binutils Binary inspection tools

Source cc -g -c → compiles to → **Object .o** nm, objdump → ar / ld → **Archive .a / ELF** readelf, size → loaded as → **Process** ldd, addr2line

CONCEPTS

Most tools read a file and never run it, so they are safe on untrusted binaries. ldd is the exception: it may execute the file.

Add -C to demangle C++ names; readelf, objdump and nm all accept it (readelf -C only for symbols).

Stripped binaries have no .symtab: use nm -D or readelf --dyn-syms for the dynamic symbols that remain.

Source lines and local variable names need a build with -g; without it addr2line only knows functions, if that.

SYMBOLS (NM)

`nm -C <file>`
List symbols, demangled

`nm -D <file>`
Dynamic symbols of a shared lib

`nm -u <file>`
Undefined symbols it needs

SYMBOLS (NM) (cont.)

`nm --defined-only <file>`
Only symbols it provides

`nm -S --size-sort -C <file>`
Biggest symbols last, with size

`nm -n <file>`
Sort by address

`nm -A <file>...`
Prefix each line with file name

`nm -g <file>`
External (global) symbols only

nm type letters

Letter	Meaning
`T` / `t`	code, global / local
`D` / `d`	initialised data
`B` / `b`	zero-filled bss
`R` / `r`	read-only data
`U`	undefined, resolved later
`W` / `w`	weak symbol

ELF STRUCTURE (READELF)

`readelf -h <elf>`
ELF header: type, arch, entry

`readelf -S <elf>`
Section headers

ELF STRUCTURE (READELF) (cont.)

`readelf -l <elf>`
Program headers (segments)

`readelf -d <elf>`
Dynamic section: NEEDED, RPATH

`readelf -s -W <elf>`
Symbol tables, wide lines

`readelf --dyn-syms <elf>`
Dynamic symbols only

`readelf -n <elf>`
Notes such as the build ID

`readelf -p .comment <elf>`
Dump a section as strings

DISASSEMBLY (OBJDUMP)

`objdump -d <file>`
Disassemble code sections

`objdump -d -C <file>`
Disassemble, demangled names

`objdump -d -M intel <file>`
Intel syntax instead of AT&T

`objdump -D <file>`
Disassemble all sections, even data

DISASSEMBLY (OBJDUMP) (cont.)

`objdump -S <file>`
Interleave source (needs -g)

`objdump -d --no-show-raw-insn <file>`
Hide the opcode bytes

`objdump -d --disassemble=<sym> <file>`
Only one function

`objdump -d --start-address=<a> --stop-address=<b> <file>`
Only an address range

`objdump -h <file>`
Section headers with sizes

`objdump -t <file>`
Symbol table

`objdump -T <file>`
Dynamic symbol table

STRINGS AND SIZES

`strings <file>`
Printable strings, 4+ chars

`strings -n <len> <file>`
Set minimum string length

`size <file>`
Text, data, bss totals

`c++filt <name>`
Demangle a C++ name

ADDRESSES AND LIBRARIES

`addr2line -e <elf> <addr>`
Address to file:line

`addr2line -e <elf> -f -C -i <addr>`
Add function, demangle, inlines

`addr2line -e <lib.so> -a <off>`
Print address, for shared lib offsets

`ldd <bin>`
Shared libs it needs

`readelf -d <bin> | grep NEEDED`
Direct dependencies, no execution

`file <bin>`
Type, arch, stripped or not

STRIP, COPY, ARCHIVES

`strip <file>`
Remove all symbols and debug info

`ar t <lib.a>`
List archive members

`ar rcs <lib.a> <obj.o>...`
Create or update static lib