第九部分　使用与管理TCP/IP网络

本部分包括：

- 协议的配置与调整
- 实现DNS
- 网络管理
- SNMP ：简单网络管理协议
- 安全的TCP/IP传送
- 疑难问题解决工具及要点

第35章　协议配置与调整

作者：Anne Carasik

本章内容包括：

- 系统初始化问题
- 配置文件

对于任何系统，用户都必须知道系统的初始化过程及配置文件所在的位置。这将有助于用户更好地调整网络配置及解决可能出现的问题。同时，本章中的某些细节与网络并不直接相关，而直接与系统本身相关，这些细节将使用户更清楚地理解网络的配置和启动过程。

35.1　系统的初始化问题

当启动Unix和Linux系统时，有大量的问题值得考虑，其中与网络相关的是网络守护进程——它们何时启动、什么触发它们运行及哪一些守护进程正在运行等等。

任何一个主流Unix都有一个“祖父”进程，称作“init”。在Unix系统中，init进程创建其他所有进程，其中包括用户shell及网络进程。

init进程及/etc/inittab文件

Unix系统由两部分组成：文件及进程。在Unix系统中，所有其他进程都由init进程并发创建——最终，所有的进程都可以通过它们的父进程追溯到init进程。图35-1说明init进程与其他进程的链接关系。

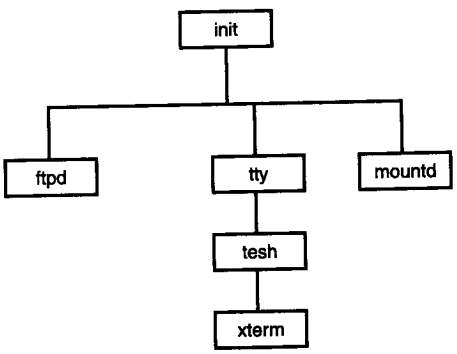

图35-1　init进程及其子进程

因为初始化进程是系统启动过程的最后一步，所以在此之前所有的文件系统都已经过检测，并且磁盘都已被加载，init进程决定进入哪一种用户模式(单用户或多用户)。某些Unix系

统使用/etc/inittab文件作为init进程的配置脚本，其内容如下：

```
# inittab       This file describes how the INIT process should set up
#               the system in a certain run-level.
#
# Author:       Miquel van Smoorenburg, <miquels@drinkel.nl.mugnet.org>
#               Modified for RHS Linux by Marc Ewing and Donnie Barnes
#

# Default runlevel. The runlevels used by RHS are:
#   0 - halt (Do NOT set initdefault to this)
#   1 - Single user mode
#   2 - Multiuser, without NFS (The same as 3, if you do not have networking)
#   3 - Full multiuser mode
#   4 - unused
#   5 - X11
#   6 - reboot (Do NOT set initdefault to this)
#
id:3:initdefault:

# System initialization.
si::sysinit:/etc/rc.d/rc.sysinit

l0:0:wait:/etc/rc.d/rc 0
l1:1:wait:/etc/rc.d/rc 1
l2:2:wait:/etc/rc.d/rc 2
l3:3:wait:/etc/rc.d/rc 3
l4:4:wait:/etc/rc.d/rc 4
l5:5:wait:/etc/rc.d/rc 5
l6:6:wait:/etc/rc.d/rc 6

# Things to run in every runlevel.
ud::once:/sbin/update

# Trap CTRL-ALT-DELETE
ca::ctrlaltdel:/sbin/shutdown -t3 -r now

# When our UPS tells us power has failed, assume we have a few minutes
# of power left.  Schedule a shutdown for 2 minutes from now.
# This does, of course, assume you have powerd installed and your
# UPS connected and working correctly.
pf::powerfail:/sbin/shutdown -f -h +2
"Power Failure; System Shutting Down''

# If power was restored before the shutdown kicked in, cancel it.
pr:12345:powerokwait:/sbin/shutdown -c
"Power Restored; Shutdown Cancelled''
  # Run gettys in standard runlevels
  1:2345:respawn:/sbin/mingetty tty1
  2:2345:respawn:/sbin/mingetty tty2
  3:2345:respawn:/sbin/mingetty tty3
```

```
4:2345:respawn:/sbin/mingetty tty4
5:2345:respawn:/sbin/mingetty tty5
6:2345:respawn:/sbin/mingetty tty6

# Run xdm in runlevel 5
# xdm is now a separate service
x:5:respawn:/etc/X11/prefdm -nodaemon
```

上述代码来自Red Hat Linux中的/etc/inittab文件，由代码可看出网络在第三运行级被启动。对于大多数主流Unix，第三运行级启动的网络进程包括 Internet守护进程(inted)、NFS、DNS及Sendmail。

rc系列脚本

为了创建并运行子进程， init进程将运行一系列启动脚本，我们称之为 rc系列脚本。这些脚本启动所需要的网络进程，从而Unix系统不再是一个“孤立的系统”。

在Red Hat Linix及基于SystemV的操作系统如HP-UX中， init进程运行/etc/rc.d/rc3目录下的所有网络启动脚本。其他系统可能运行另外的网络启动脚本如 rc.M(多用户)或rc.inet(适用于互联网)。系统运行哪些脚本与系统密切相关。

下面，为了保持一致性，我们仍采用 Red Hat Linux作为例子。因为本书主要讲解网络部分，以下仅列出了/etc/rc.d/rc3目录下的文件及子目录。

```
[root@tigerlair stripes]# ls /etc/rc.d/rc3.d/
K05innd           K20rwalld    K45arpwatch
K80random    S40atd    S85sound
K08autofs       K20rwhod    K45named    K85netfs
S40crond       S90xfsK10xntpd        K25squid
K50snmpd    K88ypserv    S50inet
S99linuxconfK15postgresql  K30mcserv    K55routed
K89portmap  S55sshd       S99local
K20bootparamd  K30sendmail    K60lpd
K96pcmcia    S72amdK20nfs           K34yppasswdd    K60mars-new    S05apmd
S75keytableK20rstatd       K35dhcpd    K75gated    S10network
S85gpm
K20rusersd      K35smb       K80nscd

   S30syslog    S85httpd
```

所有这些文件都启动某个特定的服务。对于第三运行级，除了网络部分，它还启动了一些别的特性。对于网络，我们可以通过 S##或K##来判断哪些守护进程被该文件启动 (K表示杀死，S表示启动)。其后的序列号表示某个特定的启动和停止的顺序。

同时，这些脚本还标明了当守护进程初启时，采用哪些属性。

需要注意的是，启动脚本和停止脚本并不总是成对出现，例如对于 S50inet，并没有K50inet与之对应。以下列出了S50inet的内容：

```
#! /bin/sh
#
# inet          Start TCP/IP networking services. This script
#               sets the hostname, creates the routes and
#               starts the Internet Network Daemon & RPC portmapper.
#
```

```
# Author:       Miquel van Smoorenburg, <miquels@drinkel.nl.mugnet.org>
#               Various folks at Red Hat
#
# chkconfig: 345 50 50
# description: The internet superserver daemon (commonly called inetd) \
#              starts a variety of other internet services as needed. \
#              It is responsible for starting many services, including \
#              telnet, ftp, rsh, and rlogin. Disabling inetd disables \
#              all of the services it is responsible for.
# processname: inetd
# pidfile: /var/run/inetd.pid
# config: /etc/sysconfig/network
# config: /etc/inetd.conf


# Source function library.
. /etc/rc.d/init.d/functions

# Get config.
. /etc/sysconfig/network

# Check that networking is up.
if [ ${NETWORKING} = ''no'' ]
then
        exit 0
fi
[ -f /usr/sbin/inetd ] ¦¦ exit 0

# See how we were called.
case "$1" in
  start)
        echo -n "Starting INET services:"
        daemon inetd

        echo
        touch /var/lock/subsys/inet
        ;;
  stop)
        # bringing down NFS filesystems isn't
        # inet's problem I don't know
        # why this script used to do that -- ewt

        echo -n "Stopping INET services:"
        killproc inetd

        echo
        rm -f /var/lock/subsys/inet
        ;;
  status)
        status inetd
        ;;
```

```
  restart¦reload)
        killall -HUP inetd
        ;;
  *)
        echo ''Usage: inet {start¦stop¦status¦restart¦reload}''
        exit 1
esac

exit 0
```

A kill script for K45named looks like this :

```
#!/bin/sh
#
# named
# This shell script takes care of starting and stopping
#                named (BIND DNS server).
#
# chkconfig: 345 55 45
# description: named (BIND) is a Domain
# Name Server (DNS)  that is used to resolve
# host names to IP addresses.
# probe: true

# Source function library.
. /etc/rc.d/init.d/functions

# Source networking configuration.
. /etc/sysconfig/network

# Check that networking is up.
[ ${NETWORKING} = ''no'' ] && exit 0

[ -f /usr/sbin/named ] ¦¦ exit 0

[ -f /etc/named.conf ] ¦¦ exit 0

# See how we were called.
case "$1" in
  start)
        # Start daemons.
        echo -n "Starting named: "
        daemon named
        echo
        touch /var/lock/subsys/named
        ;;
  stop)
        # Stop daemons.
        echo -n "Shutting down named: "
        killproc named
        rm -f /var/lock/subsys/named
        echo
```

```
        ;;
status)
        /usr/sbin/ndc status
        exit $?
        ;;
restart)
        /usr/sbin/ndc restart
        exit $?
        ;;
reload)
         /usr/sbin/ndc reload
         exit $?
         ;;
 probe)
         # named knows how to reload intelligently;
         # we don't want linuxconf
         # to offer to restart every time
         /usr/sbin/ndc reload >/dev/null 2>&1 || echo start
         exit 0
         ;;


  *)
         echo ''Usage: named {start|stop|status|restart}''
         exit 1
esac

exit 0
```

注意 并不是所有Unix系统的启动脚本都存放在相同的目录下。某些系统可能将它们放在/etc、/etc/rc或其他系统指定的位置。

35.2 配置文件

为了使网络正常工作，用户必须注意配置文件。在 Unix系统中，有一些非常重要的配置文件需要特别注意：

- /etc/inetd.conf
- /etc/services
- /etc/protocols
- /etc/hosts.equiv
- /etc/resolv.conf
- /etc/exports

其他Unix系统可能会包含别的配置文件，这因操作系统的不同而不同。

35.2.1 在/etc/protocols文件中定义网络协议

为了定义运行在Unix系统上的网络协议，用户必须在 /etc/protocols文件中给出说明。大部

分协议都可以很好地工作在 Unix系统中，并且用户不需要手工添加所有的协议 —— 系统将在启动时为用户完成上述工作。

注意以下仅列出了 IP协议 —— 对于一些非 IP协议如 Appletalk、NetWare及SNA并未列出。

下面是/etc/protocols文件的一个示例：

```
# /etc/protocols:
# $Id: protocols,v 1.1 1995/02/24 01:09:41 imurdock Exp $
#
# Internet (IP) protocols
#
#   from: @(#)protocols   5.1 (Berkeley) 4/17/89
#
# Updated for NetBSD based on RFC 1340, Assigned Numbers (July 1992).

Ip      0       IP          # internet protocol, pseudo protocol number
icmp    1       ICMP          # internet control message protocol
igmp    2       IGMP         # Internet Group Management
ggp     3       GGP         # gateway-gateway protocol
ipencap     4       IP-ENCAP    # IP encapsulated in IP (officially ``IP'')
st          5       ST          # ST datagram mode
tcp      6       TCP         # transmission control protocol
egp      8       EGP         # exterior gateway protocol
pup      12       PUP         # PARC universal packet protocol
udp      17       UDP         # user datagram protocol
hmp      20       HMP         # host monitoring protocol
xns-idp   22       XNS-IDP          # Xerox NS IDP
rdp      27       RDP          # ''reliable datagram'' protocol
iso-tp4   29       ISO-TP4          # ISO Transport Protocol class 4
xtp      36       XTP         # Xpress Tranfer Protocol
ddp      37       DDP         # Datagram Delivery Protocol
idpr-cmtp   39   IDPR-CMTP   # IDPR Control Message Transport
rspf   73   RSPF       #Radio Shortest Path First.
vmtp   81   VMTP       # Versatile Message Transport
ospf   89   OSPFIGP       # Open Shortest Path First IGP
ipip   94   IPIP       # Yet Another IP encapsulation
encap   98   ENCAP       # Yet Another IP encapsulation
```

35.2.2 在/etc/hosts文件中标识主机

对于一些本地系统，用户可能需要定义一些主机名，以便用户在不需要域名服务器 (DNS)提供服务的情况下，而仅靠主机名就可以找到主机。为了做到这一点，需要定义一个/etc/hosts文件。

最基本的/etc/hosts文件如下：

```
127.0.0.1  localhost  loopback
```

以上定义了localhost或loopback，它们是用户当前使用的主机的缺省主机名。 127.0.0.1为localhost的缺省IP地址。

增加其他主机的语法格式如下：

IP地址 主机名 别名

在增加其他主机时，需要列出主机的 IP地址，对应于IP地址的主机名及任意用户需要的别名。下面是一个/etc/hosts的示例：

```
# a sample /etc/hosts file
127.0.0.1   localhost       loopback
1.2.3.4      wednesday.addams.com    mymachine
1.2.3.5      pugsley.addams.com    yourmachine
1.2.3.6      gomez.addams.com    hismachine
1.2.3.7      cousinit.addams.com    itsmachine
1.2.3.8      morticia.addams.com    hermachine
```

在上例中，如果某个用户远程登录到本地主机，他将登录到 pugsley.addams.com。通过使用别名，用户不需要敲入完整的域名就可以访问目的主机。

注意 在/etc/hosts文件中，如果主机也在本地网中，填写主机名时，不需要写完整的域名，仅填写机器名即可。

35.2.3 TCP/IP与/etc/services文件

为了确定用户的Unix系统提供哪种类型的 TCP/IP服务，必须在系统中设置适当的服务。完成上述工作有两种方法：修改 /etc/services文件或修改inetd配置文件，后一种方法将在下一节讨论。

/etc/services文件为网络服务如FTP、Telnet、时间服务器、名字服务器、 Secure Shell(安全Shell)及finger等指定特定的端口。

大部分服务所对应的端口都是众所周知的特定端口，如 FTP服务端口为23，Telnet服务的端口为21等。这些服务所对应的端口号均小于 1024。

/etc/services文件中也包含一些不太通用的网络服务。 /etc/services文件的语法格式如下：

网络服务 端口号/tcp*或*udp

首先用户必须定义网络服务类型 (如Telnet、echo、SMTP)，然后指定端口号，最后指明该服务使用TCP还是UDP。

以下是/etc/services文件的示例：

```
# /etc/services:
# $Id: services,v 1.4 1997/05/20 19:41:21 tobias Exp $
#
# Network services, Internet style
#
# Note that it is presently the policy of IANA to assign a
# single well-known port number for both TCP and UDP;
# hence, most entries here have two entries
# even if the protocol doesn't support UDP operations.
# Updated from RFC 1700,``Assigned Numbers" (October 1994).

# Not all ports
# are included, only the more common ones.
```

```
tcpmux      1/tcp           # TCP port service multiplexer
echo        7/tcp
echo        7/udp
discard     9/tcp       sink null
discard     9/udp       sink null
systat      11/tcp      users
daytime     13/tcp
daytime     13/udp
netstat     15/tcp
qotd        17/tcp      quote
msp         18/tcp          # message send protocol
msp         18/udp          # message send protocol
chargen     19/tcp      ttytst source
chargen     19/udp      ttytst source
ftp-data    20/tcp
ftp         21/tcp
fsp         21/udp      fspd
ssh         22/tcp          # SSH Remote Login Protocol
ssh         22/udp          # SSH Remote Login Protocol
telnet      23/tcp
# 24 - private
smtp        25/tcp      mail
# 26 - unassigned
time        37/tcp      timserver
nameserver  42/tcp      name        # IEN 116
whois       43/tcp      nicname
domain      53/tcp      nameserver  # name-domain server
domain      53/udp      nameserver
bootps      67/tcp          # BOOTP server
bootpc      68/tcp          # BOOTP client
tftp        69/udp
gopher      70/tcp          # Internet Gopher
finger      79/tcp
www         80/tcp      http        # WorldWideWeb HTTP
www         80/udp          # HyperText Transfer Protocol
link        87/tcp      ttylink
kerberos    88/tcp      kerberos5 krb5  # Kerberos v5
kerberos    88/udp      kerberos5 krb5  # Kerberos v5
supdup      95/tcp
# 100 - reserved
pop-3       110/tcp         # POP version 3
sunrpc      111/tcp     portmapper  # RPC 4.0 portmapper TCP
sunrpc      111/udp     portmapper  # RPC 4.0 portmapper UDP
auth        113/tcp     authentication tap ident
sftp        115/tcp
nntp        119/tcp     readnews untp   # USENET News Transfer Protocol
ntp         123/tcp
netbios-ns  137/tcp         # NETBIOS Name Service
netbios-dgm 138/tcp         # NETBIOS Datagram Service
netbios-ssn 139/tcp         # NETBIOS session service
imap2       143/tcp     imap        # Interim Mail Access Proto v2
```

```
imap2       143/udp       imap
snmp       161/udp             # Simple Net Mgmt Proto
snmp-trap    162/udp       snmptrap    # Traps for SNMP
cmip-man    163/tcp              # ISO mgmt over IP (CMOT)
bgp        179/tcp             # Border Gateway Proto.
irc        194/tcp             # Internet Relay Chat
qmtp        209/tcp             # The Quick Mail Transfer Protocol
#
# UNIX specific services
#
exec       512/tcp
biff       512/udp       comsat
login       513/tcp
who       513/udp       whod
shell       514/tcp       cmd       # no passwords used
syslog       514/udp
printer       515/tcp       spooler       # line printer spooler
talk       517/udp
ntalk       518/udp
route       520/udp       router routed    # RIP
#
# Kerberos (Project Athena/MIT) services
# Note that these are for Kerberos v4, and are unofficial.  Sites running
# v4 should uncomment these and comment out the v5 entries above.
#
kerberos4    750/udp       kerberos-iv kdc    # Kerberos (server) udp
kerberos4    750/tcp       kerberos-iv kdc    # Kerberos (server) tcp
kerberos_master    751/udp             # Kerberos authentication
kerberos_master    751/tcp             # Kerberos authentication
passwd_server    752/udp            # Kerberos passwd server
krb_prop    754/tcp             # Kerberos slave propagation
krbupdate    760/tcp       kreg       # Kerberos registration
kpasswd       761/tcp       kpwd       # Kerberos "passwd"
#
# Services added for the Debian GNU/Linux distribution
poppassd    106/tcp             # Eudora
poppassd    106/udp             # Eudora
ssmtp       465/tcp             # SMTP over SSL
rsync       873/tcp             # rsync
rsync       873/udp             # rsync
simap        993/tcp             # IMAP over SSL
spop3       995/tcp             # POP-3 over SSL
socks       1080/tcp          # socks proxy server
socks       1080/udp          # socks proxy server
icp       3130/tcp          # Internet Cache Protocol (Squid)
icp       3130/udp          # Internet Cache Protocol (Squid)
ircd       6667/tcp          # Internet Relay Chat
ircd       6667/udp          # Internet Relay Chat
```

35.2.4 inetd守护进程与/etc/inetd.conf文件

在Unix系统中，许多网络服务由Internet超级守护进程inetd启动。为了使inetd守护进程知

道该启动哪些服务，它需要读取配置文件 inetd.conf，该文件通常放在/etc目录下。

注意 因为inetd可以启动其他网络服务，但是在任何时候都使用inetd并不是一个好主意，因为它将降低整个网络服务的性能。例如：Secure Shell因为其加解密过程而丧失了一部分性能。

对于大多数网络守护进程， inetd守护进程可以用来“看护”其他网络守护进程，因为inetd守护进程可以监听所有的socket，而其余网络守护进程只能监听特定的端口。

这样可以防止太多的守护进程等待一个绑定 socket，同时也可以避免socket随机进入。从某种意义上说， inetd可看作公司的安全守卫，仅让那些拜访特定人的来访者进入。这使得维护系统安全更方便，同时，可以更有效地控制流量 (带宽)。

图35-2说明在网络工作过程中， inetd如何扮演安全守卫。

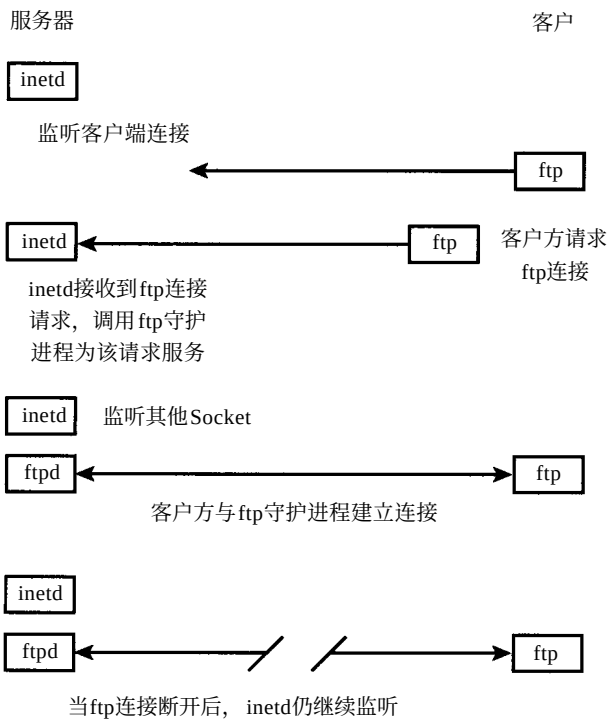

图35-2 inetd过程管理

因为inetd不知道用户需要运行哪些网络服务，所以必须在 /etc/inetd.conf文件中定义。/etc/inetd.conf中网络服务的格式如下：

网络服务名 socket类型 协议 标识 用户 服务路径名 参数

例如，对于FTP守护进程，其格式如下：

```
ftp stream tcp nowait root /usr/sbin/tcpd in.ftpd -l -a
```

ftp服务守护进程使用的 socket类型为stream，以根用户权限运行，服务器路径为/usr/sbin/tcpd，参数为in.ftpd -l -a。注意， ftp服务并未指明标识。如果某种服务不指明标识，该选项将被跳过。

以下是inetd.conf文件的部分示例：

```
# Version:   @(#)/etc/inetd.conf   3.10   05/27/93
#
# Authors:   Original taken from BSD UNIX 4.3/TAHOE.
#      Fred N. van Kempen, <waltje@uwalt.nl.mugnet.org>
#
echo    stream    tcp    nowait    root    internal
echo    dgram    udp    wait    root    internal
discard    stream    tcp    nowait    root    internal
discard    dgram    udp    wait    root    internal
daytime    stream    tcp    nowait    root    internal
daytime    dgram    udp    wait    root    internal
chargen    stream    tcp    nowait    root    internal
chargen    dgram    udp    wait    root    internal
time    stream    tcp    nowait    root    internal
time    dgram    udp    wait    root    internal

# These are standard services.
#
ftp    stream    tcp    nowait    root    /usr/sbin/tcpd    in.ftpd -l -a
telnet    stream  tcp     nowait  root     /usr/sbin/tcpd    in.telnetd
#
# Shell, login, exec, comsat and talk are BSD protocols.
#
shell    stream    tcp    nowait    root    /usr/sbin/tcpd    in.rshd
login    stream    tcp    nowait    root    /usr/sbin/tcpd    in.rlogind
#exec    stream    tcp    nowait    root    /usr/sbin/tcpd    in.rexecd
#comsat    dgram    udp    wait    root    /usr/sbin/tcpd    in.comsat
talk    dgram    udp    wait    root    /usr/sbin/tcpd    in.talkd
ntalk    dgram    udp    wait    root    /usr/sbin/tcpd    in.ntalkd
#dtalk    stream    tcp    waut    nobody    /usr/sbin/tcpd    in.dtalkd
#
# Pop and imap mail services et al
#
#pop-2    stream  tcp      nowait  root     /usr/sbin/tcpd    ipop2d
pop-3    stream  tcp      nowait  root     /usr/sbin/tcpd    ipop3d
imap     stream  tcp      nowait  root     /usr/sbin/tcpd    imapd
#
# The Internet UUCP service.
#
#uucp    stream    tcp    nowait    uucp    /usr/sbin/tcpd/usr/lib/uucp/uucico    -l
#
# Tftp service is provided primarily for booting.  Most sites
# run this only on machines acting as "boot servers." Do not uncomment
# this unless you *need* it.
#
#tftp    dgram    udp    wait    root    /usr/sbin/tcpd    in.tftpd
#bootps    dgram    udp    wait    root    /usr/sbin/tcpd    bootpd
#
# Finger, systat and netstat give out user information which may be
```

```
# valuable to potential "system crackers."  Many sites choose to disable
# some or all of these services to improve security.
#
finger    stream    tcp    nowait    root    /usr/sbin/tcpd    in.fingerd
#cfinger stream    tcp    nowait    root    /usr/sbin/tcpd    in.cfingerd
#systat    stream    tcp    nowait    guest    /usr/sbin/tcpd    /bin/ps    -auwwx
#netstat    stream    tcp    nowait    guest    /usr/sbin/tcpd/bin/netstat    -f
inet
#
# Authentication
#
auth    stream    tcp    nowait    nobody /usr/sbin/in.identd in.identd -l -e -o
#
# End of inetd.conf
```

注意，在缺省情况下，大多数网络守护进程都用 #注解。这表示除非用户已非常确信自己的修改，否则不要运行这些服务。这主要是基于安全方面的考虑，因为某些服务容易使用户获取额外的访问权限或导致系统崩溃。

注意 守护进程通常以nobody或root用户权限运行。当使用root权限运行时，要非常小心。

如果用户对/etc/inetd.conf文件做了任何修改，如注解了某个守护进程或增加了某个守护进程，都需要发送一个HUP信号重启inetd守护进程，如：

```
#killall -HUP inetd
```

35.2.5 在/etc/networks文件中设置网络

仅有少数系统使用/etc/networks文件标识系统所在的网络，这将有助于用户组织其系统连入的网络。

35.2.6 DNS客户与/etc/resolv.conf

为了获得DNS名字解析功能，用户需要正确设置 DNS客户端。要完成这一工作，需要修改/etc/resolv.conf文件。

在/etc/resolv.conf文件中，用户需要定义域名、域名服务器 (主控域名服务器或主控域名服务器和辅助域名服务器)及其他必须的指令。

一般情况下，这将允许用户的系统实现域名查询，如将 www.example.org转换为IP地址1.2.3.4。

```
domain example.com
nameserver 1.2.3.1
nameserver 1.2.3.254
search
```

注意 即使/etc/resolv.conf指明了DNS，用户仍需要检查设置是否正确。用户可以使用nsloopkup、dig或其他工具检验设置是否正确。

否则，系统将会出现与 DNS不相关的问题，如本人的系统曾经出现过因 /etc/

resolv. conf文件设置不正确而导致HP-UX X-Window CDE设置工作不正常。

35.3 小结

本章讲述了与网络管理相关的一些重要的问题，包括：系统初始化、配置文件及如何在正确的地方对配置文件做适当的修改等。

了解系统何时启动网络服务非常重要。在 Unix系统中，多个运行级保证系统能够在适当的运行级完成特定的功能，其中包含启动网络功能的运行级。在大多数系统中，网络功能在第三运行级启动。其他功能在别的运行级启动，如多用户 (定义为“ M ”)。在大多数Unix系统中，运行级配置文件为/etc/inittab。

对于网络，由于安全及带宽等方面的原因，确保配置文件的正确设置非常关键。在 Unix系统中，包括在inetd.conf中对Internet超级守护进程的配置、在 /etc/resolv.conf中对DNS客户方的配置、在/etc/networks中可访问网络的配置，在/etc/hosts中对主机的配置及在etc/services中对网络服务的配置。

一旦用户设置了TCP/IP网络服务，就必须了解如何使用 DNS管理域名服务，了解网络管理及其协议SNMP，并且了解如何加强网络安全及调试 TCP/IP网络。