NAME

init − grandparent of all processes

DESCRIPTION

The first program started by Minix is **init**. The actions performed by **init** can be summarized by this pseudo shell program:

```
# Open 0, 1, 2.
exec </dev/null >/dev/log 2>&1

# Run the system initialization script.
sh /etc/rc $bootopts

>/etc/utmp
echo reboot >>/usr/adm/wtmp

while :; do
      # Wait for a process to exit, but don’t always block.
      wait

      # Record logout.  (Not in this dumb way, of course.)
      if "pid is in my tables" $pid
      then
            echo "logout $pid" >/etc/utmp
            echo "logout $pid" >>/usr/adm/wtmp
      fi

      # Start a new session.
      while read line type getty init
      do
            if idle $line
            then
                  $init ... <$tty >$tty
                  $getty <$tty >$tty 2>&1 &
                  pid=$!
                  "add pid to tables" $pid
                  echo "login $line $pid" >/etc/utmp
                  echo "login $line $pid" >>/usr/adm/wtmp
            fi
      done < /dev/ttytab
done
```

The first action of **init** is to run **/etc/rc** to initialize the system as described in **boot**(8). **Init** then enters its main loop where it waits for processes to exit, and starts processes on each enabled terminal line. The file **/etc/ttytab** contains a list of terminal devices, their terminal types, the program to execute on them to allow one to login (usually **getty**(8)), and the program to execute first to initialize the line (usually **stty**(1)). These fields may be left out to indicate that a line is disabled or that initialization is not necessary. The commands are searched using the path **/sbin:/bin:/usr/sbin:/usr/bin**.

Init accepts several signals that must be sent to process id 1. (It is the first process, so natually its process id is 1.) The signals are:

SIGHUP

When receiving a hangup signal, **init** will forget about errors and rescan **ttytab** for processes to execute. **Init** normally rescans **ttytab** each time it feels the need to respawn a process, so the hangup signal is only needed if a line has been shut down, or after a terminate signal. Note that after turning a line off you will have to kill the process running on that line manually, **init** doesn't do that for you.

SIGTERM
Normally sent by programs that halt or reboot Minix. Causes **init** to stop spawning new processes.

SIGABRT
Sent by the keyboard driver when the **CTRL-ALT-DEL** key combination is typed. Causes **init** to run the **shutdown** command. A second abort signal makes **init** halt the system directly with a system call. The keyboard driver halts the system, without a sync, after the third CTRL-ALT-DEL.

Minix vs. Minix-vmd

There are a few differences between standard Minix and Minix-vmd on how **init** is run. The **/etc/rc** file is executed under standard Minix with input connected to **/dev/console**, but under Minix-vmd this is still **/dev/null**. This means that under Minix-vmd processes must be reconnected to **/dev/console** with the **intr**(8) program if they need user interaction. Minix-vmd passes the value of the **bootopts** boot variable to /etc/rc. Standard Minix does not.

FILES

/etc/ttytab	List of terminals devices.
/etc/utmp	List of currently logged in users.
/usr/adm/wtmp	Login/logout history.

SEE ALSO

ttytab(5), **utmp**(5), **getty**(8), **stty**(1), **boot**(8).

AUTHOR

Kees J. Bot (kjb@cs.vu.nl)