

The Bro Network Security Monitor

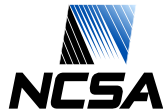


Tools of the Trade

Matthias Vallentin
UC Berkeley / ICSI
vallentin@icir.org



Bro Workshop 2011
NCSA, Champaign-Urbana, IL



Tools of the Trade

Basic Toolbox

1. awk
2. head/tail
3. sort
4. uniq
5. bro-cut



Tools of the Trade

awk

Swiss-army knife for log processing.

- ▶ Pattern-action statement: `awk 'pattern { action }'`

Tools of the Trade

awk

Swiss-army knife for log processing.

- ▶ Pattern-action statement: `awk 'pattern { action }'`
 - ▶ `awk '/start/, /stop/'`

Tools of the Trade

awk

Swiss-army knife for log processing.

- ▶ Pattern-action statement: `awk 'pattern { action }'`
 - ▶ `awk '/start/, /stop/'`
 - ▶ `awk 'length($0) > 72'`

Tools of the Trade

awk

Swiss-army knife for log processing.

- ▶ Pattern-action statement: `awk 'pattern { action }'`
 - ▶ `awk '/start/, /stop/'`
 - ▶ `awk 'length($0) > 72'`
 - ▶ `awk '$1 == "127.0.0.1" && $2 ~ /foo/'`

Tools of the Trade

awk

Swiss-army knife for log processing.

- ▶ Pattern-action statement: `awk 'pattern { action }'`
 - ▶ `awk '/start/, /stop/'`
 - ▶ `awk 'length($0) > 72'`
 - ▶ `awk '$1 == "127.0.0.1" && $2 ~ /foo/'`
 - ▶ `awk '$1 == "127.0.0.1" { x += $3 } END { print x }'`

Tools of the Trade

awk

Swiss-army knife for log processing.

- ▶ Pattern-action statement: `awk 'pattern { action }'`
 - ▶ `awk '/start/, /stop/'`
 - ▶ `awk 'length($0) > 72'`
 - ▶ `awk '$1 == "127.0.0.1" && $2 ~ /foo/'`
 - ▶ `awk '$1 == "127.0.0.1" { x += $3 } END { print x }'`
 - ▶ `awk '{ x[$1] += $3 } END { for (i in x) print x[i] }'`

Tools of the Trade

awk

Swiss-army knife for log processing.

- ▶ Pattern-action statement: `awk 'pattern { action }'`
 - ▶ `awk '/start/, /stop/'`
 - ▶ `awk 'length($0) > 72'`
 - ▶ `awk '$1 == "127.0.0.1" && $2 ~ /foo/'`
 - ▶ `awk '$1 == "127.0.0.1" { x += $3 } END { print x }'`
 - ▶ `awk '{ x[$1] += $3 } END { for (i in x) print x[i] }'`
 - ▶ `awk 'BEGIN { x["6.6.6.6"]++ } { if ($1 in x) yikes() }'`

Tools of the Trade

awk

Swiss-army knife for log processing.

- ▶ Pattern-action statement: `awk 'pattern { action }'`
 - ▶ `awk '/start/, /stop/'`
 - ▶ `awk 'length($0) > 72'`
 - ▶ `awk '$1 == "127.0.0.1" && $2 ~ /foo/'`
 - ▶ `awk '$1 == "127.0.0.1" { x += $3 } END { print x }'`
 - ▶ `awk '{ x[$1] += $3 } END { for (i in x) print x[i] }'`
 - ▶ `awk 'BEGIN { x["6.6.6.6"]++ } { if ($1 in x) yikes() }'`
- ▶ Useful functions: `length`, `substr`, `match`, `split`, `(g)sub`, `tolower`

Tools of the Trade

awk

Swiss-army knife for log processing.

- ▶ Pattern-action statement: `awk 'pattern { action }'`
 - ▶ `awk '/start/, /stop/'`
 - ▶ `awk 'length($0) > 72'`
 - ▶ `awk '$1 == "127.0.0.1" && $2 ~ /foo/'`
 - ▶ `awk '$1 == "127.0.0.1" { x += $3 } END { print x }'`
 - ▶ `awk '{ x[$1] += $3 } END { for (i in x) print x[i] }'`
 - ▶ `awk 'BEGIN { x["6.6.6.6"]++ } { if ($1 in x) yikes() }'`
- ▶ Useful functions: `length`, `substr`, `match`, `split`, `(g)sub`, `tolower`
- ▶ Useful variables:
 - `NF` Number of fields in current record
 - `NR` Number of current record

Tools of the Trade

head

`-n` Output the **first** n lines

tail

`-n` Output the **last** n lines

Tools of the Trade

head

`-n` Output the **first** n lines

tail

`-n` Output the **last** n lines

sort

(External) sorting, grouping, and duplicate filtering

- ▶ Useful options:

Tools of the Trade

head

`-n` Output the **first** n lines

tail

`-n` Output the **last** n lines

sort

(External) sorting, grouping, and duplicate filtering

▶ Useful options:

`-n` Numerical comparison

Tools of the Trade

head

`-n` Output the **first** n lines

tail

`-n` Output the **last** n lines

sort

(External) sorting, grouping, and duplicate filtering

- ▶ Useful options:
 - `-n` Numerical comparison
 - `-r` Reverse sort order

Tools of the Trade

head

`-n` Output the **first** n lines

tail

`-n` Output the **last** n lines

sort

(External) sorting, grouping, and duplicate filtering

▶ Useful options:

- `-n` Numerical comparison
- `-r` Reverse sort order
- `-u` Output each value only once (unique)

Tools of the Trade

head

`-n` Output the **first** n lines

tail

`-n` Output the **last** n lines

sort

(External) sorting, grouping, and duplicate filtering

► Useful options:

`-n` Numerical comparison

`-r` Reverse sort order

`-u` Output each value only once (unique)

`-k` Sort by column range (from[,to]; e.g., `-k 2,3`)

Tools of the Trade

head

`-n` Output the **first** n lines

tail

`-n` Output the **last** n lines

sort

(External) sorting, grouping, and duplicate filtering

► Useful options:

- `-n` Numerical comparison
- `-r` Reverse sort order
- `-u` Output each value only once (unique)
- `-k` Sort by column range (from[,to]; e.g., `-k 2,3`)
- `-S` Specify buffer size (e.g., `-S 1G`)

Tools of the Trade

head

`-n` Output the **first** n lines

tail

`-n` Output the **last** n lines

sort

(External) sorting, grouping, and duplicate filtering

► Useful options:

- `-n` Numerical comparison
- `-r` Reverse sort order
- `-u` Output each value only once (unique)
- `-k` Sort by column range (from[,to]; e.g., `-k 2,3`)
- `-S` Specify buffer size (e.g., `-S 1G`)
- `-T` Specify temporary file directory (e.g., `-T=/fast/tmp`)

Tools of the Trade

head

`-n` Output the **first** n lines

tail

`-n` Output the **last** n lines

sort

(External) sorting, grouping, and duplicate filtering

► Useful options:

`-n` Numerical comparison

`-r` Reverse sort order

`-u` Output each value only once (unique)

`-k` Sort by column range (from[,to]; e.g., `-k 2,3`)

`-S` Specify buffer size (e.g., `-S 1G`)

`-T` Specify temporary file directory (e.g., `-T=/fast/tmp`)

► Examples:

► `awk '{ print $3 }' conn.log | sort -S 1G -u`

► `sort -rn -k 9 conn.log | head -n 10`

Tools of the Trade

`uniq`

Filter repeated lines

`-c` Precede each line with count of occurrence

Tools of the Trade

uniq

Filter repeated lines

- c Precede each line with count of occurrence
- d Output lines that are repeated

Tools of the Trade

uniq

Filter repeated lines

- c Precede each line with count of occurrence
- d Output lines that are repeated
- u Output lines that are *not* repeated

Tools of the Trade

uniq

Filter repeated lines

- c Precede each line with count of occurrence
- d Output lines that are repeated
- u Output lines that are *not* repeated

Example input

A
A
A
A
B
B
B
C

Tools of the Trade

uniq

Filter repeated lines

- c Precede each line with count of occurrence
- d Output lines that are repeated
- u Output lines that are *not* repeated

Example input

```
A
A
A
A
B
B
B
C
```

Example output

- ▶ `uniq -c`
- ▶ `uniq -d`
- ▶ `uniq -u`

Tools of the Trade

uniq

Filter repeated lines

- c Precede each line with count of occurrence
- d Output lines that are repeated
- u Output lines that are *not* repeated

Example input

```
A
A
A
A
B
B
B
C
```

Example output

```
▶ uniq -c
    4 A
    3 B
    1 C

▶ uniq -d

▶ uniq -u
```

Tools of the Trade

uniq

Filter repeated lines

- c Precede each line with count of occurrence
- d Output lines that are repeated
- u Output lines that are *not* repeated

Example input

```
A
A
A
A
B
B
B
C
```

Example output

```
▶ uniq -c
    4 A
    3 B
    1 C

▶ uniq -d
A
B

▶ uniq -u
```

Tools of the Trade

uniq

Filter repeated lines

- c Precede each line with count of occurrence
- d Output lines that are repeated
- u Output lines that are *not* repeated

Example input

```
A
A
A
A
B
B
B
C
```

Example output

```
▶ uniq -c
    4 A
    3 B
    1 C

▶ uniq -d
A
B

▶ uniq -u
C
```

Tools of the Trade

bro-cut

- ▶ New awk-based field extractor for Bro logs
- ▶ List files to extract as arguments

```
bro-cut [options] <columns>
```

Extracts the given columns from an ASCII Bro log on standard input. By default, bro-cut does not include format header blocks into the output.

```
Example: cat conn.log | bro-cut -d ts id.orig_h id.orig_p
```

- c Include the first format header block into the output.
- C Include all format header blocks into the output.
- d Convert time values into human-readable format (needs gawk).
- D <fmt> Like -d, but specify format for time (see strftime(3) for syntax).

For the time conversion, the format string can also be specified by setting an environment variable BRO_CUT_TIMEFMT.

Tools of the Trade

bro-cut

- ▶ `bro-cut ts id.orig_h id.resp_p < conn.log`
1319742168.465601 192.150.187.147 80
1319742167.737945 192.150.187.147 80

Tools of the Trade

bro-cut

- ▶ `bro-cut ts id.orig_h id.resp_p < conn.log`
1319742168.465601 192.150.187.147 80
1319742167.737945 192.150.187.147 80
- ▶ `bro-cut host uri < http.log | awk '{ print $1$2 }'`
s0.2mdn.net/879366/flashwrite_1_2.js
maps.google.com/mapfiles/home3.html

Tools of the Trade

bro-cut

- ▶ `bro-cut ts id.orig_h id.resp_p < conn.log`
1319742168.465601 192.150.187.147 80
1319742167.737945 192.150.187.147 80
- ▶ `bro-cut host uri < http.log | awk '{ print $1$2 }'`
s0.2mdn.net/879366/flashwrite_1_2.js
maps.google.com/mapfiles/home3.html
- ▶ `bro-cut -d ts < conn.log`
2011-10-27T12:02:48-0700

Tools of the Trade

bro-cut

- ▶ `bro-cut ts id.orig_h id.resp_p < conn.log`
1319742168.465601 192.150.187.147 80
1319742167.737945 192.150.187.147 80
- ▶ `bro-cut host uri < http.log | awk '{ print $1$2 }'`
s0.2mdn.net/879366/flashwrite_1_2.js
maps.google.com/mapfiles/home3.html
- ▶ `bro-cut -d ts < conn.log`
2011-10-27T12:02:48-0700
- ▶ `bro-cut -D '%s' ts orig_bytes resp_bytes \
< conn.log \
| sort -n \
| awk '{ if ($1 == ts) { size+=$2+$3 } \
else { if (size != 0) print $1, size; \
ts=$1; size=0 } }'`
1319742168 33628
1319742169 22814

Caveats

Match IP addresses correctly

- ▶ `grep 1.2.3.4 conn.log`
- ▶ `fgrep 1.2.3.4 conn.log`
- ▶ `awk '$3 == "1.2.3.4" || $5 == "1.2.3.4"' conn.log`

Caveats

Match IP addresses correctly

- ▶ `grep 1.2.3.4 conn.log` ✗ 2102x3048
- ▶ `fgrep 1.2.3.4 conn.log`
- ▶ `awk '$3 == "1.2.3.4" || $5 == "1.2.3.4"' conn.log`

Caveats

Match IP addresses correctly

- ▶ `grep 1.2.3.4 conn.log` ✗ 2102x3048
- ▶ `fgrep 1.2.3.4 conn.log` ✗ 21.2.3.48
- ▶ `awk '$3 == "1.2.3.4" || $5 == "1.2.3.4"' conn.log`

Caveats

Match IP addresses correctly

- ▶ `grep 1.2.3.4 conn.log` ✗ 2102x3048
- ▶ `fgrep 1.2.3.4 conn.log` ✗ 21.2.3.48
- ▶ `awk '$3 == "1.2.3.4" || $5 == "1.2.3.4"' conn.log` ✓

Caveats

Match IP addresses correctly

- ▶ `grep 1.2.3.4 conn.log` ✗ 2102x3048
- ▶ `fgrep 1.2.3.4 conn.log` ✗ 21.2.3.48
- ▶ `awk '$3 == "1.2.3.4" || $5 == "1.2.3.4"' conn.log` ✓

Know your memory limits

- ▶ `awk '{ x[$1]++ } END { for (i in x) print x[i] }'`

Caveats

Match IP addresses correctly

- ▶ `grep 1.2.3.4 conn.log` ✗ 2102x3048
- ▶ `fgrep 1.2.3.4 conn.log` ✗ 21.2.3.48
- ▶ `awk '$3 == "1.2.3.4" || $5 == "1.2.3.4"' conn.log` ✓

Know your memory limits

- ▶ `awk '{ x[$1]++ } END { for (i in x) print x[i] }'` ✗

Caveats

Match IP addresses correctly

- ▶ `grep 1.2.3.4 conn.log` ✗ 2102x3048
- ▶ `fgrep 1.2.3.4 conn.log` ✗ 21.2.3.48
- ▶ `awk '$3 == "1.2.3.4" || $5 == "1.2.3.4"' conn.log` ✓

Know your memory limits

- ▶ `awk '{ x[$1]++ } END { for (i in x) print x[i] }'` ✗
- ▶ `awk '{ print $1 } | sort -S=2G | uniq -c'` ✓

Questions?

WE ARE THE 99%
The People are too big to fail.

