Technische Universität Berlin

Command Line Interface – I/O redirection

Training for Afghan System Administrators

Standard input, standard output, standard error

There are always three default files open, standard input or **stdin** (the keyboard), standard output or **stdout** (the screen) and standard error or **stderr** (error messages output to the screen).

Each open file gets assigned a *file descriptor*. The file descriptors for **stdin**, **stdout**, and **stderr** are **0**, **1**, and **2**, respectively. For opening additional files, there remain descriptors 3 to 9.

These, and any other open files, can be redirected. Redirection simply means capturing output from a file, command, program, script and sending it as input to another file, command, program, or script.

Redirection operators

Before a command is executed, its input and output (incl. the standard error output) may be redirected using a special notation interpreted by the shell. Redirection allows commands' file handles to be duplicated, opened, closed, made to refer to different files, and can change the files the command reads from and writes to.

Examples:

- command > file
  # Redirect the standard output (stdout) of command to a file.
- command >> file
  # Append stdout of command to a file.
- command < file
  # Redirect the contents of the file to the standard input (stdin) of command.

Redirecting standard output

Usually, if you execute a command / utility in the command line, its standard output (the “result” of your command), is printed on the screen. This is OK if you just need to see the output with your eyes. However, often it is needed to save or further process the output of a command. This can be done by redirecting the standard output of any command to a file.

“>” is the output redirection operator. It is followed by the path to the file that should capture the stdout. Bash first tries to open the file for writing and if it succeeds it sends the stdout of command to the newly opened file. If it fails opening the file, the whole command fails. Writing command >file is the same as writing command 1>file. The number 1 stands for stdout, which is the file descriptor number for standard output.

Example:

- echo 'hello world' >output
  # redirects the “results” (stdout) of the echo command to the file output
- echo 'hello world' 1>output
  # same as above

Redirecting standard input

Instead of taking the standard input from the keyboard as the user types, often it can be very useful, to take standard input from other sources, like files, instead.

Redirection of input causes the file whose name results from the expansion of word to be opened for reading on file descriptor n, or the standard input (file descriptor 0) if n is not specified.

The general format for redirecting input is:

- command < word
  # reads content of file “word” and passes it as standard input to “command”

For example (only works if “output” exists before execution!):

- cat < output
  # concatenates the contents of the file named “output”

Redirecting standard error

Usually the standard error (stderr) channel is shown on the screen, so the user can see if there was a problem running a command. If it is important to capture the stderr to view it later (for example in automated scripts which run without user interaction), stderr can be redirected to a file (similarly to redirecting stdout). It can also be redirected to stdout first, and this combined output can then be redirected to a file.

Example:

- ls file_XYZ >output_file
  # will print an error message informing the user that “file_XYZ” does not exist
- ls file_XYZ 2>output_file
  # now the standard error (2) will be redirected to the file output
- ls * file_XYZ > output_file 2>&1
  # redirects stderr to stdout, and then BOTH will be redirected to output_file (ORDER MATTERS!)
- ls * file_XYZ &> output_file
  # same as above in a shorter notation

Using pipes

Pipes let you use the output of a program as the input of another one, without having to write to any files as an intermediary step. This is very useful together with the UNIX philosophy of “do one thing and do it well”. This philosophy refers to the fact, that by using pipes, we can use one program to do one thing for us and then let another program process the results from the first step. We can then build chains of commands where each commands output is the next commands input.

Examples:

- ls | sort -r
  # reverse the order of a directory listing
- dmesg | less
  # open the output of dmesg in the less pager
- sudo cat /etc/shadow | grep stefan | cut -d : -f 2
  # show contents of /etc/shadow, then grep (search) for the word “stefan”, then print only the 2nd field (fields are delimited by the “:” character)

Exercises!

Log in as normal user and change to your home directory first. Then try these exercises:

1) Redirect the long (-l) listing (ls) of your home directory's content to file “home_dir_content”
2) *Activate routing functionality by echoing 1 to the file /proc/sys/net/ipv4/ip_forward
3) Redirect current date (with “date) to file “current_time”
4) *Append(!!!) a text string of “This is a string” with “echo” to the syslog file (/var/log/syslog)
5) Redirect the content of the syslog file as standard input to the less pager
6) List the content of /root as normal user (no sudo!) and capture the stderr to the file “ls_error”
7) List the contents of /root as normal user (no sudo!) but discard stderr (redirect to /dev/null)
8) How big is the size of every folder on the root level (“/”) of your filesystem? Capture stdout and stderr in the file size_of_root_level
9) Use grep to find all occurrences of the word “sd” in the dmesg log file
10) How many lines does the syslog file have? (2 commands: first redirect output to temp_file, then use wc on temp_file)
11) How many lines does the syslog file have? (pipe cat to wc; try different options of wc)
12) Echo your name to md5sum and cut out (with “cut”) the first 8 characters

    * these commands need root permissions, so either become root with “sudo su” and then try, or use “sudo bash -c “THE COMMAND THAT ANSWERS THE QUESTION”

Solutions to exercises

1) ls -l > home_dir_content
2) echo 1 > /proc/sys/net/ipv4/ip_forward
3) date > current_time
4) echo "This is a string" >> /var/log/syslog
5) cat /var/log/syslog | less
6) ls /root 2> ls_error
7) ls /root 2> /dev/null
8) du / > size_of_root_level 2>&1
   OR: du / &> size_of_root_level
9) dmesg | grep sd
10) cat /var/log/syslog > temp_file && wc -l temp_file
11) cat /var/log/syslog | wc -l
12) echo stefan | md5sum | cut -c 1-8