

Linux Text Utilities 101 for S/390 Wizards

SHARE Session 9220/5522

Scott D. Courtney

Senior Engineer, Sine Nomine Associates

March 7, 2002

<http://www.sinenomine.net/>

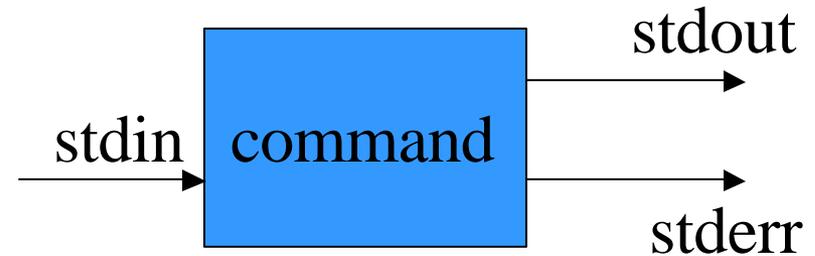
Sine Nomine
Associates

Table of Contents

- Concepts of the Linux Shell Environment
 - I/O Redirection
 - Command Piping
 - Command Substitution
- Linux Text Utilities
- Pipe Examples

Linux Shell Concepts: I/O Redirection

- Three streams: input, output, and error
- < controls input
- > or 1> controls output
- 2> controls error
- >> appends to file
- /dev/null is the Bit Bucket



Example 1: Three different files

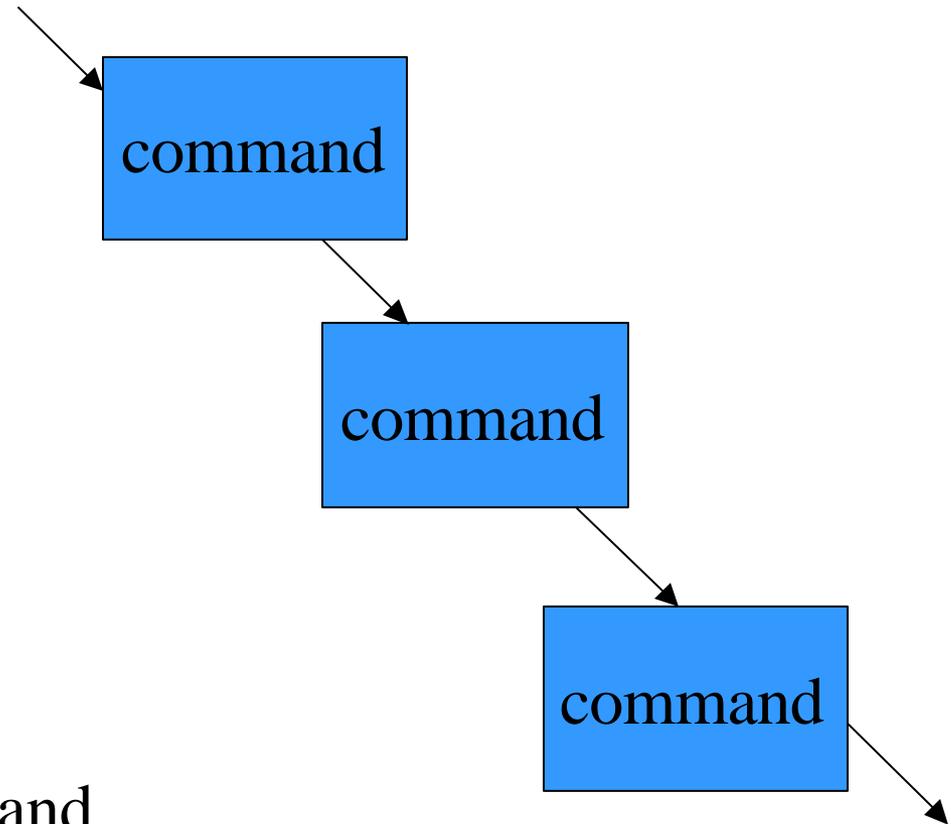
```
somecommand < input_file > output_file 2> error_log
```

Example 2: Common log file

```
somecommand < input_file > output_file 2>&1
```

Linux Shell Concepts: Command Piping

- Pipe symbol: |
- Connects output of one process to input of next
- Multiple levels
- No guarantee of timing
- Cumulative effect on data



Example 1: Simple pipe

```
somecommand | othercommand
```

Example 2: Discarding errors but catching the output

```
somecommand 2> /dev/null | othercommand > outfile
```

Linux Shell Concepts: Command Substitution



- Use back-tickmarks (accent grave) to enclose command
- Substitutes *result* back into original command line
- Can have a multistage pipe inside substitution area

Example:

```
mv `ls *.jpg | tail -3` /home/newuser/graphics/
```

Lists all JPEG files in the current directory alphabetically, picks the last three from the list, and moves them to the indicated directory. Powerful!

Text Utilities: Common Behavior

- Designed for *text*, not binary
- Treat text as variable-length lines
- May be limits (albeit very large) to length of a line
- Case-sensitive unless otherwise specified
- May buffer input, outputs, or both
- Options [usually] go before filenames, as in:
 `command -op1 -op2 file1 file2 ...`
- This presentation doesn't cover *all* options of each command.

- Short for “concatenate”
- Copies stdin to stdout
- Accepts multiple filenames
- Options:
 - b Numbers nonblank lines
 - n Numbers every line
 - s Removes consecutive blank lines

Example:

```
cat -s file1 file2 file3 > bigfile 2> /dev/null
```

Copies (without redundant blank lines) three files into one big file.
Throw any errors (such as for missing files) into the bit bucket.

- Grabs first n lines of file
- For multiple files, adds a header line before each
- Options:
 - n Indicates n lines (default 10)

Example:

```
ls ~ | head -5
```

Show me the first five files in my home directory

- Grabs last n lines of file
- For multiple files, adds a header line before each
- Options:
 - n Indicates n lines (default 10)
 - f Follow (update every few seconds)
(Great for watching logfiles be created)

Example:

```
tail -30 /tmp/something.log
```

Show me the last 30 lines of a logfile.

sort

- Many options – use “man sort” for details
- Options:
 - b Ignore leading blanks in keys
 - r Sort in reverse (descending) order
 - n Numeric (rather than alphanumeric) sort
 - f Fold to uppercase (case-insensitive sort)
 - u Unique – discard duplicate lines from output
 - k *pos1, pos2* Define key from *pos1* through *pos2* fields, inclusive (leftmost is field 1)
 - t *char* Use *char* as the field delimiter instead of blank-to-nonblank transitions

Example:

```
du -s ~/* | sort -n | tail -20
```

Lists the 20 biggest files or directories in your home directory

cut

- Cuts columns or fields from the data stream
- Discards everything not selected
- Accepts lists of ranges, e.g.:
1,3,2-7,14,39-45
- Options (-c and -f are mutually exclusive):
 - c *range_list* Return this list of column fields
 - f *range_list* Return this list of delimited fields
 - d *delimiter* Specifies field delimiter character

Example:

```
cut -f 1,5 -d: /etc/passwd
```

List all the login names and real names of users of this system.

/etc/passwd looks like “*login:x:uid:gid:real_name:home:shell*”

- Translates character-by-character
- Works much like equivalent function in REXX
- Options (partial list...see “man tr” for details):
 - d Delete the specified character(s) from input
 - s Squeeze repeated characters to one instance each
 - c Invert (complement) the input set
- Built-in special sets (again, a partial list):
 - [:alpha:] All alphabets
 - [:digit:] 0 through 9, inclusive
 - [:xdigit:] Hexadecimal digits
 - [:alnum:] Alphanumerics
 - [:upper:] and [:lower:] Upper and lower case alphabets

tr (continued)

- Example 1: Convert everything to uppercase
`tr [:lower:] [:upper:] < infile > outfile`
- Example 2: Remove all equals signs from input
`tr -d = < infile > outfile`
(Note...for many characters, put within single quotes!)
- Example 3: Remove anything except digits
`tr -d -c [:digit:] < infile > outfile`

- Searches for **regular expression** patterns in input
- Simple examples here – regular expressions are a topic unto themselves, but **grep** is too important to omit.
- Options:
 - v Find lines *not* containing the pattern
 - i Match case-insensitively
 - E Use enhanced expression syntax (see **tr** or “man grep” for examples)
 - n Show numbers of matching lines
 - l List filenames containing matches, not actual text

grep (continued)

- Example 1: Search for lines beginning with “A”
`grep ^A somefile`
- Example 2: Search for lines not ending with “A” or “a”
`grep -v -i a$ somefile`
- Example 3: Name all C source files in the current directory that contain “MyFunction” anywhere in the file
`grep -l “MyFunction” *.c`
- Example 4: What files in the /tmp directory begin with something other than a letter or digit?
`ls /tmp | grep -iv ^[0-9A-Z]`
- Many, many other possibilities. Read up on this one!

Example Pipe 1: Show me the piggies!

- Situation: I want to know the logins of the three users who have the largest home directories, assuming everyone's home directory is under /home. Show the size of the directories in kilobytes. Put the worst offender at the top of the list, not the bottom.
- Solution:
`du -sk /home/* | sort -rn | head -3 | cut f1,3 -d/ | tr -d /`
- Output sample:
4537 piggy
2301 oink
1896 glutton
- Advanced tip: **sed** (not covered today) is a better way to get rid of the “/home/” part of the home directory pathname, or just run the commands from /home as the current directory, and omit the full path from **du**, also dropping **cut** and **tr** from the pipe.

Example Pipe 2: Get rid of test files

- Situation: You've been working on some source code lately, and in a fit of poor administrative practice, you mixed the prototypes in with your production source files! Luckily you know that the word "TEST" (upper case) appears in the bogus source files but not in any of the others. Delete all the bogus source files from the current directory in one fell swoop.
- Solution:

```
rm `grep -l TEST *.c *.cpp *.h`
```
- Important safety tip:
Doh! This is an easy way to zap a lot of data by mistake. Run the stuff inside the back-ticks, separately, to see what the list of filenames will be before you actually delete anything.

Example Pipe 3: Data query

- Situation: You have input records that look like this:
name→*address*→*city*→*state*→*zipcode*
You need to extract the name and zipcode (only) of all the records beginning with A through M. Also, it is desired to have the output be only uppercase because it will be fed to an application that does not understand lowercase letters. Sort the output by zipcode, ascending. Output records should be *name*→*zipcode* only.
- Solution:

```
grep -i ^[A-M] infile | cut -f5,1 | sort -n | cut -f2,1 > outfile
```
- Advanced tip:
Tabs are **cut**'s default delimiter. Note that we first pull the fields zipcode-to-the-left to make the **sort** syntax simpler, then use **cut** again to reverse the fields (of which there are now only two).

Conclusion

- There are dozens of other text-oriented commands
- They can be combined in virtually infinite ways
- Modularity is the key
- Look in `/usr/bin` and `/bin` for many of these. The names may give you a clue as to what they do.
- Type “`man command`” to get syntax help.
- Test things that obtain filename lists before doing things to the files. (“Let’s be *careful* out there!”)
- Have fun, and experiment!

Contact Information



Scott D. Courtney
Senior Engineer
Sine Nomine Associates
43596 Blacksmith Square
Ashburn VA 20147-4606
scourtney@sinenomine.net
<http://www.sinenomine.net/>

The logo for Sine Nomine Associates is a blue rectangular box with a white horizontal line. The text "Sine Nomine" is written in white, serif font above the line, and "Associates" is written in white, serif font below the line. The box is set against a light gray background with a subtle drop shadow.

Sine Nomine
Associates

Presentation Copyright © 2001 by Sine Nomine Associates. All Rights Reserved.

“S/390: The Linux Dream Machine” Copyright © 2000 by Scott D. Courtney. All Rights Reserved.