# x86_32 Shellcode-Lab Cheat Sheet

by therealdash via cheatography.com/196544/cs/41321/

## GDB - Gnu Debugger - Initiation

| | |
|---|---|
| gdb -q ./<file> | Start GDB in quiet mode |
| gdb -p <pid> | Attach to process-id |
| gdb -c <core> ./<file> | Load up a core file and the program |

Those commands are executed to start GDB.

## GDB - Commands - Run a program

| | | |
|---|---|---|
| run | r | Start the program |
| run testarg | r testarg | Start with an argument |

## GDB - Commands - Registers

| | | |
|---|---|---|
| info registers | i r | Show default registers |
| info registers all | i r a | Show all registers |
| info registers eax | i r eax | Show EAX register |

Commands for showing the content of registers.

## GDB - Commands - Examine

| | |
|---|---|
| x $eax | Examine address in EAX |
| x/i $esp | Examine address at ESP interpret as instruction |
| x/s 0xffffffab | Examine address interpret as string |
| x/4s 0xffffffab | Print from that address 4 times |
| x/4xb | Examine in HEX repeat 4 times show in Bytes |
| disass-emble / disas | Disassemble at current position |
| disas _start | Disassemble from label _start |

## GDB - Commands - Examine (cont)

| | |
|---|---|
| print / p system | Print address of libc system |

Note: Examine needs valid addresses to function. Unit sizes: b, Bytes; h, Halfwords (two bytes);w, Words (four bytes); g, Giant words (eight bytes).

## GDB - Commands - Breakpoint

| | | |
|---|---|---|
| break _start | b _start | Set a breakpoint at the label _start |
| break 5 | b 5 | Breakpoint at source line 5 |
| break *0x443-32211 | b *0x443-32211 | Breakpoint at address/offset |

## GDB - Commands - Stepping

| | | |
|---|---|---|
| step | s | Step per line of source. |
| stepi | si | Step per machine instru-ction |
| continue | c | Continue program execution |

## GDB - Commands - Set and Call

| | |
|---|---|
| call (int) mprotect(-0xDEADBEEF, 0x1000, 1) | Execute mprotect() in debugee context. |
| call strcpy(0xdea-dbeef, "hacky") | Write hacky to addr 0xdeadbeef |
| set follow-fork-mode child | Follow newly created childs |
| set *(char [SIZE]* ) 0xdeadbeef = "my_-new_array" | Write data to address |
| set {int}0xdeadbeef = 4 | Set value at address to 4 |
| set $eax = 0xdeadbeef | Set value of register EAX to 0xdeadbeef |

## GDB-GEF - Overview

| | | |
|---|---|---|
| gdb-gef | | Start gdb-gef at commandline |
| gef help | | Show help of GEF |
| start | | Start program with auto breakpoints set |
| kill | | Kill current process |
| context | ctx | Show context |
| checksec | | Check security features |
| vmmap | | Show virtual memory map |
| python-in-teractive | pi | Start Python Interp-reter |
| python-in-teractive 23*5 | pi 23*5 | Use python interp-reter and calculate |

## GDB-GEF - Configuration

| | |
|---|---|
| gef config | Show running configuration |
| gef config context | Configure GEF context |
| gef config context.show-_opcode_size 8 | Set the opcode output to length of 8 |
| gef config context.layout "legend regs stack memory" | Set only for widgets as output |
| gef save | Save running configuration |

Extra configurations for GDB-GEF

## GCC - Overview

| | |
|---|---|
| gcc -m32 <input> -o <output> | Compile source for x86_32 arch. |
| gcc -m32 <input> -o <output> -z execstack | Compile with executable stack |
| gcc -m32 <input> -o <output> -g | Compile with debug symbols |




Sponsored by **CrosswordCheats.com**
Learn to solve cryptic crosswords!
http://crosswordcheats.com

## NASM - Overview

| | |
|---|---|
| nasm -f elf32 <input> -o <output>.o | Creates x86_32 object file from assembly. |
| ld -m elf_i386 <input>.o -o <output> | Create x86_32 ELF from object file |

## OBJDUMP - Overview

| | |
|---|---|
| objdump -d -M intel <file> | Dump the opcodes in Intel Syntax |
| objdump -s -j <section> <file> | Dump only named section |

## STRACE - Overview

| | |
|---|---|
| strace <filename> | Starts program and tracing it |
| strace -p <pid> | Attaches at process-id |
| strace -o log.txt <filename> | Writes output into a logfile |
| strace -f <filename> | Also log child processes |

## PWNtools

| | |
|---|---|
| pwn asm nop | Write NOP opcode |
| pwn asm nop 'mov eax, 1' | Write NOP and MOV opcode |
| pwn asm -f string nop | Outputs in \x Notation |
| pwn disasm 909090 | Output the disassembly of three NOPs |

## PERL - Basics for exploits

| | |
|---|---|
| perl -e '{print "A"x"1024"}' | Print 1024 times A |

## Student Files

| | |
|---|---|
| lessons/ | Assembler files, aimed at teaching x86_32 basics |
| shellcode/ | Collection of bad shellcodes, students have to improve |
| skeletons/ | Skeleton Code files |

## Student Files (cont)

| | |
|---|---|
| exploits/ | Exploits shellcode is ran against |
| tools/ | Support tools for the training |



Sponsored by **CrosswordCheats.com**
Learn to solve cryptic crosswords!
http://crosswordcheats.com