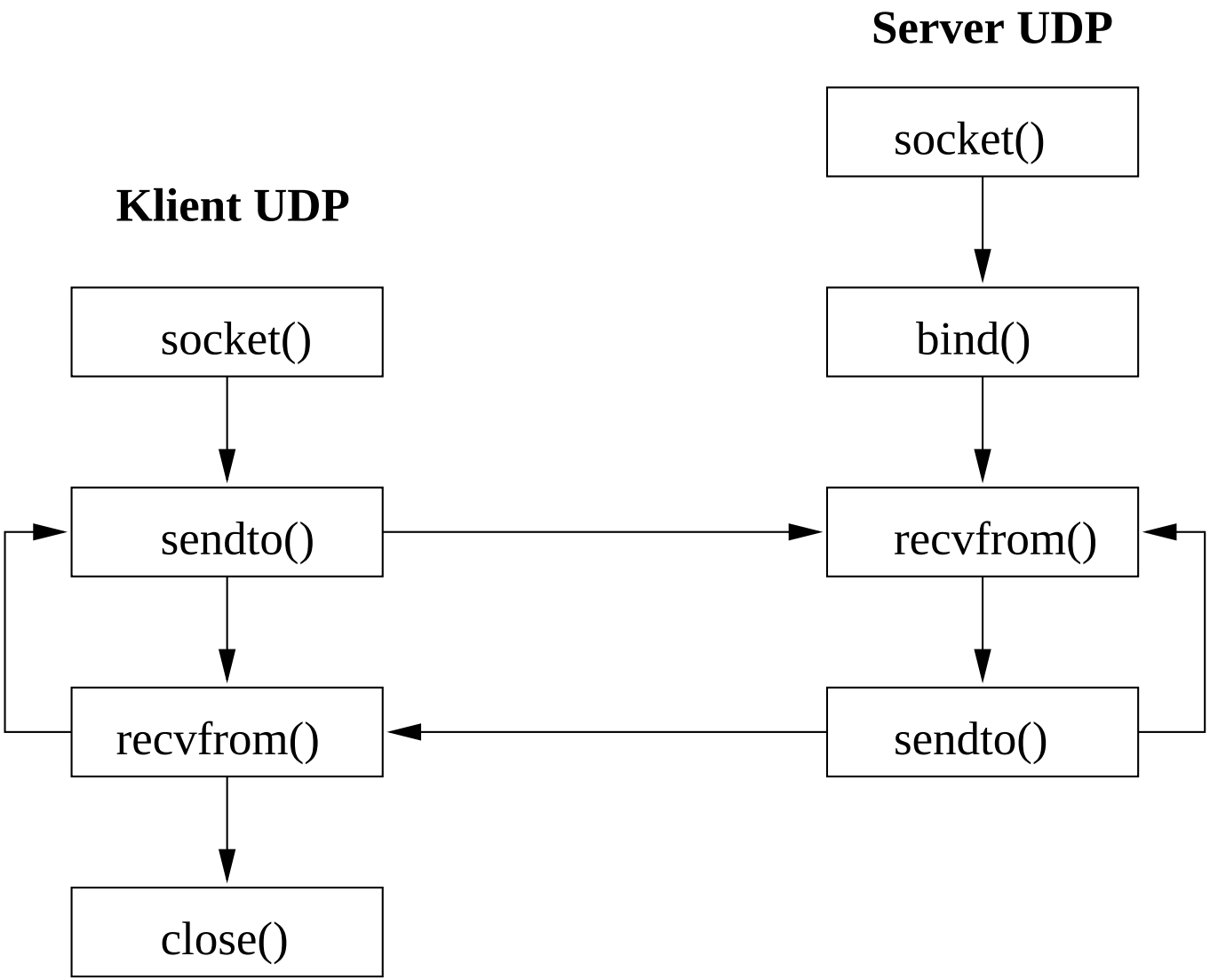
Łódź, 3 kwietnia 2006

Wstęp

Ze względu na różnice w protokołach TCP i UDP sposób korzystania z gniazd UDP różni sie znacznie od sposobu korzystania z gniazd dla protokołu TCP.

Większość różnić wynika z podstawowych cech protokołu UDP, czyli braku niezawodności i bezpołączeniowości.

Typowe wywołanie funkcji dla gniazd UDP



Funkcja socket

Funkcja `socket()` służy do utworzenia deskryptora (nowego socketu). Funkcja ta zadeklarowana jest następująco:

```
#include <sys/types.h>
#include <sys/socket.h>

int socket(int domain, int type, int protocol);
```

Dla socketu typu UDP funkcja `socket()` przyjmuje postać:

```
int sock = socket (PF_INET, SOCK_DGRAM, 0);
```

Funkcja recvfrom

Deklaracja funkcji `recvfrom()`:

```
#include <sys/types.h>
#include <sys/socket.h>

ssize_t recvfrom(int s, void *buf, size_t len, int flags,
    struct sockaddr *from, socklen_t *fromlen);
```

Uwaga: Z każdym socketem UDP związany jest niejawny bufor odbiorczy zrealizowany w postaci kolejki FIFO.

Funkcja sendto

Deklaracja funkcji `sendto()`:

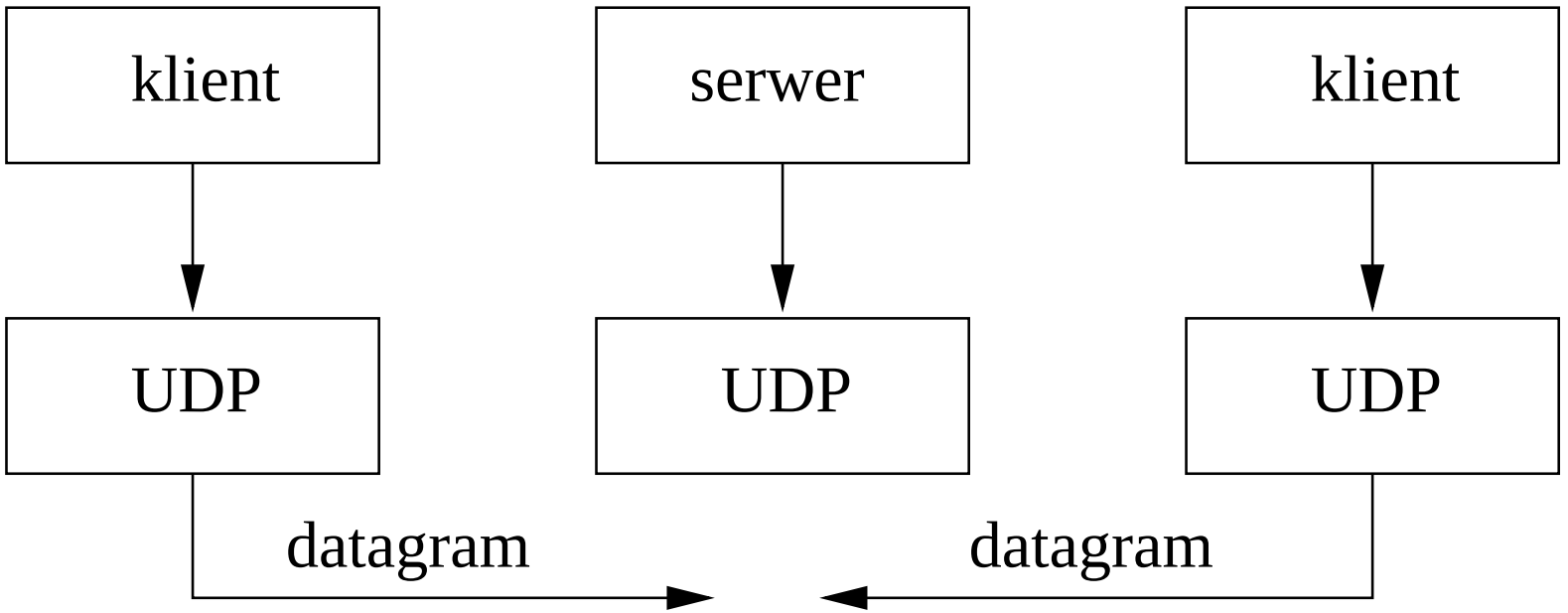
```
#include <sys/types.h>
#include <sys/socket.h>

ssize_t sendto(int s, const void *buf, size_t len, int
    flags, const struct sockaddr *to, socklen_t tolen);
```

Uwaga: Możliwe jest wysłanie datagramu o długości równej zero.

Serwer iteracyjny

Funkcje `sendto()` i `recvfrom()` w naturalny sposób tworzą wielodostępowy serwer iteracyjny:



Tracenie datagramów

Korzystając z protokołu UDP należy zawsze mieć świadomość, że datagram może zostać bezpowrotnie stracony. Co więcej utrata datagramu jest kłopotliwa ponieważ należy ustalić czy utracony został datagram wysłany, czy może utracony został datagram zawierający odpowiedź.

Utrata datagramu może nastąpić na skutek:

- przepełnienia bufora (np. komputer nadający jest znacznie szybszy od komputera odbierającego),
- odrzucenia pakietu przez jeden z routerów pośredniczących w ruchu sieciowym.

Błąd asynchroniczny

Błąd asynchroniczny występuje, gdy klient próbuje połączyć się z serwerem, który nie został uruchomiony (port docelowy jest zamknięty). Klient wysyła datagram lecz system odpowiada mu komunikatem ICMP *port unreachable*. Błąd asynchroniczny spowoduje funkcja `sendto()` ponieważ po wysłaniu datagramu zakończy ona swoje działanie w sposób normalny. Ponieważ błąd ICMP przekazany jest dopiero po pewnym czasie to klient nie ma możliwości dowiedzenia się o tym.

Podstawowa zasada mówi, że błędów asynchronicznych nie przekazuje się dla gniazd UDP (odstępstwem od tej zasady są gniazda UDP *połączone*).

Gniazda UDP połączone

Specyfikacja gniazd (socketów) rozróżnia dwa rodzaje gniazd UDP:

- Gniazda UDP niepołączone — są to gniazda tworzone domyślnie na których nie wywołano funkcji `connect()`.
- Gniazda UDP połączone — są to gniazda UDP na rzecz których wywołano funkcję `connect()`.

Porównanie gniazd UDP połączonych i niepołączonych

1. Po połączeniu gniazda UDP (wywołaniu funkcji `connect()`) nie można już określić adresu IP ani numeru portu dla operacji wyjścia. Oznacza to, że zamiast funkcji `sendto()` używa się funkcji `send()` lub `write()`.
2. Jądro przekazuje dla socketu jedynie te datagramy, których adres źródłowy odpowiada adresowi podanemu funkcji `connect()`. Oznacza to, że nie stosuje się funkcji `recvfrom()`, a funkcje `read()` lub `recv()`.
3. Błędy asynchroniczne są przekazywane do procesu dla gniazd połączonych.

Wielokrotne wywołanie funkcji connect

Proces może wywołać wielokrotnie funkcję `connect()` na rzecz gniazda UDP w celu:

- określenia nowego docelowego adresu IP i portu,
- rozłączenia gniazda.

Uwaga: Aby rozłączyć gniazdo należy ustawić pole `sin_family` struktury adresowej na wartość `AF_UNSPEC`.

Kiedy używać protokołu UDP

Aby odpowiedzieć na to pytanie należy być świadomym jakie są zalety protokołu UDP w stosunku do protokołu TCP:

- Protokół UDP udostępnia rozgłaszanie i rozsyłanie grupowe.
- Ta sama komunikacja daje się zrealizować za pomocą mniejszej liczby pakietów UDP niż TCP.
- Implementacja programowa protokołu UDP jest dużo bardziej prosta i zwięzła niż protokołu TCP.

Ćwiczenia

- Błąd asynchroniczny.
- Serwer iteracyjny.
- Serwer wielobieżny.