sed & awk
70s tools still hip today

Michael Stroucken
(2014-04-15)

Where do they come from?

- Bell Labs, the people that brought us Unix
- The Unix philosophy
- sed (stream editor) first released in 1974, developer was Lee McMahon (1931-1989)
- awk first released in 1977, named after developers
  - Alfred Aho
  - Peter Weinberger
  - Brian Kernighan

sed

- "stream editor"
- related to ed (underlying vi)
- Unfortunately many incompatible versions
- GNU and BSD sed probably most popular
- grep does better grepping, awk does better processing, perl does almost all better

Simple sed

- Find and replace
  - s/^M// (dos2unix)
  - s/^count: //
  - s/Snowden/REDACTED/g
- Drop matching lines
  - /BK/d

more sed commands

- branching
  - t/T, : label
- transform
  - y/abcdef/ABCDEF/
- store/load/exchange
  - h/g/x
- replace
  - c

awk

- Arguably more useful
- Record at a time
  - usually a line
- more convenient data access
  - fields, variables, functions
- pattern - action pairs
  - both can be empty
  - Like perl's -n option: for every record read, check for pattern match, then run the associated action
- Or function definitions

awk handles fields

- Default field separator is whitespace
- $0 contains the entire record
- $1, $2, $3, $(1+3),$(five)

awk system variables

- NR, FNR, FS, OFS, NF, RS, ORS, SUBSEP
  - Can be written to, but the record won't be reparse
  - If you touch NF, the effects are apparently undefined
  - Variables beginning with 'O' are for output
  - Field separator can be regular expression, so can the record separator
  - SUBSEP enables some syntactic sugar

awk variables

- standard variables
  - no type strictness
  - foo=bar, baz="bat", quux=$3
  - C style truth values
- arrays
  - actually a key -> value mapping
  - foo[3] = $3, bar[NR] = NF
  - if (2 in foo), for (bar in foo)
  - delete foo[3]
  - but not foo = bar!
- multidimensional arrays
  - foo[3, 1] = 1
  - The key here is actually: 3 SUBSEP 1
  - SUBSEP is ASCII FS [chr(28)] by default

awk functions

- Variables and functions share name space
  - Can't have variable foo and function foo()
- Variables are global-scoped by default, but car have private scope with function arguments
  - function foo(arg) { arg++; print arg }
  - modified value of arg will not be seen in caller, eve if a variable with that name exists

awk patterns

- Regexp
  - /^foobar$/
- Computed expressions
  - $1 > 31337
  - ($1 ~ /^row:/)
- Special keywords
  - BEGIN, END
- Empty
  - matches on any record
- pattern1,pattern2
  - matches to the record matched by pattern1 until the record matched by pattern2

awk actions

- Action part enclosed in curly braces
- Commands on same line separated by semicolon
- Very C like, with additional niceties
  - for (key in list)
  - next jumps to next record
  - nextfile jumps to next file
  - ARGC, ARGV, ENVIRON

Additional awk capabilities

- awk can split text into an array
- awk can call subprocesses with the pipe character
  - Don't forget to close!
- awk can redirect output to a file
  - > initially will overwrite, subsequently append

Suggestions

- You'll see shell scripts that let sed and awk do processing the shell can't do internally
  - Perhaps in a shell if statement, perhaps in a loop
  - Process launches are expensive and shell scripts are error-prone
- Larry Wall: Perl proposed as a "replacement" for sed and awk
  - use strict, better error handling, traditional file operations
  - Syntax designed for easy adaptation
  - Perl regular expressions are gold standard
- I use both, depending on whether I just want an answer, or want regular, automated processing
- They excel for
  - awk: extracting fields
  - sed: inline find and replace on a stream

Howtos

- AFS listvol output
  - Header lines, backup and readonly volumes
  - Want statistics on readwrite volumes
  - min, max, mean, standard deviation
  - support sample and population standard deviation
- Passphrase generator
  - Self-contained program that can generate assigned passphrases given a word list