Re: how to set timeout for 'read' command

Source: http://unix.derkeiler.com/Newsgroups/comp.unix.shell/2005−07/0230.html

---

From: Icarus Sparry (*usenet_at_icarus.freeuk.com*)
Date: 07/08/05

```
Date: Thu, 07 Jul 2005 23:37:11 GMT
```

On Thu, 07 Jul 2005 07:03:16 −0700, nasir wrote:

>
>
> Icarus Sparry wrote:
>
>> > I changed it to /dev/tcp/10.100.1.1/29 where tcp/29 is an unused port
>> > and no service is running on that. Ideally it should timeout after 15
>> > seconds, but I get the following output immediately:
>> >
>>
>> If the host 10.100.1.1 is accessible, then this is the correct behavior.
>> The shell will attempt to connect to that TCP port, get an error response,
>> and will therefore fail the shell 'read' command. If you remove the
>> s>/dev/null on line 9 you will probably get an error message telling you
>> this.
>
> No luck, I tried with IP(10.100.1.1) live and Port(29) blocked. And to
> dead non responding IP as well. Same behaviour. It should not be the
> same. I dont have any problem if the IP do no exist (do not respond at
> all) in my original script. The problem I am having is with an IP which
> is live/responding but not responding to certain port (in my case
> SSH(tcp/22)).
>
> P.S. Mind that I am originally working on SSH or tcp/22 port, the port
> 29 I am using just for testing to simulate a non−responding
> service/port.
>
> thanks.

I am not sure what you mean by port 29 being 'blocked'

There are 6 cases I can see.

1) There is no IP connectivity between the source and destination machine on the particular port. This case includes one machine being down if it is not on the local network, or if there is a firewall between the machines which is blocking the port, but not giving any ICMP errors. After

about four minutes the connection will fail. The desired behavior of the program is to wait 15 seconds as the TCP retries, and then abort.

2) The remote machine running but has nothing listening on that port, or there is a firewall inbetween which sends back ICMP messages saying that you can not connect. The desired behavior of the program is to abort instantly, in response to the refusal to open the connection.

3) The remote machine has something listening on the port, but it is not responsive. The desired behavior of the program is to wait 15 seconds for a response, and then abort if none is received.

4) The remote machine is listening on the port, and it sends a response quickly. The desired behavior of the program is to print out this response and more on without waiting 15 seconds.

5) The sending machine doesn't know the route to the receiving machine. The desired behavior is to abort instantly.

6) The sending machine knows that the receiving machine should be on the local network, but it is off or disconnected. The desired behavior is to abort as soon as possible.

You seem to be expecting that the behavior in the first two cases will be the same, but they are very different at the network level. In the first case packets are sent out, but none are received in return. This will give you your 4 minute delays as it continues to retry. In the second case packets are sent out, and a response is returned from the IP stack straight away, telling you that the connection is refused. In the fifth case the sending machine will not send out any packets, and will report an error. In the sixth case the sender will send out ARP packets in an attempt to find the MAC address of the remote machine, but will get no answer and after a few seconds it will report an error. In all these cases an error will be returned to the 'connect' system call, which the shell read will make.

In the third and fourth cases the TCP 3 way handshake completes, and it is then up to the application to send something or not. So the connect system call will complete.

You say you don't have a problem in your original script if the "IP does not exist", but you don't say if this is a local (case 6) or remote (case 1) IP.

You say your problem is "with an IP which is live/responding but not responding to certain port", but you don't say if this lack of response is because the application is not responding (case 3), or there is no application (case 2).

Consider the following. I have tidies up the redirection a little, and I am using $(...) instead of `...`. tcpserver is program that runs

a command, a bit like inetd, with its standard input and output connected
to the network. I am outputing the return code of the assignment so you
can see if it worked or not.

```
$ tcpserver 192.168.1.99 9999 bash −c "sleep 5 ; echo boo" &
$ ./j1
Thu Jul 7 16:22:06 PDT 2005
0 boo
Thu Jul 7 16:22:11 PDT 2005

$ tcpserver 192.168.1.99 9999 bash −c "sleep 500 ; echo boo" &
$ ./j1
Thu Jul 7 16:22:52 PDT 2005
271
Thu Jul 7 16:23:07 PDT 2005
$ cat j1
#!/bin/ksh
exec 2>/dev/null

date
var=$(ksh −c '(sleep 5 ; kill $$)&
        read fred < /dev/tcp/192.168.1.99/9999
        kill $!
        echo "$fred"
        ')
echo $? $var
date
```