プログラミング環境 第8回

久野 靖 *

1991.11.19

11 計算機ネットワーク

11.1 計算機ネットワークとその目的

計算機ネットワークとは、何のことだと思いますか? 物理的には、複数の (たぶん多数の) 計算機システムが、互いに通信できるように (つまりデータがやりとりできるように) 接続されたもの、ということになろう。しかし、例えば (皆様がよくやるように)PC に端末ソフトを入れ、電話線などを通して計算機に接続してもそれは普通ネットワークとは言わない。つまり、ネットワークと

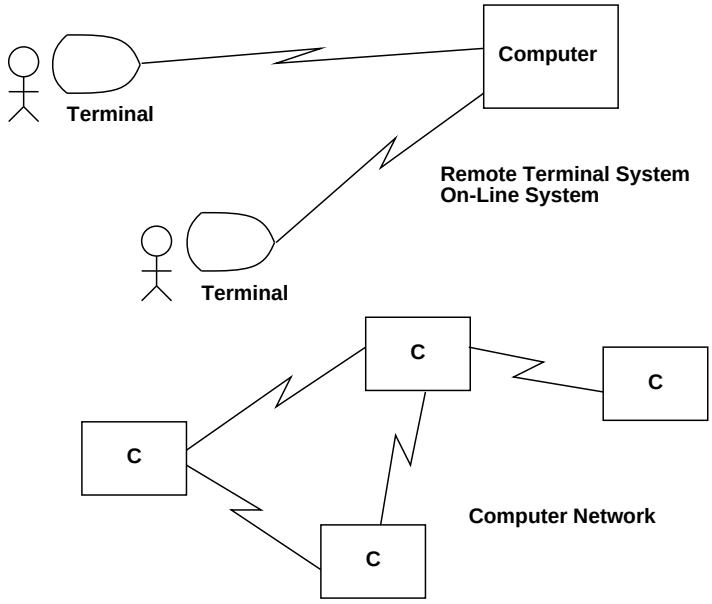

図 1: リモート端末システムとネットワークシステム

言うからには

- 接続された各計算機が
- 主従関係なしに、それぞれ自律的に動作し、
- 接続された他の計算機と協調動作しながら
- 有用な仕事を進めていく

*筑波大学経営システム科学専攻

ようなものを指すのが普通である。ということは、あるシステムが計算機ネットワークと呼ぶのに値するかどうかは、物理的な形態だけでなく、ソフトを含めてどのように運用されているかまで見ないと分からない。例えば電話線で PC と他の計算機をつないだシステムでも、PC 上で自律的に動作するソフトが動いていて、向こう側の計算機と協調して仕事をするようなシステムであれば立派な計算機ネットワークである。

では次に、**何のために**計算機ネットワークを作るのだと思いますか?(間) 「遠くの計算機システムに入っているデータを見たり操作したりするため」というのだったら×である (なぜなら、リモート端末システムでもそれはできるから)。Tanenbaum 先生によれば、[1] 計算機ネットワークを構成する目的としては次のようなものがある:

1. 資源の共有 — 例えばある計算機に入っているデータを、その計算機で処理を行なう際だけでなく、別の計算機で処理を行なう際にも利用できるようにする、ある計算機の CPU 能力が不足したらデータの一部を別の計算機で処理する、など。

2. 信頼性 — 1 台の計算機であればそれが止まってしまえば処理は停止してしまうが、複数台の計算機をネットワークで結合したものであれば、1 台が壊れても残りで処理を進めて行くようにできる。

3. 経済性 — 大きな計算機 1 台で何もかもやらせるより、複数の PC や WS をネットワーク結合したシステムの方がコスト的に安い。

4. 段階的成長 — 1 台の計算機で能力が不足したら、より大きいマシンにリプレースするしかないが、ネットワークシステムなら何台かマシンを追加する形で成長して行ける。

5. 通信媒体 — 距離的に離れたシステムどうしを接続することにより、新しいタイプの応用が可能になる。

もちろん、どの目的を主とするかによって、ネットワークの形態は大幅に異なる。例えば信頼性や経済性のために 1 台の計算機に代えてネットワークシステムを構成するのなら、その各計算機間の距離は比較的近く、それらの間は高速な通信方式で結ばれることになるだろう (LAN – Local Area Network)。一方、地域的に離れたところにあるデータの共有や個人間の通信が目的なら、そのネットワークは長い距離を結ぶものになるわけである (WAN – Wide Area Network)。

11.2 計算機ネットワークの通信媒体

ネットワークを構成するには、当然何らかの方法で計算機と計算機を結ぶ必要がある。計算機のコネクタにつなげるのは銅線でできたケーブルだが、その先が**論理的に**どうなっているかによって次のように分類できる。

1. 2 点間 (point-to-point) リンク: そのケーブルの先が直接別の計算機のコネクタにつながっている。当然そのケーブルへ出た信号は相手の計算機にしか着かない。

2. 放送型 (broadcast) リンク: そのケーブルには複数の計算機がつながり、それら全てが同時に信号を受ける。

これらは物理的な媒体の構成と深く関わっているし、一方でそれぞれについて様々な物理媒体が利用できる。例えば 2 点間リンクとしては次のような選択肢がある。

a. 銅線。RS-232C 規格などの信号ケーブルを自分で敷設することもできるし、既設の電話線などを利用することもできる。コストと速度に応じて多数の選択肢がある。

[1] Tanenbaum, Computer Networks 2nd ed., Prentice-Hall, 1988.

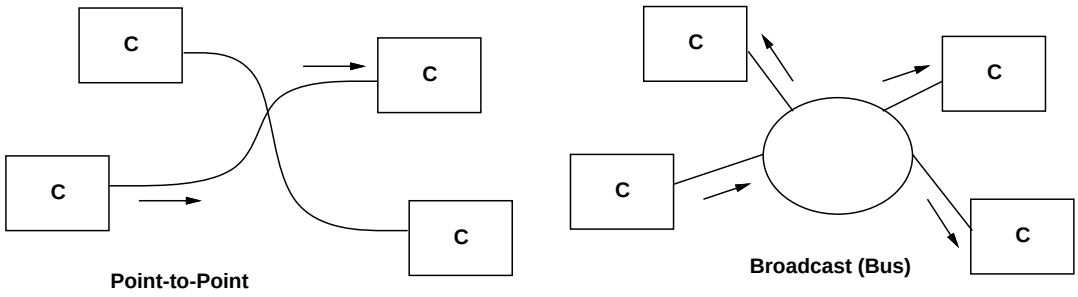

図 2: 2 点間リンクと放送 (バス) 型リンク

b. 光ファイバ。高速な 2 点間リンクを張ることができる。

c. マイクロ派や赤外線。道路をまたいだり、建物が違うなどしてケーブルを敷設するのが困難なときに有効。

一方、放送型の媒体には次のようなものがある。

d. 銅線。ただし、2 点間に張るのでなく、あちこちケーブルを引き回して、多数の計算機をつなぎ込む。同軸ケーブルを使用する Ethenet などがこの代表であり、LAN の標準的媒体でもある。

e. 通信衛星。パラボラアンテナさえ建てればどんな僻地でも使え、通信容量も多い (通信時間はやや掛かる)。

ただし、WAN の場合には自前で長距離のケーブルを張ったり衛星を持てる訳がなく、NTT など通信業者から回線を買うのが普通である。その場合、専用線を買えばそれは 2 点間にケーブルを張ったのと同じである (通信業者の中でどうやって通信を処理しているかは別問題)。また、衛星のチャネルを買えばそれは e. と同じわけである。

さて、計算機の側からみるとネットワーク装置も他の入出力装置とさして変わりはない。つまり、媒体そのものはコントローラ (通信制御装置、Communication Control Processor などと呼ばれる) が制御し、CPU はそれに対して「読め」とか「書け」とか指令を出すだけである。この読み書きの単位をデータフレームと呼ぶ。データフレームの大きさは媒体の性質によって許される範囲が異なり、1 バイトのフレームしか使えないこともあれば、数千バイトまで許されるような媒体もある。2 点間リンクの場合にはある CPU が送り出したデータフレームはつながっている相手

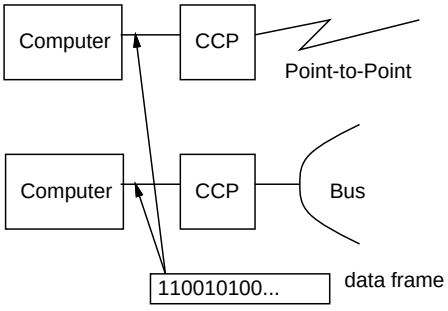

図 3: CCP とデータフレーム

の CCP(一意的に決まる!) に受けとられ、相手の CPU が CCP に「データがきたらここに格納せよ」という指令を予め出しておくことで指定された主記憶上の場所におさまる。一方、バス型の

ネットワークでは原理的には送られたデータフレームはそのネットワークに物理的につながっている全ての CCP によって受けとられるが、いつもそうだと困るのでデータフレームの決まった場所に「相手の番号 (アドレス)」を入れておき、受ける CCP の方では自分あて以外のフレームは無視するようになっているのが普通である。

11.3 ネットワークソフトウェアとプロトコル

さて、あなたがネットワークを構成するのに必要なハードウェア (計算機、媒体、通信制御装置、etc.) を入手したとする。もちろん、「ソフトなければただの箱」である。では、どんなソフトウェアが必要だと思うか? (間)

では、Top-Down に考えてみよう。例えばあなたが遠くにある計算機システムから自分の計算機システムに必要なデータの入ったファイルを転送したいものとする。すると....

a. まず、あなたは相手計算機の名前 (アドレス?) とファイル名を指定して「ファイル転送」という指令を起動する。すると、その指令は相手計算機の「ファイル転送機能」と接続して必要な情報を知らせ、相手計算機の「ファイル転送機能」は指定されたファイルの中身を最初から 1 バイトずつ順番に送ってくる。こちらの指令は送られてきた中身をこちらのファイルに順番に格納していく。

b. さて、中身を順番に送ってくる、と言ったが、送っている途中でデータの一部が欠落したり、順番が入れ代わったりしては困ることになる。そこで、こちらから 1 バイトずつ送ったものが向うでも確かに欠落なく、その順番で受けとられるように制御する、という機能が必要である。

c. さらに、もし相手計算機が USA の計算機だったりすると、各データはまず向うの計算機から日米の橋渡しになっている計算機へやってきて、橋渡しに衛星を経由し、こっちがわの橋渡し計算機から出て自分の計算機に着く (実際にはもっとずっと沢山の中継点がある) ことになるので、そういう通過経路を制御する、という機能が必要である。

d. そして、ある中継点 (または出発点) から次の中継点 (または最終目的地) への転送に当たっては、当然 CCP に命令を出してデータを転送させる、という制御が必要である。

e. もちろん、一番下では CCP と媒体が個々のデータフレームを運ぶわけである。

こうしてみると、ネットワークのソフトウェアというのは一番上の我々が利用したい操作 (ファイルを転送する、etc.) から一番下のハードウェアまで多数の階層が積み重なって作られていることが分かる。ISO ではネットワークの標準規格 (OSI — Open System Interconnect) を制定するに当たって、このような階層化の標準モデルを提案している。これは 7 つの階層から成り、OSI 参照モデルとか 7 層モデルとか呼ばれている。その構成を図 4 に示す。ここで左側に先の各機能がどこに入るか、を示した。a. については OSI モデルではさらに細かく分けられているが、そこまで細かく考える必要はないと思うので先の例ではまとめたものである。

ところで、このモデルで同じレベル (層) にある機能どうしがやりとりするのに使う約束ごとを「プロトコル」と呼ぶ。例えば先のファイル転送機能では「まず相手側にファイル名の長さを送り、続いてファイル名を送る。すると相手側はファイルの長さを送り、続いてその長さぶんだけファイル本体を送って終る。もし指定のファイルがなければファイル長さとして-1 を送る。」という約束ごと (つまりファイル転送プロトコル、ですね) を使っているかも知れない。

また、各層が上の層に提供する機能の集まりを「インタフェース」と呼ぶ。例えば、伝達 (Transport) 層は「ホストアドレスとサービス名 (例えば “FileTransfer”) を指定すると、そのホストの指定されたサービスを行なっている相手を探し、接続番号を返す」「接続番号、バイト数、バッファ

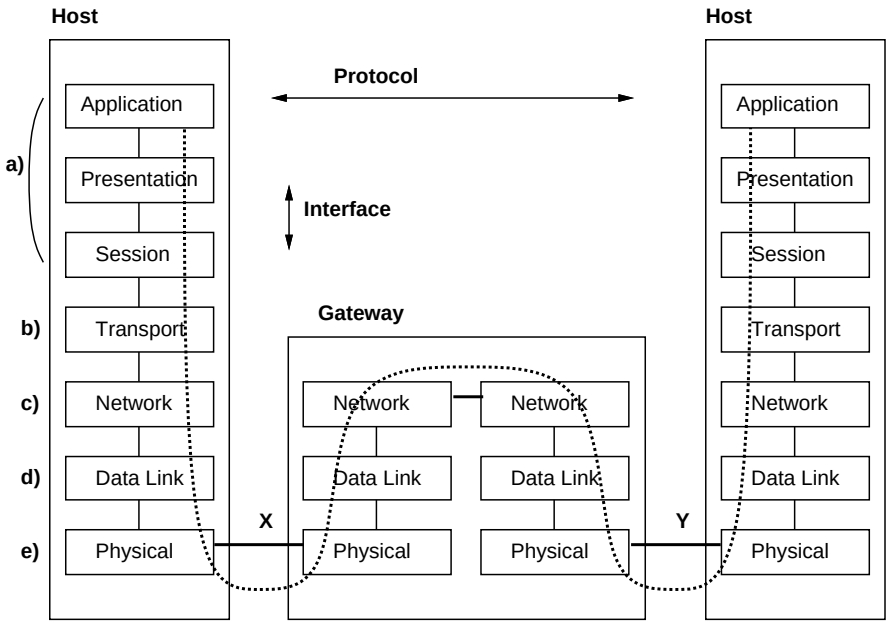

図 4: OSI 参照モデル

アドレス、読み/書きの別、を指定すると、その接続に対して指定バイト数の転送を行なう」などのインタフェースを持つのが普通である。

各層は自分が提供する機能を実現するのに、相手ホストの対になっている層とプロトコルを用いてやりとりするが、そのやりとりのためには自分の下にある層に対してインタフェースを通じて頼むことを行なうわけである。例えば伝達層は相手ホストのトランスポート層から送られてくるデータに抜けがないことをチェックするために、「各データには 1 から始まる 16 ビットの 1 連番号を先頭に付加する」という「プロトコルを使うかも知れない。一方、データを受けとるには例えばネットワーク層の「自分あてに来たデータの 1 かたまりを指定したバッファに入れ、同時に送ってきた相手のアドレスも通知する」という「インタフェース」を利用することになるだろう。言い替えれば、各層は直接相手の同レベルの層と話している「つもりになっている」が、それは幻想で実際のデータは下へ下へと送られ、物理層で相手へ渡って、そこから上へ上へと上ってくる、という大迂回を生じているのであった。もちろん、物理的な媒体を通ることなしに他のホストへデータが行けるわけがないので、当たり前ではある。

ところで、先に出てきた「中継点」ではどんなことが起きているだろう。当然、送っているファイルを一旦全部中継点に蓄えてしまう、というのは賢くない (そんな余分なディスクがどこにある?)。一方、中継点では「米から衛星で来たデータ 1 かたまりを見ると、それは GSSM 宛である。では東京方面への専用線に送り出そう」という制御は必要である。だから中継点ではネットワーク層が動いている必要はある。ときに、中継点の両側では物理層、データリンク層はそれぞれ別ものであることに注意。というのは X は専用線、Y は衛星なのだからその媒体も制御のしかたも全然異なるからである。というわけで、中継点まで考えるとデータそのものの流れは点線のようになるわけである。

11.4 プロトコル群、TCP/IP

さて、ネットワークの目的は複数の計算機間で互いに通信することにあるわけだが、そのためには通信し合うモジュールどうしでプロトコルが同じでないと通信ができない (あたりまえ)。そして、ある層だけ共通のプロトコルだがその下は全然別、というのは不自然なので、結局図 47 に

あるような各階層にわたってそれぞれが共通のプロトコルを喋る計算機間で通信する、というのが普通なわけである。そのような**プロトコル群**としては、例えば次のようなものがある。

- SNA プロトコル群。IBM のシステムで使われている。もちろん、IBM 互換各社でもそれぞれ FNA とか HNA とかいうのをやっている。
- Decnet プロトコル群。DEC のシステムで使われている。これもわりと古くからある。
- Internet プロトコル群。TCP/IP とも呼ばれている。もともとは米国 ARPANET というネットワークの実験 (広域ネットワークの元祖) に起源するが、4.2bsd Unix の標準付属品となったので、その結果 Unix の広まりとともに世の中で多く使われるようになった。Unix そのものと同様、多くのベンダーで共通に使われているのが特徴。
- MAP/TOP。GM が最初に開発したもので、Factory Automation / Office Automation のため各種機器を接続することを目的としている。その方面の分野では多く使われるようになっている。
- OSI。これは前述のように ISO が規格として標準化を進めているもので、まだ開発途上にあるが、今後採用が増えるものと思われている。

その他にも Macintosh の Appletalk など、PC/LAN レベルのプロトコル群は多数ある。

さて、抽象的なお話はそれくらいにして、具体的な話へ進もう。我々のところではプロトコル群として TCP/IP を採用しているので、以下では当然これを題材に取り上げることにする。プロトコル群は多数のプロトコルの集まりだと先に述べたが、TCP/IP の場合は図 5 のような構成になっている。

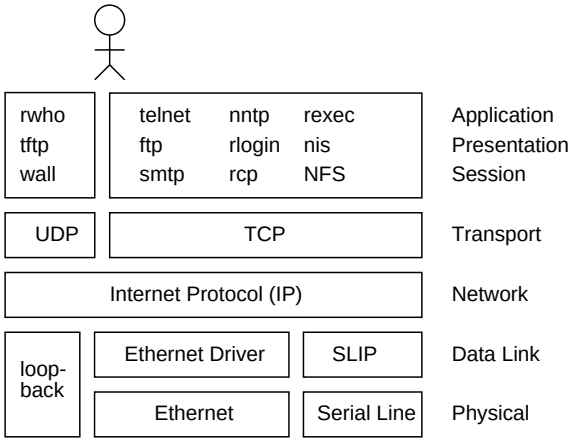

図 5: TCP/IP のプロトコル構成

まず、我々の手元では LAN の媒体として Ethernet を使用しているので (マシンの裏を這っている黒いケーブルや廊下を這っている色のついたケーブルがそうです)、物理層がこれで、それを制御するデータリンク層もそのためのものが OS に組み込まれている。また、遠隔地との接続は専用線を使っていて、その制御は SLIP と呼ばれるモジュールによっている。ところで、**1 つのマシンの上で**あるプロセスとあるプロセスが通信したい場合もありますよね? Ethenet や専用線はそこを通ると他へ行ってしまうので、自分あての通信には使えない。そこで、ループバックという特別なデバイスがあるつもりになって、そこへデータを流すと自分自身に返ってくる、というのもおまけとして用意してある。ループバックの場合には「媒体」は要らないので物理層はからっぽである。

さて、それらの上にあるネットワーク層プロトコルは IP と呼ばれ、これから上ではどのような媒体を使用しているかに関わらず同様の機能が提供される。その上の伝達層には UDP(User Datagram Protocol) と TCP(Transmission Control Protocol) という 2 種類のプロトコルがあり、それぞれ違う種類のサービスを提供している。その上は各種応用ごとに様々なプロトコルがあるが、代表的なものを挙げると次のようなものがある。

- telnet — 他の計算機に login するためのプロトコル。
- rlogin — 同じだが、Unix マシンどうし専用。その分機能も多い。
- ftp — 他の計算機との間でのファイル転送プロトコル。
- rcp — 同じく、ただし Unix 専用でより簡便。
- smtp — 電子メール搬送用プロトコル。
- nntp — 電子ニュース搬送用プロトコル。
- NFS — 他のマシンのファイルシステムを読み書きする。
- rwho — 誰がどのマシンにいるかの情報を交換し合う。
- rwall — マシン停止などのお知らせメッセージ伝達用。

では、以下で物理層から始めてもう少し具体的に見てみよう。

11.5 物理層とデータリンク層 — Ethernet

我々の所で LAN のために使用している媒体は Ethernet である。その原理は次のようなものである。まず、1 本の同軸ケーブルに多数の計算機を接続する。各接続点では、計算機からケーブルに信号 (というのは、電圧の変化ですね) を送り出すことと、ケーブル上の信号を感知することができる。ケーブルは 1 本だから、誰かがケーブル上に信号を出せば (つまりデータフレームを送信すると)、それは他の全員に感知できる (つまりデータフレームが受信できる)。もちろん、実際には特定の相手だけに送信したいのだから、12.2 節で説明したようにフレームには相手のアドレスが入っている。[2]

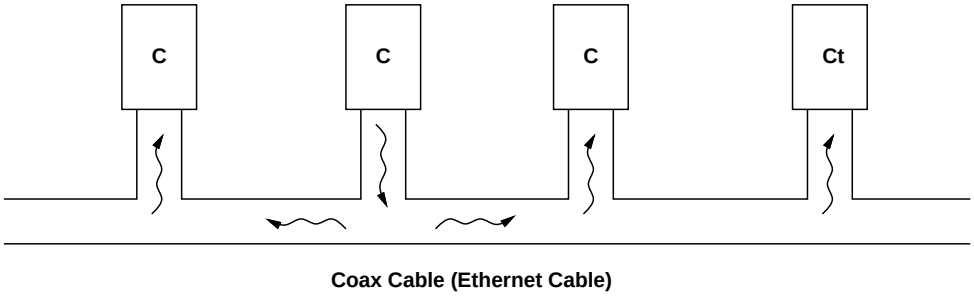

図 6: Ethernet の原理

ただここで問題なのは、たまたま 2 台の計算機が同時に信号を送ろうとすると、信号が同軸ケーブル上でぶつかって (衝突)、でたらめな値になる、ということである。そこでどうするかという

[2] このアドレスは 48 ビットの数値で、世の中にある全ての Ethernet 機器は別のアドレスになるように、上 24 ビットが製造業者固有番号、その下が業者内部の固有番号にしてある。だからどっかで買ってきた 2 つの Ethenet 機器をつなげたらアドレスがおんなじで混乱する、という心配はない。

と、送る計算機は送るのと同時にケーブル上の信号の感知も行ない、衝突が起きたのが分かったら送るのを止めて**ランダムな時間だけ**待ち、改めて送り始める。2 つの計算機上の乱数が一致することはまずないから、今度は衝突せずに送れる可能性が高いわけである。もちろん、Ethernet がひどく混んでいる時などは再送のときまた衝突が起きることもあるが、その時はランダム時間を計算してその倍待ってから再試行する。また衝突したらさらに倍... とする。こうすると、衝突が起きている限りは各計算機の送信確率は半分ずつに減っていくから、確率的に見ていつかは必ず送れるようになるわけである。

11.6 ネットワーク層 — IP と IP アドレス

ネットワーク層の役割は「〜と通信したい」と言われたら、そのデータをできる限りうまく指定の相手に向かって送ることである。では、その「指定」はどのようにしたら行なえるだろうか? 例えば郵便であれば住所と氏名、ということになるが、計算機ネットワークの場合にはいちいちそういう長い文字列を引きずって歩きたくない。そこで IP では 32 ビット (4 バイト) の数値をアドレスとして使用している。このうち、上の何バイトかはネットワーク番号、残りがホスト番号、と言う風に分かれている。例えば我々[3]のところは上 3 バイトがネットワーク番号であり、赤と黄色の Ethernet ケーブルがそれぞれ

```
192.50.17.
192.50.18.
```

という番号を持っている (この書き方は、各バイトをそれぞれ 10 進数だと思って読んで、それを . でつなげる、という流儀で、IP アドレスの表記は伝統的にこの流儀によっている)。それぞれにつながっているホストのアドレスはネットワーク部分は同じで最後のホスト部分で区別される。例えば

```
192.50.17.1    sma
192.50.17.3    smra
```

といった具合である。

これからも分かるように、ネットワーク番号というのはそれぞれ 1 つの物理的なネットワーク (例えば Ethenet ケーブル) に対応している。そして、世の中には同じネットワークアドレスは 2 つとないように割り当てを (人手で) 管理しているので、IP の世界では世界中どこからでも `192.50.17.1` といえばうちの `sma`、ということになるわけである。

なお、より大きな組織でホストを多数持っている場合には、ネットワーク部分が 1 バイトないし 2 バイトで残りがホスト部分になる。これらは最初の 1 バイトを見れば分かるようになっていて、

```
1〜127      ネットワーク部 1 バイト
128〜131    ネットワーク部 2 バイト
132〜       ネットワーク部 3 バイト
```

となっている。[4]

各サイトで具体的なホスト名とアドレスの対応表を調べる指令は `ypcat hosts` である。しかしこれは

```
% ypcat hosts
192.50.17.150   smac00  # room 4E408
192.50.17.121   smpc21  # terano
...
```

[3] というのは、筆者が所属する筑波大学大塚地区経営システム科学専攻の話であり、東工大では当然それなりの違いがあります。

[4] 実は東工大は外部的にはネットワーク 2 バイトの部に属するが、内部では多数の Ethernet ケーブルをゲートウェイで接続しているので、内部的にはネットワーク部 3 バイト (厳密には 2 バイトと 7 ビット) として扱うようになっている。こういうのをサブネットという。

のようにひどく沢山、しかもデタラメな順に出てくるのでやりにくいのだが、ファイルに保存してソートしてから見るとよい。なお、ここに出てくるのは「名前だけですぐに検索できる」ものだけで、その他にもドメイン形式 (「sma.gssm.otsuka.tsukuba.ac.jp」のように階層構造になった長い名前) が使える場合もあるし、使えなくても何らかの方法で数字表記を調べればそれでアクセスできるようになっているサイトもある。

ところで、実は sma は黄色のケーブルにもつながっているが、192.50.17. というのは赤色のケーブルの方のアドレスだから、黄色のケーブルにつながっている部分 (これを Unix 用語でネットワークインタフェースと呼ぶ。先のプロトコル層のインタフェースと混同しないでね) には

```
192.50.18.20    sma-gw
```

という別のアドレスがつけてある。この辺りは IP のちょっと分かりにくい部分ではある。

この辺の情報を調べたい場合には netstat -i、および ifconfig という指令を使う。上記 sma でこれをやって見たところを示す。

```
% netstat -i
Name  Mtu   Net/Dest    Address      Ipkts   Ierrs Opkts   Oerrs Collis Queue
ie0   1500  smr-net     sma          32544167 1218  33319129 18887 135    0
ie1   1500  sme-net     sma-gw       19806065 1582  45885580 5     570404 0
lo0   1536  loopback    localhost    4091620 0     4091620 0     0      0
% ifconfig ie0
ie0: flags=63<UP,BROADCAST,NOTRAILERS,RUNNING>
        inet 192.50.17.1 netmask ffffff00 broadcast 192.50.17.255
% ifconfig ie1
ie1: flags=43<UP,BROADCAST,RUNNING>
        inet 192.50.18.20 netmask ffffff00 broadcast 192.50.18.255
```

つまり、netstat -i はそのマシンがどんなインタフェースを持つか、また各インタフェースで合計何個パケットをやりとりしたか、などを知ることができる。また ifconfig ではインタフェースのアドレスや状態が分かる。さらに、ある特定のインタフェースでのパケット数を毎秒繰り返し表示させるには

```
netstat -I インタフェース名 1
```

でできる。例えば次のような具合である。

```
% netstat -I ie1 1
    input   (ie1)     output           input  (Total)    output
packets errs  packets errs  colls  packets errs  packets errs  colls
19806649 1582  45886380 5     570404 56446851 2800  83303260 18892 570539
0       0     0       0     0      15      0     17      0     0
0       0     0       0     0      1       0     2       0     0
0       0     0       0     0      1       0     2       0     0
0       0     0       0     0      2       0     2       0     0
0       0     0       0     0      1       0     2       0     0
0       0     0       0     0      2       0     3       0     0
0       0     0       0     0      38      0     41      0     0
^C
%
```

1 行目はこれまでの総計だからひどく大きな値になっているが、あとは毎秒の累計であるので、これを動かしながらファイル転送などやるとパケットが定常的に飛んでいる状態が観察できる。

さて、それではこのアドレスを使ってどうやって「うまく相手に送る」のかを見てみよう。IP ではデータのかたまり (パケット、と呼ぶ) ごとにその頭の部分 (ヘッダ、と呼ぶ) に送り先、送り元の IP アドレス、その他の制御情報を入れておく。そこで、あるホスト A に図 7 のような具合に IP パケットがやってきたものとする。このパケットの行き先を見ると、192.50.21.31 とある。すると、ネットワーク番号は 192.50.21. ということになる。ときに、このホストは 3 つネットワークインタフェースを持っていて、その中に 192.50.21.1 というのがある。ということは、こ

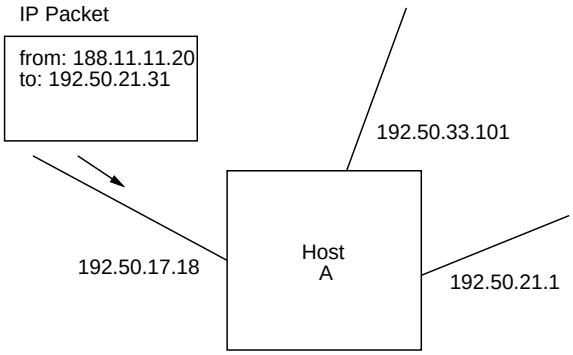

図 7: IP の経路制御 (1)

のインタフェースは 192.50.21. のネットワークにつながっているのだから、ここへパケットを送り出せば目指すホストに行きつくに違いない! [5]

このように、IP の体系では「ネットワーク番号が同じ」ならば「隣接している」ことになるので、ホストがどこにあるかを個別に勘案しなくてもネットワーク番号だけ見て経路制御を行なうことができる。もちろん、上の例はたまたま目的地の「すぐ手前の」ホストにパケットが来たからこれで済むのだが、一般の場合には「どのネットワーク番号ならどっちへ送る」という表 (ルーティングテーブル) を持っている必要がある。例えば図 8 のように、ホスト B にさっきと同じよう

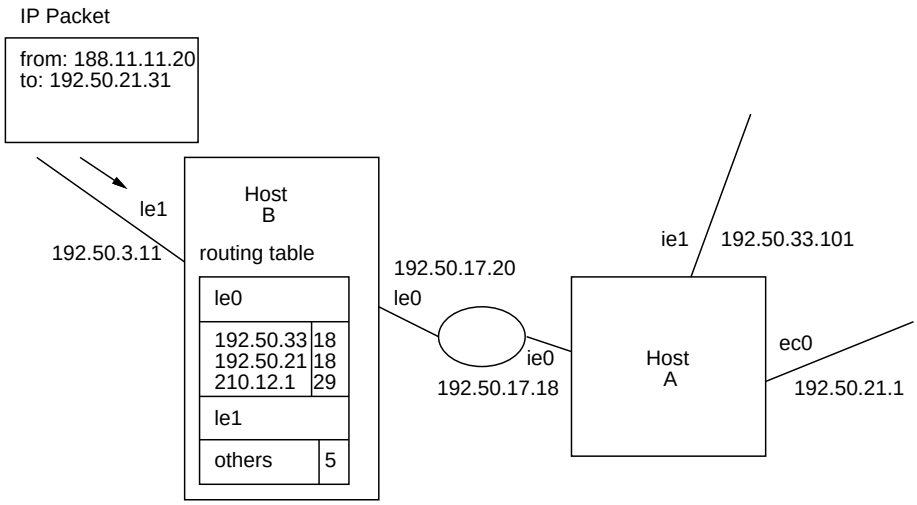

図 8: IP の経路制御 (2)

なパケットが来た場合にはテーブルを見てインタフェース le0 経由でホスト A へ送ればよい、とあるのでその通りにする (と、A はさっきと同様にして送ることができる) わけである。当然、これらのテーブルはどうやって用意されるのか、という質問が出ると思うが、答えは 2 通りある。1 つはシステム管理者が手で用意する、というもので、これは比較的小規模のサイトで行なう方法である。もう 1 つは、隣合うホストどうしでテーブルの情報を交換することを繰り返して、どちらへ行けばどのネットワークへ通じるかの情報を自動的に伝達する方法で、これは多数のネットワークの接続経路に当たるホストで用いられる。

[5]実際はネットワークが Ethernet であれば、Ethernet アドレスと IP アドレスの対応表が別にあって、それを見て目的地にパケットを転送するだけのことである。

ところで、B のルーティングテーブルには「192.50.33, 192.50.21, 210.12.1 については le0、それ以外は le1」と書いてある。これはちょうど、函館駅のルーティングテーブル (?) に「千歳線と室蘭本線と宗谷本線と...(以下北海道各線の名前) 行きは函館本線経由、その他は全部津軽海峡線経由」と書いてあるようなもので、表の大きさを減らす効果がある。実際、我々のサイトでは外の世界への出口は専用線が 1 つあるだけなので、我々のサイト以外のアドレスはすべて「その他」としてこの専用線へ送っている。

ルーティングテーブル情報は netstat -r という指令で調べることができる。

```
% netstat -r
Routing tables
Destination          Gateway              Flags    Refcnt Use        Interface
localhost            localhost            UH       1      2554       lo0
default              smc                  UG       0      129578     ie0
smr-net              sma                  U        57     28400864   ie0
sme-net              sma-gw               U        2      21848955   ie1
```

これはつまり、smr-net(192.50.17.) へは直接つながっていて ie0 インタフェースへ出ればよく、また sme-net(192.50.18.) へはやはり直接つながっていて ie1 から出ればよいが、それ以外 (うちのサイト外) へ行く場合にはすべて smc というマシンへ ie0 経由で行くべし、と書いてあるわけである。

さて、このような経路情報を使ってパケットが転送されていくわけだが、具体的にあるホストから別のホストへ至る経路を教えてくれるソフトがあると便利である。traceroute というのがそれで、例えば sma からとあるマシンへの経路を追跡してみた結果が次のものである。

```
% traceroute nsd.is.titec.ac.jp
traceroute to nsd.is.titech.ac.jp (131.112.40.4), 30 hops max, 40 byte packets
 1  smc.gssm.otsuka.tsukuba.ac.jp (192.50.17.50)  4 ms  3 ms  2 ms
 2  * * *
 3  wnoc-tyo.wide.ad.jp (133.4.3.2)  261 ms  261 ms  267 ms
 4  nirvana.cs.titech.ac.jp (131.112.16.14)  296 ms  293 ms  297 ms
 5  titcs-gw.cs.titech.ac.jp (131.112.16.254)  326 ms  289 ms  297 ms
 6  * * *
 7  nsd.is.titech.ac.jp (131.112.40.4)  301 ms  295 ms  327 ms
```

このように traceroute はパケットが通過するゲートウェイの一覧を、そこまでパケットが到達するのに掛かる時間とともに順に表示してくれる。なお、「* * *」と出ているのは OS のバージョンが古いその他の理由で「見えない」(traceroute に応答してくれない) ゲートウェイである。なお、筆者はこの経路について一応知っているのでそれを図 9 に示しておく。

11.7 伝達層 — TCP と UDP

伝達層のプロトコルには TCP と UDP の 2 つがある、と述べたが、その違いについてまず説明しておこう。まず、TCP は先のファイル転送の例に出てきたように、接続元と相手先の間に仮想的な「回線」を用意する機能を持つ。一旦「回線」がつながれば、片方がそこに送り込んだデータはその順番で重複や損失なく他方が受けとることが保証される。このようなサービスを (本当に線を敷設するわけではないから)「仮想回線」と呼ぶ。一方、UDP は送り先アドレスと送りたいデータを指定すると、そのアドレスにデータを転送する、という機能のみを持つ。ただし、途中の経路に障害があったり混雑があった場合にはそのデータは着かないかも知れないが、ともかくできる範囲で努力する (best effort)。その分 TCP よりオーバーヘッドは小さい。このようなサービスを「データグラム」と呼ぶ。現実の世界に例えると、仮想回線は電話、データグラムは小包に相当する、とよく言われる。[6]

[6]「着かないかも知れない」サービスが役に立つのか、と思われるかも知れないが、たとえば音声をディジタル転送して電話のように使う場合には、時々データが落ちてノイズが聞こえても、時間的遅れがない方が望ましいからデータグラムが向いている。要は適材適所である。

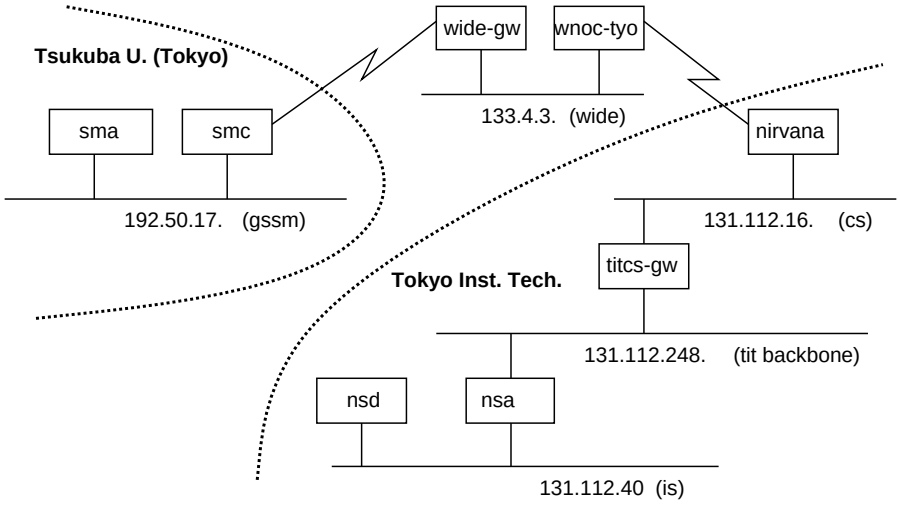

図 9: sma から nsd への経路

ところで、IP アドレスはあくまでもホスト単位につくのであった。実際には各ホストで沢山のプログラムが動いているだろうから、ネットワークを使って仕事をするには「どのホストの、どのプログラム」という指定ができないと役に立たない。TCP/IP ではこの「どのプログラム」というのも IP アドレスと同様の考えに基づき、16 ビットの数値 (ポート番号) で表している。この、ポート番号の機能を提供するのも UDP や TCP の重要な役割である。ポート番号には、予めサービスの種類によって決められているものと、プログラムを動かしていて必要になった時適当に割り当ててもらうものの 2 種類がある。予め決められているポート番号の情報は ypcat services で次のように見ることができる。

```
% ypcat services
skkserv         1178/tcp
phone           1167/udp
new-rwho        550/udp         new-who         # experimental
talk            517/udp
...
```

また、現在接続中の TCP ポートの一覧は netstat -f inet により次のように見ることができる。

```
% netstat -f inet
Active Internet connections
Proto Recv-Q Send-Q  Local Address      Foreign Address        (state)
tcp        0      0  sma.2180          sma.669                TIME_WAIT
tcp        0      0  sma.22273         utogw.1603             ESTABLISHED
tcp        0      0  sma.22273         sma.2177               ESTABLISHED
tcp        0      0  sma.2177          sma.22273              ESTABLISHED
...
```

11.8 各種応用プロトコルとネットワークサービス

伝達層より上の層については、ネットワーク伝送そのものは TCP や UDP に頼んでしまえるため、各プロトコルそのものは比較的単純である。たとえばリモートログイン (ネットワーク経由で他のマシンに login する) プロトコルの場合、TCP の接続を行なってから、まず最初に login するユーザの名前を転送するが、その後パスワードを答えて入ってしまえば、あとはキーボードから打ち込んだものを相手先に転送し、向うのプログラムの出力をこちらへ転送してくるだけなので、ほとんど TCP の機能そのものに過ぎない。というわけで、以下では応用プロトコルというよりは、Unix の場合を例に取り、どのようなネットワークサービスがあるか、について紹介していく。

11.8.1 遠隔ログイン — telnet/rlogin

最初に述べたように、計算機ネットワーク以前から、手元の端末から遠隔地にある計算機に通信回線を通じて接続する、という利用形態が多く存在した。ので、ネットワーク時代になっても同様に「手元の計算機から向うの計算機に接続する」ことがまず求められた。この場合、手元の計算機はそれ以前の遠隔端末の「まねをする」わけなので、この機能を「ネットワーク仮想端末(Network Vritural Terminal、NVT)」などと呼ぶこともある。

我々の Unix では仮想端末ソフトとして Internet 文明共通の telnet と Unix 固有に特化した rlogin の 2 つが存在する。これらの使い方は

```
telnet  ホスト名
telnet  IP アドレス
rlogin  ホスト名
```

である。telnet の場合は、手元のホスト名データベースにアドレスが載っていないホストにも接続できるようにするため、IP アドレスでも相手が指定できる (例えば telnet 192.50.17.1 のように)。いずれの方法でも接続ができると普通に login プロンプトが出て、ユーザ名とパスワードを入れると接続できる。接続は向うのホストから exit すると切れるが、それ以外にも^] を押すと telnet の指令モードに入り、そこで「q[ret]」とやってこちらから切ることもできる。

一方、rlogin は Unix システム間専用であるが、まずこちらのユーザ名を自動的に向うに転送するので、自分のユーザ名を打ち込まずに済む。また、管理者の設定によって一群のホストどうしの間ではパスワード問い合わせを省略するようにもできる (うちの専攻内ではそうなっている)。その他、端末の種別も自動的に設定される (telnet では接続した後に端末タイプを設定してやらないと画面制御を使うプログラムが動かせないのでやや不便である)。従って、Unix システム間で行き来する場合には rlogin を使うのが便利である。

11.8.2 ファイル転送 — ftp/rcp

遠隔ログインとならんで多くの需要があるサービスは、やはりファイル転送であろう。これも遠隔ログインと同様、Internet 文明共通の ftp と Unix 固有に特化した rcp の 2 つが存在する。これらの使い方は

```
ftp  ホスト名
ftp  IP アドレス
rcp  ホスト名:パス名  ホスト名:パス名
```

である。ftp では相手ホストに接続した後、ユーザ名とパスワードを聞いてくる。これらを正しく入れると ftp>とプロンプトが出てくるので、次のような指令を使ってファイルを転送する。

```
cd  パス名       --- 「向こう側で」現在位置を移動
lcd パス名       --- 「こちら側で」  〃
type text        --- テキスト転送モードにする
type binary      --- バイナリ転送モードにする
get ファイル名   --- 向うからこちらにファイルを転送
put ファイル名   --- こちらから向うにファイルを転送
bye              --- ftp を終る
```

一応例を示そう。

```
% ftp smc                    ←接続先を指定して開始。
Connected to smc.
220 smc FTP server (SunOS 4.0) ready.
Name (smc:kuno):             ←向うでのユーザ名 (同じなら [RET])。
331 Password required for kuno.
Password:                    ←パスワード (常に必要)。
230 User kuno logged in.
```

```
ftp> cd /tmp              ←向うの/tmp デイレクトリに
250 CWD command successful.
ftp> put t1               ←こちらの./t1 を転送
200 PORT command successful.
150 ASCII data connection for t1 (192.50.17.1,2184).
226 Transfer complete.    ←完了。
local: t1 remote: t1
3836 bytes sent in .11 seconds (34 Kbytes/s)
ftp> bye                  ←おしまい。
221 Goodbye.
```

一方、rcp はこのような「対話」は不要で、いきなり両ホストのパス名を指定してコピーを行なう。その代わり、パスワードを打ち込む機能はないので、前項で述べた rlogin でのパスワード不要の設定にしてあるホスト間でのみ使うことができる。ちなみに現在入っているホストのホスト名は省略してよい。

11.8.3 遠隔指令実行 — rsh

これは全く Unix 固有の機能で、

```
rsh  ホスト名  '指令 ...'
```

とやると、指定したホストで指令が実行できる、というものである。これも rcp と同様、パスワード不要の設定になっている場合のみ利用できる。これは何の役に立つか分かりますか?

11.8.4 遠隔ファイルアクセス — NFS

ところで、これまで皆様はいくつかあるマシンのどれかに login して作業をして来られたわけだが、どれに login しても自分のホームディレクトリ以下にあるファイルは同じようにアクセスできた。これは不思議だと思いませんか? 実は、皆様の個人ファイルはすべて大容量ディスクのついている sma に入っていて、ls とか cat とか言うたびにファイルの内容が NFS(Network File System) 機能により**遠隔アクセス**されて持って来られるように設定されていたのでした。NFS の設定状況は

```
/etc/mount
```

により表示できる。例えばホスト smea で実行すると次のように表示される。

```
% /etc/mount
/dev/sd0a on / type 4.2 (rw,nosuid)
/dev/sd0g on /usr type 4.2 (rw)
/dev/sd0h on /t1 type 4.2 (rw)
/dev/sd0d on /root type 4.2 (rw)
/dev/sd0e on /swap type 4.2 (rw)
utogw-gw:/sb on /sb type nfs (rw,bg)
sma-gw:/usr/share on /usr/share type nfs (rw,bg)
sma-gw:/sf on /sf type nfs (rw,bg)
sma-gw:/se on /se type nfs (rw,bg)
sma-gw:/sd on /sd type nfs (rw,bg)
sma-gw:/sc on /sc type nfs (rw,bg)
sma-gw:/sa on /sa type nfs (rw,bg)
sma-gw:/ma on /ma type nfs (rw,bg)
sma-gw:/uc on /uc type nfs (rw,bg)
sma-gw:/ub on /ub type nfs (rw,bg)
sma-gw:/ua on /ua type nfs (rw,bg)
%
```

これらのうち、最初が/dev/...で始まるのは自分のディスクに入っているファイルシステムを意味する。一方、ホスト名:...で始まるのはNFSアクセスが指定されていることを意味する。例えば「/ub」をアクセスしようとした場合は、実際にはNFS経由でsma-gw(というのは、smaのこのマシン側のアドレスだった)の/ubをアクセスする、という設定になっているわけである。なお、このように常時マウントしておく代わりに最初にアクセスが起きた時に自動的にマウントが行なわれる仕組み(オートマウンタ)を用いることもある。

11.8.5 遠隔ホスト情報 — rwho、ruptime、rmap

ネットワークでつながっている他のホストがどのくらい混雑しているか、またそこでどんなユーザがloginしているかを知りたいことがある。これらの情報は次のような指令で見ることができる。(このようなサービスは負荷が大きいとして動かしていないサイトも多いが。)

```
ruptime   ---  動いているホストと負荷の一覧。
rwho      ---  loginしているユーザの一覧。
rmap      ---  上2つの機能を併せ持つ。画面端末版。
```

これらのサービスは実際にはコマンドを起動するたびに情報を探しに行くわけではなく、各マシンで定期的に負荷やユーザ状況を報告するプログラム(デーモン)がずっと走っていて、これが決まったホスト(我々の所ではutogw)のデーモンに状況をUDPパケットで送りつけてくるので、utogwのデーモンがその状況をファイルに保管する。各コマンドはこのファイルを読んで必要な情報を抽出して表示しているだけのことである。

11.8.6 情報交換サービス — mail、news、phone

電子メールと電子ニュースについてはもはやそれが何であるかの説明は不要と思うが、その中身がどうなっているかは簡単に説明しておこう。まず、我々のサイト内ではすべてのメール送出やニュース投稿はUnixの遠隔実行機能を用いてホストutogwで実行される。ローカルなニュースやメールはそれが直ちにutogw内にある格納用ファイルに入って、各自はそれをNFSにより読み出すのでこれでおしまいである。一方、外部とのやりとりはメールについては複数サイト、ニュースは1つのサイトと接続しあっていて、これらのサイトが新しいメール/ニュースを入手した場合にはutogwにネットワーク経由で接続してその内容を転送してきて、管理プログラムを通じてutogwのファイルに格納する。こちらから出ていくべきものの場合は逆にこちらからこれらのサイトの管理プログラムに接続して転送する。

さて、メール(お手紙)とニュース(新聞)に加えて「電話」もあるのはご存知だろうか。これは

```
phone 相手のユーザ名@ホスト名
```

により起動し、指定した相手が指定したホストにloginしていれば相手の画面に

```
phone: connection requested by ユーザ名@ホスト名
phone: respond with "ユーザ名@ホスト名"
```

と表示される。ここで相手が

```
phone
```

とだけ応答すると、接続が確立して画面が上下2つに分かれ、それぞれがキーボードから打ち込んだものが両方の画面に表示されて「会話」が行なえる、というものである。なお、サイトによってはphoneが用意されてなくて代わりにより古いソフトであるtalkというのが使えることも多い。

11.8.7 ネットワーク情報サービス — NIS

ここまでに述べてきたように、ネットワークにおいてはホスト名と IP アドレスの対応表、サービス名とポート番号の対応表、ホスト名と Ethernet アドレスの対応表、など様々な対応表が必要である。また、各ホストにはそこに登録したユーザの情報を管理する表もある。昔の Unix これらを各ホスト個別のファイルに入れていた。しかし、そもそもこれらの情報はあるサイト内ではすべて同じなので、情報に変更があるたびに各ホストのファイルを更新する手間が掛かり大変であった。その対策として考えられたのがネットワーク情報サービス (NIS --- Network Infomation Service) で、この機能を使うことにより情報は 1 箇所 (我々のサイトでは sma) だけに置き、他のホストからはこれを遠隔アクセスして情報を参照するようにできる。この情報をユーザが見たい場合には

```
ypcat 情報名
```

により「表」の内容が (ランダムな順ではあるが) 表示できる。現在のところ次のような情報がサポートされている。

```
ypcat passwd     --- ユーザ登録情報
ypcat hosts      --- IP アドレス- ホスト名対応表
ypcat networks   --- IP アドレス- ネットワーク名対応表
ypcat services   --- サービス名- ポート番号対応表
ypcat ethers     --- Ethernet アドレス- IP アドレス対応表
```

A 演習、課題

A.1 11 節の演習

11-1. rlogin、telnet、ftp、rcp、rsh などを実際に使ってみよ。これらの方法で確かに他のホストに接続できたかどうかを確認するには、どうしたらいいと思うかも考えてみよ。

11-2. 何らかの方法で、自分がいるホストの IP アドレスを調べてみよ。また調べた IP アドレスが正しいことを確認してみよ。(ヒント: ホスト名は hostname 指令で、一覧表は ypcat hosts で調べられる。また telnet IP アドレスでアドレスを指定して接続して、login した後で ps などで確かに「元の」ホストであることを確認すればよい。)

11-3. あなたがいるマシンから学内の各マシンへの「地図」を分かる範囲で作成してみよ。ypcat hosts で学内のホスト群の目安をつけて、traceroute を使えばある程度分かる。[7]

11-4. rlogin、telnet、ftp、rcp、rsh などを使っている状態で netstat -f inet を実行し、どれが自分が現在使っているサービスに対応するリンクかを探してみよ。また、そのサービスを終った時対応するリンクもなくなることを確認せよ。

11-5. netstat 1 でネットワークの通信状況を表示させた状態で、別の窓でネットワークアクセスを行ない、パケットが飛ぶのを確認せよ。できればアクセスする対象を工夫して 1 パケットは何バイトかを推察してみよ。[8]

A.2 8 回目の課題

この回の報告を出される場合には 11-1〜11-5 から 2 個以上選択して下さい。

[7] なお、traceroute では一部のマシン (OS が古いもの) については表示がおかしいことがあるので、そのつもりでどうぞ。また、学外へは出れません。そしてこれはあくまでも勉強のためであるので、今後お遊びにこういう指令を使って探索しまくったりしないように。なお NEWS では traceroute は/usr/sony/bin あたりに置いてある。

[8] こういう目的には ftp で何バイトかファイルを転送してみる、というのが適しているが、別のやり方でもよい。なお、自分のホームディレクトリにアクセスすると NFS パケットが飛ぶので注意。だいたい/tmp というディレクトリは各マシンのローカルディスクにあるので NFS アクセスにならないから、ここにテスト用のファイルを置くとよい。なお他人と一緒にならないよう空いた時間にやらないと測れないのは当然である。