

Re: unix shell script ignores 'exit' when in function that is piped to tee

Source: <http://unix.derkeiler.com/Newsgroups/comp.unix.shell/2005-01/1100.html>

From: Nunya Bizness (nunyabizness36_at_hotmail.com)

Date: 01/28/05

Date: Fri, 28 Jan 2005 00:49:25 -0700

onealwc@agedwards.com wrote:

> *Dear Lew Pitcher,*
>
> *Thank you for taking the time to reply and share your knowledge. I*
> *must agree with you in that I misunderstood how functions, pipes and*
> *sub-shells work.*
>
> *I thought calling a function within a script did NOT create sub-shells.*
>
> *This new enlightenment goes to show I asked the wrong question. The*
> *question I should have asked is; In a sh script, can a child kill a*
> *parent, and if so, how is this done?*
>
> *I ask this because your example code fix does not help me with my real*
> *world app.*
>
> *The real world app must log for auditors what the admin see and how the*
> *admin responds, each with date/time stamps. Thoughtout most of the*
> *script this is done by calls to aa function I have written called LOG*
> *which echos its arguments to the screen and to an audit file preceeded*
> *by a date/time stamp. In the script I have two section where I found*
> *using LOG either impossible -or- impractical. For these sections, I*
> *turned them into functions and piped them to tee. If a child cannot*
> *kill it's parent, then I will re-org the script such that there is*
> *only one function that is tee'ed and that it is the last call within*
> *the script.*
>
> *Look forward to reading your reply.*
>

Regardless of the method used to capture the keyboard input and screen output, the first question that comes to mind is "How secure does your audit log need to be?" If you or your auditors need assurance that everything that was typed/printed is in the log, without modification, you need to take into account the fact that the admin, presumably root, is going to be able to alter the log file to cover up any inappropriate activity.

comp.unix.shell: Re: unix shell script ignores 'exit' when in function that is piped to tee

In order to protect against this, you might want to consider writing your own logging program in c and using the crypt function to tie each entry in the log to the previous entry. The easiest way to use crypt to do this is to combine the 'key' from the previous entry with the current time, the text of the current log entry and an additional 'salt' value that is not publicly known and run it thru crypt. The resulting key should be output as part of the line that you write to your log file so it can be used as input to the next invocation.

Verification of the integrity of the log file is a simple matter of running the log file thru the same encryption algorithm with the exception that the date/time stamp is taken from the log file instead of the system. (Note that source code security is a must else it would be a simple matter to write a separate c program that could be used to reprocess the log and make everything consistent if the key value is known.)