Smart Start: Linux/Networking Command-line processors

Sergii Kudriavtsev

Agenda

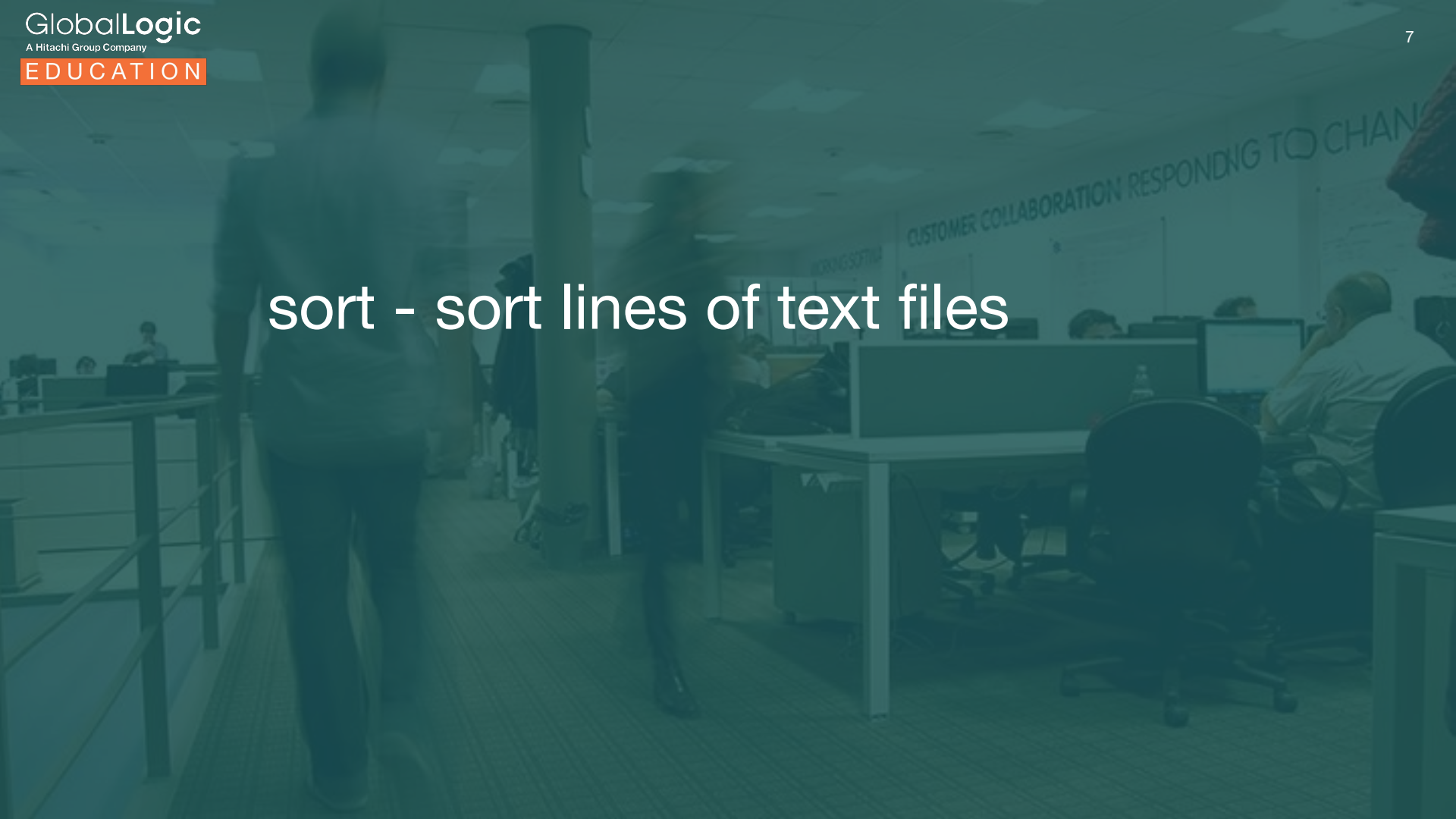
1. cut
2. tr
3. sort
4. sed
5. awk
6. grep

cut - remove sections from each line of files

- cut
  - ```
    $ echo 192.168.0.100 | cut -d. -f 2
    168
    ```
  - ```
    $ echo 192.168.0.100 | cut -d. -f 3,1
    192.0
    ```

tr - translate or delete characters

- tr
  - `$ echo belly | tr 'l' 'r' # transforms "belly" to "berry"`
  - `$ echo belly | tr '[A-Za-f]' 'r' # transforms "be lly" to "rxrllyx"`
  - `$ echo belly | tr 'l' 'r' | tr 'b' 'f' # transforms "belly" to "ferry"`

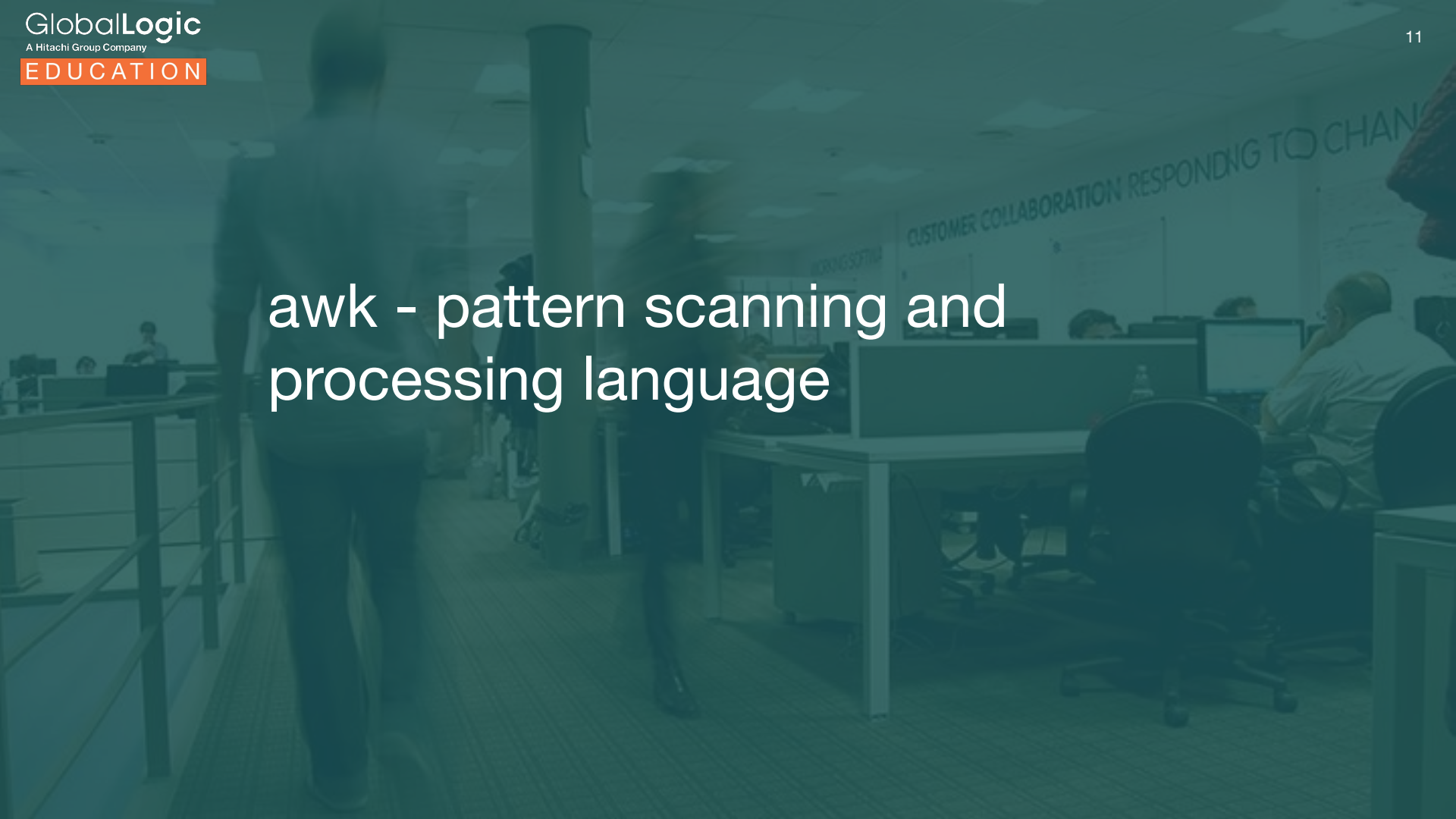


sort - sort lines of text files

- sort
  - `$ sort file.txt`
  - `$ cat file.txt | sort    # sort alphabetically (a, b, b, c, d)`
  - `$ cat file.txt | sort -u # sort alphabetically and remove duplicates (a, b, c, d)`
  - `$ cat file.txt | sort -r # reverse alphabetical sort (d, c, b, b, a)`
  - `$ cat file.txt | sort -n # sort numerically (1, 2, 5, 9, 11, 17)`
  - `$ ls -l | sed 1d | sort -k5 -n -r`

sed - stream editor for filtering and transforming text

- sed
  - Replacement
    - Instruction **'s/old/new/'** replaces only the first **old** substring with **new** substring.
    - Instruction **'s/old/new/g'** replaces all **old** substrings with **new** substrings.
  - Special characters:
    - ^ - beginning of line
    - $ - end of line
    - https://www.gnu.org/software/sed/manual/html_node/Regular-Expressions.html
  - `$ echo belly | sed -e 's/l/r/' # transforms "belly" to "berly"`
  - `$ echo belly | sed -e 's/l/r/g' # transforms "belly" to "berry"`
  - `$ echo belly | sed -e 's/l/r/g' -e 's/b/f/g' # transforms "belly" to "ferry"`
  - `$ echo belly | sed -e 's/l/r/g' -e 's/b/ch/g' # transforms "belly" to "cherry"`
  - `$ echo overlook | sed -e 's/look/view/g' # transforms "overlook" to "overview"`
  - `$ echo overlook | sed -e 's/look//g' # transforms "overlook" to "over"`
  - `$ echo berry | sed 's/\([a-f]\)/_\1_/g' # surrounds each letter from interval a-f with underscore`
  - `$ cat .profile | sed '5d' # remove line number 5`
  - `$ cat .profile | sed '5!d' # remove all except line number 5`



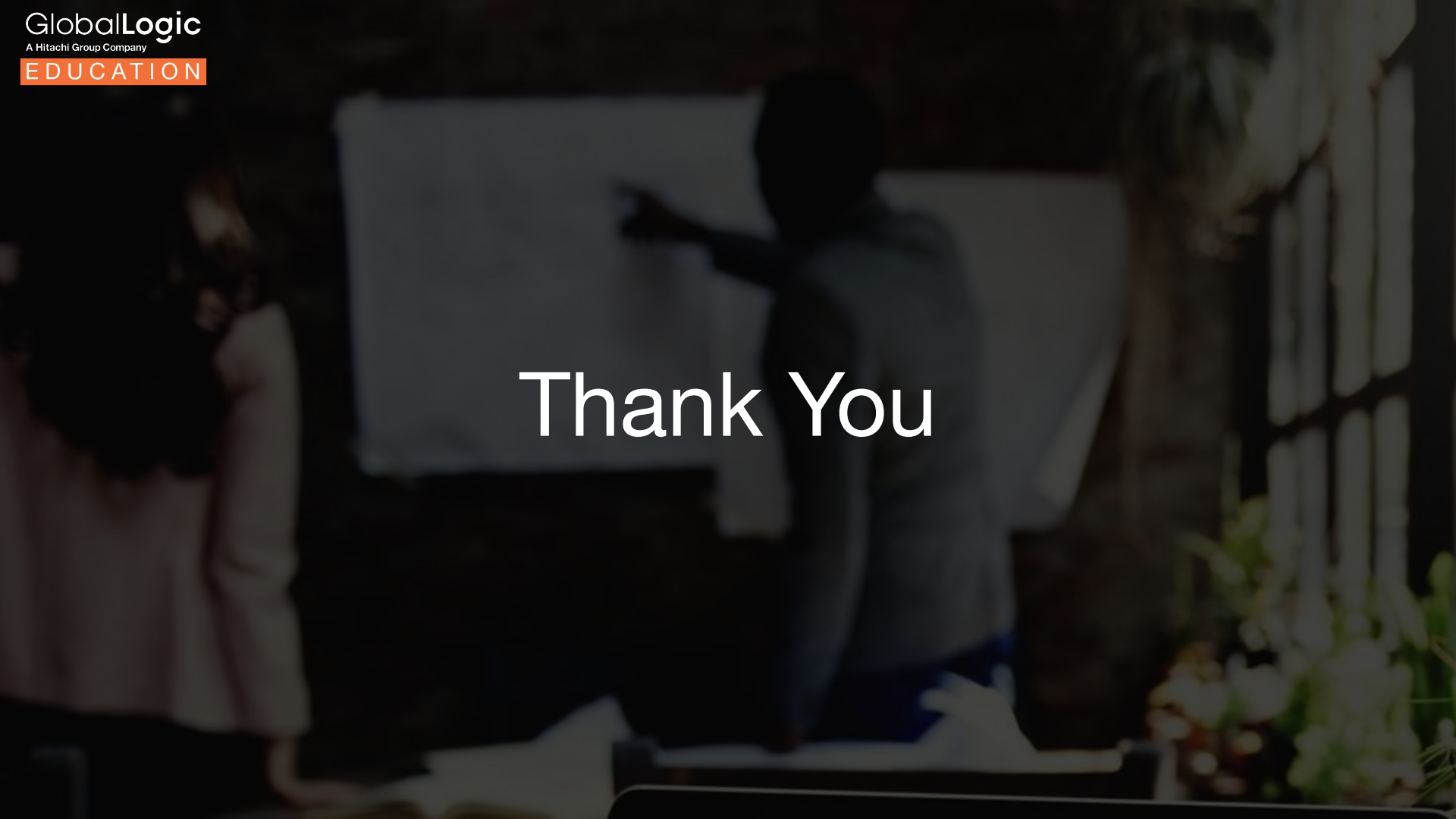
awk - pattern scanning and processing language

- awk
  - `$ ls -l | awk '{print $5}' # print 5th field`
  - `$ ls -l | awk '{printf "Test: %s %s %s\n", $6, $7, $8}' # print 6th, 7th and 8th fields`
  - `$ ls -l | awk '{print $NF}' # print the last field`
  - `$ cat /etc/passwd | awk -F : '{print $3}' # print the 3rd field. Fields separated by :`

grep - print lines matching a pattern

- grep
  - `$ grep 2014 file1.txt file2.txt file3.txt`
    - `$ grep -H 2014 file1.txt file2.txt file3.txt #print file name with output lines`
    - `$ grep -HR 2014 dir1 #likewise -r, but follow all symlinks`
  - `$ grep 2015 file.txt`
  - `$ cat file1.txt file2.txt file3.txt | grep 2014`
  - `$ cat file.txt | grep 2015`
  - `$ grep -v pattern file.txt`
  - `$ grep -i paTTeRn file.txt`

- egrep (grep -E)
  - `$ egrep # is the same as grep -E`
  - `$ man egrep`
    - Sections
      - REGULAR EXPRESSIONS
  - `$ egrep "201[45]|pattern2" file.txt`
    - [a-z0-9ABCFT] - matches any 1 character from the set
      - [a-z0-9ABCFT]{2,} - matches any 2 characters from the set
      - [a-z0-9ABCFT]{2,4} - matches any 2, 3 or 4 characters from the se
        - a0A, zc
      - [a-z0-9ABCFT]* - matches any number of characters from the set
    - [^a-z0-9ABCFT] - matches any 1 character NOT from the set
    - [[:digit:]], [[:alnum:]] and so on (see man grep)
    - | - union of 2 regular expressions
- fgrep (grep -F)
  - `$ fgrep # is the same as grep -F`
    - `$ fgrep -f patterns.txt log.txt`


GlobalLogic
A Hitachi Group Company
EDUCATION
Thank You