```
int hi_ping(char *ips, int timeout, int max_times)
{
    struct timeval timeo;
    int sockfd;
    struct sockaddr_in addr;
    struct sockaddr_in from;

    struct timeval *tval;
    struct ip            *iph;
    struct icmp          *icmp;

    char sendpacket[128];
    char recvpacket[128];
    char from_ip[32];
    int n;
    int ping_times = 0;
    int ret = 0;
    pid_t pid;
    int maxfds = 0;
    fd_set readfds;

    if (ips == NULL || strcmp(ips, "") == 0)
        return 0;
    bzero(&addr,sizeof(addr));
    addr.sin_family = AF_INET;
        inet_pton(AF_INET, ips, &addr.sin_addr);

    sockfd = socket(AF_INET, SOCK_RAW, IPPROTO_ICMP);
    if (sockfd < 0)
    {
                return HI_RETURN_FAIL;
    }

    timeo.tv_sec = timeout / 1000000;
```

```
    timeo.tv_usec = (timeout % 1000);

    if (setsockopt(sockfd, SOL_SOCKET, SO_SNDTIMEO, &timeo, sizeof(timeo)) ==
-1)
    {
        close(sockfd);
                sleep(5);
        return HI_RETURN_FAIL;
    }

    memset(sendpacket, 0, sizeof(sendpacket));

    pid=getpid();
    int packsize = 0;

    icmp=(struct icmp*)sendpacket;
    icmp->icmp_type=ICMP_ECHO;
    icmp->icmp_code=0;
    icmp->icmp_cksum=0;
    icmp->icmp_seq=0;
    icmp->icmp_id=pid;
    packsize=8+56;
    tval= (struct timeval *)icmp->icmp_data;
    gettimeofday(tval,NULL);
    icmp->icmp_cksum=cal_chksum((unsigned short *)icmp,packsize);

    n = sendto(sockfd, (char *)&sendpacket, packsize, 0, (struct sockaddr
*)&addr, sizeof(addr));
    if (n < 1)
    {
                close(sockfd);
                sleep(5);
        return HI_RETURN_FAIL;
    }
```

```
    while(ping_times++ < max_times)
    {
        FD_ZERO(&readfds);
        FD_SET(sockfd, &readfds);
        maxfds = sockfd + 1;
        n = select(maxfds, &readfds, NULL, NULL, &timeo);
        if (n <= 0)
        {
            ret = HI_RETURN_FAIL;
            continue;
        }

        memset(recvpacket, 0, sizeof(recvpacket));
        int fromlen = sizeof(from);
        n = recvfrom(sockfd, recvpacket, sizeof(recvpacket), 0,
                                        (struct sockaddr *)&from, (socklen_t
*)&fromlen);
        if (n < 1)
        {
                        ret = HI_RETURN_FAIL;
            continue;
        }

                inet_ntop(AF_INET, &from.sin_addr, from_ip, sizeof(from_ip));
        if (strcmp(from_ip,ips) != 0)
        {
                        ret = HI_RETURN_FAIL;
            continue;
        }

        iph = (struct ip *)recvpacket;

        icmp=(struct icmp *)(recvpacket + (iph->ip_hl<<2));

        if (icmp->icmp_type == ICMP_ECHOREPLY && icmp->icmp_id == pid)
```

```
        {
            ret = HI_RETURN_OK;
            break;
        }
        else
        {
            ret = HI_RETURN_FAIL;
            continue;
        }
    }

    close(sockfd);
    return ret;
}
```