

# Reverse Engineering @ unfug

Timo “bluec0re“ Schmid && Sergej “winnie“ Schmidt

26. Mai 2011

# Inhaltsverzeichnis

- 1 Reverse Engineering
  - What and Why
  - Law
- 2 Assembly
  - Registers
  - Memory Layout
  - Calling Conventions
  - Name Mangling
  - ELF-Header
- 3 Other Languages and Tools

# Reverse Engineering

## what is that?

Reverse engineering is the process of discovering the technological principles of a human made device, object or system through analysis of its structure, function and operation. From wikipedia

## Why

The bigger picture:

Most programmers learn the language from the top down and never see the big picture. Hackers get their edge from knowing how all the pieces interact within this bigger picture. To see the bigger picture in the realm of programming, simply realize that C code is meant to be compiled. From Hacking - The Art of Exploitation.



# Reverse Engineering

Law

§ 69e Dekompilierung @ Urhg



# Assembly

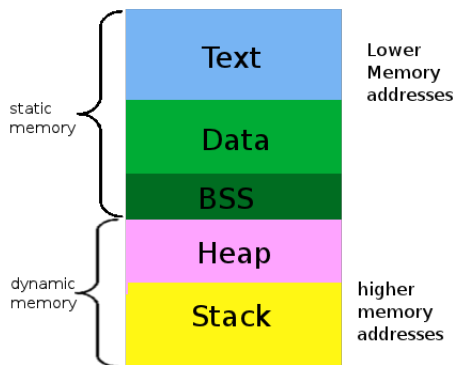
# Registers

- ESP
- EBP
- EIP
- eflags

## data registers

- EAX
- ECX
- EDX
- EBX

# Memory Layout



# Calling Conventions

## cdecl

- stack based call
- caller cleans stack
- linux standard

## stdcall

- stack based call
- function cleans stack
- windows standard

# Calling Conventions

## fastcall

- register based call
- out of regs? → use stack
- linux standard for syscalls and for amd64



# Name Mangling

- @c++
- overloading



## ELF-Header

0 - 7	8-15	16-23	24-31
ident			
type		machine	
version			
entry			
phoff			
shoff			
flags			
ehsize		phentsize	
phnum		shentsize	
shnum		shtrndx	

## Other Languages and Tools

### java

- jad
- jd-gui / jd-eclipse

### .NET

- .reflector

## Other Languages and Tools

### debuggers

- edb
- ollyDbg
- Immunity Debugger