FÖRORD

Denna handbok vänder sig till systemadministratörer som ska driftsätta och utnyttja DIAB:s Netman, vilket är program varan som ligger till grund för nätverket D-NET och kommunikation via gateways mot andra leverantörers datorer.

I handboken förutsättes att läsaren är någorlunda familjär med datakommunikation och operativsystemet DNIX. Lämplig grundkunskap kan inhämtas i DIAB:s kurser i respektive ämne.

Handboken beskriver hur Netman installeras och används. Detaljer angående parametersättning för olika leverantörers datakommunikationsprotokoll finns inte med i denna handbok utan återfinnes i respektive leverantörs parametersättningshandbok.

Beskrivning utav hårdvaran för detta koncept finner du under handboken "Nätuppbyggnad - Hårdvara".

Konceptet Netman bygger på dels DS90 i ett lokalt nät, dels DS90 på lokalt skilda platser förbundna med någon av televerkets förbindelser. Det kan också vara en singel-DS90, som ska kopplas mot någon fjärrbelägen dator. I handboken kallas datorer i nätverket för **NÄTVERKSDATORER**, och datorer som nås via televerkets tjänster kallas för **FJÄRRDATORER** till skillnad från den lokala datorn dit användare är direktanslutna med en terminal.

Bilaga 1 innehåller en konfigurationsbild där den terminologi, som används i handboken, visas med bokstäver i motsvarande utrustning.

Operativsystemet DNIX kallas i handboken för OS, samt D-NET kallas för nätverket. Datorer i familjen DS90 kallas för datorer.

Du ska nu börja med att LÄSA INTRODUKTIONEN, för den ligger TILL grund för resterande delar av HANDBOKEN.

INNEHÅLLSFÖRTECKNING

1. INTRODUKTION

1.1. INLEDNING

Netman består av ett antal program, som ligger till grund för all kommunikation mellan datorer i nätverket, fjärrdatorer eller mot andra leverantörers datorer via gateways.

Netman och erforderlig hårdvara, ger möjlighet att på ett flexibelt sätt från en terminal nå nätverksdatorer eller fjärrdatorer med kommandon eller inloggning. Det finns också möjlighet att expandera separata datorers filsystem till ett gemensamt. Netmans tjänster bygger på att dess programvara är aktiverat i samtliga sammankopplade datorer.

Mellan datorerna ger ett lokalt nätverk hög överföringskapacitet, så att användarna inte upplever någon större skillnad om en nätverksdator utnyttjas. Om en fjärrdator däremot utnyttjas, blir dock överföringsmediets kapacitet begränsat av televerkets tjänster.

Det finns också möjlighet att nå en del andra leverantörers datorer via gateways från någon terminal som loggar in sig i den dator som har rätt uppsatt gateway för ändamålet.

1.2. ANVÄNDNINGSOMRÅDE

Det finns 4 olika grupper av tjänster i NETMAN-konceptet. Dessa tjänster har följande namn:

- NCU
- RX
- RACCESS
- GATEWAY

1.2.1 NCU - Network Call UNIX

NCU ger möjlighet att från en terminal koppla upp sig som en virtuell sådan mot godtycklig nätverksdator eller fjärrdator av samma familj, om säkerheten och uppsättningen så tillåter.

Användaren upplever sig, med hjälp av "NCU" direktansluten till den andra datorn, och kan utföra allt precis som om han vore lokalt ansluten till denna. Det finns dock hela tiden möjlighet att utföra kommandon i den lokala datorn, under tiden som man utnyttjar NCU. Filöverföringar kan utföras i båda riktningarna, men RX och RACCESS lämpar sig bättre för detta.

Kommandot är mycket enkelt till sin uppbyggnad och ser ut på följande sätt:

```
$ ncu system-namn
```

där systemnamn motsvaras av namnet på den dator man önskar nå. Ytterligare information om funktion och optioner, för NCU, finner du i kommandobeskrivningen och avsnitt 5.1.

Användningsområde för NCU

NCU är avsett för att utnyttjas i de fall användaren har behov av att få en virtuell terminal, till en annan dator, än den han är direktansluten till.

Ska ett fåtal kommandon utföras lämpar sig RX bättre, efter- som inloggningsförfarandet vanligtvis är automatiskt.

1.2.2 RX - Remote Execute

RX erbjuder användare en möjlighet att utföra kommandon i andra datorer än den lokala. RX förutsätter att användaren har användaridentiteter i samtliga datorer där kommandon önskas utförda. Den dator som kommandot ska utföras i auktoriseras vanligtvis genom filen ".netkey", där RX också hämtar användar- identitet och password, vilket matchas mot ".netkey" i den dator där kommandot ska utföras. Ett annat sätt att aktivera RX, som ej kräver information från filen ".netkey", och även ignorerar ".netkey" om den finns, är med hjälp av optionen "-l". I detta fall beaktas enbart information i /etc/password, för inloggning, i den andra datorn. För mer information om denna option kan bilagan RX(1) och avsnittet 5.2 läsas.

Hjälpprogrammet "ls" skriver ut filerna i ett directory. Vill man utföra det i t ex en nätverksdator ser kommandot ut på följande sätt:

```
$ rx system-namn!ls
```

resultatet av kommandot kommer på terminalen (standard output) där kommandot gavs. Exemplet förutsätter ".netkey" i dels den lokala datorn, dels i den andra datorn.

Ytterligare information om rx finner du i bilagan RX(1) och avsnittet 5.2.

Användningsområde för RX

RX är väl lämpat för enstaka kommandon i nätverksdatorer eller fjärrdatorer. Exempel på användningsområden är följande:

- o Utskrifter på skrivare anslutna till andra datorer än den lokala
- o Filöverföringar (även backuper)
- o Status frågor på andra datorer (who, ps, etc)

1.2.3 RACCESS - Remote file ACCESS

RACCESS erbjuder användaren ett utbyggt filsystem, dvs filsystemet för flera datorer upplevs som ett enda.

RACCESS kräver en ".netkey", vars innehåll matchas mot ".netkey" i den dator som man önskar nå, innan filåtkomsterna kan verkställas. Flerdator-filsystemet är inte permanent utan varje gång en filåtkomst ska utföras ombesörjs automatiskt en inloggning i den andra datorn och filåtkomsten möjliggörs, om inte en tidigare uppkoppling utförts inom en viss tidsrymd, vanligen 30 sekunder. Denna tidsrymd kan styras med hjälp av optionen "t", när RACCESS aktiveras, se vidare bilagan RACCESS(1).

Användare kan se innehållet i password-filen i t ex en fjärrbelägen dator med följande kommando:

```
$ cat /net/system-id/etc/passwd
```

Resultatet kommer ut på terminalen där kommandot utfördes.

Ytterligare information finner du i bilagan RACCESS(1) och avsnittet 5.3.

Användningsområde RACCESS

RACESS är den rätta funktionen om man vill se innehållet i en fil eller låta ett program utföra åtkomster på en fil i en nätverksdator eller en fjärrbelägen dator. Exempel på användningsområden är följande:

- o Filåtkomster
- o Filöverföringar
- o Åtkomst till databaser i andra datorer

1.2.4 Gateways

Gateways är utgångar som ger användare möjlighet att från en terminal, ansluten till någon dator i nätverket, få tillgång till andra leverantörers datoringångar. Ett 4004-kort fungerar vid detta tillfälle som en gateway. Ytterligare programvara, som finns som option, krävs för att klara den andra leverantörens protokoll för kommunikationen via gatewayen.

En uppkoppling mot en IBM-dator med SNA/SDLC 3270, utförs på följande sätt:

```
$ sna3270
```

förutsatt att allt är specificerat på ett korrekt sätt, vilket beskrivs i avsnittet "Installation av nätverk" och i handboken "NÄTVERK OCH DATAKOMMUNIKATION" under avsnittet "IBM SNA/SDLC 3270".

1.3. UTRUSTNING

Netman består av en diskett med grundfiler. Protokoll för GATEWAYS, dvs program som emulerar andra leverantörers kommunikationsprotokoll, ingår inte i Netman utan kan köpas till på separata disketter.

För att du ska kunna använda funktionerna i Netman krävs följande hårdvara:

Produkt

- Dator
- Kommunikationsprocessor 4004 (001-7095-00)
- Linje-adapter och kabel till 4004-kort (001-7094-00)
- Nätverkskabel
- Bootprom för 4004-kort (endast fjärrbootade 4004)

Om gateways ska utnyttjas krävs följande:

- Modem
- Modemkabel (medföljer 4004)
- Linje (om fast förbindelse)
- Telefonanslutning (uppringbar förbindelse)
- NRZI-adapter (SDLC med NRZI)

Om Datex ska utnyttjas för gateway krävs följande:

- Datex-anslutning
- DCE
- X.21-adapter (012-7090-10)
- Skarvkabel DB15 (004-6151-05)

2. INSTALLATION AV NÄTVERK

Det här avsnittet beskriver hur man installerar konceptet Netman med tillhörande hårdvara. Avsnittet innehåller en hel del detaljinformation, som krävs i vissa fall. För att inte behöva tränga in i alla detaljer, har installationsavsnittet delats upp i tre delar enligt följande:

- o Grundkonfiguration
- o Tilläggskonfiguration
- o Fjärrboot-konfigurering

Grundkonfigurationen avser de fall då användarna enbart ska använda datorer som är anslutna i nätverket.

Tilläggskonfigurationen avser de fall då flera 4004-kort ska utnyttjas, och eller "GATEWAYS" mot fjärrdatorer eller andra leverantörers datorer ska definieras.

Fjärrboot-konfigurering avser de fall då 4004-kort ska fjärrbootas från någon dator i nätverket. 4004-kortet sitter i detta fall i någon annan typ av dator, t ex en mätdator, eller i en egen box (expansionsrack), som är ansluten till nätverket.

Denna installationsanvisning avser icke nätverkskabel och linje-adapter, dvs den hårdvara som krävs för att ansluta en dator till nätverket. Detta beskrivs i handboken "Nätuppbyggnad - Hårdvara", vilken också anvisar hur linjeadapters ska monteras.

Installationsanvisningen sträcker sig från linjeadapterns kontakt mot 4004-kortet, till en dator med tillhörande filer för Netman eller en separat box med 4004-kort.

I bilaga 1 finns en komplex konfigurationsbild, med bokstäver i varje del, som ger referenser, till den terminologi som finns i handboken.

2.1. GRUNDKONFIGURATION

Grundkonfigurering omfattar de arbetsmoment som krävs, för att användare ska kunna utnyttja datorer anslutna lokalt i ett nätverk. Efter att ha utfört denna kan tjänsterna "NCU", "RX" och "RACCESS" användas, om behörighetsfiler har satts upp. Behörighetsfilerna finns beskrivna i avsnittet "SÄKERHETSSYSTEMET".

Att installera en grundkonfiguration av Netman med tillhörande hårdvara är mycket enkel.

Arbetsgången består av två steg:

1) Bygla och montera hårdvara

2) Läsa in Netmanfiler

Hårdvaran för en grundkonfiguration består av (artikel-nr mellan parenteser):

a) Kommunikationsprocessor 4004 (001-7095-00)

b) Linjeadapter för nätverket och kabel till 4004-kort (001-7094-00)

c) Nätverks-kabel (partvinnad två-tråd eller koax)

2.1.1 Kommunikationsprocessorn 4004

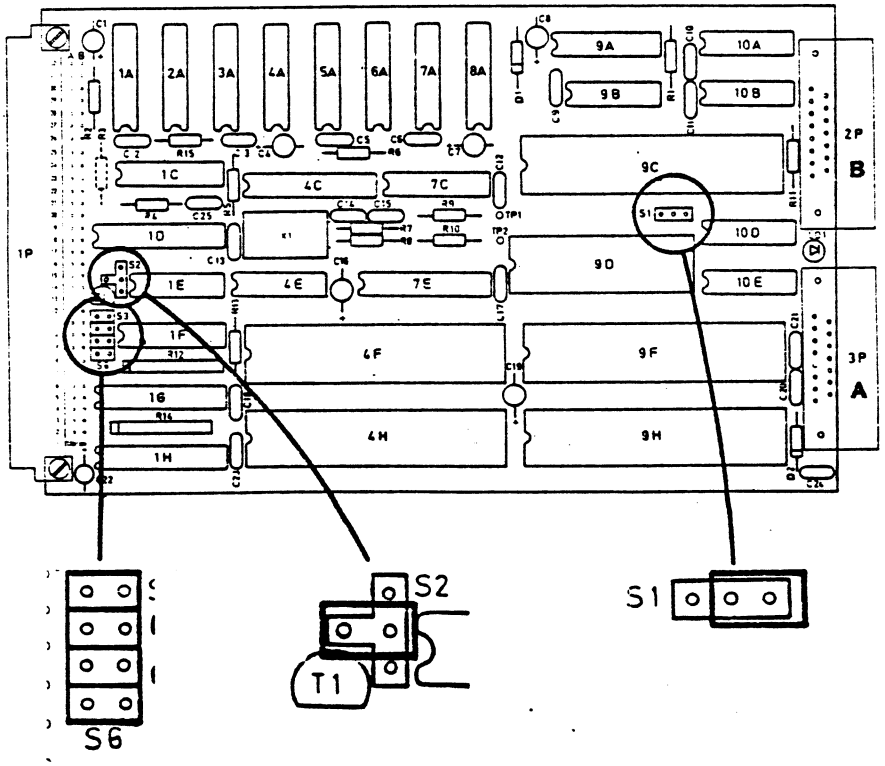
Kommunikationsprocessorn 4004 är placerad på ett kretskort (enkelt europakortsformat) som monteras i lediga utrymmen bak på datorn. För att kunna ha flera 4004-kort i en dator, måste kortadresser kunna varieras, vilket utförs med byglingar.

Nedan finner du en beskrivning av arbetsgången för bygling och montering av 4004.

Bygling av 4004

Kommunikationsprocessorn 4004 ska före montering byglas på två ställen.

I figur 1 finns en bild av ett 4004-kort med A och B kanal samt byglar utmärkta:



Först ska man bygla adressen på bygel s3, se bilden ovan. På ett av 4004-korten ska samtliga byglar vara slutna, vilket ger adressen noll.

Bygel s1 ska alltid vara i högerläge.

Det finns också en tredje bygel s2, vilken för närvarande inte används. Vid leveransen är den satt i vänsterläge.

Montering av 4004

Före monteringen måste datorn stängas av och spänningen slås av. Detta måste göras genom att man loggar in som root och ger kommandot /etc/shutdown. När meddelandet "System Halted" kommer på huvudterminalen ska spänningen slås av.

Därefter placeras 4004-kortet på valfri plats i expansionsutrymmet bak på datorn.

När 4004-kortet har monterats, startas datorn enligt företagets rutiner.

Kabel mellan 4004 och nätverksadapter

Kabeln ansluts till kanal A på 4004-kortet och till kontakten märkt "computer" på linje-adaptern.

Inläsning av Netman-filer

Vid leverans av Netman följer det alltid med "NETMAN RELEASE NOTICE". Detta dokument beskriver ett korrekt införande av alla filer som hör till Netman.

2.2. TILLÄGGSKONFIGURATION

Tilläggskonfigurering krävs när fler 4004-kort än huvudkortet ska utnyttjas i en dator och även när "GATEWAYS" mot andra leverantörers datorer ska definieras.

Tilläggskonfigurationen kräver en djupare kunskap i system-administratörens arbetsuppgifter och att man följer vissa rekommendationer enligt detta avsnitt. Rekommendationerna gör det enklare att underhålla en installation och förändra en konfiguration.

Tilläggskonfigurering består av följande steg:

1 Bygla och montera nya 4004-kort

2 Skapa/ändra filen "/usr/lib/net/cards"

3 Skapa node i OS för nya 4004-kort

4 Parametersättning via compar-programmet

5 Starta om Netman

2.2.1 Bygla och montera nya 4004

När du ska sätta i nya 4004-kort kan du följa avsnitten ovan om bygling och montering 4004, med undantag för adressbyglingen.

Det första 4004-kortet, huvudkortet, bör ha adressen 0. Till de efterföljande rekommenderar vi att ni väljer adresserna 1, 2, 3 t o m 15, i tur och ordning. OS:et är dock genererat för att hantera tre kort, med vilka som helst av adresserna ovan. Antalet kort kan utökas, men detta kräver en omgenerering av operativsystemet. Fler 4004-kort än fyra, kräver ett expansionsrack.

Adressen byglas binärt på s3 med fyra byglar. Om ni håller 4004-kortet med kontakterna A och B till höger, så är den minst signifikanta biten nederst. En öppen bygel är lika med binär etta, en sluten är lika med en binär nolla. Dvs om nedersta biten är öppen och de övriga slutna fås adressen 1.

Samtliga möjliga adresser fås enligt följande:

nedersta			översta	
sluten	sluten	sluten	sluten	= 0
öppen	sluten	sluten	sluten	= 1
sluten	öppen	sluten	sluten	= 2
öppen	öppen	sluten	sluten	= 3
sluten	sluten	öppen	sluten	= 4
öppen	sluten	öppen	sluten	= 5
sluten	öppen	öppen	sluten	= 6
öppen	öppen	öppen	sluten	= 7
sluten	sluten	sluten	öppen	= 8
öppen	sluten	sluten	öppen	= 9
sluten	öppen	sluten	öppen	= 10
öppen	öppen	sluten	öppen	= 11
sluten	sluten	öppen	öppen	= 12
öppen	sluten	öppen	öppen	= 13
sluten	öppen	öppen	öppen	= 14
öppen	öppen	öppen	öppen	= 15

2.2.2 Skapa/ändra i filen "/usr/lib/net/cards"

Filen /usr/lib/net/cards är underlag för Netmans bootrutiner. När ni satt i ett nytt kort skapar eller ändrar ni filen "cards" med en editor. Filen "cards" består av rader med två fält. Det första fältet är korttyp (för närvarande endast 4004), och det andra fältet är namnet som systemadministratören väljer till 4004-kortet. Namnet ska också användas som nodenamn, när motsvarande enhet skapas i OS:et.

Har ni enligt avsnittet "Bygla och montera nya 4004" valt någon adress, t ex 1, skriver ni på en ny rad i filen "cards" följande:

```
4004 n400401
```

För de två sista positionerna i fältet "n400401", rekommenderar vi att ni väljer den sifferkombination som motsvarar adressen på 4004-kortet.

Filen "cards" måste finnas för att Netman ska känna till att andra kort än det första, dvs huvudkortet, existerar vilket krävs för att boot av dessa ska fungera.

2.2.3 Skapa node för 4004-kort i OS:et

En node, dvs en adress i operativsystemet och en knytning till den programvara som krävs, för 4004-kortet måste skapas i OS:et, vilket utförs med kommandot "/etc/mknod". Fortsätter vi även här med adressen 1 får kommandot följande utseende:

```
$ /etc/mknod /dev/n400401 c 10 1
```

Första argumentet är nodenamnet som önskas, 01 i namnet, motsvarar adressen på 4004-kortet (bygel s3). Andra argumentet "c" anger att det är en teckenenhet, till skillnad från blockenhet. Tredje argumentet "10" är OS:ets modul för näthantering. Sista argumentet måste vara adressen som är byglad på 4004-kortet (bygel s3). I det här fallet 1, eftersom vi valde det exemplet. Adressbyglingen finns beskriven i avsnittet "Bygling av 4004-kort", och "Bygla och montera nya 4004-kort".

2.2.4 Parametersättning via "compar"

Compar är ett program som är till för att skapa och/eller sätta parametrar för 4004-kort i en Netman-konfiguration. Filen som skapas är till för bootning av ett 4004-kort. Compar arbetar mot ett 4004-korts parameter- och konfigurationsfil i taget.

För att starta compar måste systemadministratören ha superuser-privilegier, vilka fås genom att starta programmet su eller logga in som root.

Compar består av en huvudmeny med valmöjligheter enligt följande:

```
COMMUNICATION PARAMETERS FOR 4004/4204
-------------------------------------------------------------------
Main menue

                    1. Installation parameters
                    2. Port definitions
                    3. Definition of com-servers
                    4. Com-server references
                    5. Definition of other servers
                    6. Update next board
                    7. Write
                    0. Exit
-------------------------------------------------------------------
Selection (0 - 7):
```

Om man utför någon förändring med compar MÅSTE man ovillkorligen välja "7. Write" innan man avslutar compar eller väljer "6. Update next board", annars utförs inte de förändringar man gjort. Eftersom "compar" har denna write- funktion, kan man prova programmet utan att spara det man har gjort.

Innan man startar "compar" (observera SUPER-USER privilegier) måste man byta directory till /usr/lib/net, vilket utförs med kommandot:

```
# cd /usr/lib/net
```

Start av "compar" utförs med följande kommando:

```
# compar
```

Efter start av compar får man först frågan "Update parameters for board # :". Där svarar man "n4004nn", fast istället för nn skriver man den adress man har byglat på 4004-kortet. Skall parametrar ändras för huvudkortet anger man n4004. Trycker man därefter på return får man upp huvudmenyn.

För att aktivera de förändringar man utfört via "compar" måste Netman startas eller återstartas, vilket beskrivs i avsnittet "Start eller återstart av Netman", avsnitt 2.4.

Avsnitten som följer beskriver de olika menyerna.

Avslutning och markörförflyttning i "compar"

I compar använder man sig av CTRL-d för att avsluta menyer.

Markörförflyttning utförs enligt följande:

Vänster	CTRL-h
Höger	CTRL-l
Nedåt ett steg	CTRL-j eller return
Uppåt ett steg	CTRL-k

"1. Installation parameters"

Nedan följer meny 1:

```
                COMMUNICATION PARAMETERS FOR 4004/4204
-------------------------------------------------------------------
Installation parameters

Boardtype: 4004 Host connection (Y/N):
```

På "Installation parameters" fråga "Host connection (Y/N):", kan man svara Yes (Y) eller No (N).

Svarar man Y indikerar man att det aktuella 4004-kortet är ett huvudkort, som nås från bussen i datorn. "N" indikerar att 4004-kortet är ett tilläggskort.

"2. Port Definition"

Nedan följer meny 2:

COMMUNICATION PARAMETERS FOR 4004/4204

Port definitions

Port id	Usage	Group	Linetype	Phone number
SIO A	D			
SIO B	G	0	0	

Linetype	Usage
0 Leased or manual operated	D DNET
1 RS232 Autoanswer - DTE	G Gateway
2 RS232 Autoanswer - DCE	U Unused
3 Autodial - Mikroteknik	
5 X21	

Under menyn "2. Port definition" anger man hur port A och B på 4004-kortet ska användas.

Den nedre halvan visar valmöjligheterna för den övre halvans kolumner "USAGE" och "LINETYPE".

"GROUP" används för att kunna dela in kortets SIO (Seriell Input Output) i grupper, främst avsett för framtida bruk.

"PHONE NUMBER" används för återuppringning i Datex X.21, här skrivs abonnemangsnumret för den egna DCE:n.

"3. Definition of com-servers"

Nedan följer meny nummer 3:

COMMUNICATION PARAMETERS FOR 4004/4204

Definition of com-servers

Server	Count	Suffix	Portgroup	Autostart	Reference
SNACLUSTE	1	R	0	N	0

I denna meny väljer man vilket protokoll som ska användas i den "GATEWAY" som går via port B mot någon annan leverantörs datorer. Även port A kan utnyttjas som en "GATEWAY", om den inte är kopplad mot nätverket.

Dokumentet "BESKRIVNING AV FILER FÖR BOOTNING AV KOMKORT" (bilaga 2) informerar om tillgängliga protokoll. Alla "com- servers" (protokoll) har beteckningen "boota05nn", där nn kan vara 10-19, och motsvarar olika leverantörers protokoll.

Server-kolumnens innehåll styrs av vilka protokoll som finns inlästa. Låt oss ta exemplet att "boota0512", vilket motsvarar protokollet SNA/SDLC 3270, finns inläst i den dator som ni håller på att generera. Då står det "SNACLUSTE" i kolumnen "Server".

I kolumnen "Count" anger man hur många samtidiga cluster man vill ha tillgång till i samma 4004-kort. Det maximala antalet är tre. Valida värden är 0, 1, 2 och 3. Det interna minnet på kortet är begränsat till 64 Kbytes, så ju fler rutiner man lägger ut i kortet desto mindre utrymme är det kvar för externa anrop från t ex nätverket (NCU, RX eller RACCESS). Antalet terminaler som kan kopplas upp mot i kortet laddad programvara minskar också.

Kolumnen "Suffix" ger möjlighet att nå olika cluster som finns definierade. Det absolut vanligaste är att ett cluster definieras i varje 4004-kort. SNA 3270 t ex tar automatiskt "R", som suffix-värde. Vi rekommenderar därför "R", för det första clustret. Anger man att två eller tre cluster önskas, fås automatiskt "A" för det andra och "B" för det tredje. Väljer man själv ett tecken till "Suffix" för det första clustret, får efterföljande cluster nästa möjliga tecken i ASCII-alfabetet osv. Observera att suffix måste vara unikt i ett nätverk för cluster av samma typ.

"Portgroup" refererar till "group" under menyn "2. Port definition". Detta ger möjlighet att gruppera portar, vilket mest är avsett för framtida bruk.

I kolumnen "Autostart" anger man Y (Yes) eller N (No). Y ger möjlighet att starta ett cluster utan att någon terminal är aktiverad Det innebär att DTR (108) läggs hög till modemet så att en uppkoppling tillåts.

"Reference"-kolumnen ger möjlighet att med en siffra ge ett referensnummer till det cluster man definierar. Detta utnyttjas under menyn "4. Com-server references", där man ger kommunikationsparametrar för varje "GATEWAY".

"4. Com-server references"

Nedan följer meny 4:

COMMUNICATION PARAMETERS FOR 4004/4204

Com-server references

Ref	Options FDX	DLC	NAE	SHM	Number to call	Handler dependent data (hex)
0	N	N	N	N		00 00 00 00 00 00 00 00 00 00 00
1	N	N	N	N		00 00 00 00 00 00 00 00 00 00 00
2	N	N	N	N		00 00 00 00 00 00 00 00 00 00 00
3	N	N	N	N		00 00 00 00 00 00 00 00 00 00 00
4	N	N	N	N		00 00 00 00 00 00 00 00 00 00 00
5	N	N	N	N		00 00 00 00 00 00 00 00 00 00 00

Under meny 4 anger man kommunikationsparametrar för "GATEWAYS". Fördefinierat finns 6 st parametergrupper 0-5, motsvarande de som finns under meny 3 i kolumnen "Reference". De övriga fälten i denna meny är uppdelade i 3 grupper enligt följande:

- Options
- Number to call
- Handler dependent data (hex)

"Options" avser förbindelseparametrar och förkortningarna finns förklarade nedan.

FDX Full DupleX kolumnen ger möjlighet att välja halv eller full duplex, dvs om man utnyttjar en förbindelse med fast bärvåg i båda riktningarna (FDX), eller om förbindelsen "vänds" med RTS (105) och CTS (106) (HDX). Y ger full duplex, N ger halv duplex.

DLC Disconnect on Lost Carrier ger möjlighet att koppla ner Datelförbindelsen om bärvågen (109) försvinner gränssnittet.

NAE No Auto Enable är en drivrutinsberoende parameter (för GATEWAYS), som inte används för närvarande. Den ska vara "N" (NO).

SHM Short Hold Mode ger möjlighet att utnyttja Datexgränssnitt på ett effektivt sätt.

I kolumnen "Number to call" skriver man det nummer som automatiskt ska skickas, när det är förbindelser som kopplas upp till skillnad från fasta förbindelser. Den senare tjänsten kräver automatisk uppringare eller Datex X.21. Rutinen är anpassad för Micro-tekniks automatiska uppringare.

"Handler dependent data" avser sådan information, som den installation som man ska koppla upp sig mot tillhandahåller. I SNA-fallet krävs bl a XID (exchange id), och i UTS krävs "PID" (poll-id).

"5. Definition of other servers"

Denna meny ger möjlighet att få tillgång till andra servers än sådana som är avsedda för "GATEWAYS". Nedan följer meny 5:

COMMUNICATION PARAMETERS FOR 4004/4204

Definition of other servers

Server	Include	Parameter
COMMUX	N	
COMBOOT	N	

De två som idag finns i en Netman-miljö är Commux och Comboot, vilka ger tilläggstjänster i nätverket.

Include-kolumnen ger systemadministratören möjlighet att ta med eller utesluta servern. Detta verkställs med Y (Yes) eller N (No).

Parameter-kolumnen används ej för commuxen.För comboot styrs prioritet för fjärrbootning, valida värden är 0-3, där 0 är högsta prioritet.

"6. Update next board"

Detta val under huvudmenyn ger möjlighet att direkt från compar börja generera ett nytt 4004-kort. Man får efter val 6 samma fråga som när man startar compar, dvs "Update parameters for board # :". Man svarar där med node-namnet, t ex n400401 för det första tilläggskortet och n400402 för det andra tilläggskortet.

"7. Write"

Val nummer 7 "Write" verkställer de inmatningar man utfört med hjälp av compar.

Det är VIKTIGT att AVSLUTA med WRITE, annars går man miste om alla inmatade värden.

"0. Exit"

"Exit" avslutar compar-programmet.

2.3. FJÄRRBOOT-KONFIGURERING

I de fall 4004-kort inte är placerade i någon dator utan i en egen box (expansions-rack) ansluten till nätverket, krävs laddning (boot) från någon dator i nätverket.

Arbetsgången för att konfigurera för en sådan installation består av tre moment:

1) Parametersättning via "compar"

2) Förse 4004-kort med boot-prom

3) Bygling och montering av 4004 i expansionsrack

2.3.1 Parametersättning via "compar"

Parametrar för ett fjärrbeläget 4004-kort sättes i de datorer som ska ombesörja fjärrboot av detsamma. Arbetsgången är den vanliga:

- Sätt parametrar med hjälp av "compar"
- Avsluta "compar" med write
- Starta om Netman

För ett fjärrbootat 4004-kort väljs serienumret till namn för kortet. På frågan, efter start av "compar", "Update parameters for board # :" skrivs serienummer där serienummer ska vara aktuellt serienummer på 4004-kortet.

För att man ska kunna aktivera ett fjärrbeläget 4004-kort, måste nedanstående menyer ändras:

1. Installation parameters
2. Port definition
3. Definition of com-servers (om någon gateway önskas)
4. Com-server references (om någon gateway önskas)
5. Definition of other servers

När dessa menyer ändrats enligt efterföljande avsnitt, kan 4004-kortet bootas via någon dator i nätet. Ska andra portar på 4004-kortet parametersättas, följs beskrivning i avsnittet "TILLÄGGSKONFIGURATION".

"1. Installation parameters"

Under denna meny svarar man N (No), på frågan "Host connection (Y/N):".

"2. Port definition"

Under "USAGE" i denna meny anges ett "D" (nätanslutning) för port A på 4004-kortet.

Parametrar för andra utgångar på 4004-kortet sätts enligt önskemål. Valmöjligheter för dessa beskrivs under avsnittet "TILLÄGGSKONFIGURATION".

"5. Definition of other servers"

När ett fjärrbotat kort ska installeras måste meny 5 förändras för huvudkortet i den dator som ska utföra booten och även för den dator som ska fungera som reservdator för boot. Under denna meny för huvudkortet, dvs när man svarar n4004 på frågan "Update parameters for board # :", anges att 4004-kortet ska fjärrbootas genom att man skriver Y (Yes) efter "Comboot".

När 4004-kort fjärrbootas ska man ha alternativa datorer som utför denna, förutsatt att det finns flera datorer i nätverket. Samtidigt kan bara en dator ombesörja fjärrbooten. Detta styrs genom en prioritering som nås via parameterkolumnen. Valida värden i denna kolumn är 0, 1, 2 och 3, där 0 är högsta prioritet.

För datorn med huvudkortet, som normalt sköter fjärrbooten ges värdet 0 i parameterkolumnen. Alternativdator nummer 1 får en 1:a, alternativdator nummer 2 en 2:a och alternativdator nummer 3 en 3:a.

Netman måste startas om efter det att ändringar införts via compar, i de datorer som ska ombesörja boot. Se avsnittet 2.4 för information om hur det går till.

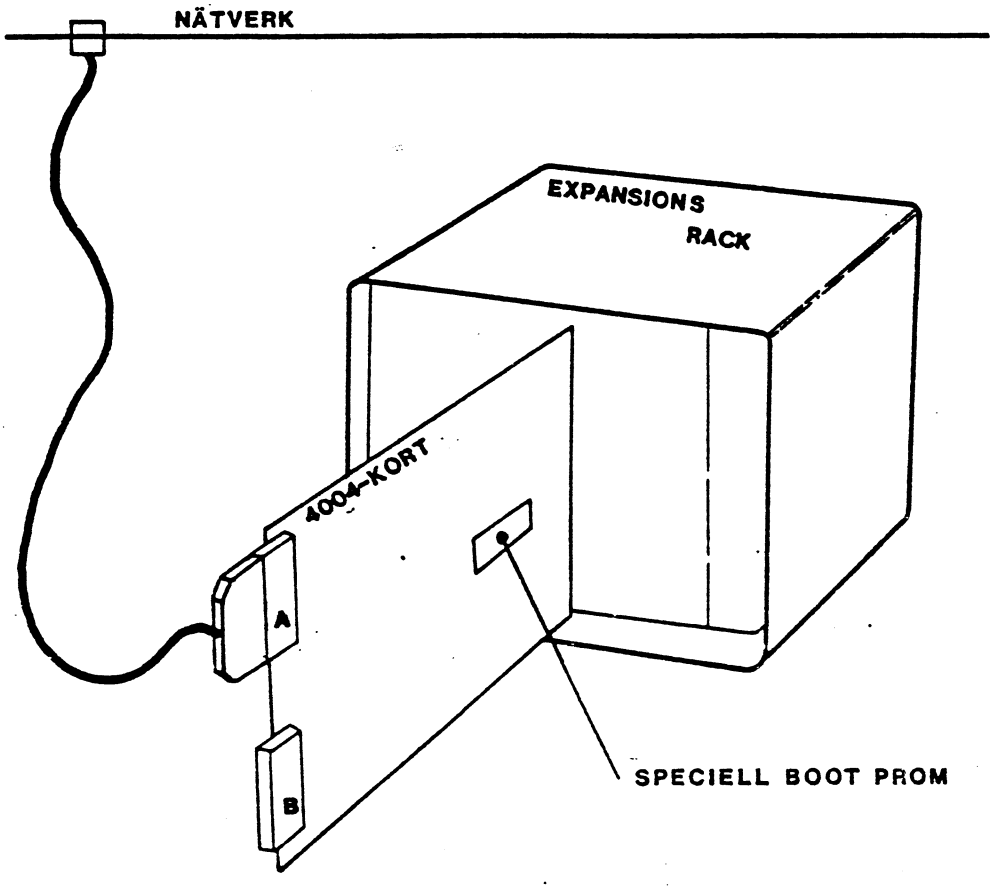
2.3.2 Förse 4004-kort med boot-prom

Ett 4004-kort som ska fjärrbootas måste vara försedd med en speciell krets (boot-prom), till skillnad från den som finns när kortet sitter i en dator. Kretsen beställs från leverantören eller när 4004-kortet beställs, påtalas det att kortet är avsett för fjärrboot.

2.3.3 Bygling och montering av 4004 i expansionsrack

Den enda bygel som måste kontrolleras är s1 (se figur1 i avsnitt 2.1), som ska vara satt i högerläge.

Montering sker enligt nedanstående bild på expansionsrack och 4004-kort. När detta är utfört ansluts port A på 4004-kortet till en nätadapter med därför avsedd kabel.



2.4. START ELLER ÅTERSTART AV NETMAN

För att aktivera de förändringar man utfört via compar måste Netman startas om. Netman återstartas enklast enligt följande:

```
$ /usr/lib/net/netman -k1
```

vilket också ser till att alla utestående aktiviteter för Netman avslutas.

För att manuellt avsluta och starta Netman används följande kommandon:

```
# /usr/lib/net/netman -k
```

efter kommandot kontrollerar man med kommandot:

```
# ps -al
```

om det finns någon netman i kolumnen längst till vänster. Finns det ingen startar man Netman med kommandot enligt följande:

```
# /usr/lib/net/netman
```

2.4.1 Problem att avsluta Netman

Finner man att Netman fortfarande finns kvar när man utför kommandot "ps -al", kan man prova att avsluta Netman med kommandot:

```
# /usr/lib/net/netman -k9
```

Skulle inte heller detta fungera måste systemet återstartas enligt företagets rutiner.

Om systemet återstartas blir netman automatiskt startad om en rad finns i /etc/rc enligt följande:

```
nice -14 /usr/lib/net/netman
```

Vill man starta Netman manuellt är kommandot:

```
# /usr/lib/net/netman
```

2.5. SÄKERHETSSYSTEMET I NETMAN

Säkerhetssystemet i konceptet Netman bygger på två delar:

- Skyddsmekanismer i operativsystemets filsystem
- Netmans skyddsmekanismer

Det är omöjligt att beskriva säkerhetssystemet genom att bara beskriva Netmans skyddsmekanismer. Därför måste också operativsystemets skydd i filsystemet beskrivas för att användare av flera datorers gemensamma filsystem skall få en helhetssyn av säkerhetssystemet.

2.5.1 Skyddsmekanismer i operativsystemets filsystem

Skyddsmekanismerna i filsystemet är uppdelade i tre delar:

- Användare
- Directories
- Filer

Användarna är uppdelade i tre grupper:

- Ägare
- Gruppmedlemmar
- Alla användare

Skydd på filnivå

För att se vilket skydd man kan få i filsystemet ser ni nedan resultatet av kommandot "ls -l" på ett directory med namnet **Bok**, och en fil med namnet **"del1"**.

```
drwxrwxr-x   2 pert     stats        256 Nov 29 12:22 Bok
-rwxrw-r--   1 pert     stats       7272 Nov 26 13:39 del1

-!--!--!--
t u  g  o          t=type  u=user  g=group o=other
```

Den första tecknet på varje rad visar vilken typ av fil det är. Bok har ett "d" i första positionen vilket visar att det är ett directory. Ett "-" visar att det är en vanlig fil, dvs icke ett directory eller hårdvaruenhet.

Nästa nio tecken är skyddsbitarna. De tre första av dessa nio avser ägaren, nästa tre avser gruppmedlemmar, och de tre sista avser alla användare. Grupper definieras i filen /etc/group, och används för att dela in alla användare i olika grupper, så att man flexibelt kan styra läs-, skriv- och exekveringsmöjligheter för gruppmedlemmar.

För en fil betyder "r" läsrättighet (read), "w" skrivrättighet (write) och "x" exekveringsrättighet (execute).

Skydd på directorynivå

För ett directory har skyddsbitarna följande betydelse:

r directoriet kan läsas med ls-kommandot.

w filer får skapas och tas bort i directoriet.

x ger rättighet att förflytta sig till directoriet att kopiera filer från detsamma (om filerna är läsbara).

Med kommandot "chmod" kan användare ändra dessa skyddsbitar för sina filer eller directories. Kommandot nedan:

```
$ chmod g-w del1
```

tar bort skriv rättighet för gruppmedlemmar på filen del1. Första tecknet i "g-w" avser användarnivån och kan vara u = user, g = group eller o = other, eller en kombination av dessa. Sista tecknet i "g-w", w = skrivbar (write) kan också vara r = läsbar (read) eller x = exekveringsbart (execute).

För att ytterligare klarlägga möjligheterna med skyddsbitarna för filsystemet på directory-nivå kan ni studera tabellen nedan.

Skyddsbitar för Testdir	-wx	r-x	rw-
pwd = pert ls "ser" Testdir	ja	ja	ja
pwd = pert cd Testdir	ja	ja	nej
pwd = Testdir "ls" ser testinfo	nej	ja	ja
pwd = Testdir cat testinfo	ja	ja	nej
pwd = Testdir rm testinfo	ja	nej	nej
pwd = Testdir > testinfo	ja	nej	nej

2.5.2 Netmans skyddsmekanismer

Netman har skyddsmekanismer för att skapa ett säkert nätverk. Förutom de skydd som operativsystemet ger, har Netman möjlighet att styra åtkomster från fjärrdatorer, eller nätverksdatorer. Detta styrs på tre nivåer enligt nedan:

- Användare utan ".netkey"
- Användare via ".netkey"
- Anropstyp via "servtab"

De olika tjänsterna "NCU", "RX" och "RACCESS" mellan olika datorer använder sig av olika principer för att klara sina anrop och filkopieringar.

"RACCESS" använder sig av en kombination av nätverksdatorns och den lokala /etc/passwd för kopiering av filer, och via ".netkey" kontrolleras behörigheten, för användare, att utnyttja RACCESS till andra datorer än den lokala.

NCU använder sig enbart av /etc/passwd i den dator som försöker nås för kontroll av behörighet vid inloggningen.

RX kan utnyttja ".netkey" och /etc/passwd i datorerna som är inblandade i kommandot, eller enbart /etc/passwd för kontroll av inloggning.

Optionen "-l" i RX används för att undvika ".netkey". För att inte blanda ihop användaridentiteter och användarnummer (u-id) vid filkopieringar, rekommenderar vi systemadministratören att se till att användare har samma användarnummer i olika datorer, dvs i lokaldatorn, nätverksdatorer och alla fjärrdatorer som användaren finns definierad i.

".netkey"

FÖR ATT INTE ÄVENTYRA SÄKERHETEN I ETT NÄTVERK SKA ROOT EJ HA NÅGON ".netkey". Om inte så är fallet kommer root- användare i lokala datorer in även i nätverksdatorer, vilket inte alltid är självklart på en installation.

Filen ".netkey" bör varje användare ha i sitt hem-directory, för att underlätta användandet av "RX" och möjliggöra utnyttjandet av "RACCESS". Filen skapas med en editor, och innehåller rader som består av tre fält enligt följande:

System-namn Användar-identitet Lösen-ord

Systemnamn är namnet på det system som användaren ska kunna nå med Netmans funktioner. Metatecken går att utnyttja i detta fält. En "*" ger användaren möjlighet att anropa alla system i ett nätverk, om denne finns med i fjärrdatorernas respektive nätverksdatorernas /etc/passwd.

Användaridentitet motsvarar login-namn för användaren, dvs första fältet i /etc/passwd.

Lösenord motsvarar ej det som skapats av användaren med hjälp av programmet "passwd", utan ".netkey" har egna lösenord vilka jämförs i olika datorer. Är de identiska tillåts funktioner via "RX" (utan l-option) och "RACCESS". Lösenordet ska skrivas in i klartext, till skillnad från den krypterade form som finns i /etc/passwd.

För en användare med användaridentiteten "pert" och lösenordet "venus" och som vill komma åt datorerna alfa, beta och gamma, skulle ".netkey" kunna se ut på följande sätt: alfa pert venus beta pert venus gamma pert venus

Vill man underlätta ".netkey"-hanteringen kan filen ha följande utseende:

```
* pert venus
```

vilket resulterar i att nya datorer kan läggas till utan att ".netkey" ändras, men den måste dock kopieras till användarens HOME-directory i den nya datorn.

Om ".netkey" inte skulle vara läs- och skrivskyddad vid något tillfälle, så ändras den så fort den utnyttjas till enbart läs- och skrivbar för ägaren. Detta för att skydda lösenordet, så att obehöriga inte ska kunna läsa det. Filen får alltså, efter någon access, behörighet enligt nedan:

```
-rw-------  1 pert      stats          43 Nov 26 13:00 .netkey
```

"servtab"

Filen servtab finns i directoriet /usr/lib/net, och beskriver vilka lokala tjänster som understöds. Servtab innehåller rader med tre fält vardera. Dessa anger servicenummer, servicenamn och programnamn.

Servicenummer kan bestå av ett nummer eller ett minustecken. Det vanligaste är ett minustecken, som innebär att ett unikt nummer ges vid varje uppstartstillfälle av Netman.

Servicenamnet används av program som vill använda sig av tjänsten. Om fältet innehåller "$S", ersätts det av systemnamnet som finns i /etc/systemid. Vanligtvis har man ett prefix som indikerar vilken typ av tjänst det är.

Programnamn är ett pathname till ett program, dvs ett fullständigt filnamn uttryckt från roten i filsystemet. Motsvarande program aktiveras när tjänsten används.

Servtab ser ofta ut på följande sätt, om "NCU", "RX" och "RACCESS" är tillåtna anrop från nätverksdatorer eller fjärrdatorer:

```
6       L_$S    /usr/lib/net/ncu
-       X_$S    /usr/lib/net/rx
-       A_$S    /usr/lib/net/raccess
```

2.5.3 Säkerheten på directory-nivå för RACCESS

Säkerheten på directory-nivå för "RACCESS" motsvarar det som finns för operativsystemets filsystem, avseende de funktioner som finns i tabellen i avsnitt 2.5."Skydd på directorynivå".

RACCESS har inte alla "system calls" implementerade. Se kommandobeskrivningen för information om vilka funktionskoder som vidarebefordras till andra datorer.

Skapa directories samt ändra skyddsbitarna för filer/directories är exempel på funktioner som inte kan utföras via RACCESS. Vill man uppnå sådana funktioner tar man i stället NCU eller RX till hjälp.

3. PROGRAMVARA OCH HJÄLPFILER

Netman består av ett antal program och hjälpfiler för att få ett nätverk att fungera och ge den säkerhet som krävs i en nätverksmiljö.

Program och hjälpfiler distribueras på en diskett med "tar-format" dvs filerna är nedlästa på disketten med UNIX hjälpprogram "tar".

Finns en diskett tillgänglig kan en innehållsförteckning erhållas med följande kommando:

```
$ tar tf /dev/sf0

boota0000
boota0100
boota0102
boota0300
boota0400
boota0500
boota0520
boota0521
bootman
bootp4004
bootpnet
compar
compar.bac
install
ioctltab
ncu
netman
raccess
rx
```

85-12-04 fanns ovanstående filer med på Netman-disketten. En aktuell innehållsförteckning fås med motsvarande tar- kommando på den senaste versionen av Netman-disketten.

Nedan följer avsnitt som beskriver de olika programmen och hjälpfilerna.

3.1 "install"

Install är Netmans installationsprogram.

För ytterligare information om hur man installerar programvaran kan dokumentet "Netman Release notices" läsas.

3.2. "bootannnn"

Gruppen "bootannnn" är ett antal filer, som dels är program som kommunicerar med hårdvaran (drivers), dels konfigurationsfiler. För ytterligare information om "boota"-filer kan bilaga 1 dokumentet "Beskrivning av filer för bootning av komkort" läsas.

3.3. "netman"

Netman är nätverksövervakaren som ombesörjer trafik till och från nätverket, samt startar processer för att ta hand om inkommande anrop från andra leverantörers datorer, fjärrdatorer och nätverksdatorer.

Netman i sin tur startar bootman som ombesörjer laddning (boot) av 4004-kort.

3.4. NCU, RX och RACCESS

NCU, RX och RACCESS är funktioner som fungerar mellan datorer från samma familj i ett nätverk. Funktionerna och dess användningsområde beskrivs i introduktionen till denna handbok.
Detaljinformation finns i respektive kommandobeskrivning.

3.5. "bootman", "bootpnet" och "bootp4004"

Bootman är ett program som laddar (bootar) 4004-kort i en dator eller i nätverket (fjärrboot). Finns det endast ett 4004-kort i en dator och inga som fjärrladdas via nätverket avslutas bootman efter laddningen. I annat fall är bootman kontinuerligt aktiverat och kontrollerar om något kort behöver laddas om.

Bootp-filer är program som laddas i 4004-kort för att senare ombesörja den verkliga laddningen via bootman ute i 4004-kortet.

Bootp4004 aktiverar lokalt 4004 komkort, till skillnad från bootpnet som aktiverar 4004-kort i nätverket.

3.6. Compar

Compar är ett program för konfigurering och sättning av parametrar till program, som kan laddas i 4004-kort. Programmet består av ett antal menyer, som är väl beskrivna i avsnittet "2.2. Tilläggskonfiguration".

3.7. "ioctltab"

Ioctltab innehåller beskrivning av funktioner som sköter I/O-kontrollen för RACCESS. Formatet för "ioctltab" är beskrivet i kommandobeskrivningen för RACCESS.

4. DRIFT AV NÄTVERK

Netmans drift kan vara helt automatiserad, dvs aktiverad via demoner i stället för av någon systemadministratör eller användare. Detta är också det normala för att datorer i ett nätverk ska vara tillgängliga även om systemadministratören inte är det.

I vissa fall kan det dock vara önskvärt att starta Netman manuellt, t ex om systemadministratören vill starta Netman med en trace. En trace ger kontinuerlig utskrift av status eller överförda data och den manuella starten är att föredra, eftersom utskriften tillfälligt önskas till någon speciell terminal eller skrivare.

Det är två program som ska aktiveras för att Netmans tjänster ska kunna utnyttjas. Det är dels "netman", dels "raccess". De andra programmen "rx" och "ncu" är program som varje användare startar själva när dessa funktioner önskas.

4.1. AUTOMATSTART

Automatstarten kan utföras på två sätt. Det ena sättet, avser en installation som önskar Netman aktiverad kontinuerligt, från det att operativsystemet går upp i fleranvändarmode tills systemet desaktiveras. Det andra sättet avser de installationer som önskar Netman aktiverad vissa tidsperioder, dvs ej kontinuerligt under dag, vecka, månad eller år.

4.1.1. Kontinuerlig aktivering av Netman

I de fall då en kontinuerlig aktivering av Netman önskas, dvs dygnet runt när operativsystemet är i fleranvändarmode, använder man sig av /etc/rc. Shellproceduren rc är avsedd för att aktivera filsystem (mount), debiteringsrutiner etc. Programmet är placerat i directoriet /etc, som innehåller även andra, för systemadministratören viktiga program och filer.

För att starta "netman" och "raccess", krävs att "rc" kompletteras via en vanlig editor enligt följande:

```
nice -14 /usr/lib/net/netman
nice -14 /usr/lib/net/raccess
```

När operativsystemet går upp i fleranvändarmode, startas automatiskt proceduren "rc", som i sin tur aktiverar "netman" och "raccess", om ovanstående rader finns i "rc".

4.1.2 Aktivering av Netman under vissa tidsperioder

I de fall när Netman endast ska vara aktiverat under dagtid, eller någon annan ej kontinuerlig tidsperiod, används cron; crontab, sed och cut. Dessa finns med i utvecklingspaketet, vilket krävs för att man på ett smidigt sätt ska kunna uppnå de aktiveringsperioder som önskas.

Programmet "cron" är en sk demon, som alltid ska vara startad om man vill ha program startade via "crontab". Programmet "cron" tittar med jämna tidsintervaller (vanligtvis varje minut) i "crontab", som är en tidtabell för program som återkommande ska startas.

För att få "netman" och "raccess" aktiverad respektive desaktiverad vid olika tillfällen, krävs fyra rader i filen /usr/lib/crontab. Raderna i "crontab" består av sex fält enligt nedan:

```
n n n n n program-namn
```

Alla "n" står för tidpunkter, i tur och ordning minuter, timmar, dag i månaden, månaden och slutligen veckodagen. För att kunna aktivera något flera gånger under t ex en timme kan "n" upprepas åtskilt av ett komma enligt nedan:

```
0,10,20,30,40,50 n n n n programnamn
```

vilket gör att aktuellt program startas var tionde minut. En "*" kan också ersätta "n", vilket betyder samtliga tillfällen för fältet. Valida värden för veckofältet är 1- 7, vilket motsvarar måndag-söndag.

Vill man ha "netman" och "raccess" aktiverad kontorstid mellan 08.00 och 17.00 krävs följande fyra rader:

```
0 8 * * 1-5 /usr/lib/net/netman
0 8 * * 1-5 /usr/lib/net/raccess
0 17 * * 1-5 /usr/lib/net/netman -k9
0 17 * * 1-5 kill -9 pid
```

På fjärde raden motsvarar "pid" en shellprocedur med en pipeline enligt följande:

```
ps -al ö grep raccess ö sed "s/Ö Ö */ /g" ö cut -f3 -d""
```

vilken ska vara placerad i en fil med namnet "pid" under directoriet /usr/bin. För att detta ska fungera måste hjälpprogrammen (UNIX utilities) ps, grep, sed och cut finnas i hjälpprogramdirectoriet "/bin", samt cron vara aktiverad av /etc/rc, med följande rad:

```
nice -14 /etc/cron
```

4.2. MANUELL START

Systemadministratören startar "netman " manuellt med följande kommando:

```
# /usr/lib/net/netman
```

Observera att superuser-behörighet krävs. Om en trace eller någon annan funktion önskas, finner ni ytterligare information om "netmans" optioner i kommandobeskrivningen för "netman".

4.3. LOGFILER

Netman:s logfil har namnet netlog????? under directoriet /tmp. ????? motsvarar ett löpnummer internt i Netman. Logfilen innehåller felmeddelanden om t ex varför netman inte kunnat aktiverats eller varför det har avbrutits. Systemadministratören behöver inte tömma logfilen, för detta tas om hand av /etc/rc vid systemstart.

5. ANVÄNDARDIALOG

UNIX* kommandoöversättare shell är utvecklat för programutvecklare i huvudsak. Programutvecklare kan dra stor nytta av kommandoöversättaren när de har byggt upp en kunskapsbas för sitt arbete genom att kontinuerligt använda operativsystemet.

För ovana användare är dock kommandoöversättaren minst sagt kryptisk. Kommandot nedan visar hur det kan se ut i värsta fall:

```
$ find / Ö(-name a.out -o -name '*.o' Ö) -atime +7 -exec rm ää
```

Kommandot tar bort alla filer med namnet a.out och/eller *.o som inte har använts under den senaste veckan.

En annan egenskap i operativsystemet kompenserar det kryptiska användargränssnittet med råge. Det är möjligheten att skräddarsy ett användargränssnitt för varje användare eller grupp av användare.

Kommandoöversättaren kan bytas mot någon annan utan något större ingrepp. Hjälpprogram (utilities) kan anpassas till användare enligt deras önskemål.

Detta möjliggörs eftersom kommandoöversättaren och hjälpprogram inte är integrerade i operativsystemets kärna. Förutsättningen för att detta ska fungera är en lyhörd och kunnig systemadministratör.

Netman:s funktioner erbjuder motsvarande möjligheter i ett nätverk av datorer, vilket kan realisera det framtida kontoret idag.

Även här krävs att miljön anpassas till användares förkunskaper. Följande avsnitt visar formatet på kommandon utan anpassning, samt användarvänliga shellprocedurer som kan anpassas efter användares önskemål. För att kunna utföra en användaranpassning i någon större omfattning krävs utvecklingspaketet med alla dess hjälpprogram.

De exempel på shellprocedurer som finns längre fram i detta avsnitt beskrivs inte närmare, utan vi hänvisar till boken "THE UNIX SYSTEM" skriven av S.R. Bourne, för information om hur man skriver shellprocedurer.

5.1. NCU

NCU har en mycket enkel kommandoutformning och efterföljande dialog enligt nedan:

```
$ ncu system-namn
Connected

login: pert
password:

                SYSTEMETS VÄLKOMSTMEDDELANDE

$ Disconnected
$
```

systemnamn ersätts med namnet på det system användaren önskar nå. Efter att ncu-kommandot utförts, erhåller användaren meddelandet "Connected" och därefter ett "login:". När användaren vill avsluta NCU, ges CTRL-D, vilket resulterar i att meddelandet "disconnected" kommer i den position som användaren har markören. På nästa rad uppträder kommandoöversättarens prompt (vanligen "$"), från den dator användaren startade NCU. För ytterligare information om NCU, se bilagan NCU(1).

5.1.1 Användaranpassning av ncu

Det som en användare kan råka ut för när NCU används, är att ett icke åtkomligt system anropas. Det kan t ex vara ett system som inte finns, ett som användaren inte är auktoriserad för eller ett som inte är aktiverat.

NCU, till skillnad från RACCESS, tar inte hänsyn till ".netkey", utan endast /etc/password. För att en användare, med hjälp av NCU, ska kunna logga in i en annan dator, måste alltså dennes användaridentitet vara inlagd i /etc/password på den dator som önskas.

Användare som inte så ofta loggar in i andra datorer, glömmer lätt bort vad de andra datorerna i ett nätverk har för namn. Filen ".netkey" kan då ge en fingervisning, men inte definitivt besked, om vilka datorer som kan nås med hjälp av NCU. Om systemadministratören har valt att skriva ut systemnamnen i ".netkey", kan följande shellprocedur underlätta användandet av NCU:

```
if test $# -eq 0
        then echo "

        Vilket system vill du nå:

"
        sed "s//TAB/                                    TAB=tabtecken
        s/Ö .*$//" $HOME/.netkey
        echo -n "

        (tryck på DEL om ni önskar avbryta ncu)

        SVARA MED MOTSVARANDE SYSTEMNAMN: "

        read syssvar
        echo

        /usr/bin/ncu $syssvar

        else echo;/usr/bin/ncu $1
fi
```

Proceduren kan anropas med eller utan systemnamn. Om systemnamn anges, anropas NCU som vanligt. Utesluter man systemnamn kontrolleras användarens ".netkey", och användaren får en lista på valida namn, och kan därefter angé önskat systemnamn vilket resulterar i ett motsvarande NCU-kommando.

Namnet "ncu" rekommenderas på den här proceduren, och den ska placeras i användarens eget directory "bin".

5.2. RX

RX är en av Netmans funktioner, som erbjuder användaren möjligheter att utföra godtyckligt kommando i en nätverksdator eller fjärrdator. Syntaxen ser ut enligt nedan:

```
$ rx system-namn!kommando ÄoptionsÅ ÄargÅ ... Ä> >> < öÅ ÄargÅ
```

systemnamnet efterföljs av ett utrops-tecken (!), vilket indikerar att en dator avses. Efter utropstecknet skrivs kommandot enligt gällande syntax. Kommandots utdata kan styras till användarens bildskärm (underförstått), till en fil med redirection eller till en pipe. Vid redirection och pipes avbryts utförandet i den andra datorn, och resultatet verkställs i den lokala datorn, se exempel senare i avsnittet.

Exempel på rx-anrop följer nedan:

```
$ rx alfa!ls

.netkey
.profile
bin
```

kommandot ls utförs i nätverksdatorn "alfa", med standard output utskrivet på den terminal som användaren utförde kommandot "rx" på. Formatet på rx-kommandot ovan, kräver en ".netkey" både i datorn där kommandot initieras, och i datorn alfa där kommandot verkställs. Vid användandet av en pipeline, om man t ex önskar räkna antalet filer i directoriet på datorn alfa med hjälp av "wc -l" (word count), skulle kommandot se ut enligt följande:

```
$ rx alfa!ls ö wc -l
```

Programmet ls utförs i alfa, medan "wc -l" utförs i lokaldatorn.

Vid användandet av redirection enligt följande:

```
$ rx alfa!ls > tmp
```

skapas filen tmp i lokaldatorn, och resultatet av kommandot "ls" på datorn alfa, skickas till filen.

Det finns möjlighet att utnyttja "RX" utan ".netkey", vilket kräver optionen "-l" enligt nedan:

```
$ rx -luid:passwd tilde!ls
```

På datorn "tilde" kontrolleras att användaridentiteten (uid) finns i /etc/passwd, och om passwd finns, att det stämmer överens med det som är angivet på kommandoraden. För ytterligare detaljer om "RX", se bilagan RX(1).

5.2.1 Användaranpassning av RX

För "RX" rekommenderas en separat shellprocedur för varje ändamål, som i sin tur i och för sig kan kombineras i en gemensam procedur. Det finns flera exempel på användningsområden för RX, t ex utskrift på fjärrskrivare, filöverföringar och statusfrågor.

Vi väljer här att visa en procedur för fjärrutskrifter, dvs utskrift av filer på en annan dators skrivare än den man är direktansluten till. Filen hämtas från den lokala datorn.

Proceduren följer nedan:

```
echo -n "
Nedan följer tillgängliga datorer:

"
sed "s//TAB/                                    TAB=tabtecken
s/ö .*$//" $HOME/.netkey
echo -n "

(tryck på DEL om ni önskar avbryta rlpr)

VILKET SYSTEM VILL DU SKRIVA UT FILEN PÅ: "

read syssvar

echo -n "
VAD HETER FILEN: "
```

```
read filsvar

rx $syssvar!lpr < $filsvar
```

Ett lämpligt namn på den här proceduren kan vara "rlpr", som är en förkortning av Remote Line PRinter. Proceduren "rlpr" placeras lämpligen i directoriet "/usr/bin" eller användarens eget "bin", beroende på vilka som ska nå den.

5.3. RACCESS

RACCESS är ett program för ett filsystem, som är expanderat över flera datorer i ett nätverk.

Kommandon som utnyttjar "RACCESS" ser ut på följande sätt:

```
$ ls /net/system-namn/path-name
```

I det här exemplet utförs kommandot "ls" i den lokala datorn och ser en annan dators filsystem som om det vore lokalt. Första delen "/net" indikerar att RACCESS utnyttjas, vilket är monterat på directoriet, dvs ett expanderat filsystem utnyttjas. Path-name är det fullständiga filnamnet, uttryckt från root, enligt följande exempel:

```
$ ls /net/alfa/usr/pert

bin
```

Directoriet /usr/pert listas på datorn alfa med hjälp av RACCESS.

5.3.1 Användaranpassning av RACCESS

Exemplet för RACCESS bygger på att en användare önskar utföra cat, ls och file på andra datorers filer, till skillnad från den dator de är direktanslutna till. Vi skapar som vanligt en shellprocedur för detta, enligt nedan:

```
echo -n "
Nedan följer tillgängliga datorer:

"
sed "s//TAB/                                    TAB=tabtecken
s/Ö .*$//" $HOME/.netkey
echo -n '

(tryck på DEL om ni önskar avbryta rfil)

VILKET SYSTEM VILL DU UTFÖRA KOMMANDOT PÅ: '

read syssvar

echo -n '
Följande kommandon är tillgängliga:

        cat
        file
        ls
```

```
VILKET ÖNSKAS: '

read cmdsvar

echo -n '
VILKEN FIL ELLER DIRECTORY ÖNSKAS
ANGE FULLSTÄNDIGT PATH-NAME: '

read filsvar

$cmdsvar /net/$syssvar$filsvar
```

Proceduren placeras i directoriet "/usr/bin" eller användarens "bin", beroende på vilka som ska nå den. Ett lämpligt namn kan vara "rfil", som är en förkortning av Remote Fil, dvs kommandon som berör filer i fjärrdatorer eller nätverksdatorer med hjälp av RACCESS.

5.4. SAMMANFATTNING

Användardialogen är vad systemadministratören gör den till. Funktioner finns för att effektivt utnyttja datorer i ett nätverk, och detta avsnitt visar, dels syntaxen i sin oförändrade form, dels användarvänliga shellprocedurer.

Shellprocedurerna är enbart exempel på hur det går att användaranpassa ett system. Systemadministratören kan ta till vara ideerna och ytterligare anpassa procedurerna för sin egen organisation.

Att skapa shellprocedurer går både snabbt och smidigt. Det är också enkelt att göra dem tillgängliga för användarna i ett system av den här typen.

Se det som en välkommen service för företagets användare.

6. NÄTVERKSKONFIGURATIONER

Det här avsnittet ska ytterligare åskådliggöra hur man går till väga för att skapa de mest vanliga konfigurationerna. Avsnittet innehåller tre olika typkonfigurationer enligt nedan:

1 Singeldator med en SNA 3270 GATEWAY

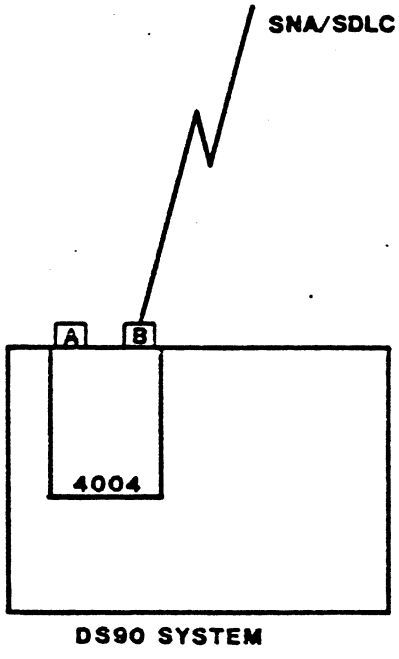
2 Två datorer i ett nätverk med en X.25 GATEWAY

3 Tillägg av ett 4004-kort med en UTS GATEWAY

Dessa konfigurationer bildar slutligen tillsammans en komplex konfiguration utifrån tre grundläggande.

6.1. SINGELDATOR MED EN SNA 3270 GATEWAY

Konfigurationen i det här avsnittet visas i figur1.



En konfiguration av en singeldator med en SNA 3270 GATEWAY mot en IBM dator iordningställs med 7 steg enligt nedan:

1 Skapa en grundkonfiguration

2 Läsa in SNA 3270 från diskett

3 Parametersättning via comparprogrammet

4 SNA 3270 parametersättning

5 Montera kabel mellan 4004-kort och DCE

6 Välj adapter vid behov (X.21 och/eller NRZI-adapter)

7 Starta Netman

6.1.1. Referensanvisning konfiguration 1

Att skapa en grundkonfiguration beskrivs i avsnitten 2.1.1. och 2.1.2.

Parametersättning via comparprogrammet beskrivs i avsnittet 2.2.4. och de menyer som man behöver ändra i är följande:

2. Port definition (se avsnitt 2.2.4)
3. Definition of com-servers (se avsnitt 2.2.4)
4. Com-server references (se avsnitt 2.2.4)

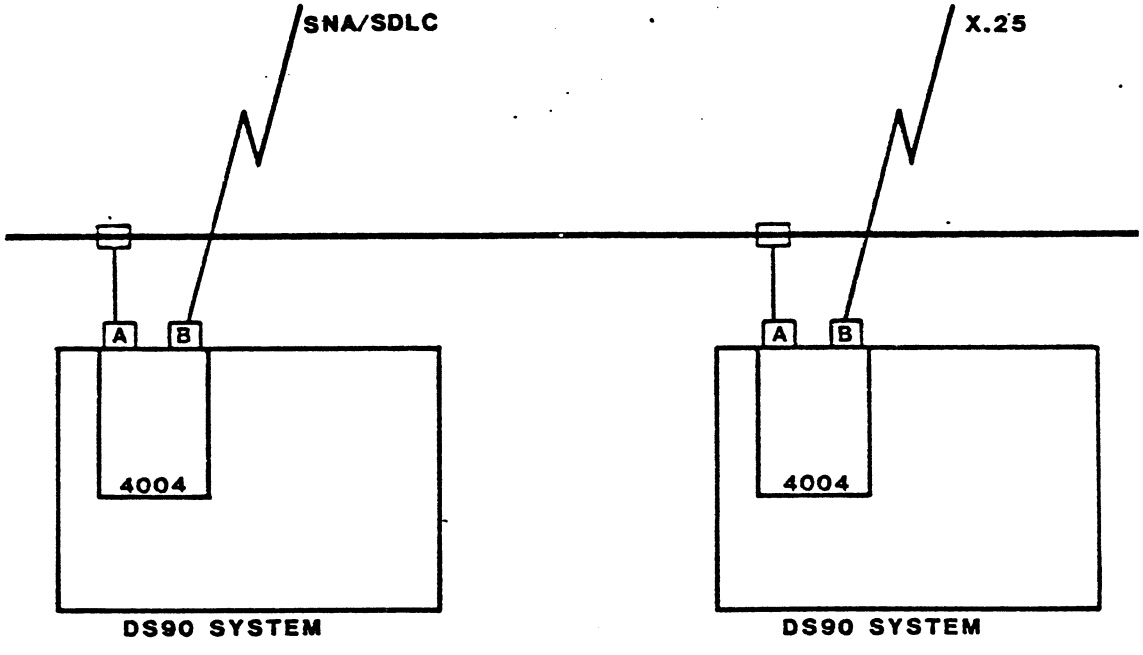
Compar avslutas med att först välja "7. write" i huvudmenyn, och därefter "0. Exit" också det i huvudmenyn.

Anvisningar för parametersättning av SNA 3270 finner du i handboken "NÄTVERK OCH DATAKOMMUNIKATION" under avsnittet "IBM SNA/SDLC 3270". De värden som ligger till grund för detta måste inhämtas från den installation som man ska koppla upp sig mot.

Hur man startar Netman finns beskrivet i avsnittet 2.4.

6.2. TVÅ DATORER I ETT NÄTVERK MED EN X.25 GATEWAY

Konfigurationen för det här avsnittet visas i figur 2.



För den här konfigurationen utgår vi från att den ena datorn är den som vi gjorde i ordning i det föregående avsnittet. Det enda som ytterligare krävs är anslutning till nätverket med en kabel mellan port A på 4004-kortet och kontakten märkt "computer" på linjeadaptern till nätverket.

Åtta steg ska utföras på denna dator enligt följande:

1 Skapa en grundkonfiguration

2 Läsa in X.25 från diskett

3 Parametersättning via comparprogrammet

4 X.25 parametersättning

5 Montera kabel mellan 4004-kort och DCE

6 Montera kabel mellan 4004-kort port A och kontakt märkt computer på linjeadaptern till nätverket.

7 Välj adapter vid behov (X.21-adapter)

8 Starta Netman

6.2.1. Referensanvisning konfiguration 2

Att skapa en grundkonfiguration beskrivs i avsnitten 2.1.1. och 2.1.2.

Parametersättning via comparprogrammet beskrivs i avsnittet 2.2.4. och följande menyer behöver ändras:

2. Port definition (se avsnitt 2.2.4)
3. Definition of com-servers (se avsnitt 2.2.4)
4. Com-server references (se avsnitt 2.2.4)

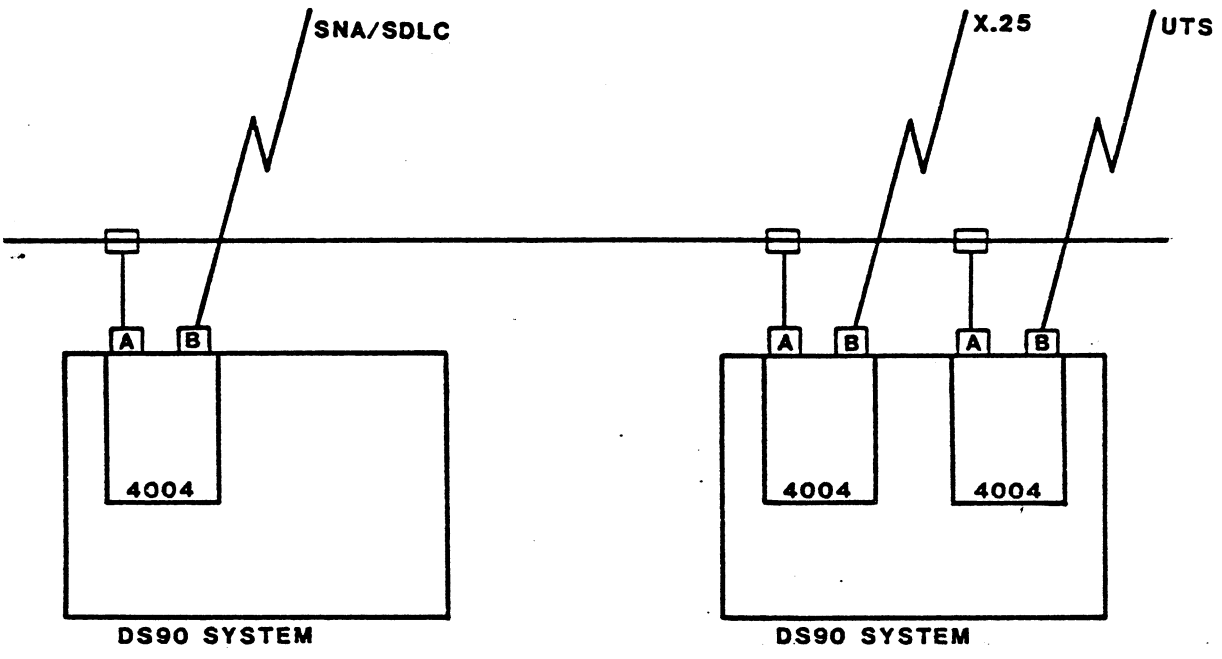
Compar avslutas med att först välja "7. write" i huvudmenyn och därefter "0. Exit" också det i huvudmenyn.

Anvisningar för parametersättning av X.25 finner du i hand- boken "NÄTVERK OCH DATAKOMMUNIKATION" under avsnittet "X.25 DATAPAK". De värden som ligger till grund för detta måste inhämtas från Televerket och/eller den installation som GATEWAYEN ska koppla upp sig mot.

Hur man startar Netman finns beskrivet i avsnittet 2.4.

6.3. TILLÄGG AV ETT 4004-KORT MED EN UTS GATEWAY

Den sista konfigurationen visas i figur 3.



Den här konfigurationen innehåller också en UTS GATEWAY. För att få det i den tidigare skapade konfigurationen utför man följande steg:

1 Lägg till en ny rad för 4004-kortet i filen "/usr/lib/net/cards"

2 Läs in UTS från diskett

3 Parametersättning via comparprogrammet

4 UTS parametersättning

5 Montera kabel mellan 4004-kort port B och DCE

6 Välj adapter vid behov (X.21-adapter)

8 Starta om Netman

6.3.1. Referensanvisning konfiguration 3

Filen "/usr/lib/net/cards" beskrivs utförligt i avsnittet 2.2.2.

Parametersättning via comparprogrammet beskrivs i avsnittet 2.2.4. och de menyer som man behöver ändra i är följande:

2. Port definition (se avsnitt 2.2.4)
3. Definition of com-servers (se avsnitt 2.2.4)
4. Com-server references (se avsnitt 2.2.4)

Compar avslutas med att först välja "7. write" i huvudmenyn, och därefter "0. Exit" också det i huvudmenyn.

Anvisningar för parametersättning av UTS finner du i handboken "NÄTVERK OCH DATAKOMMUNIKATION" under avsnittet "UNIVAC UTS 4000". De värden som ligger till grund för detta måste inhämtas från den installation som man ska koppla upp sig mot.

Hur man startar om Netman finns beskrivet i avsnittet 2.4.

7. KOMMANDON

För att få en utförlig beskrivning av varje netman-kommando cmnämnt i den här handboken finns det ett antal bilagor bifogade. Bilagorna täcker inte de hjälpprogram som finns i operativ-systemets grundsystem och utvecklingspaket, utan dessa återfinnes i systemmanualen under fliken "6 DNIX KOMMANDON".

Kommandot netman finns beskrivet i bilagan NETMAN(1). De övriga kommandon som finns tillgängliga i konceptet Netman, "NCU", "RX" och även programmet "RACCESS", finns i detalj beskrivna i bilagorna NCU(1), RX(1) och RACCESS(1).

För programmerare finns det också ett antal bilagor. De har istället för (1) efter programnamnet beteckningen (5), t ex NETMAN(5).

BESKRIVNING AV FILER FÖR BOOTNING AV KOMKORT

Samtliga filer som beskrivs i detta dokument finns i biblioteket /usr/lib/net/boot. Dom filnamn som har en förklaring är filer som finns idag eller också är de planerade att tas fram. Övriga filnamn är för framtida bruk.

Filerna är indelade i olika grupper vilket innebär att filnamnen är uppdelat i tre delar, del 1 är en konstant som skall vara "boot" , del 2 är en gruppbeteckning om ett tecken, del 3 är ett unikt namn inom gruppen.

ex. boota0100

Gruppbeskrivning

bootpnet Preboot för komkorten 4004 och 4204 när dem skall laddas från nätverket.

bootp4004 Preboot för lokalt komkort av typen 4004

boot0nnnnn Konfigureringsfil för komkortet nnnnn, där nnnnn är komkortets serienummer eller devicenamnet komkortet är kopplat till.

boot1nnnnn

boot2nnnnn

boot3nnnnn

boot4nnnnn Debugger för komkortet nnnnn, där nnnnn är serienumret på kortet.

boot5nnnnn Konfigureringsfil för host till komkortet nnnnn som är placerat i en annan dator (t.ex DB4680), där nnnnn är komkortes serienummer.

boot6nnnnn

boot7nnnnn

boot8nnnnn

boot9nnnnn

bootannnn boota gruppen är en blandning av olika typer av filer (drivers, konfigfiler etc.) som med hjälp av nnnn (som är ett nummer) är indelat i nya grupper.

boota0000 JMPS *

boota0100 Default konfigurerings fil för komkortet 4004, denna fil används för att skapa en ny konfigfil i gruppen boot0nnnnn.

boota0101 Debugger för komkortet 4004.

boota0102 Basprogram utan services till komkortet 4004.

boota0103

boota0104

boota0105

boota0300 Standard DNET driver och protokoll till 4004

!

boota0309

boota0400 Standard Host driver och protokoll (HDLC) för 4004

!

boota0409

boota0500 Standard Gateway-hanterare

!

boota0509

boota0510 Standard UTSCLUSTER service

boota0511 Standard 2780-3780 service

boota0512 Standard SNACLUSTER service

boota0513 Standard X25 service

boota0514 Standard BSCCLUSTER service

boota0515 Standard ASYNKRON service

!

boota0519

```
boota0520   COMMUX service
boota0521   DNET Boot Master (DBM) service
        !
boota0529

boota1000   Default konfigfil för Databoard 4680 host
boota1001   Default OS.8 fil för Databoard 4680 host

boota1100   Remote operator task OS.8 (accessed via ncu)
        !    Övriga tasks som kan laddas till DB4680 host
boota1199
```

netman(1)

NAMN

netman - nätverks transport manager på DNIX-system

SYNTAX

netman (option ...)

BESKRIVNING

Nätverksmanagern är en process som sköter om trafiken till och från DNET och startar processer för att hantera inkommande anrop.

Service för att kontakta det egna systemet eller ett avlägset system identifieras med namn (Symbolic Service String, SSS) enligt DNET.

Nätverksmanagern kopplar sig själv till filsystemet (normalt under namnet /netphys) för att bevaka och ta hand om utgående anrop, den initierar också bootningen av nätverkscontrollern och bevakar och tar hand om inkommande anrop från den.

Optioner används huvudsaklingen för debugging och dom innebär:

-mnamn
: Får netman att koppla sig själv till namn.

-knn
: Sänder signalen nn till nätverks managern som finns kopplad till /netphys. Om inget argument anges sänds SIGTERM (vilket innebär att aktuell nätverksmanager omedelbart terminerar, och detta orsakar i sin tur att alla aktiva uppkopplingar fallerar.

-xnn
: Sätter debugnivån till nn (om DEBUG har kompilerats in)

-f
: Används endast i samband med debugging (motverkar fork när det körs under adb; gäller endast om DEBUG har kompilerats in).

-L
: Förhindrar netman att logga på konsolen.

Inkommande anrop:

Lokala servicar beskrivs i filen /usr/lib/net/servtab. Varje tjänst beskrivs på en rad som innehåller:

(1) ett minustecken eller service nummer
(2) service namn
(3) pathname till ett program som startar för att ta emot det inkommande anropet.

Service nummer används sällan direkt i tillämpningar eftersom servicarna är tillgängliga med namn. Ett minustecken här anger att netman ska välja nummer automatiskt. Vissa tjänster har dock kända nummer (se SSS/04-06-28/al) och i så fall kan tjänstens nummer bli använt om man specificerar det här.

Servicenamn används av tillämpningar som önskar utnyttja servicen. Om strängen i servicetabellsfilen innehåller '$S' så ersätts '$S' av systemnamnet som finns i /etc/systemid så att samma fil kan utnyttjas i olika system. Om en tjänst är identisk på flera system och att det inte spelar någon roll till vilket system anropet går, kan namnet vara helt identiskt på flera system, normalt är dock (för remote login och liknade) att servicenamnet innehåller systemnamnet med en prefixsträng för att identifiera typen av service.

Pathname används för att starta en hanterare för varje inkommande anrop. När ett inkommande anrop anländer startar en process med användarid och gruppid tagna från filerna uid och gid. Processen ges "-B" och tjänstens namn som parametrar. (Tjänstenumret används enbart som en adress, följaktligen ges den inte som parameter till processen). Processen får också ett virtuellt anrops fd som dess fd #0. Alla andra D-NIX fd:s stängs och aktuellt bibliotek sätts till "/".

Utgående anrop.

Utgående anrop sänds ut genom att man utför en öppning på det kopplade namnet (dvs /netphys/tjänstenamn). Om öppningen blir lyckad retuneras en virtuell anrops fd och en hanterare har startats i något system som också har en virtuell anrops fd uppkopplad i den andra änden av det virtuella anropet. Kan servicen inte nås misslyckas öppningen E- NOENT.

Utgående anrop kan också genomföras med numerisk adress genom att använda ett namn som består av numret föregånget av ett nummertecken (#). Numret används direkt i ett anropspaket på DNET som bara kontrollerar att det är <= 15 siffror. DNET mjukvara skickar tal med 1-2 siffror till ursprungliga systemet med service nummer lika med det anropade numret, och tal med 5-7 siffror skickas till det system vars 4004-kort har ett serienummer som motsvarar de fem första siffrorna och varvid servicenummer tas från de resterande siffrorna. Numerisk adressering tillhandahålls huvudsakligen för debugging.

Omhändertagna signaler:

Netman fångar vissa signaler:

SIGHUP får netman att vänta tills den sista uppkopplingen har gått ner. När så skett startas det en ny kopia av programmet. Detta kan vara användbart när man skall installera en ny version utan att behöva avbryta pågående nätverks arbete.

SIGINT får netman att uppfatta kortet som kraschat vilket innebär att det blir ombootat. Används mest vid debugging.

Bootning

Netman använder en annan process bootman för utföra den egentliga bootningen. Denna process kan också användas som chef för remote-bootning av andra system. Den kan också boota andra 4004-kort för andra ändamål i samma maskin. För att ha möjlighet att vara chef för remotebootning måste servicen finnas laddad i 4004-kortet. Om så är fallet använder bootman den automatiskt.

För att möjliggöra bootning av andra kort måste deras typ och namn listas i en fil med namnet /usr/lib/net/cards. Varje kort beskrivs på en rad. Först korttypen (för närvarande är endast 4004 möjligt) åtföljt av minst en tab och/eller blank och sen kortets device namn (utan det inledande /dev/).

FILER

/etc/systemid	Maskinens namn.
/usr/lib/net/servtab	Servicetabell för lokala servicar.
/netphys	Default kopplingsplats.
/tmp/netlog????	Här skrivs felmeddelanden.
/tmp/netswap????	Används för att spara buffrar när många buffrar används.
/usr/lib/net/bootman	Program för att boota kom-kort.
/usr/lib/net/cards	Fil innehållande namn på andra kort som skall bootas.
/usr/lib/net/boot/*	Bibliotek som innehåller data att boota kort med.

HÄNVISNING

netman(5), ncu(1), rx(1), raccess(1)

ANM

Netman skall endast startas av root och normalt görs detta från filen /etc/rc.

Netman skall ägas av root men skall inte ha setuid-biten satt .

ncu(1)

NAMN

ncu - nätverksanrop i DNIX

SYNTAX

ncu (option ...) systemid

FUNKTION

Ncu anropar ett annat DNIX-system via ett nätverk. Det hanterar en interaktiv dialog med möjlighet till filöverföring. Systemid är maskinens namn. En kopia av ncu startas på remotemaskinen och denna kopia startar en login-process och agerar terminal till denna remotelogin. På den anropande sidan körs ncu som två processer: sänd processen läser standardinput och skickar det mesta vidare till remotemaskinen, mottagarprocessen läser från remote-systemet och skickar alla data till standard output. Ncu använder stty/gtty möjligheterna i den lokala terminalen och skickar data bara när en läsning kan lyckas. Detta medför vissa konsekvenser:

- Linje redigering görs lokalt.
- Eftersom data överförs som om det skulle läsas kan 8-bitars binärdata (exekverbara program t.ex) flyttas med fileöverförings-möjligheten.
- Läses tomma tecken, överförs de som sådana, INTE som specialtecken.

Sänd processen tolkar en rad som börjar med 'ü' som följer:

ü.	avslutar konversationen
üEOT	avslutar konversationen
ü<fil	skicka innehållet i fil till remote systemet, som om den skrevs på terminal.
ü!	startar en interaktivt shell på det lokala systemet
ü!kommando	kör kommandot på det lokala systemet (via sh -c)
üü...	skickar raden éü...'.

Filer kan läggas upp på remotesystemet med följande sekvens: (när inloggning på remotesystemet har gjorts, behövs det avslutande EOT som en tom läsning till cat och som därmed avslutar kopieringen till destinationen):

```
cat >remote_dest
ü<lokal_fil
EOT
```

Filer kan tas från ett remote system genom att först anropa det systemet med ncu och sedan (efter inloggning) åter anropa det lokala systemet. Efter återinloggning på det lokala systemet ges kommandon som följer (Det dubbla ü:et (tilde) används som citerat ü på den lokala maskinen för att få den remote startade ncu-processen att ta sin input från önskad fil):

```
cat >lokal_dest
üü<remote_fil
EOT
```

Optioner används huvudsakligen för debugging och dom innebär:

-s I stället för systemid ges ett servernamn. Default login server prefixet 'L_' undertrycks.

-B Detta kommando används av netman för att starta B-sidan. Skall aldrig ges explicit.

-mnamn
Får ncu att använda den nätverksmanager som kopplad till namn.

-xnn Sätter debugnivån till nn (om DEBUG kompilerats in).

FILER

/netphys	Ncu använder en netman kopplad här.
/tmp/ncuXXXXXX	B-sidan av ncu ansluter sin pseudoconsole här.
/bin/login	Startas på B-sidan.
/bin/sh	Exekverar ü! kommandon.

HÄNVISNING

netman(1), rx(1), ncu(5), raccess(1)

ANM

ncu skall ägas av root men skall ej ha setuid-biten satt.

TIOCSETN hanteras för närvarande som TIOCSETP.

Följande funktionskoder understödjs och skickas vidare till remote-maskinen: F_OPEN, F_CREATE, F_LOCATE, F_LINK, F_UNLINK, F_CLOSE, F_READ, F_WRITE, F_CANCEL, F_IOCR, F_IOCW.

Iocontrols med en buffer innehållande mer än en byte måste beskrivas filen /usr/lib/net/ioctltab för att raccess skall tillåtas utföra byteswapping.

Stat-anrop (ioctl subfunktioner) hanteras speciellt: remote anropets fält st_uid och st_gid översätts om möjligt till symboler vid sändning och sen tillbaks till dom numeriska på utsändarsidan. Om det inte finns någon användare med samma namn försöker man med netxgt och som en sista utväg antar man 0xffff. Om det inte finns någon grupp med samma namn antar man other och som en sista utväg antar man 0xffff.

Fältet st_dev är översatt för att försäkra konventionen att paret (st_dev,st_ino) definerar en unik fil (sett från en maskin). Observera att en och samma fil har olika st_dev fält när den accessas från olika maskiner.

Optioner används huvudsakligen för debugging och de innebär:

-B Optionen används av netman för att starta B-sidan. Skall aldrig anges explicit.

-tnn Sätter tiden man skall hålla en uppkoppling uppe utan någon fil öppning till nn sekunder (Default är 30 sekunder).

-mnamn
Får raccess att koppla sig själv till namn (används huvudsakligen vid debug).

-s Förhindrar insättning av server-prefixen (används huvudsakligen vid debug).

-sx Sätter första karaktären i anropsservernamnet till x (används vid debug).

-xnn Sätter debugnivån till nn (om DEBUG har kompilerats in. Sätter samma debugnivå på den anropade sidans program.

Formatet på filen /usr/lib/net/ioctltab är som följer:
Varje rad har formatet

SUBFN (fmt1 (fmt2))

SUBFN är ioctl-subfunktioner i decimalt värde. Om endast fmt1 anges används det för att konvertera dataöverföringen i båda riktningarna om det går. Om två format anges används fmt1 för WRITE och fmt2 för READ (för READ-WRITE används ett formet för varje riktning).

Varje format innehåller tecken typer dessa kan inledas med en siffra som anger antal

l innebär long (32-bit) kvantitet.
s innebär short (16-bit) kvantitet.
c innebär character (8-bit) kvantitet.

avslutande tecken av typen c i ett format behöver inte specificeras. Det kan endast finnas tecken av typen c efter de 64 första byten om ett format angivits och efter 30 byte om två format används. Ex:

```
1234    16c4l8s4l    : I detta exemplet på format kan det efter de
                       sista 4l endast följa tecken av typen c.
```

FILER

/etc/systemid	Maskin namn.
/usr/lib/net/ioctltab	Twist tabell för iocontrols.
/netphys	Raccess använder netman kopplad här.
/tmp/rdevxlate	Översättningstabell för st_dev fält. (underhålls automatiskt)

HÄNVISNING

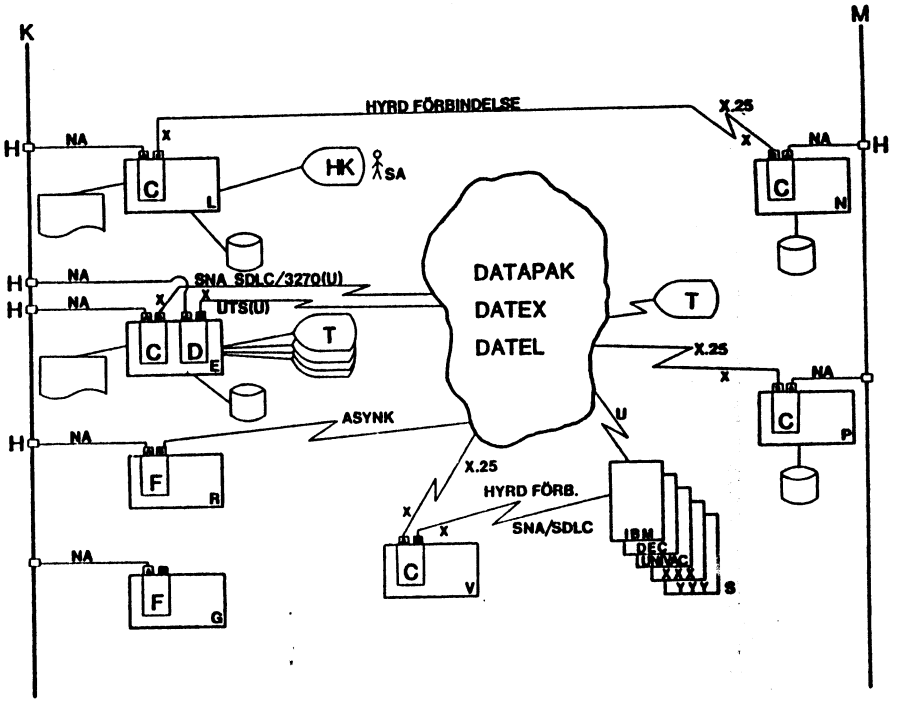
netman(1), ncu(1), rx(1), raccess(5)

ANM

Raccess skall ägas av root men skall inte ha setuid bit satt.
Raccess skall normalt startas automatiskt från /etc/rc

TERMINOLOGILISTA

HK	HUVUDKONSOL	Den terminal som systemadministratören är verksam från
NA	NÄTVERKSANSLUTNING	Från A-kanal på 4004-kort till linje-adapter
SA	SYSTEMADMINISTRATÖR	SA är en person som läser Netmanhand-boken och ombesörjer installation av konceptet
A	A-KANAL	A-kanal på 4004-kort
B	B-KANAL	B-kanal på 4004-kort
C	4004-KORT	Huvudkort
D	4004-KORT	Tilläggskort
E	4004-KORT	Fjärrbootat kort
F	NÄTVERKSDATOR	Utifrån systemadministratörens placering, en dator ansluten till nätverket.
G	MÄTDATOR	Kan t ex vara en Databoard, som fjärr-laddas från någon dator med sekundär-minne. Laddningen bör utföras från L eller E, men kan också laddas från N, P eller V
H	LINJEADAPTER	Anslutningsdosa till nätverket.
K	LOKALT NÄTVERK	Lokalt avgränsat nätverk t ex inom en kontorsbyggnad
L	LOKAL DATOR	Den dator som systemadministratören eller annan användare är direktansluten till via V24/V28 gränssnitt med asynkron terminal
M	FJÄRRBELÄGET NÄTVERK	Utifrån systemadministratörens place-ring; ett nätverk placerat på annan ort dvs anslutet med någon typ av förbind-else hyrd från televerket
N	FJÄRRDATOR	Fjärrbelägen dator ur samma familj (DS90) ansluten till ett fjärrbeläget nätverk (M)
P	NÄTVERKSDATOR I FJÄRRBELÄGET NÄTVERK	



R	EXPANSIONSRACK	En låda som man placerat 4004-kortet i för att förse detta med ström
S	ANDRA LEVERANTÖRERS DATORER	Dessa ansluts via en gateway
T	ASYNKRON TERMINAL	
U	ANNAN LEVERANTÖRS DATAKOMMUNIKATIONS-PROTOKOLL	
V	SINGEL FJÄRRDATOR	V har inget eget nätverk
X	GATEWAY	Port via vilken användaren kan kommuni-cera med andra leverantörers datorer

rx(1)

NAMN

rx - exekverar ett kommando på en annan maskin

SYNTAX

rx (option ...) systemid!kommando (parametrar ...)

FUNKTION

Rx loggar automatiskt in på en annan maskin och exekverar ett kommando där. Om remoteprogrammet läser standard input kopieras det lokala rx-kommandots standard input via nätverket och matas som standard input till remotekommandot. Remotekommandots standard output och standard error kopieras på det lokala rx-kommandots standard output och standard error. Om remotekommandot slutar normalt (med exit(2)) avslutar rx med samma kod. Om remote kommandot avbryts avslutar rx med exit kod 1.

Det normala är att rx letar i aktuellt HEMbibliotek efter filen .netkey för att hitta information om inloggningsrutinerna på remote systemet. Filen .netkey innehåller rader med tre fält på varje:

- ett systemid eller '*' för att ange vilken maskin som helst.

- användarnamn (uid) att använda vid login på remotemaskinen.

- ett nyckel som skall matcha på remotemaskinen.

Ursprungssidan genomsöker den lokala .netkey filen efter ett namn som matchar angiven remotemaskin och plockar sedan uid och nyckel från den posten. På remotesidan startas en annan kopia av rx som först letar i det systemets passwordfil efter angivet uid och sedan i dess $HOME/.netkey fil efter systemid och jämför därefter uid och nycklar. Om nycklarna är lika utförs remote-exekveringen. Eftersom man jämför både uid och nyckeln är det inga problem att ha fler användare som har samma login-bibliotek men olika privilegier.

Eftersom .netkey filerna genomsöks efter destinations systemid på båda sidor, kan en användare ha identiska filer på flera maskiner, även om dennes uid och/eller nyckel varierar från maskin till maskin. Användaren kan hantera sin .netkey-fil efter behag utan att ha root-privilegier.

Som en säkerhetsåtgärd ändrar rx skyddet till 0600 på alla .netkey filer den använder, detta utifall användaren skulle glömma göra sin nyckel hemlig.

Följande signaler fångas av rx och överförs till det remote exekverade kommandot: SIGINT, SIGQUIT, SIGPIPE och SIGTERM. Går linjen ner får SIGHUP skickas till remote kommandot. Andra signaler på lokalsidan får rx att avsluta och därmed går linjen ner och SIGHUP ges på andra sidan.

Environment kopieras till remote-maskinen med två undantag:

HOME sätts till loginbiblioteket på den andra maskinen.

PATH modifieras för att återge det nya HOME: på alla ställen där hembiblioteket förekommer i vägvalen ersätts de med HEMbiblioteket på den andra maskinen (om inte hembiblioteket helt enkelt är på den ursprungliga sidan).

Optioner och deras betydelse:

-luid(:(password))

Läser över den automatiska inloggningen och loggar in som uid med password på den andra maskinen. Detta password är normala som används av login(1) och kräver inte närvaron .netkey fil på remote-maskinen.

-ddir

Ändrar aktuellt bibliotekt till dir innan kommandot exekveras Normalt exekveras kommandona i login-biblioteket på remote maskinen.

-B Denna option används av netman för att starta B-sidan. Skall aldrig anges explicit.

-mnamn

Får rx att använda den nätverksmanager som är kopplad till namn.

-xnn

Sätter debugnivån till nn (om DEBUG har kompilerats in)

FILER

/netphys	Rx använder en netman kopplad här.
/tmp/rxXXXXXX	B-sidan av rx ansluter sin pseudoconsole här
/etc/systemid	Maskinnamn

HÄNVISNING

netman(1), ncu(1), rx(5), raccess(1)

ANM

rx skall ägas av root men skall ej ha setuid-biten satt.

Interaktiva program bör ej exekveras med rx. Använd istället ncu.

Filer kan kopieras till och från system genom att använda dd med respektive option if= eller of=. tar(1) kan också användas, men om förbindelsen går ner under överföringen skapas det en ej fullständig fil.

Komplicerade kommandon kan exekveras på anropssidan genom att använda sh med optionen -c. Exempel:

```
rx -d/usr/lib/net cpu1!sh -c "l *04 ö lpr"
```

listar alla filer på maskinen cpu1 som slutar med 04 på sin skrivare.

EXEMPEL

Ex 1. rx cpu1!ps lx

Anropet innebär att man i en maskin (med namnet cpu1) på nätverket utför kommandot för att lista processtatus. I exemplet är det förusatt att det finns en .netkey-fil med ett tänkbart utseende som följer på båda maskinerna.

```
cpu0        user1        mitt_nyckelord
cpu1        user1        mitt_nyckelord
cpu2        user1        mitt_nyckelord
```

x 2. rx -lroot:ds90 cpu3!cat etc/passwd

Detta innebär att man anropar en maskin med namnet cpu3 och där listar ut password filen på standard output. Man har här skickat med login-parametrar för superuser-privilegier på cpu3.

NAME

netman - format on requests on network transport server

SYNOPSIS

```
#include <net/pk_transport.h>
```

DESCRIPTION

The network manager deals with data messages that can have infinite length (implemented with the X25 M-bit) and interrupt messages that can have a length of zero or one byte. The data messages have a qualifier bit that can be 0 or 1. (Implemented with the X25 Q-bit. Can be used as a command/data bit). Note: there is only one flow of data so messages with or without Q-bit are delivered in order. Interrupt messages have a separate flow that has priority over the data flow. Interrupt messages are less efficient and are being used mainly to convey state changes independently of a possibly stuck data flow.

Data is transferred by READ and WRITE requests on the virtual call fd. The buffer used by these calls contains a leading control byte in which two bits are currently being used. PH_Q is the qualifier bits mentioned above. PH_M means that more data in the same message follows in another buffer. (PH_M is used only in the communication between the application and netman, it does not affect splitting between packets on the network.) PH_Q should be constant in all write requests that make up a message, otherwise the result is unpredictable.

Interrupt packets are sent with an IOCW request with subfunction NIO_WOB and received with an IOCR request with subfunction NIO_ROB.

The above values as defined in the include file <net/pk_transport.h> are:

```
/* Interface between level 3 and level 4.
 * that is between netman and its applications*/

#define PH_M    1          /* Indicates more data after this request *
#define PH_Q    2          /* Message phase*/

#define PACKFD  0          /* Virtual call fd on the called side of
                            * a connection*/

#define NIO_ROB ('n'<<8 | 1)/* ioctl to read out of band data */
#define NIO_WOB ('n'<<8 | 2)/* ioctl to write out of band data */
#define NIO_CHAN     ('n'<<8 | 3)/* ioctl get channel number*/
```

Write and IOCW(NIO_WOB) requests complete as soon as the data is moved out of the user space, not when it has reached its destination.

When the connection is broken all requests give error EREQAB. Cancelled requests (by signals or CANCEL requests) give error EINTR.

When a virtual call fd is closed the connection is broken immediately and any message not yet delivered is lost.

NOTES

If one side wants to write some information and then hangup without loosing information it should write the information and then follow it by a message meaning invitation to hangup and wait for the other side to hangup. This is determined by reading the virtual call fd (ignoring anything received) until it returns error EREQAB and then closing the virtual call fd. When a reading program receives an invitation to hangup it knows that no more information will arrive and it can close the virtual call fd immidiately. (see ncu(5) for an example of this protocol).

It may be useful to have an IOCR(NIO_ROB) request active even if no interrupt message is expected just to be able to see if/when then connection goes down. This is particulary true in applications that does not constantly have a read or write request to the virtual call fd.

SEE ALSO

netman(1), ncu(5)

NAME

packet protocol for terminal access

SYNOPSIS

#include <net/pk_ncu.h>

DESCRIPTION

This protocol is specified rather close to the UNIX version III terminal behaviour. It does not in any way restrict the capabilities of the UNIX terminal interface.

It is beleived that the protocol could very well be useful as the communication between the background and the foreground parts of the terminal interface within the same computer as well as communicatinmg between remote sites over a X25 or DNET virtual circuit.

The protocol assumes that there is a sequenced data flow in both directions of at least two datastream types: data and control. These are implemented with the Q bit in X25.

The protocol further assumes that there exists a means for expediated data transfer where short messages (at least one byte) can be transferred even if the normal sequenced data flow is blocked. This is implemented with interrupt messages in X25.

In the following the word message is used for one or more packets (as the packet size requires). In X25 message boundaries are indicated by the absence of the M-bit in the sequenced flow and interrupt messages are always one packet.

DATA

Data is transferred as it would appears in read and write requests under UNIX. In X25 data is transferred in the sequenced message stream with the Q bit off.

Data from a terminal is transmitted whenever a read from a local terminal would unblock, that is when a complete line has been assembled in normal mode or after a single character in RAW and CBREAK modes. The contents of a message correspond to what one read with a large bytecount would return.

Data to a terminal is transmitted asyncronously with write requests from the writing task. Whenever possible as much data as possible is collected into the same message.

Data is handled only when the READ_FLUSH counter (described below) is zero, otherwise data is ignored.

FLUSHING

In order to be able to reliably discard data in one or both directions there is a flush protocol:

Each side maintains two counters: the READ_FLUSH counter and the WRITE_FLUSH counter.

While the READ_FLUSH counter is non-zero all data is discarded but commands are obeyed. The READ_FLUSH count is incremented when interrupt messages arrive that have the FORWARD_FLUSH bit set or when the local side wants to flush incoming data. The READ_FLUSH count is decremented when a command packed called FLUSH_MARK (described below) arrives.

While the WRITE_FLUSH counter is non-zero FLUSH_MARK commands shall be sent and the WRITE_FLUSH counter shall be decremented for each FLUSH_MARK successfully sent. The WRITE_FLUSH counter could be incremented for several reasons: The local side just sent an interrupt message with the FORWARD_FLUSH bit set, or an interrupt message is received with the REVERSE_FLUSH bit set, or a FLUSH_MARK_REQUEST command is received in the datastream.

To flush the outgoing stream immideately: Send an interrupt message with the FORWARD_FLUSH bit set, then increment the WRITE_FLUSH counter and consequently send a flush mark. The other side will see the interrupt message and increment its READ_FLUSH counter and ignore data until the corresponding FLUSH_MARK causes the READ_FLUSH counter to return to zero.

To flush the ingoing stream immideately: Send an interrupt message with the REVERSE_FLUSH bit set, then increment the READ_FLUSH counter, and start ignoring input while the READ_FLUSH counter is non-zero. The remote side will see the interrupt message and increment its WRITE_FLUSH counter and consequently send a FLUSH_MARK which will return the local READ_FLUSH counter to zero.

To flush both streams: Simply send an interrupt message with both flush bits (FORWARD_FLUSH bit and REVERSE_FLUSH bit) set, and then increment both READ_FLUSH counter and WRITE_FLUSH counter. Both streams will drain as described above.

To let output drain and then flush input: Send a FLUSH_MARK_REQUEST command and increment the local READ_FLUSH counter. When all data has reached the destination the other side will see the FLUSH_MARK_REQUEST and therefore increment its WRITE_FLUSH counter and thus send a FLUSH_MARK. The local READ_FLUSH counter will return to zero when the FLUSH_MARK arrives.

COMMANDS

In X25 commands is transferred in the sequenced message stream with the Q bit on. Each message carry one command. Note that commands are interpreted even when data is ignored due to the READ_FLUSH counter. The commands as defined in the include file <net/pk_ncu.h> are:

```
/* level 4 definitions for remote login (ncu)*/

#define Q_CLRINV      0     /* Invitation to clear*/
#define Q_SIGNAL      1     /* Signal*/
#define Q_FLUSH       2     /* Flush mark*/
#define Q_FMRQ        3     /* Flush mark request*/
#define Q_ERROR       0x0f  /* Bad command received*/

#define Q_STTY        0x10  /* gtty/stty*/
#define Q_SETC        0x11  /* setc/getc (special chars)*/

#define Q_TERM3    0x18 /* SYSTEM III termio extended by some extras *
#define Q_T3NF     0x19 /* Like Q_TERM3 but no flushing */

#define Q_P_PAR     0x20 /* Protocol parameters A Larsson <alEdiab> */
#define Q_P_START   0x21 /* Start/Stop protocol A Larsson <alEdiab> */

/* Error codes  */

#define QE_USPEC     0     /* Unspecified error*/
#define QE_CMD       1     /* Illegal command*/
#define QE_PAR       2     /* Illegal parameter*/

/* Bits in interrupt message*/

#define I_SIGNAL    0x1f  /* An interrupt message can carry a signal */
#define I_BACK      0x40  /* Flush back direction*/
#define I_FORW      0x80  /* Flush forward direction*/
```

INVITATION_TO_CLEAR (no parameter). This command tells the remote side to clear the connection. If a connection is simply cleared any message not yet received is lost. Therefore if one side wants its output to drain and then clear the connection it can send INVITATION_TO_CLEAR. (This function can also be implemented by sending FLUSH_MARK_REQUEST, waiting for the corresponding FLUSH_MARK and then clear the connection, but in this case the remote side would not know why the connction went down)

SIGNAL (with parameter) Sent from a controlling terminal to stop looping processes. The parameter is the signal number, typically SIGINT or SIGQUIT. SIGHUP is typically not sent this way, it is generated when the connection is cleared. An interrupt packet can also carry a signal.

FLUSH_MARK (can have parameter) decrements the READ_FLUSH counter and thus cause the following data to be accepted. If there is a parameter it is interpreted as a signal number (see above).

FLUSH_MARK_REQUEST increment the WRITE_FLUSH counter and thus cause a FLUSH_MARK to be returned.

STTY (can have parameter) This message is used to implement the ioctl subfunctions TIOCGETP and TIOCSETP (gtty and stty in older UNIX versions) If there is no parameter the message shall be interpreted as an enquiry. If there is a parameter then it shall be interpreted as an indication of the current terminal parameters if the message goes from the terminal (as a reply to an inquiry) or as an order to modify the terminal parameters if the message goes in the other direction. If there is a parameter, it shall have the same format as the ioctl block as described in <sgtty.h>. Multiple byte quantities shall be HIGH-endian (high order first).

SETC (can have parameter) This message is used to implement the ioctl subfunctions TIOCGETC and TIOCSETC. If there is no parameter the message shall be interpreted as an enquiry. If there is a parameter then it shall be interpreted as an indication of the current special characters if the message goes from the terminal (as a reply to an inquiry) or as an order to modify the special characters if the message goes in the other direction. If there is a parameter, it shall have the same format as the ioctl block as described in <sgtty.h>. Multiple byte quantities shall be HIGH-endian (high order first).

ERROR (with parameter) If a command is not understood an ERROR message shall be sent. The parameter field is one byte error code followed by at least 10 bytes copied from the beginning of the offending command. ERROR commands must never generate an ERROR commad: a bad ERROR command shall be ignored. If several errors occur and the protol handler doesn't have storage to keep all causes before the error packet is sent the it is permissible to send an ERROR command without parameter. This is legal ONLY when there are several errors and must not be used as a 'cheap' error handling.

SEE ALSO

ncu(1), netman(5)

NAME

packet protocol for remote file access

SYNOPSIS

#include <net/pk_raccess.h>

DESCRIPTION

When the originating side of raccess gets a request that has to be sent to the other party it sees if there is already a call to the other side. If not – it makes one.

The originating side issues commands that the slave side obeys rather blindly. Each connectioncan have 31 request channels (numbered 1–31) for each with a private buffer allocated on the slave side. The master side controls the selection of which channel to use and also chooses buffer size when a new channel is allocated. There is also a special request channel (numbered 0).

For each request there is one message going from the master to the slave to initiate it, and then a message going from the slave to the master telling that the request is done. It is illegal for the master to issue another request on the same channel before the reply has been received. All messages are sent with teh X25 "Q" bit off. Requests consists of a struct omsg optionally followed by data (Filename or data to write/IOCW). Replies consists of a struct imsg optionally followed by data (read data or IOCR). There are three exceptions to this sceme:

- F_CLOSE messages get sent on channel 0. The slave does not send any reply on this message.
- F_uxlate messages get sent on channel 0. The slave does not send any reply on this message.
- F_CANCEL messages can be sent on a busy channel. It is a hint to the slave that it may cancel the operation in progress. (Note that the slave may already have terminated the request and sent a reply. This means that not all F_CANCEL messages are acted upon by the slave.)

Certain requests require that an uid is known on the slave side (those with filenames). In these cases it is the masters responsibility to transmitt a message about the translation before the actual request. The slave is required to keep these translations indefinitely so the master may remember for which uids a translation is known by the slave. The uid field in the request is always expressed in terms of the originating system and the slave should always have a translation ready for that uid except when the keys differed in which case the slave immediately replies with status EACCESS.

Structures as defined in the include file <net/pk_raccess.h> is:

```
/*
 * Request sent from A-side to B-side
 *      */
struct omsg {
        char om_thdr ;     /* Transport level header byte*/
        unsigned char om_chnum ;/* Channel id*/
        unsigned char om_fc ;/* Remote function code*/
        unsigned char om_fd ;/* Remote file descriptor*/
        long om_bc ;       /* Byte count*/
        long om_bsize ;    /* Buffer size*/
        long om_upar ;     /* User par or random address*/
        long om_nph,om_npl ;/* Network wide requester id*/
        union   {
                struct {
                     short omun_uid, omun_gid ;
                     char omun_mytwist ;
                     char omun_debug ;
                } omu_norm ;
                struct {
                     unsigned long omus_m[2] ;
                } omu_spec ;
        } om_u ;
} ;
#define TWNORM  "4c5iss"
#define TWSPEC  "4c7l"

/*
 * Reply sent from B-side to A-side upon request complete
 *      */
struct imsg {
        char im_thdr ;     /* Transport level header*/
        unsigned char im_chnum ;/* channel id */
        unsigned char      im_1,im_2 ;/* spare */
        long im_kpar ;     /* Open privileges, append address*/
        long im_rpar ;     /* Return parameter*/
} ;
#define TWREPLY "4cll"

/* Direction bits in chnum call*/
#define DATAOUT 1
#define DATATWIST    0x20 /* Data should be twisted on b side
                           * also selects special format of
                           * om_u*/
#define DATAIN       0x40
#define DATARND      0x80 /* Random address in request*/
#define CHMSK        0x1f /* Channel number mask*/

/* Error values from B-side to A-side*/
#define Erabuf  (-900)     /* Cannot allocate buffer*/
#define Eseq    (-901)     /* Sequence error*/
#define Ebcnt   (-902)     /* Request overflows buffer*/
```

```
/* Special function codes used only over the net*/
#define F_uxlate      253   /* Uid info translation*/
#define F_stat        254   /* Stat request with translation */
#define F_symstat     255   /* Stat - symbolic version*/
/*
 *      Symbolic stat stuff.
 */
#define NIO_SYMSTAT   ('n'<<8 | 100)
#define SY_IDLEN      20

#define Devxlate      "/tmp/rdevxlate"
#define Defuname      "netxqt"
#define Defgname      "other"

struct symstat  {
        unsigned short    sy_dev ;
        unsigned short    sy_ino ;/* Not twisted to match dir entry */
        unsigned short    sy_mode ;
        unsigned short    sy_nlink ;
        unsigned short    sy_rdev ;
        unsigned short    sy_1 ;/* Pad to make align 4 */
        long    sy_size ;
        long    sy_atime ;
        long    sy_mtime ;
        long    sy_ctime ;
        char    sy_sysname[SY_IDLEN] ;
        char    sy_uname[SY_IDLEN] ;
        char    sy_gname[SY_IDLEN] ;
} ;
#define TWSYST  "s2c4s4l"
```

All messages (requests and replies) are expressed in the masters byteorder. Twisting is done by the slave side based on the omun_mytwist field in the first request when a connection is first established. How requests and replies are twisted is built into raccess. Filenames and data for F_READ and F_WRITE is expected to be a stream of bytes and thus not twisted. Data for F_IOCR and F_IOCW is twisted according to a table in the file /usr/lib/net/ioctltab on the maser side if there is a more than one byte of data. In order to enforce portability this data is required even when communication between computer of the same family. Twisting info for ioctl is encoded into the omus_m field.

There are some special function codes and error codes used only between master and slave:

F_uxlate
: is used in the messages that transfer uid translation. The data part contains null-terminated argument string each beginning with a character indicating how to interpret the rest of the string. Currently the characters can be 'U' for user name and 'K' for key to match in .netkey.

F_stat
: is used instead of F_IOCR and SF_STAT and makes the slave send symbolic (struct symstat) data as a reply instead of a struct stat. This allows the master to reply with a status structure that fulfills the convention that st_ino and st_dev together form an unique serial number for a file even if the files reside on different systems (it also handles the rare cases when a request passes through several instances of raccess).

F_symstat
: is currently unused. It is a synonym of F_stat that will possibly go away.

Erabuf
: is generated by the slave if it cannot allocate the requested buffer. The master considers the channel as un-allocated and the error E2BIG is reported to the user.

Eseq is geneated by the slave when the master has issued a command on a bad ot busy channel. Should not happen.

Ebcnt
: is generated by the slave when the master attempts to make a commad with a buffer that is bigger than allocated. Should not happen.

SEE ALSO

raccess(1), netman(5)

NAME

packet protocol for remote execution

SYNOPSIS

```
#include <net/pk_rx.h>
```

DESCRIPTION

The commands as defined in the include file <net/pk_rx.h> are:

```
#define Q_EXIT          1     /* Exit*/
#define Q_JOB           2     /* Job definition*/
#define Q_SIERR         3     /* Standard error read failed*/
#define Q_STDERR        4     /* Data to write on stderr*/

#define I_RDON          1     /* Start read standard input*/
#define I_RDOFF         2     /* Close standard input*/
#define I_SIG           0x40  /* Signal*/
#define I_SOERR         0x80  /* Error on stdout*/
#define I_SEERR         0xc0  /* Error on stderr*/

#define I_FMT           0xc0  /* Format bits*/
```

First a JOB message is transferred to the called machine. It contains several null-terminated argument strings each beginning with a character defining how to interpret the rest of the string. Currently the characters can be:

U uid for login

K key to match in .netkey for automatic login

P password as in login(1) when -l switch is used in rx(1)

D current directory

C command name

A argument. Each argument is passed as a separate string each beginning with 'A'

E environment,. Each environment variable is passed as a separate string each beginning with 'E'

When the remote side reads standard input for tyhe first time an interrupt message with I_RDON is sent to start reading standard input on the local side. Standard input data is moved with the Q-bit off. End of file is indicated with an empty message with the Q-bit off. If reading standard error fails a Q_SIERR message is moved with errno as data, and the reading of standard input stops. When standard input is closed in the receiving side an interrupt message with I_RDOFF is sent to the local side.

Standard output data is moved without the Q-bit set. End of file is indicated with an empty message without Q-bit. Errors writing on standard output is reported in the other direction by interrupt messages with data I_SOERR+errno.

Standard error data is moved in Q-packets beginning with Q_STDERR. End of file is indicated with a message containing only Q_STDERR and no data. Errors writing on standard error is reported in the other direction by interrupt messages with data I_SEERR+errno.

Signals are transferred by interrupt messages with data I_SIG+signal_number

When the remote command terminates a Q_EXIT message is sent with exit code as parameter (or 1 if the command aborted).

SEE ALSO

rx(1), netman(5)

DOCUMENTATION COMMENT REPORT Nr: _____

Originator: _____ Date: _____
(Name, Address, _____
Telephone) _____
_____

Manual/Datasheet: _____
_____
Version & Date: _____

Related products _____
and versions: _____
_____

Used in computer: _____
Operating system _____
version: _____
_____
Delivered by: _____

Application: _____
_____
_____

Describe below the deviations found. Give the page, the error and if possible, your suggested correction.

Describe also how the deviation was found and what influence it had on your system.

_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____

PLEASE SEND THIS REPORT TO DataIndustrier DIAB AB
Documentation
Box 2029
S-183 02 Täby, Sweden

****

ACTION BY DOCUMENTATION STAFF.

Comment verified Date: _____ Sign: _____

COMMENTS FIXED:

Release notice:	Version _____	Date: _____	Sign: _____
Manual/Datasheet:	Version _____	Date: _____	Sign: _____
Media:	Version _____	Date: _____	Sign: _____
Hardware:	Version _____	Date: _____	Sign: _____

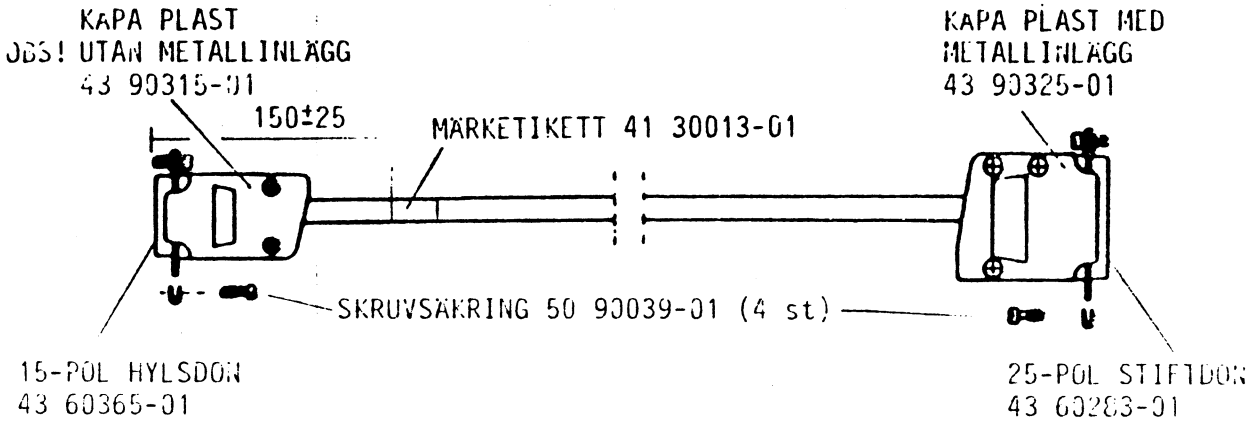
Transferred to distribution: Date: _____ Sign: _____

DataIndustrier DIAB AB

LUXOR

KABEL NETCENTRAL-LINJE
INGÅR I LUX-NET 230 8460

43 71779-01
Blad 2 av 2



15-POL D-SUB	MOTSTAND	FÄRG	25-POL D-SUB	LINJE ANSL	FUNKTION
1					
2		BRUN	14		TXD
3	3,3K	GUL	15		RXD
4		ROSA	16		RTS
5	3,3K	RÖD	17		CTS
6					
7		VIT	2		GND
7		GRÅ	4		GND
7		SVART, BLÅ	7	7,11,12,13	GND SIGNALJORD
8	3,3K	VIOLETT	20		DCD
9		GRÅ/ROSA BLÅ/RÖD	24, 25		+12V
10		5 x 3,3K			-12V
11					
12					
13					
14	3,3K	BRUN/GRÖN	22		RXC
15	3,3K	BRUN/GUL (VIT/GÖN)	21		TXC
-		SKÄRM	KAPA		SKÄRM
		VIT/GRÖN, VIT/GUL			AVKLIPPS
		GRÖN			AVKLIPPS

Ritad	Godkänd 840823	Reg 840918	METOD 840823	43 71779-01
	215L/HK	215/210L	03/403L	

H 113 A4