AWK is a data driven programming language designed for processing text-based data, either in files or data streams. It is an example of a programming language that extensively uses the string datatype, associative arrays (that is, arrays indexed by key strings), and regular expressions. WIKI

Here are the most Kick ass voted AWK commands.

1) List of commands you use most often

history | awk '{a[$2]++}END{for(i in a){print a[i] " " i}}' | sort -rn | head

2) Display a block of text with AWK

awk '/start_pattern/,/stop_pattern/' file.txt

I find this terribly useful for grepping through a file, looking for just a block of text. There's "grep -A # pattern file.txt" to see a specific number of lines following your pattern, but what if you want to see the whole block? Say, the output of "dmidecode" (as root):

dmidecode | awk '/Battery/,/^$/'Will show me everything following the battery block up to the next block of text. Again, I find this extremely useful when I want to see whole blocks of text based on a pattern, and I don't care to see the rest of the data in output. This could be used against the '/etc/securetty/user' file on Unix to find the block of a specific user. It could be used against VirtualHosts or Directories on Apache to find specific definitions. The scenarios go on for any

text formatted in a block fashion. Very handy.

3) Graph # of connections for each hosts.

netstat -an | grep ESTABLISHED | awk ‘{print $5}’ | awk -F: ‘{print $1}’ | sort | uniq -c | awk ‘{ printf(“%s\t%s\t”,$2,$1) ; for (i = 0; i < $1; i++) {printf(“*”)}; print “” }’

Written for linux, the real example is how to produce ascii text graphs based on a numeric value (anything where uniq -c is useful is a good candidate).

4) Check your unread Gmail from the command line

curl -u username:password –silent “https://mail.google.com/mail/feed/atom” | tr -d ‘\n’ | awk -F ” ‘{for (i=2; i<=NF; i++) {print $i}}’ | sed -n “s/\(.*\) <\/title.*name>\(.*\)<\/name>.*/\2 – \1/p”

Checks the Gmail ATOM feed for your account, parses it and outputs a list of unread messages.

For some reason sed gets stuck on OS X, so here’s a Perl version for the Mac:

curl -u username:password --silent "https://mail.google.com/mail/feed/atom" | tr -d '\n' | awk -F '<entry>' '{for (i=2; i<=NF; i++) {print $i}}' | perl -pe 's/^<title>(.*) <\/title>.*<name>(.*)<\/name>.*$/$2 - $1/'If you want to see the name of the last person, who added a message to the conversation, change the greediness of the operators like this:

curl -u username:password --silent "https://mail.google.com/mail/feed/atom" | tr -d '\n' | awk -F '<entry>' '{for (i=2; i<=NF; i++) {print $i}}' | perl -pe 's/^<title>(.*) <\/title>.*?<name>(.*?)<\/name>.*$/$2 - $1/'

5) Remove duplicate entries in a file without sorting.

awk ‘!x[$0]++’ <file>

Using awk, find duplicates in a file without sorting, which reorders the contents. awk will not reorder them, and still find

and remove duplicates which you can then redirect into another file.

6) find geographical location of an ip address

lynx -dump http://www.ip-adress.com/ip_tracer/?QRY=$1|grep address|egrep 'city|state|country'|awk '{print $3,$4,$5,$6,$7,$8}'|sed 's\ip address flag \\'|sed 's\My\\'

I save this to bin/iptrace and run "iptrace ipaddress" to get the Country, City and State of an ip address using thehttp://ipadress.com service.

I add the following to my script to get a tinyurl of the map as well:

URL=`lynx -dump http://www.ip-adress.com/ip_tracer/?QRY=$1|grep details|awk '{print $2}'`

lynx -dump http://tinyurl.com/create.php?url=$URL|greptinyurl|grep "19. http"|awk '{print $2}'

7) Block known dirty hosts from reaching your machine

wget -qO – http://infiltrated.net/blacklisted|awk '!/#|[a-z]/&&/./{print "iptables -A INPUT -s "$1" -j DROP"}'

Blacklisted is a compiled list of all known dirty hosts (botnets, spammers, bruteforcers, etc.) which is updated on an hourly basis. This command will get the list and create the rules for you, if you want them automatically blocked, append |sh to the end of the command line. It's a more practical solution to block all and allow in specifics however, there are many who don't or can't do this which is where this script will come in handy. For those using ipfw, a quick fix would be {print "add deny ip from "$1" to any}. Posted in the sample output are the top two entries. Be advised the blacklisted file itself filters out RFC1918 addresses (10.x.x.x, 172.16-31.x.x, 192.168.x.x) however, it is advisable you check/parse the list before you implement the rules

8) Display a list of committers sorted by the frequency of commits

svn log -q|grep "|"|awk "{print \$3}"|sort|uniq -c|sort -nr

Use this command to find out a list of committers sorted by the

frequency of commits.

9) List the number and type of active network connections

netstat -ant | awk '{print $NF}' | grep -v '[a-z]' | sort | uniq -c

10) View facebook friend list [hidden or not hidden]

lynx -useragent=Opera -dump 'http://www.facebook.com/ajax/typeahead_friends.php?u=4&__a=1′ |gawk -F'\"t\":\"' -v RS='\",' 'RT{print $NF}' |grep -v '\"n\":\"' |cut -d, -f2

There's no need to be logged in facebook. I could do more JSON filtering but you get the idea…

Replace u=4 (Mark Zuckerberg, Facebook creator) with desired uid.

Hidden or not hidden… Scary, don't you?

11) List recorded formular fields of Firefox

cd ~/.mozilla/firefox/ && sqlite3 `cat profiles.ini | grep Path | awk -F= '{print $2}'`/formhistory.sqlite "select * from moz_formhistory" && cd – > /dev/null

When you fill a formular with Firefox, you see things you entered in previous formulars with same field names. This command list everything Firefox has registered. Using a "delete from", you can remove anoying Google queries, for example ;-)

12) Brute force discover

sudo zcat /var/log/auth.log.*.gz | awk '/Failed password/&&!/for invalid user/{a[$9]++}/Failed password for invalid user/{a["*" $11]++}END{for (i in a) printf "%6s\t%s\n", a[i], i|"sort -n"}'

Show the number of failed tries of login per account. If the user does not exist it is marked with *.

13) Show biggest files/directories, biggest

first with ‘k,m,g’ eyecandy

du –max-depth=1 | sort -r -n | awk ‘{split(“k m g”,v); s=1; while($1>1024){$1/=1024; s++} print int($1)” “v[s]”\t”$2}’

I use this on debian testing, works like the other sorted du variants, but i like small numbers and suffixes :)

14) Analyse an Apache access log for the most common IP addresses

tail -10000 access_log | awk ‘{print $1}’ | sort | uniq -c | sort -n | tail

This uses awk to grab the IP address from each request and then sorts and summarises the top 10

15) copy working directory and compress it on-the-fly while showing progress

tar -cf – . | pv -s $(du -sb . | awk ‘{print $1}’) | gzip > out.tgz

What happens here is we tell tar to create “-c” an archive of all files in current dir “.” (recursively) and output the data to stdout “-f -“. Next we specify the size “-s” to pv of all files in current dir. The “du -sb . | awk ?{print $1}?” returns number of bytes in current dir, and it gets fed as “-s” parameter to pv. Next we gzip the whole content and output the result to out.tgz file. This way “pv” knows how much data is still left to be processed and shows us that it will take yet another 4 mins 49 secs to finish.

Credit: Peteris Kruminshttp://www.catonmat.net/blog/unix-utilities-pipe-viewer/

16) List of commands you use most often

history | awk ‘{print $2}’ | sort | uniq -c | sort -rn | head

17) Identify long lines in a file

awk ‘length>72′ file

This command displays a list of lines that are longer than 72 characters. I use this command to identify those lines in my scripts and cut them short the way I like it.

18) Makes you look busy

alias busy='my_file=$(find /usr/include -type f | sort -R | head -n 1); my_len=$(wc -l $my_file | awk "{print $1}"); let "r = $RANDOM % $my_len" 2>/dev/null; vim +$r $my_file'

This makes an alias for a command named 'busy'. The 'busy' command opens a random file in /usr/include to a random line with vim. Drop this in your .bash_aliases and make sure that file is initialized in your .bashrc.

19) Show me a histogram of the busiest minutes in a log file:

cat /var/log/secure.log | awk '{print substr($0,0,12)}' | uniq -c | sort -nr | awk '{printf("\n%s ",$0) ; for (i = 0; i<$1 ; i++) {printf("*")};}'

20) Analyze awk fields

awk '{print NR": "$0; for(i=1;i<=NF;++i)print "\t"i": "$i}'

Breaks down and numbers each line and it's fields. This is really useful when you are going to parse something with awk but aren't sure exactly where to start.

21) Browse system RAM in a human readable form

sudo cat /proc/kcore | strings | awk 'length > 20′ | less

This command lets you see and scroll through all of the strings that are stored in the RAM at any given time. Press space bar to scroll through to see more pages (or use the arrow keys etc).

Sometimes if you don't save that file that you were working on or want to get back something you closed it can be found floating around in here!

The awk command only shows lines that are longer than 20 characters (to avoid seeing lots of junk that probably isn't "human readable").

If you want to dump the whole thing to a file replace the final '| less' with '> memorydump'. This is great for searching through many times (and with the added bonus that it doesn't overwrite any memory…).

Here’s a neat example to show up conversations that were had in pidgin (will probably work after it has been closed)…

sudo cat /proc/kcore | strings | grep '([0-9]\{2\}:[0-9]\{2\}:[0-9]\{2\})'(depending on sudo settings it might be best to run

sudo sufirst to get to a # prompt)

22) Monitor open connections for httpd including listen, count and sort it per IP

watch “netstat -plan|grep :80|awk {‘print \$5’} | cut -d: -f 1 | sort | uniq -c | sort -nk 1″

It’s not my code, but I found it useful to know how many open connections per request I have on a machine to debug connections without opening another http connection for it.

You can also decide to sort things out differently then the way it appears in here.

23) Purge configuration files of removed packages on debian based systems

sudo aptitude purge `dpkg –get-selections | grep deinstall | awk ‘{print $1}’`

Purge all configuration files of removed packages

24) Quick glance at who’s been using your system recently

last | grep -v “^$” | awk ‘{ print $1 }’ | sort -nr | uniq -c

This command takes the output of the ‘last’ command, removes empty lines, gets just the first field ($USERNAME), sort the $USERNAMES in reverse order and then gives a summary count of unique matches.

25) Number of open connections per ip.

netstat -ntu | awk ‘{print $5}’ | cut -d: -f1 | sort | uniq -c | sort -n

Here is a command line to run on your server if you think your

server is under attack. It prints our a list of open connections to your server and sorts them by amount.

BSD Version:

netstat -na |awk '{print $5}' |cut -d "." -f1,2,3,4 |sort |uniq -c |sort -nr

And there you have it killer awk usages. Now I know you might be thinking these are NOT awk commands. Maybe not, but awk was used to filter out data.
Did I make a mistake?,
did I leave something cool behind?
Please feel free to comment.

AWK

PREVIOUS POST

25 Best SSH Commands / Tricks

NEXT POST

Webmasters Block These IP Addresses

8 THOUGHTS ON "25 BEST AWK COMMANDS / TRICKS"

Pingback: Tweets that mention 25 Best AWK Commands / Tricks -- Topsy.com

Jackie

NOVEMBER 27, 2010 AT 12:17 AM

Awesome Article, You just keep coming up with great stuff. Keep up the good work

REPLY

Pingback: da non perdere « Uno scrittoio digitale …

Mark

NOVEMBER 27, 2010 AT 10:27 AM

while trying this one: curl -u username:password –silent “https://mail.google.com/mail/feed/atom” | tr -d ‘\n’ | awk -F ‘’ ‘{for (i=2; i<=NF; i++) {print $i}}’ | sed -n “s/\(.*\)\(.*\).*/\2 – \1/p”

i get “bash: syntax error near unexpected token `(‘ ”
any reason why it doesnt like the command?

REPLY

Isaiah

NOVEMBER 27, 2010 AT 11:59 PM

hey mark, try this one

curl -u username –silent “https://mail.google.com/mail/feed/atom” | perl -ne ‘print “\t” if //; print “$2\n” if /< (title|name)>(.*)< \/\1>/;’
for some reason wordpress decided to create a space between < and (in <(title|name)>
just erase the extra space

notice what happens when there is more than one unread message in a thread…

also people please dont hardcode the password when you use curl. Leave it out and curl will ask you when it runs. Please…?

but if you must try this one
curl -u username:password –silent “https://mail.google.com/mail/feed/atom” | tr -d ‘\n’ | awk -F ‘’ ‘{for (i=2; i< =NF; i++) {print $i}}' | sed -n "s/