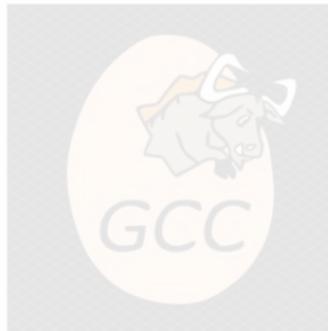




# PRELUDE



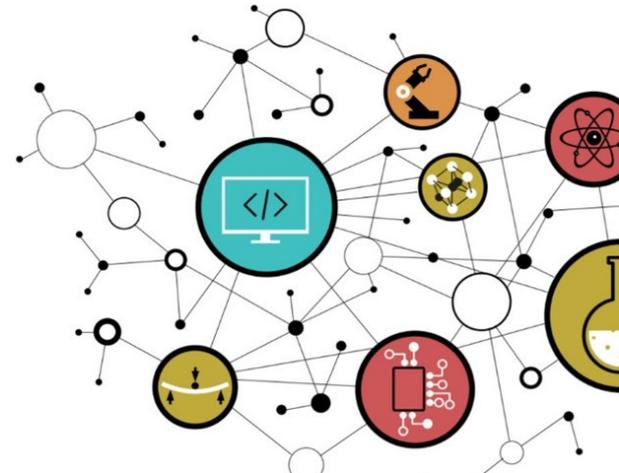
```
kernel text=0x30000 data=0x30470 bss=0x22000 sym=0x0+0x47000+0x4+0x500
Copyright (c) 1992-2000 The FreeBSD Project.
Copyright (c) 1979, 1989, 1993, 1996, 1999, 2003, 2004, 2006, 2007, 2009, 2012, 2013, 2014
The Regents of the University of California. All rights reserved.
FreeBSD is a registered trademark of the FreeBSD Foundation.
FreeBSD 0.4 RELEASE-02: Mon Jan 5 22:14:32 CST 2011
root@babel:~# pwd; cat /dev/urandom | tr -dc 'a-z0-9' | fold -w 64 | xargs sha1sum
timecounter "125254" frequency 1155152 Hz quality 0
CPU: Genesys[IM] Integrated Processor by National Semi (266.00-MHz 506-class CPU)
0x10000 = "Genesys by NSE" id = 0x0000, classing = 0
Features=0x000111470: TS, MMU, L2, CPUID, MMX
real memory = 208433456 (204 MB)
avail memory = 236531712 (225 MB)
class: no ACPI policy registered
ats: bus: 0.0.0.0.3 (AR0210, AR0211, AR0212, RP0111, RP0112, RP0211, RP0212)
CPM on Motherboard
class: class is PCI bridge pciBus 0 on Motherboard
class: -PCI bus on pci0
class: -Nvidia NV031150 18/10800000 port 0x200 BusID=0x0100 0x00000000 0x00000111 10
class: Silicon Revision: 0P030104
class: -PCI bus on pci4
class: -AMD64 00:00:00 00:00:00 media interface on xlibus4
class: -AMD64 00:00:00 00:00:00 10800000, 10800000, 10800000, auto
class: Ethernet address: 98:00:20:00:07:00
class: -Nvidia NV031150 18/10800000 port 0x200 BusID=0x0100 0x00000111 10
class: Silicon Revision: 0P030104
class: -PCI bus on pci1
```

Électronicien spécialisé en Systèmes Embarqués, je suis à votre écoute et ferais de mon mieux pour vous accompagner et vous aiguiller dans votre projet professionnel !

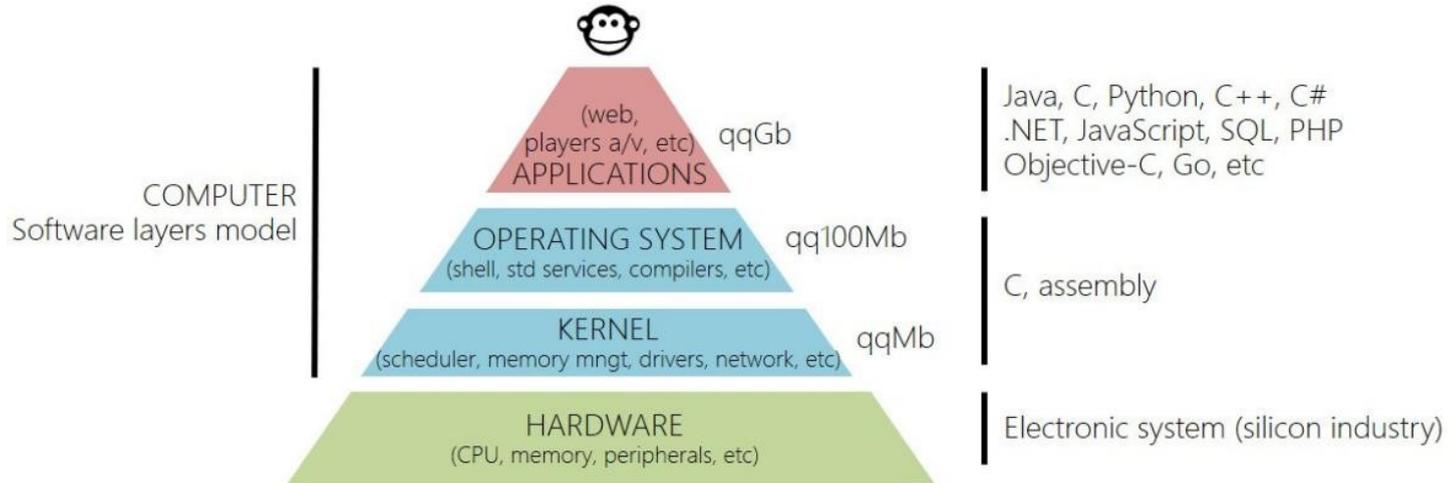
- Bureau en A202 (accès digicode par la salle A203 - *carré + A203 + triangle*) - 02 31 45 27 61
- [hugo.descoubes@ensicaen.fr](mailto:hugo.descoubes@ensicaen.fr)



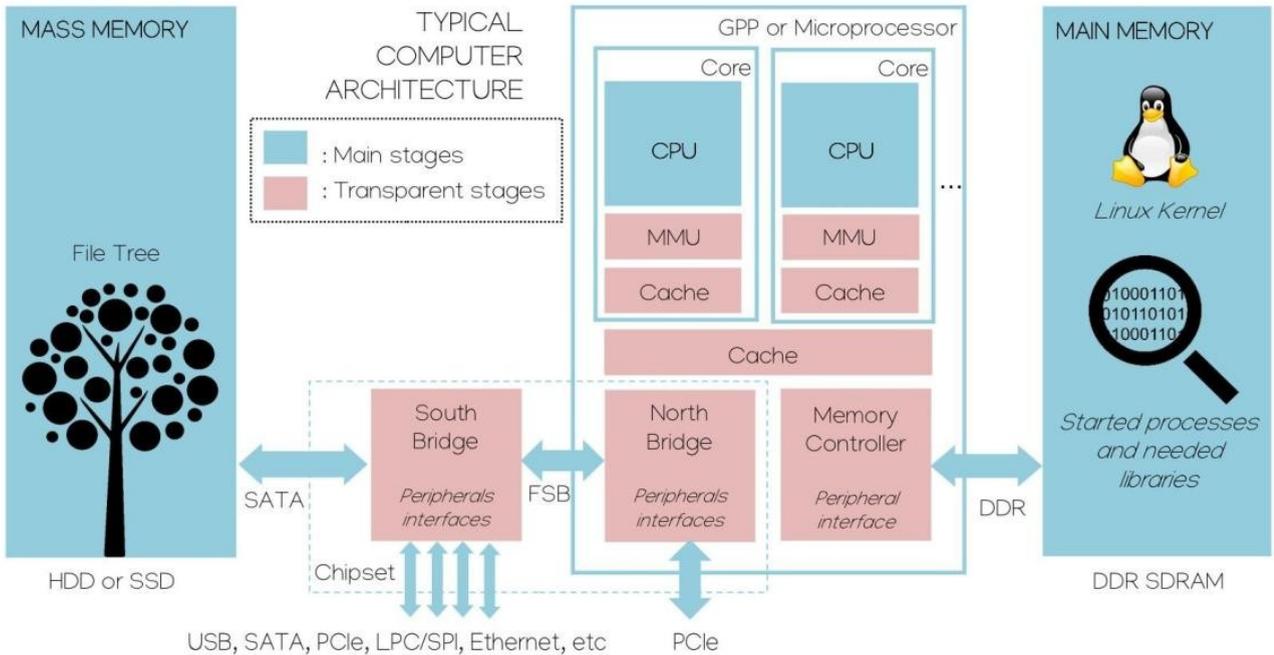
- Objectifs
- Ressources pédagogiques
- Évaluations des compétences
- Pré-requis



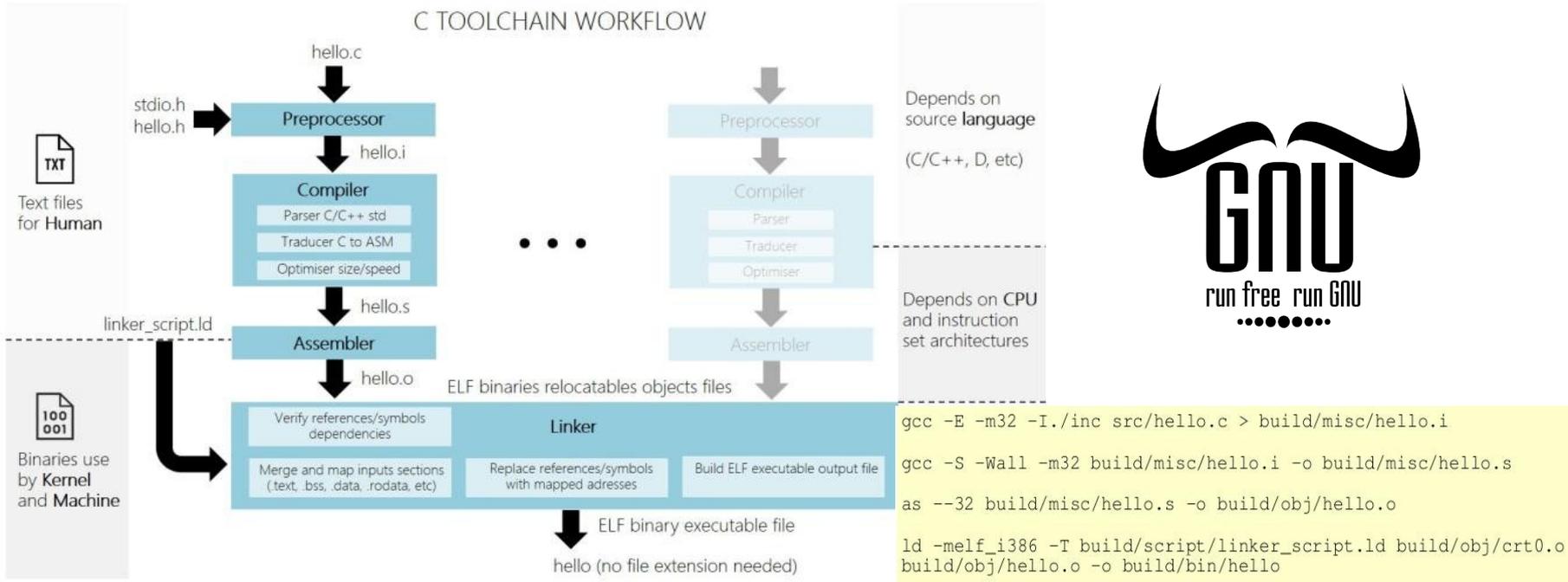
Une bonne maîtrise des langages de programmation système (C et ASM), du processus de compilation et d'édition des liens, du travail de cloisonnement du noyau système au chargement d'un programme en mémoire principale (segmentation et pagination), ainsi que la compréhension des rôles des principaux composants matériels et du fonctionnement global de l'ordinateur sont des bases fondamentales pour un développeur logiciel, même haut niveau (JAVA, C++, etc).



- Connaître les principaux composants matériels d'un ordinateur. Comprendre l'architecture et le fonctionnement global d'un système numérique d'information géré par un système d'exploitation évolué exploitant une MMU et un espace mémoire virtualisé (Linux, NT, etc)



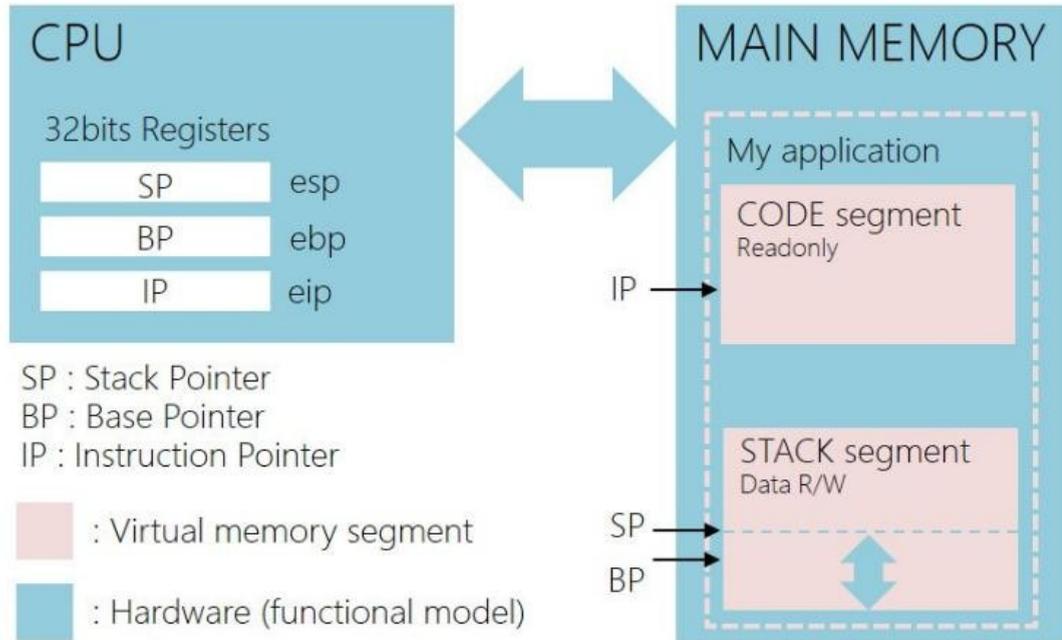
- Comprendre le processus de compilation et d'édition des liens. Connaître les principaux outils d'analyse et de compilation du projet GNU (gcc, as, ld, objdump, readelf, strip, ar, etc)



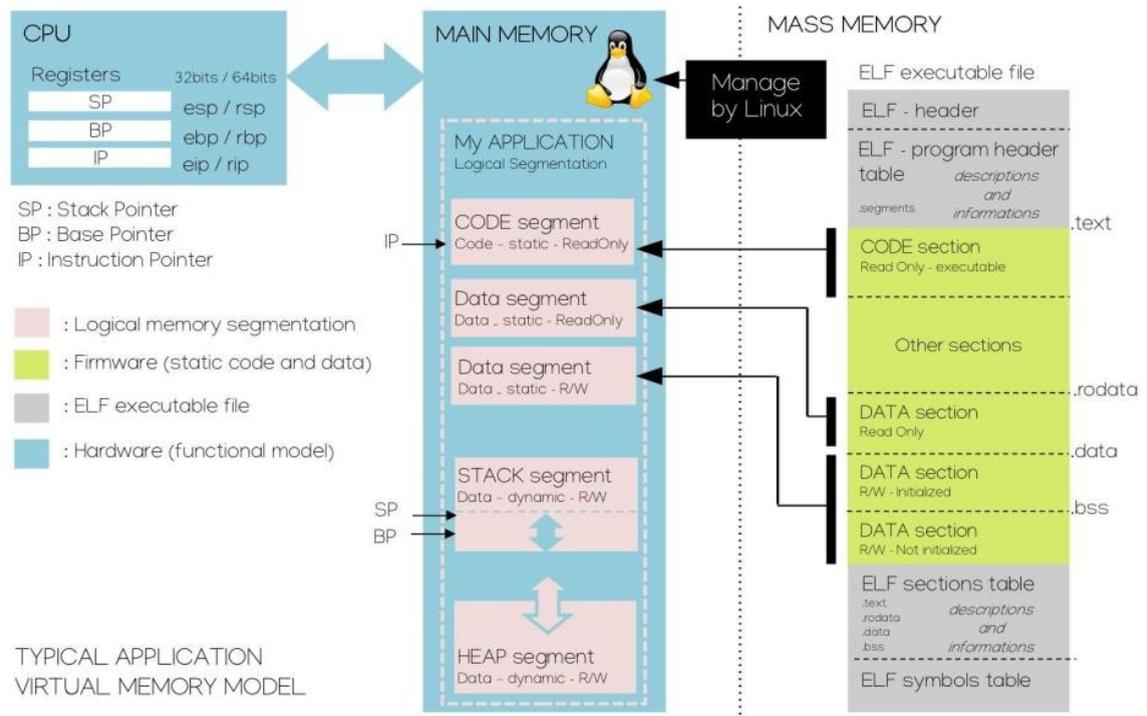
- Analyser un programme ASM x86\_64. Comprendre le mécanisme de gestion des variables locales sur la pile réalisé par les outils de compilation, par le système ainsi que la machine

```
int main (void)
{
return 0 ;
}
```

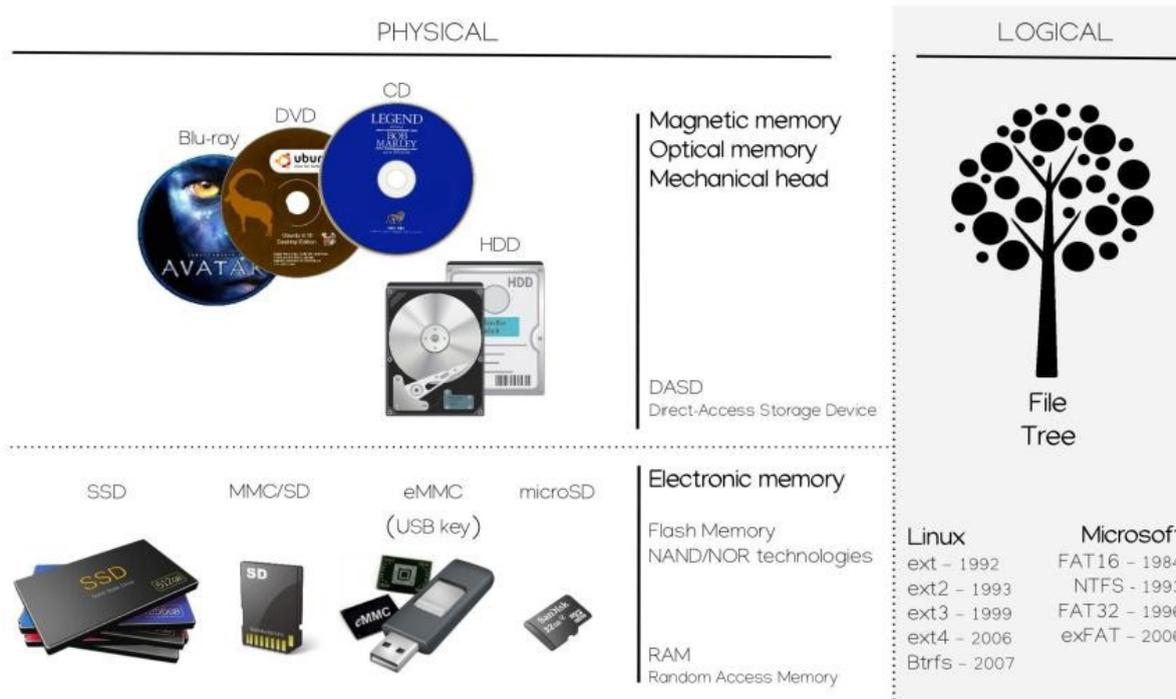
```
main:
    pushl   %ebp
    movl   %esp,%ebp
    movl   $0,%eax
    popl   %ebp
    ret
```



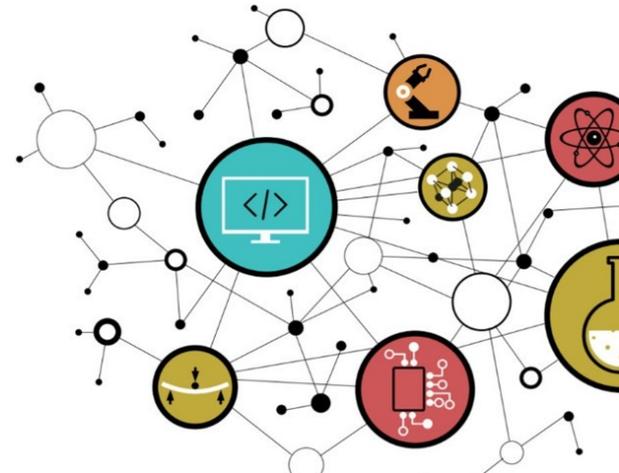
- Comprendre le processus et l'environnement d'exécution d'une application logicielle. Connaître les limitations mémoire imposées par le système et l'ordinateur (segmentation et pagination)



- Comprendre l'architecture logique des supports de stockage de masse (table des partitions, systèmes de fichiers, etc). Être apte à préparer un média de masse pour un besoin spécifique.



- Objectifs
- **Ressources pédagogiques**
- Évaluations des compétences
- Pré-requis



- Archive complète de travail Cours/TP sur la plateforme moodle ENSICAEN : **arch.zip**



<https://foad.ensicaen.fr/course/view.php?id=99>

- Polycopiés séparés de cours et de TP



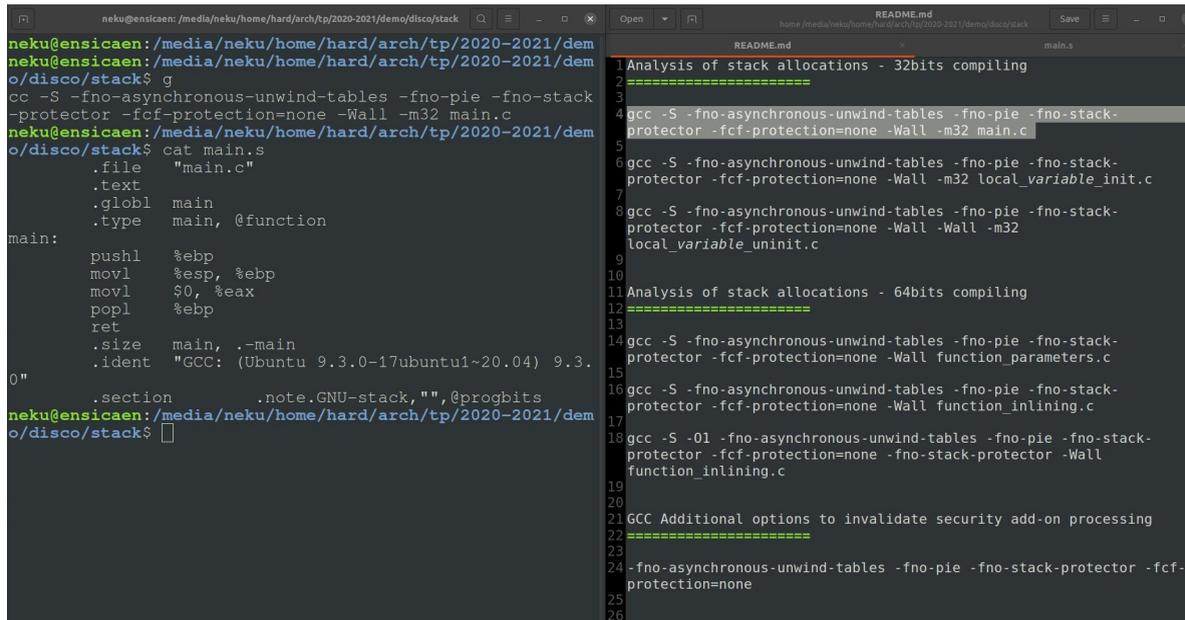
- Les TP peuvent être réalisés dans les salles informatique du bâtiment E ou en salles A203 et A201 (digicode carré + A203 ou A201 + triangle)
- Nous vous conseillons néanmoins d'utiliser vos machines personnelles avec un système GNU/Linux 64bits. Afin d'utiliser les même configurations système que l'école, nous vous conseillons d'installer un système **Ubuntu 20.04 LTS**. Vous trouverez ci-dessous un tutoriel ENSICAEN pour vous aiguiller dans vos installations

<https://ubuntu.com/download/desktop?version=20.04&architecture=amd64>

<https://foad.ensicaen.fr/mod/page/view.php?id=22125>

<https://foad.ensicaen.fr/mod/page/view.php?id=25095>

- L'environnement de travail se vaudra volontairement minimaliste. Un système GNU/Linux 64bits, quelques programmes sources C et ASM x86\_64 à analyser, le shell, une toolchain GNU GCC, etc et nous pourrons explorer les entrailles de notre ordinateur ...



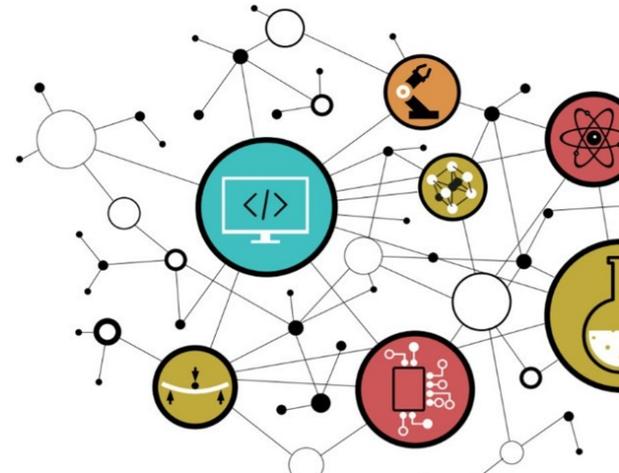
```

neku@ensicaen: /media/neku/home/hard/arch/tp/2020-2021/demo/disco/stack
neku@ensicaen: /media/neku/home/hard/arch/tp/2020-2021/demo/disco/stack$ g
cc -S -fno-asynchronous-unwind-tables -fno-pie -fno-stack-protector -fcf-protection=none -Wall -m32 main.c
neku@ensicaen: /media/neku/home/hard/arch/tp/2020-2021/demo/disco/stack$ cat main.s
.file "main.c"
.text
.globl main
.type main, @function
main:
    pushl   %ebp
    movl   %esp, %ebp
    movl   $0, %eax
    popl   %ebp
    ret
.size   main, .-main
.ident  "GCC: (Ubuntu 9.3.0-17ubuntu1~20.04) 9.3.0"
.section .note.GNU-stack,"",@progbits
neku@ensicaen: /media/neku/home/hard/arch/tp/2020-2021/demo/disco/stack$
  
```

```

1 Analysis of stack allocations - 32bits compiling
2 =====
3
4 gcc -S -fno-asynchronous-unwind-tables -fno-pie -fno-stack-protector -fcf-protection=none -Wall -m32 main.c
5
6 gcc -S -fno-asynchronous-unwind-tables -fno-pie -fno-stack-protector -fcf-protection=none -Wall -m32 local_variable_init.c
7
8 gcc -S -fno-asynchronous-unwind-tables -fno-pie -fno-stack-protector -fcf-protection=none -Wall -Wall -m32 local_variable_uninit.c
9
10
11 Analysis of stack allocations - 64bits compiling
12 =====
13
14 gcc -S -fno-asynchronous-unwind-tables -fno-pie -fno-stack-protector -fcf-protection=none -Wall function_parameters.c
15
16 gcc -S -fno-asynchronous-unwind-tables -fno-pie -fno-stack-protector -fcf-protection=none -Wall function_inlining.c
17
18 gcc -S -O1 -fno-asynchronous-unwind-tables -fno-pie -fno-stack-protector -fcf-protection=none -fno-stack-protector -Wall function_inlining.c
19
20
21 GCC Additional options to invalidate security add-on processing
22 =====
23
24 -fno-asynchronous-unwind-tables -fno-pie -fno-stack-protector -fcf-protection=none
25
26
  
```

- Objectifs
- Ressources pédagogiques
- **Évaluations des compétences**
- Pré-requis



- Partiel 30mn QCM sur moodle :

- Compilation et édition des liens
- Assembleur 32bits x86 et 64bits x86\_64



- Examen 1h30 à l'écrit sur table :

- **7pts** : 3-4 questions ouvertes d'ordre général. Illustrer à l'aide d'un schéma exhaustif avec définitions et fonctionnement du système à présenter. Peut-être vu comme un échange d'ingénieur à ingénieur.
- **13pts** : Exercice de traduction et de retro-ingénierie ASM x86\_64 vers langage C. Analyse d'un programme ASM x86\_64 et du fichier objet associé au format ELF. Analyse du comportement et trace du contenu de la pile

Merci !