第５話（ＳＮＳ通信の基礎）

タヌキ、コンピュータのポイント理解できたかな。
次からは、木（詳細項目）を見て行くことになるぞ。
タヌキ同士もスマホを使ってメールやらチャット、さらにはリモート会議などをやっているだろう。その基礎がプロセス間通信と言うんだよ。プロセスとはコンピュータ内で実行されているプログラムを意味するんだ。スマホでも実行されているアプリのようなものね。
日々使っているものだから多少は興味があるだろう。

あるある。いつも不思議だな、と思っていたんだ。
リモート会議には無限に多くのタヌキが参加できるのかな、などとね。
キツネ、教えてくれる。

では、ゆるりと始めるか！
我々が使っているネットワークは主に TCP/IP というプロトコル（通信の約束ごと）を使っているのさ。その TCP/IP は、通信の段階毎に使われるプロトコルが異なり、その段階（階層）を４階層で考えたのが DARPA（４階層）モデルというものだ。それを図示するね

［DARPA（４階層）モデルで考える TCP/IP の階層］

アプリケーション層 Telenet、Http、Ftp SMTP、NFS、 X ウィンドウ
トランスポート層 TCP、UDP、ICMP
インターネット層 IP v 4、IP v 6
データリンク層 （個別ネットワーク層） WAN、LAN、無線

ソケット：トランスポート層へのインタフェース
TCP ポート、UDP ポート

・ＴＣＰ/ＩＰ

ＩＰ　：Internet Protocol

ＴＣＰ：Transmission Control Protocol　（コネクション型という意味）

・ＵＤＰ/ＩＰ

ＵＤＰ：User Datagram Protocol（データグラム型、つまりコネクションレス型という意味）

次は、ソケットの話になるぞ。ロケットでは無いぞ。
プロセス間通信にはソケットというものが使われているんだ、それはトランスポート層での通信の通り道ことだ。そのことを覚えておいてくれ。

タヌキ、まずは実習だ！
CentOS という Linux の OS を起動してくれ。CentOS が無ければ、Ubuntu、Fedora でもいいぞ。
OS というのは、コンピュータの中心になるプログラム（ソフトウェア）だ。マイクロソフトの Windows が一番有名だが、Linux も OS だ。オイラは、CentOS が好きだ。その OS を起動してくれ。

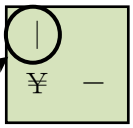
次に端末とやらを起動してくれるか。
コマンドラインで使用する１つ１つのコマンドもプログラムでできているのだ。小さいプログラムだけどね。
であるから、コマンドを実行すれば、そのコマンドはプロセスということになる。複数のコマンドを実行し、コマンド間でデータの送受が行われれば、プロセス間通信になるのだ。簡単な例を示す。

［簡単なプロセス間通信の例］

・シェル（使えるコマンドが限定される）内のコマンドを利用
・パイプがプロセス間通信の通信路となる。

オプション
$ ls –l | grep “txt” | more
プロセス　パイプ　プロセス　パイプ　プロセス
リスト表示　該当文字を含む　１画面づつ（「space」で次画面、「enter」で１行づつ）

パイプってどうやって入力するんだ？

キーボードの右上側にある パイプ  キーを

［Shift］キーを押しながら押せば入力できるよ！

次は、ソケットにつながるファイルディスクリプタの話だ。
プロセス（実行中のプログラム）は、外部の機器とデータのやり取りをする出入り口を用意しています。その出入り口には０から１０２３までの番号が順番に割り振られます。つまり、この１０２４個の出入り口に割り振られた整数がファイルディスクリプタです。ただ、外部機器のようなハードだけでなく、自分（プロセス）以外のプロセス（ソフトウェア）も外部とみなすことに注意してください。
ファイルディスクリプタの内、0 は、キーボードなどの標準入力、１はモニターやプリンターなどの標準出力、２は標準エラー出力に固定されています。それ以外の番号は接続順に割り振られます。故に１個のプロセスで、理論上約５００個のプロセスと同時に相互通信ができることになります。

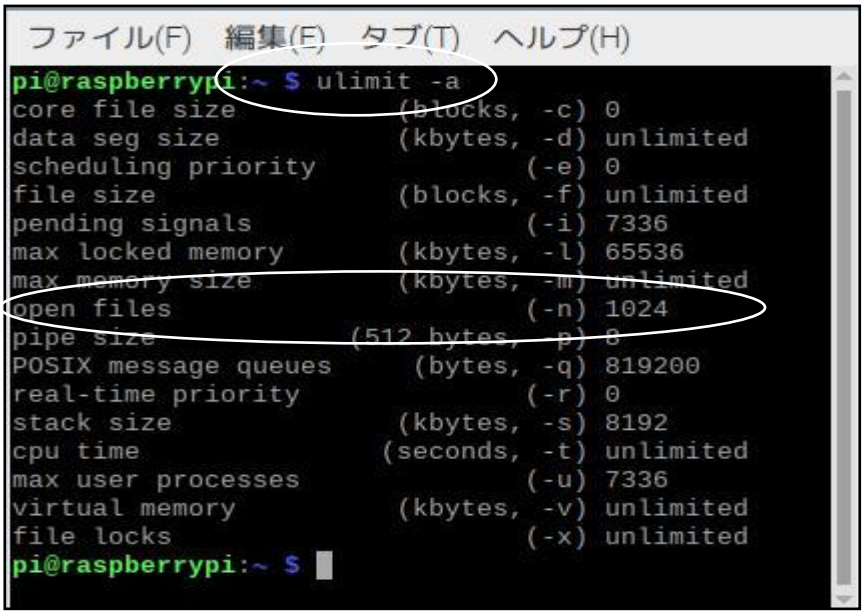
なるほど、理屈上は、５００人のタヌキと同時にリモート会議ができるというわけか。でも同時に音声でも使うから１００から２００人くらいかな。
何となく納得したぞ。キツネ、お前すげえな！

タヌキに褒められると照れるな！
じゃあ、説明を図で表わすと次のようになるよ。

［ファイルディスクリプタ］

プロセスと外部装置との相互作用

・１つのプロセスあたりの出入り口の数（ファイルオープン数）の確認



次は、明示的にファイルディスクリプタを使うプログラムを作ってみようか。Linux のＯＳはＣ言語で作られているので、Ｃ言語のコンパイラは標準でインストールされているので、それを使おう。Ｃ言語は、コンパイラ言語なので、コンパイル（翻訳）という作業が必要になるから忘れないでね。ソース（元になる）プログラム名を mfd.c とするね。作成にはエディッタというプログラムが必要になるよ。オイラは gedit というエディッタを使うよ。

プログラムを作る為のプログラムが必要なのか。複雑だな！

[mfd.c の作成から実行まで]

・mfd.c を gedit で作成し、保存。（mfd.c：実行時はプロセス）

```
#include <stdio.h> /*標準入出力*/
int main()
{
        int fd;
        char buf[10];
        fd = open("data",0);  /*0:標準入力:ファイルﾃﾞｨｽｸﾘﾌﾟﾀ*/
        read(fd,buf,5);    /*fd=3:ファイルﾃﾞｨｽｸﾘﾌﾟﾀ*/
        write(1,buf,5);    /*1:標準出力:ファイルﾃﾞｨｽｸﾘﾌﾟﾀ*/
        close(fd);
}
```

・mfd.c がプログラム内で使う data ファイルを gedit で作成し、保存。

```
abcdefghijklmn
```

・コンパイル作業（端末のコマンドラインで実行）

ただし、mfd.c ソースファイルが自分のディレクトリに保存され、そこがカレントディレクトリになっているものとする。

```
$ gcc  –o  mfd  mfd.c
```

・実行

```
$ sudo ./mfd
```

・結果（５文字読み込み、５文字出力）

```
abcde
```

キツネ、カレントディレクトリって何だ？

ディレクトリは、Windows のフォルダのことです。カレントディレクトリはユーザが今、存在しているフォルダのことです。「ls」というコマンドを実行し、内容が表示される場所がカレントディレクトリになります。そこに mfd.c というファイルがあれば、コンパイルできます。ファイルが存在しません、というメッセージが表示されたら、カレントディレクトリに保存されていない、ということです。

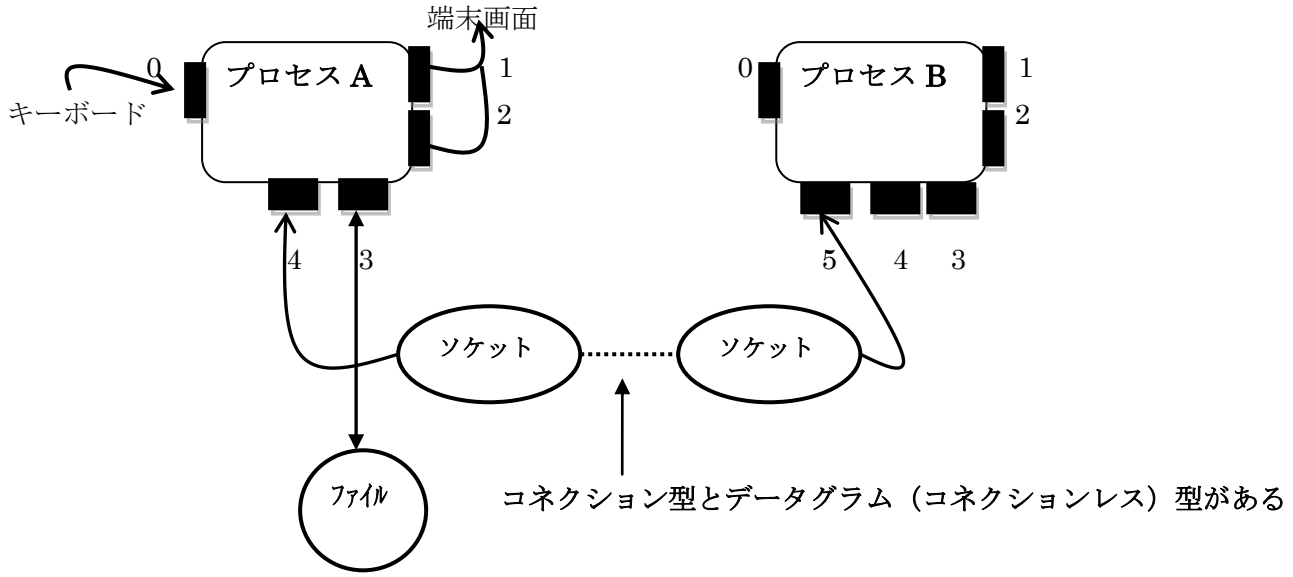
mfd.cというソースファイルの「close(fd); 」の前に「printf(“ファイル番号の表示 %d”,fd); 」という1行を挿入すれば、データファイルの読み込みに使われているファイルディスクリプタの正数値を表示させることができるよ！
タヌキ、やってみてよ。C言語のコンパイルから実行までの練習にもなるよ。

OK、やってみるよ。

最後は、第５話の中心のソケットの話だ。ソケットは OS に存在するプロセスが通信を行う時に利用できるソフトウェアインタフェース（通信仕様）なのだ。最近はブラウザにもプロセス間通信を行う為のインタフェース機能を持たせているので、ブラウザ上でリモート会議やチャットができるようになって便利なのだ。
さて、次にソケットの解説図を書いておくね！

［ソケットとは］

ファイル入出力とプロセス間通信



コネクション型：通信路が確保（電話回線のイメージ）

データグラム（コネクションレス）型：恒常的な結びつきは無い（郵便のイメージ）

クライアント・サーバモデル

サーバ：接続を待つプロセス

サーバプログラム：ソケット接続を受け付ける役目を担当するプログラム

クライアント：接続を要求するプロセス

クライアントプログラム：サーバにソケットを接続するプログラム

上記の図でプロセス A とプロセス B は、1 つのパソコンの中で実行されていても OK だし、ネットワークで接続されていれば、プロセス A が家のパソコンで実行され、プロセス B が職場のパソコンで実行されている、というリモートの状態でも成り立ちます。
じゃあ、第 6 話では、実際にチャットのプログラムを作ってみるか・・・・・。

第 6 話へ続く