Linuxネットワークプログラミング(socket)

書籍「Linuxネットワークプログラミング」ソフトバンククリエイティブ(株)発行 の1〜3章が参考になる。（下記リンク）

1. インターネットの仕組み 2. TCP通信の基礎 3. UDP通信の基礎 4. 目次 5. はじめに

UDPソケット通信サンプルプログラム

以下のサンプルは、システムコール時のエラー判定やタイムアウト監視など不十分なので、実際に使う場合はエラー処理などもよく吟味して利用する事。

UDPパケットを受信待ちして返信

上記PDFのList3-17プログラムを参考。このサンプルは、受け取った相手に対してUDPでそのまま返信を繰り返すプログラム。

udp-test-server.c

```
#include <stdio.h>
#include <unistd.h>
#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>

int
main(int argc, char *argv[])
{
    int sock;
    struct sockaddr_in addr;
    struct sockaddr_in senderinfo;
    socklen_t addrlen;
    char buf[2048];
    char senderstr[16];
    int n;

    printf("*** UDP test(server) program start ***\n");

    /* AF_INET+SOCK_DGRAMなので、IPv4のUDPソケット */
    sock = socket(AF_INET, SOCK_DGRAM, 0);

    /*待ち受けポート番号を12345にするためにbind()を行う */
    addr.sin_family = AF_INET;
    addr.sin_port = htons(12345);
    addr.sin_addr.s_addr = INADDR_ANY;
```

```
    bind(sock, (struct sockaddr *)&addr, sizeof(addr));

    while (1) {
        memset(buf, 0, sizeof(buf));

        /* recvfrom()を利用してUDPソケットからデータを受信*/
        addrlen = sizeof(senderinfo);
        n = recvfrom(sock, buf, sizeof(buf) - 1, 0,
                    (struct sockaddr *)&senderinfo, &addrlen);

        /*送信元に関する情報を表示*/
#if 1
        inet_ntop(AF_INET, &senderinfo.sin_addr, senderstr,
sizeof(senderstr));
#else
        strcpy(senderstr, inet_ntoa(senderinfo.sin_addr));
#endif
        printf("recvfrom : %s, port=%d\n", senderstr,
ntohs(senderinfo.sin_port));

        /* UDPで返信*/
        sendto(sock, buf, n, 0, (struct sockaddr *)&senderinfo,
addrlen);

        /*送信元に関する情報をもう一度表示*/
        printf("sent data to : %s, port=%d\n", senderstr,
                ntohs(senderinfo.sin_port));
    }

    /*このサンプルコードはここへは到達しません*/
    close(sock);

    return 0;
}
```

UDPパケットの送信&受信

上記PDFのList3.18プログラムを参考。上記のUDPアプリケーションに対してUDPパケットを送信して返信を受け取るプログラム。例えば../udp-test-client 127.0.0.1のように宛先を指定して実行する。

udp-test-client.c

```
#include <stdio.h>
#include <unistd.h>
#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
```

```
#include <arpa/inet.h>

void
print_my_port_num(int sock)
{
    struct sockaddr_in s;
    socklen_t sz = sizeof(s);
    getsockname(sock, (struct sockaddr *)&s, &sz);
    printf("%d\n", ntohs(s.sin_port));
}

int
main(int argc, char *argv[])
{
    int sock;
    struct sockaddr_in addr;
    struct sockaddr_in senderinfo;
    socklen_t senderinfolen;
    int n;
    char buf[2048];

    if (argc != 2) {
        fprintf(stderr, "Usage : %s dstipaddr\n", argv[0]);
            return 1;
    }

    sock = socket(AF_INET, SOCK_DGRAM, 0);

    addr.sin_family = AF_INET;
    addr.sin_port = htons(12345);
#if 1
    inet_pton(AF_INET, argv[1], &addr.sin_addr.s_addr);
#else
    inet_aton(argv[1], &addr.sin_addr);
#endif

    n = sendto(sock, "HELLO", 5, 0, (struct sockaddr *)&addr,
sizeof(addr));
    if (n < 1) {
        perror("sendto");
        return 1;
    }

    print_my_port_num(sock);

    memset(buf, 0, sizeof(buf));
    senderinfolen = sizeof(senderinfo);
    recvfrom(sock, buf, sizeof(buf), 0,
            (struct sockaddr *)&senderinfo, &senderinfolen);

    printf("%s\n", buf);
```

```
    close(sock);
    return 0;
}
```

TCPソケット通信サンプルプログラム

以下のサンプルは、システムコール時のエラー判定やタイムアウト監視など不十分なので、実際に使う場合はエラー処理などもよく吟味して利用する事。

単純なTCPサーバ

上記PDFのList2.5プログラムを参考。このサンプルは、接続してきたクライアントに対して"HELLO"という文字列を送信して終了。

tcp-test-server.c

```
#include <stdio.h>
#include <unistd.h>
//#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>

int
main(int argc, char *argv[])
{
    int sock0;
    struct sockaddr_in addr;
    struct sockaddr_in client;
    socklen_t len;
    int sock;

    printf("*** TCP test(server) program start ***\n");

    /* ソケットの作成 */
    sock0 = socket(AF_INET, SOCK_STREAM, 0);

    /* ソケットの設定 */
    addr.sin_family = AF_INET;
    addr.sin_port = htons(12345);
    addr.sin_addr.s_addr = INADDR_ANY;
    bind(sock0, (struct sockaddr *)&addr, sizeof(addr));

    /* TCPクライアントからの接続要求を待てる状態にする */
```

```
    listen(sock0, 5);

    /* TCPクライアントからの接続要求を受け付ける */
    len = sizeof(client);
    sock = accept(sock0, (struct sockaddr *)&client, &len);

    /*5文字送信*/
    write(sock, "HELLO", 5);

    /* TCPセッションの終了*/
    close(sock);

    /* listen するsocketの終了*/
    close(sock0);

    return 0;
}
```

単純なTCPクライアント

上記PDFのList2-6プログラムを参考。このサンプルは、サーバに接続すると文字列を受信して表示。

tcp-test-client.c

```
#include <stdio.h>
#include <unistd.h>
#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>

int
main(int argc, char *argv[])
{
    struct sockaddr_in server;
    int sock;
    char buf[32];
    int n;

    /*ソケットの作成*/
    sock = socket(AF_INET, SOCK_STREAM, 0);

    /*接続先指定用構造体の準備*/
    server.sin_family = AF_INET;
    server.sin_port = htons(12345);

    /* 127.0.0.1はlocalhost */
```

```
#if 1
    inet_pton(AF_INET, "127.0.0.1", &server.sin_addr.s_addr);
#else
    inet_aton("127.0.0.1", &server.sin_addr);
#endif

    /* サーバに接続 */
    connect(sock, (struct sockaddr *)&server, sizeof(server));

    /* サーバからデータを受信 */
    memset(buf, 0, sizeof(buf));
    n = read(sock, buf, sizeof(buf));

    printf("%d, %s\n", n, buf);

    /* socketの終了 */
    close(sock);

    return 0;
}
```

関連記事

- msys2Windowsネットワークプログラミングwinsock
- wireshark

参考書籍

---

1. Linuxネットワークプログラミング(SBクリエイティブ) 2. TCP/IPソケットプログラミング C言語編(オーム社) 3. ふつうのLinuxプログラミング 第2版 4. Linuxネットワークプログラミングバイブル(秀和システム)

上記[4.]の、小俣氏と種田氏が書かれた本が、実践的でとても参考になる。残念ながら冊子版は絶版になっているようだがAmazoneから電子書籍は購入可能。尚、上リンクから、誌面のサンプルソースもダウンロードできる。

参考

---

1. UNIXネットワークプログラミングに登場する構造体の紹介と正しい使い方 2. C言語で学ぶソケットAPI入門 第4回 UDPサーバ/クライアント編 #1 3. UDPを使う 4. 第 2 章 ソケットインタフェース

From:
https://www.yokoweb.net/dokuwiki/ - **新石器Wiki**

Permanent link:
https://www.yokoweb.net/dokuwiki/programing/linux-programing/network-programing/linux-socket-network-programing?rev=1626350877

Last update: **2021/07/15 21:07**