ENSIIE 3A – S5 SEC
INTRODUCTION AU REVERSE ENGINEERING

Support de cours

Evry, 17 Novembre 2023

ensiie
école nationale supérieure d'informatique pour l'industrie et l'entreprise

REVERSE ENGINEERING

- Introduction
- Format ELF
- Techniques et outils
- Exercice incrémental

Introduction

Introduction

Bases

- Rétro-conception : comprendre comment fonctionne un produit/programme sans disposer des étapes de conception (plans, code source…)
- Utilisation de techniques et d'outil permettant d'analyser un système jusqu'à être capable de le dupliquer, voire de l'améliorer.
- Pas réservé aux programmes informatiques :
  - ingénierie mécanique
  - recettes industrielles (CocaCola, Nutella…)
- Néanmoins, couvre un périmètre très large dans le domaine de l'informatique :
  - Exécutables écrit en C
  - Applications complexes en langages object (Java, C++)
  - Code embarqué dans un microcontroleur

Introduction

Utilité du *reverse engineering*

- Il peut être utilisé pour comprendre en détail le fonctionnement d'un logiciel,
- Améliorer la sécurité et qualité d'un logiciel (recherche de failles, étude d'un virus pour l'éradiquer, ...),
- Permettre de contourner les protections logiciels (numéro de licence, etc...),
- Reproduire le comportement d'un logiciel en outrepassant les DRM et restrictions des logiciels propriétaires.
- Permettre l'interfacage avec un logiciel ancien (retro-compatibilité)

Introduction

Approche statique et approche dynamique

- Approche statique :
  - “lecture de code”
  - Pas d’execution (fichier en lecture seule)
- Approche dynamique :
  - Exécution du programme
  - Utilisation d’un debugger
  - Manipulations de registres et du flot d’execution
  - Pas toujours utilisable
- Approche intermédiaire : execution symbolique

- Approches complémentaires et à utiliser selon les circonstances

Cadre

- Programme écrit en C
- Compilation pour une architecture Intel x86
- Système d'exploitation Linux : executable ELF

Format ELF

Format ELF

Compilation

- Tranformation d'un fichier de code source en executable binaire
- Différents compilateurs produiront des binaires (légèrement) différents
- Note : un langage informatique "n'existe" que lorsqu'un compilateur pour ce langage existe
- GCC : CNU C Compiler, compilateur *open source* pour le C
- Très nombreuses options de compilation

- Autres compilateurs : Clang (utilisé pour le noyau Android depuis peu)

Format ELF

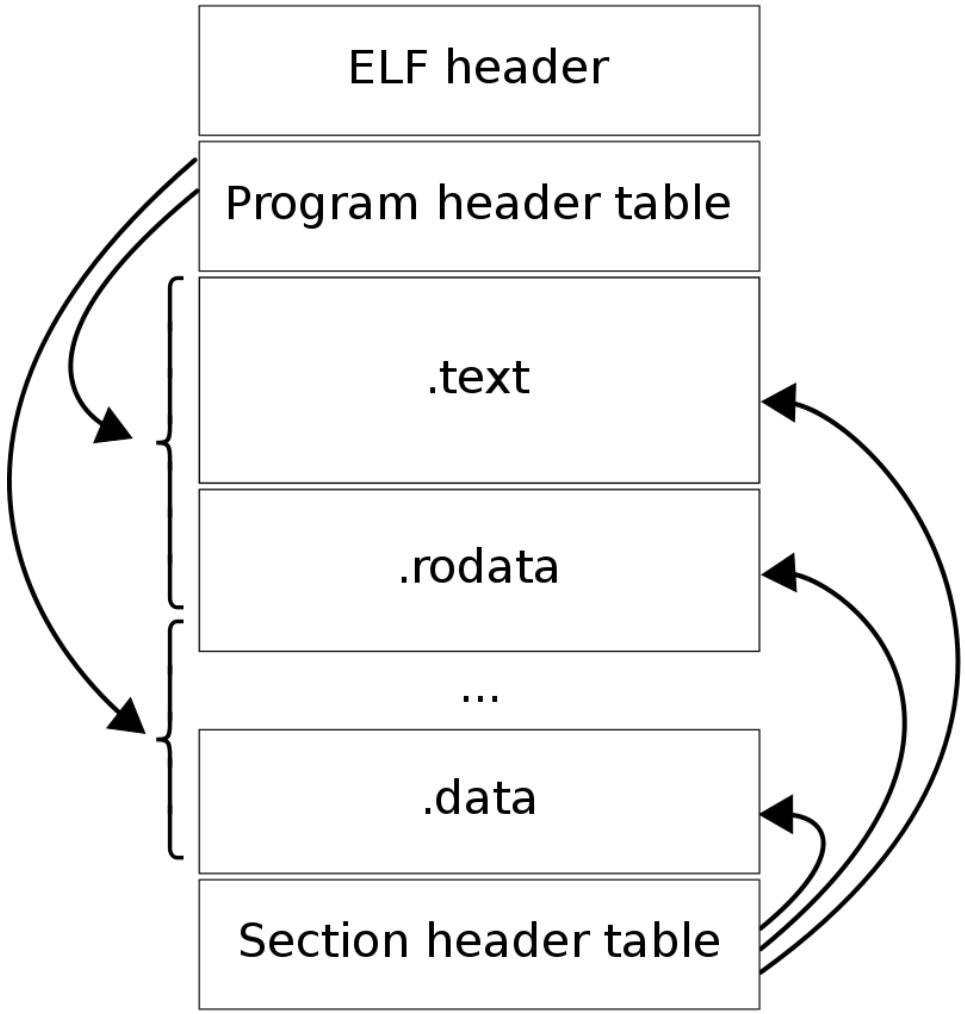
Exercice

- Implémentation d'une verification de mot de passe
- Programme en C
- Compilation avec GCC en 32 bits

```
$ ./a.out testpwd123
Wrong password
```

Format ELF

Sections

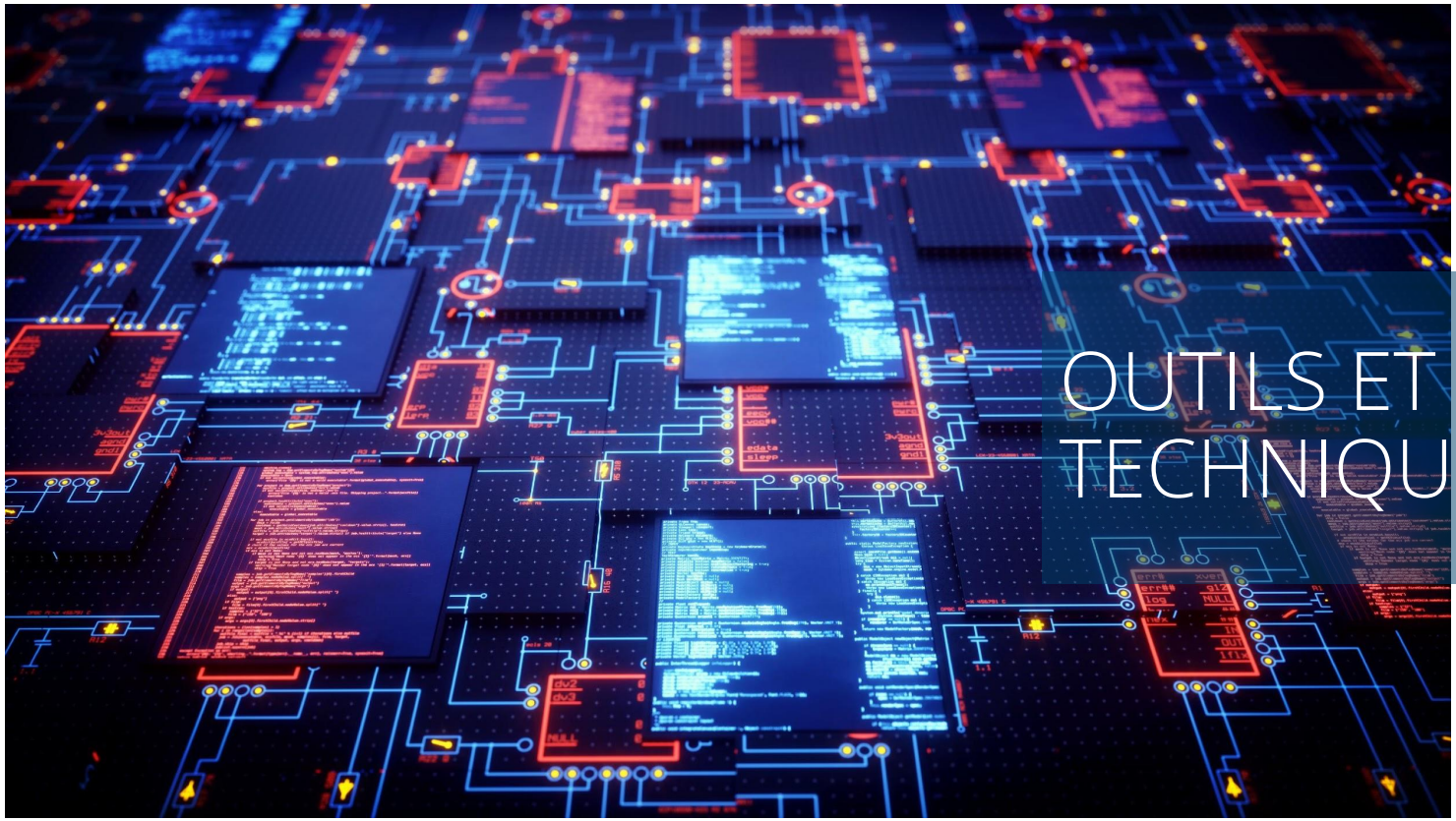


Format ELF

Code assembleur

- Spécifique à chaque architecture matérielle
- Plus courant : x86 (32 bits) et x64 (ou x86-64, 64 bits)
- Smartphones, circuits embarqué : ARM et ARM64
- Autres : MISP

- Jeu d'instructions relativement limité
- Traduction directe du code machine pour chaque instruction
- Registres + Pile
- Différentes syntaxes de représentation



OUTILS ET TECHNIQUES

Outils et techniques

Readelf, strings, objdump

- Analyse statique
- Lecture de différentes sections du fichier binaire
- Strings : extraction des chaînes de caractères imprimables
- Objdump et readelf : lecture de différentes sections du binaire

Outils et techniques

ltrace et strace

- Approche dynamique
- Tracer les appels vers des fonctions exérieures au programme
- ltrace : “library trace” □ function dans des bibliothèques
- strace : “system trace” □ appels système (syscalls)

Outils et techniques

gdb

- GNU Debugger
- Très utile au développemement / débbuging, également en reverse
- Progression pas à pas dans le programme
- Etude de l'état des registres et de la mémoire
- Disposition de "*breakpoints*" spécifiques

- Gdb est open-source mais d'autre debuggers existent

Outils et techniques

IDA et Ghidra

- Analyse statique + dynamique (debugger intégré)
- Ghidra : gratuit et open source
- IDA : payant mais avec une version gratuite

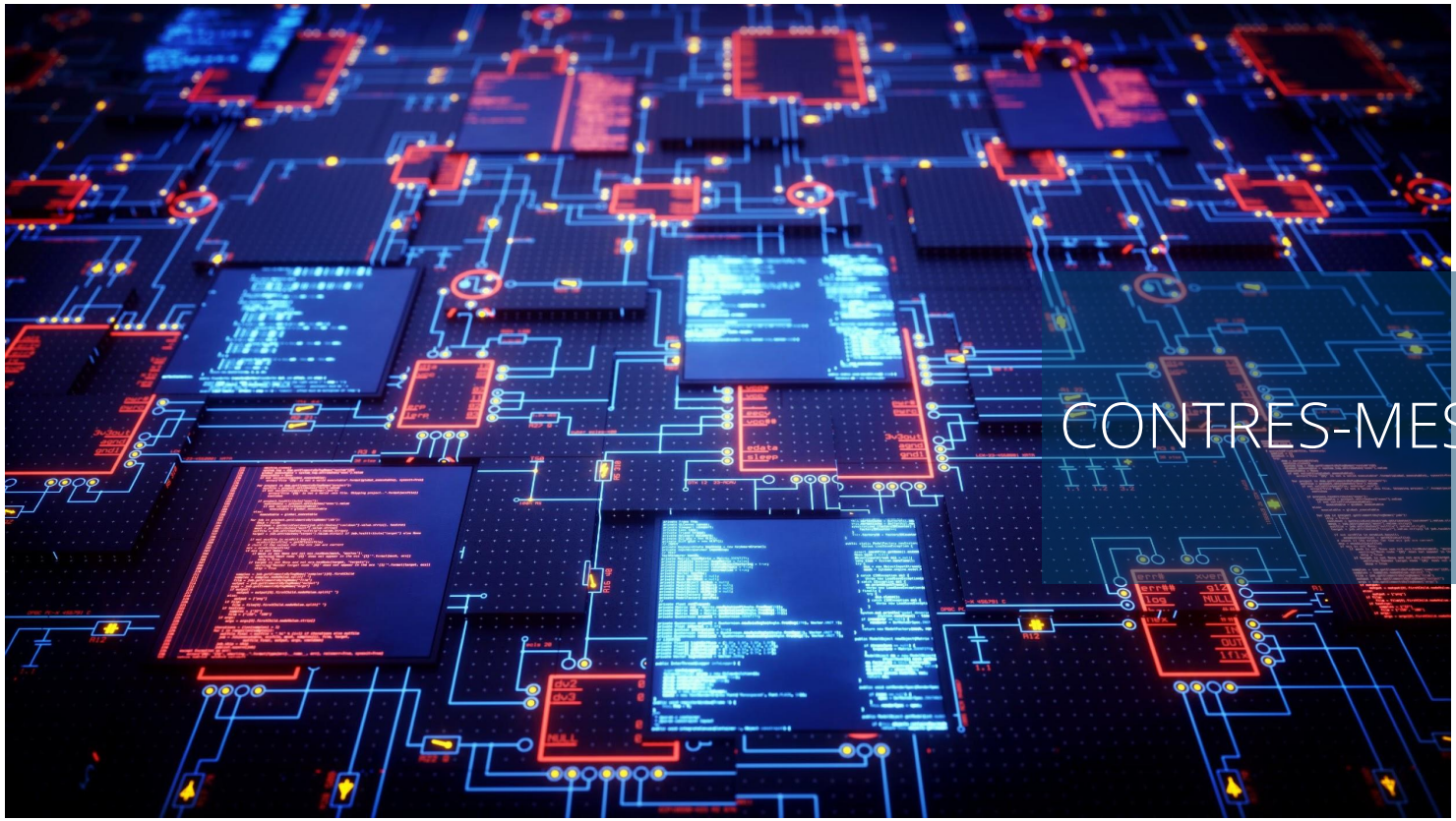
- Graphe de flot de contrôle
- Crucial pour comprendre le fonctionnement d'un programme plus complexe

- Desassembleur : retrouver un code C à partir du code assembleur
- Résultat pas toujours très lisible mais peut aider à la compréhension

Outils et techniques

Bruteforce

- Identification de conditions non suffisantes
- Tests sur toutes les possibilités à partir de ces conditions
- Scripts en python lançant l'execution du programme en C



CONTRES-MESURES

Contres-mesures

Obfuscation

- Flot de contrôle arbitrairement complexe
- Obfuscation à la compilation
- Packers

Contres-mesures

Gêner l’analyse dynamique

- Machines Virtuelles
- ptrace()
- Anti-debug

Outils et techniques

Expérience

- Reverse Engineering est un domaine très vaste
- Ne s'apprend pas en théorie pour être pratiqué ensuite