Introduccion a AWK

Jesús Espino García

Grupo de usuarios de Linux
Universidad Carlos III de Madrid.

10 de Octubre de 2008

Un poco de historia

Historia

- Desarrollado por Alfred V. Aho, Peter J. Weinberger y Brian W. Kernighan
- Version original escrita en los laboratorios de AT&T en 1977.
- En 1985 aparece una nueva version con muchas nuevas caracteristicas.
- En 1986 aparece la implementacion gnu de AWK, GAWK.

¿Qué es AWK?

Definición (AWK)

AWK es equivalente a una navaja del ejercito suizo, que es útil para modificar archivos, buscar y transformar bases de datos, generar informes simples y otras muchas cosas.

Gestión de la entrada

Tratamiento de la entrada

- Cada fichero se divide en registros. Normalmente una registro por linea.
- Cada registro se divide en campos. Normalmente un campo por palabra.
- Los ficheros se procesan registro a registro.
- Para cada registro se generan las variables `$0` para todo el registro y `$1`, `$2`, `$3`... para cada campos.

Reglas en AWK

- En AWK el código se compone de reglas, divididas en:

Patron

Un patron que se ajuste al registro. Ej: `/hola/`.

Accion

Un codigo a ejecutar. Ej: `{print 'hola'}`.

- Si el patron coincide, se ejecuta al accion.

Patrones

Patrones existentes

- `/expresion regular/`: Ejecuta la accion si el texto de entrada concuerda con la expresion regular. Ej: `/^192.168/{print $0}`.
- `expresion`: Ejecuta la acción si la expresión es verdadera. Ej: `$1 == 192.168.0.1{print $0}`.
- `pat1,pat2`: Ejecuta la acción para todas las cadenas entre la cadena que coincida con el primer patron y la que siguiente que coincida con el segundo patron. Ej: `/^a/,/^b/{print $0}`.
- `BEGIN`: Se ejecuta la acción antes de procesar ninguna entrada. Ej: `BEGIN{print 'Principio de programa'}`.
- `END`: Se ejecuta la acción después de haber procesado toda la entrada. Ej: `END{print 'Fin de programa'}`.
- `null`: Ejecuta siempre la acción. Ej: `{print $0}`

Comparadores

Comparadores

- `x >` y: Si x es mayor que y.
- `x >=` y: Si x es mayor o igual que y.
- `x <` y: Si x es menor que y.
- `x <=` y: Si x es menor o igual que y.
- `x ==` y: Si x es igual a y.
- `x !=` y: Si x es diferente de y.
- `x ~` y: Si x encaja en la expresión regular y.
- `x !~` y: Si x no encaja en la expresión regular y.

Operadores

Operadores

- `x + y`: Suma.
- `x - y`: Resta.
- `- x`: Negación.
- `x * y`: Multiplicación.
- `x / y`: División.
- `x % y`: Resto.
- `x ^ y`: Potencia.
- `x ** y`: Potencia.
- `x = y`: Asignacion.

Sentencias de control

Sentencias de control

- `if`: Bifurcacion if.
- `while`: Bucle while.
- `for`: Bucle for.
- `break`: Sale del bucle actual.
- `continue`: Salta al proximo ciclo del bucle.
- `next`: Hace que pase a la siguiente cadena sin tener en cuenta el resto de las reglas.
- `exit`: Hace que termine la ejecucion.

Variables predefinidas para el sistema

Variables predefinidas para el sistema

- `IGNORECASE`: Sensibilidad a mayúsculas.
- `FS`: Separador de campos.
- `RS`: Separador de registros.
- `OFS`: Separador de campos a la salida.
- `ORS`: Separador de registros a la salida.

Variables predefinidas para el usuario

Variables predefinidas para el usuario

- `ARGC`: Numero de argumentos pasados.
- `ARGV`: Argumentos pasados.
- `ENVIRON`: Array con las variables de entorno.
- `FILENAME`: El fichero que esta leyendo awk actualmente.
- `FNR`: Numero del registro actual.
- `NF`: Numero de campos del registro actual.
- `NR`: Numero de registros procesados hasta el momento.

Acciones

Acciones

- Cada acción es un código de lo que debe hacer.
- Una de las principales funciones y mas usadas es `print`.
- Existen muchas otras funciones predefinidas, podemos encontrar algunas en `man awk`.
- Se pueden definir funciones propias.

Ejemplos simples

Ejemplo (Codigos de una linea)

- `{print $0}` == `cat`.
- `/cadena/{print $0}` == `grep cadena`.
- `(NR%2)==0{print $0}`: *Imprime las lineas impares.*
- `BEGIN{FS=','}`: *Establece que el separador de campo es ","*.
- `END{print 'Programa terminado'}`: *Muestra un fin de programa al final*

Ejemplo complejo I

Ejemplo (Analisis de logs de apache)

```
BEGIN{r200=0; r301=0; r304=0; r404=0;}
$9==200{r200++; next;}
$9==301{r301++; next;}
$9==304{r304++; next;}
$9==404{r404++; next;}
END{print "OK: " r200;
    print "Moved Permanently: " r301;
    print "Not Modified: " r304;
    print "Not Found: " r404;
    print "Others: " NR-r200-r301-r304-r404;
    print "Total: " NR}
```

Ejemplo complejo II

Ejemplo (Analisis de diccionario)

Ver el ejemplo dict.awk en la carpeta ejemplos.

Referencias

- `man awk`: Pagina del manual.
- http://www.gnu.org/software/gawk/manual/: Manual de AWK de GNU.

Dudas

. . .

Fin