

# Efficient and Playful Tools to Teach Unix to New Students

Matthieu Moy

Grenoble-INP, Ensimag

Updated : June 2014

# Outline

- 1 Unix Course in Ensimag
- 2 Treasure Hunt
- 3 (Fully Automated) Lab Exam
- 4 Conclusion

# Context of Unix Introduction

- Start of 1<sup>st</sup> year
- Students don't expect to work hard (Prep. school Vs Engineering school)
- Few total beginners with Unix, some experienced users

# Context of Unix Introduction

- Start of 1<sup>st</sup> year
- Students don't expect to work hard (Prep. school Vs Engineering school)
- Few total beginners with Unix, some experienced users
- Teaching material *must* be attractive!
- Let students progress at their own pace
- Grading is (unfortunately) needed

## Target Students

### Experienced



- Already know what they need
- Can help/teach others

### Willing



- Don't know much about Unix
- But want to learn

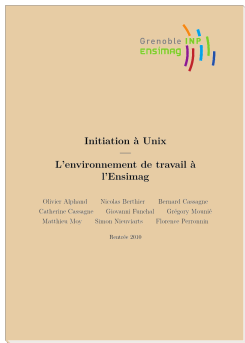
### Lazy



- Not interested in learning Unix

# Basic Teaching Material

## Textbook



- Linear structure
- Beginner-oriented + remarks for advanced users

## Wiki



- Web of articles
- Used through 3 years of study

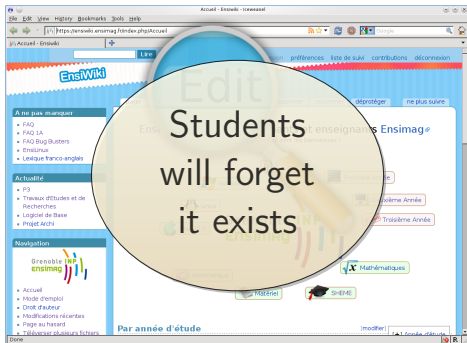
# Basic Teaching Material

## Textbook



- Linear structure
- Beginner-oriented + remarks for advanced users

## Wiki



- Web of articles
- Used through 3 years of study

# Outline

- 1 Unix Course in Ensimag
- 2 Treasure Hunt
- 3 (Fully Automated) Lab Exam
- 4 Conclusion



# Treasure Hunt: Goals & Ideas


- Old style:
  - ▶ Typical question in textbook:

Try typing the command `blah`. What do you see?

# Treasure Hunt: Goals & Ideas

- Old style:
  - ▶ Typical question in textbook:



Try typing the command `blah`. What do you see?

- ▶  "I already know that, I'll skip to the next question"

# Treasure Hunt: Goals & Ideas

- Old style:
  - ▶ Typical question in textbook:




Try typing the command `blah`. What do you see?

- ▶  "I already know that, I'll skip to the next question"
- ▶  "I already know that, it's boring, I'll stop reading the book"

# Treasure Hunt: Goals & Ideas

- Old style:
  - ▶ Typical question in textbook:




Try typing the command `blah`. What do you see?

- ▶  “I already know that, I’ll skip to the next question”
- ▶  “I already know that, it’s boring, I’ll stop reading the book”
- ▶  types something else, sees something else, doesn't notice

# Treasure Hunt: Goals & Ideas

- Old style:
  - ▶ Typical question in textbook:

Try typing the command `blah`. What do you see?

- ▶  “I already know that, I’ll skip to the next question”
  - ▶  “I already know that, it’s boring, I’ll stop reading the book”
  - ▶  types something else, sees something else, doesn’t notice
- “Treasure hunt”-style:

You’ll get the next question using command `blah`. See you there.

# Treasure Hunt: Example Questions

## Step A1

follow this link to reach step A2 (warm up):

<http://www-verimag.imag.fr/~moy/jeu-de-piste/etape-A2.txt>

## Step A2

Good, you've solved the step A1.

For the next step (A2), here it is, but it is rot13-encoded.

At the point where you are, you should be able to find what rot13 is, and a way to decode it.

`lbvyn, gur ebg13 vf qrpbrq.`

`Vs lbh'er fzneg, lbh'ir cebonoyl abgvprq gung gur jro vf shyy bs bayvar rapbqre/qrpbrqre sbe ebg13,[...]`

# Treasure Hunt: Example Questions

## Step A1

follow this link to reach step A2 (warm up):

<http://www-verimag.imag.fr/~moy/jeu-de-piste/etape-A2.txt>

## Step A2

Good, you've solved the step A1.

For the next step (A2), here it is, but it is rot13-encoded.

At the point where you are, you should be able to find what rot13 is, and a way to decode it.

Ibvyn, gur ebg13 vf qrpbrq.

Vs lbh'er fzneg, lbh'ir cebonoyl abgvprq gung gur jro vf shyy bs bayvar rapbqre/qrpbrqre sbe ebg13,[...]

⇒ Google “rot13”, first answer is a rot13 decoder  
 (“think outside the box”-question)

## Treasure Hunt: Topics

- A **Internet**: search engines, wiki, email
- B **Basics**: Navigate in filesystem, manipulate files (`cd`, `cp`, `mv...`), compile programs &  $\text{\LaTeX}$ , execute commands (`./script.sh`)
- C **Useful applications**: open files (PDF, OpenOffice, image)
- D **Text editor**: search & replace, fix syntax error in big program
- E **Commands**: extract TAR archive, hidden files, `diff`,...
- F **Bash**: redirections (`>`, `<`, `|`), wildcards
- G **Remote access**: SSH, sftp
- H **Bonus questions**: scripting, Git, `strace`, `/proc/`, ssh keys

total = 36 steps

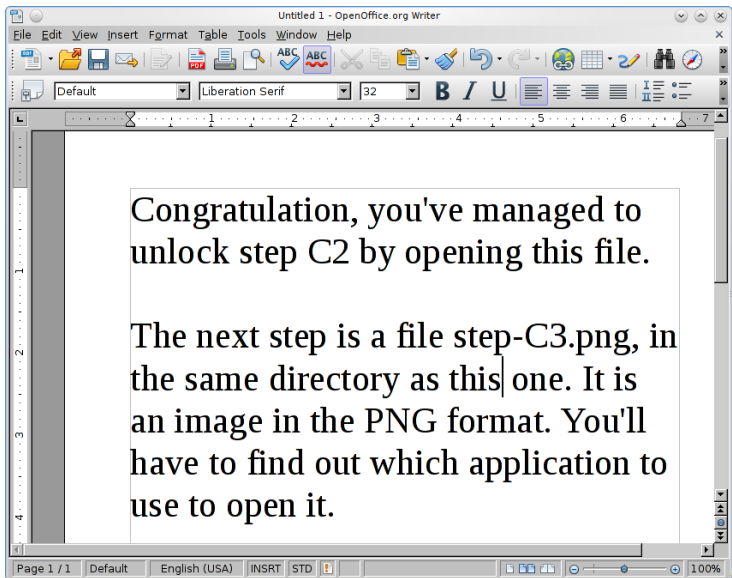


## Basic Ideas Shared by Most Steps

- Essentially short manipulations to get the answer
- Step  $N + 1$  explains better the answer(s) of step  $N$
- Obfuscate the instructions to next step
  - ▶ Files in non-listable directory (HTTP or `chmod -r dir/`)
  - ▶ Obfuscated program to compile/execute
  - ▶ Actual information hidden in random garbage

Common obfuscation scripts grouped in an open-source “generation library”

# Generation Library: Example Step (Student-side)



## Generation Library: Example Step (Teacher-side)

```
#!/bin/bash
```

```
./imglib.sh
```

```
echo "Congratulation, you've managed to open this PNG image.
```

You can edit it with The Gimp (command `gimp`), view it with Eye of Gnome (command `eog`), for example.

We'll now continue with the part on text editing. You'll find the next file here:

```
http://www-verimag.imag.fr/~moy/jeu-de-piste/abc/etape\_d1.adb
```

You'll have to fix a few errors to compile it.

```
" | txt2img step-C3.png
```

# Treasure Hunt: Pros and Cons

- Pros:
  - ▶ Students **love** it
  - ▶ Students can't skip or mistakenly think they solved one step
  - ▶ Teachers have fun preparing it
- Cons:
  - ▶ Students can be blocked on a step
  - ▶ May divert attention from booklet and wiki

# Treasure Hunt: Necessary Conditions to Make it Work

- Remind students about its existence
  - ▶ Frequent references in textbook
  - ▶ Informal surveys by teacher “who started treasure hunt?”, “Who went past step B?”, ...
- Be pro-active helping students
  - ▶ Look over student’s shoulders
  - ▶ Make whole-group demo to solve hardest steps

# Percentage of students reaching steps

Big brother is watching ...

- Some steps are monitored (student success recorded in a database)
- Results:

step	Percentage students reaching this step	Average grade at exam
A5	98%	16.2/20
E9	90%	16.6/20
F2	80%	16.8/20
G2	71%	17.0/20
H8	10%	18.8/20

(note: hardest step is E7)

# Outline

- 1 Unix Course in Ensimag
- 2 Treasure Hunt
- 3 (Fully Automated) Lab Exam
- 4 Conclusion

## Lab Exam: Why?

- Treasure Hunt = good for motivated students . . .  
... but most (?) students didn't complete it
- Long tradition of “teammate effect” among students  
⇒ Evolution towards individual exam in Ensimag



## Lab Exam: Goals (Student-side)

- Don't penalize beginners
  - ▶ Should be entirely doable without prior knowledge
  - ▶ Most questions easy once the treasure hunt is done and understood
  - ▶ All documents allowed
- Test both knowledge and speed
  - ▶ Many questions (28 wasn't enough for 1h)

## Lab Exam: Goals (Teacher-side)

- Easy to maintain:
  - ▶ Rudimentary technology (plain PHP, SQL, bash)
  - ▶ Few dependencies
  - ▶ Small codebase ( $\approx$  1000 LOC of PHP+bash)
- Prevents cheating:
  - ▶ Give password to a friend
  - ▶ Copy neighbor's answers
- Automatic grading:
  - ▶ Exam added without increasing teacher's workload
  - ▶ Unambiguous, unique answers required

Give the students no excuse to fail

# Lab Exam: Example

The screenshot shows a web browser window titled "Demo - Mozilla Firefox" with the address bar containing "http://www.verimag.imag.fr/~moy/demos-unix-training/demo\_ensimag2010-enf". The page content is as follows:

**(1 points)** The answer for this question is f7b6029c.  
 ✘

**(2 points)** The answer for this question is located in the file 5db35d41.txt in your working directory (this is a text file).  
 ✔ Correct answer validated

**(2 points)** What is the size, in bytes number without unit.

An overlaid terminal window titled "demo\_zsh" shows the following commands and output:

```
~/demo$ cat 5db35d41.txt
The answer is : 24eb3767
~/demo$
```

## Exam Generation Library

```
$ cat exam.sh
#! /bin/bash
[... around 500 LOC for a 1h exam ...]
EXAM_DIR=../unix-training.git/gen-exam
. "$EXAM_DIR"/exam-main.sh
$ ./exam.sh
Generating the exam in SQL mode
[...]
Number of questions: 28
Total coefficients: 100
Generated files in exam/exam_genere
- questions.sql
- 1/ and 2/ : files to put on students
                account for sessions 1 and 2
- php/ : PHP files to put on the server.
$ rsync -av php/ web-server:/var/www/exam/
$ mysql -h web-server < questions.sql
```

# Exam Generation Library: Example

You write...

```
all_questions () {
    smart_question latex 2
    [...]
}

desc_question_latex () {
    echo "The answer is in the
        file <tt>latex.tex</tt>."
}

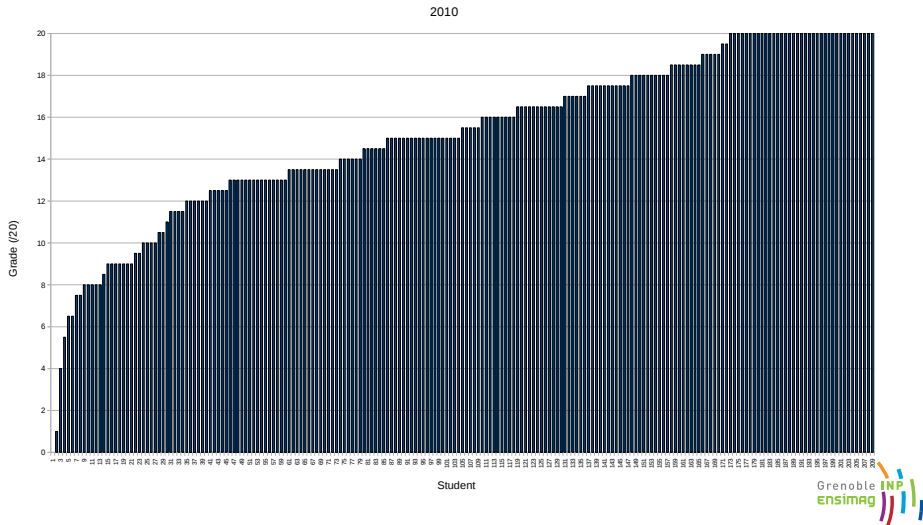
gen_question_latex () {
    (
        echo '\documentclass{article}'
        latexable
        echo '\begin{document}'
        echo "The answer is: $1" \
            | latexencode
        echo '\end{document}'
    ) > latex.tex
}
```

... The library takes care of

- Calling functions for each student
- Choosing different answer for each student
- Filling-in SQL database

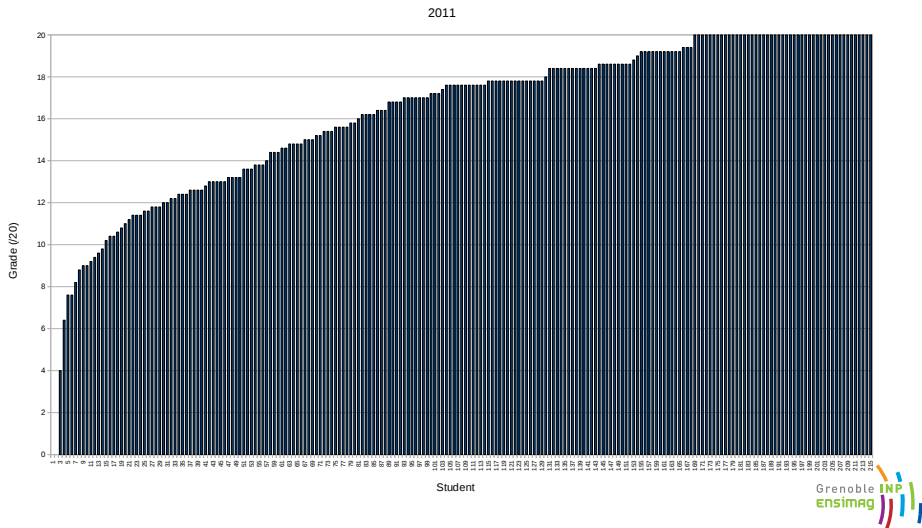
# Lab Exam: Results in 2010

1 hour, 28 questions / far better results than expected!



# Lab Exam: Results in 2011

1 hour, 5 more questions / far better results than expected!



# Lab Exam: Adaptation to Algorithmics Exam (Python)

The image shows a browser window with a quiz question and a terminal window overlaid on it.

**Quiz Question (3 points):** Combien vaut  $\sum_{i=1}^{11} i$  ? Entrez une valeur numérique décimale.

**Answer:** 66

**Quiz Question (1 point):** Voici maintenant un QCM. Cette fois-ci, vous ne pouvez pas valider une réponse si elle n'est pas correcte pendant l'examen (même si elle est la bonne). Vous pouvez valider votre réponse autant de fois que vous voulez. Soit le morceau de code suivant:

```
x = 42
def f(y):
    print(x)
f(12)
```

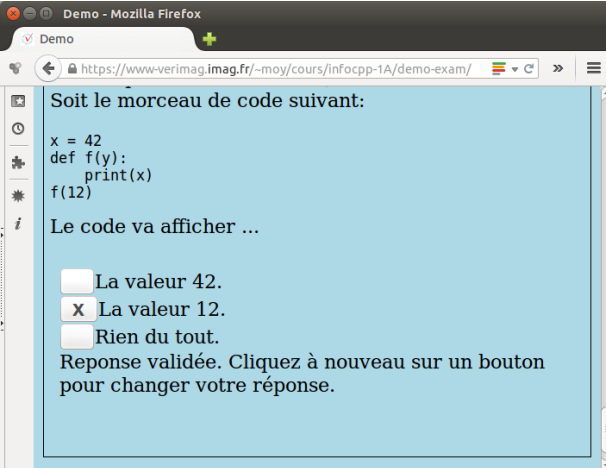
**Terminal Output:**

```
moy@tablets:~$ python
Python 3.4.0 (default, Apr 11 2014, 13:05:11)
[GCC 4.8.2] on linux
Type "help", "copyright", "credits" or "license()"
>>> s = 0
>>> for i in range(1, 12): s += i
...
>>> print(s)
66
>>>
```

<http://www-verimag.imag.fr/~moy/cours/infocpp-1A/demo-exam/>



# Lab Exam: Multiple-Choices Questions



The screenshot shows a Mozilla Firefox browser window with the address bar displaying `https://www.verimag.imag.fr/~moy/cours/infocpp-1A/demo-exam/`. The page content is as follows:

Soit le morceau de code suivant:

```
x = 42
def f(y):
    print(x)
f(12)
```

Le code va afficher ...

- La valeur 42.
- La valeur 12.
- Rien du tout.

Reponse validée. Cliquez à nouveau sur un bouton pour changer votre réponse.

(No feedback to student, of course!)

## Lab Exam: Pros and Cons

- Pros:
  - ▶ “Motivates” students to read textbook and finish treasure hunt
  - ▶ Students can't escape the “Unix” thing anymore
- Cons:
  - ▶ Learning Vs Cramming
  - ▶ Motivate Vs “Motivate”



Treasure Hunt + Exam

?

Balance between positive motivation and forced work

# Outline

- 1 Unix Course in Ensimag
- 2 Treasure Hunt
- 3 (Fully Automated) Lab Exam
- 4 Conclusion

## Exam and Treasure Hunt: Conclusion

- Treasure Hunt: very appreciated from students, but insufficient
- Exam: the necessary complement
- *I* had fun writing this . . . and hope to share the fun!

Download (open-source): <http://gitorious.org/unix-training>  
(Google: “unix-training Ensimag”)

(Ask me if you want the complete exam)



## Sources



<http://en.wikipedia.org/wiki/File:CGIYoda.jpg>  
(© 2002 Lucasfilm Ltd. All Rights Reserved—fair use)



<http://en.wikipedia.org/wiki/File:LukeSkywalkerROTJTV2Wallpaper.jpg>  
(© shot from the Star Wars franchise—fair use)



<http://www.flickr.com/photos/starwarsblog/793008715/>  
(CC BY 2.0)



<http://fr.wikipedia.org/wiki/Fichier:Prisonlabor1.jpg>  
(public domain)