Arhitektura i zbirni jezik procesora x86

Sveučilište u Zagrebu, Fakultet elektrotehnike i računarstva
Zavod za automatiku i računalno inženjerstvo
ReversingLabs

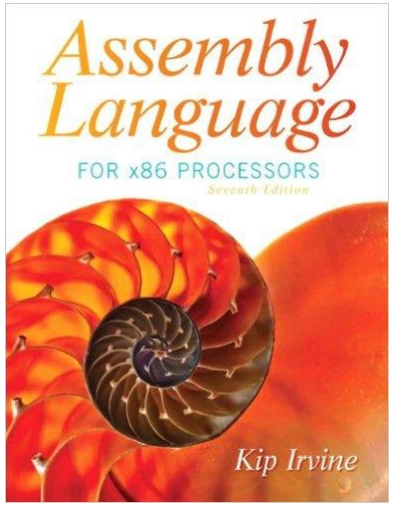
2022.

Sadržaj

1. Arhitektura x86 procesora
2. Programiranje u x86 zbirnom jeziku
3. Osnove debugginga
4. Instrukcijski skup
5. Sistemski pozivi i API
6. Zaobilaženje jednostavne programske zaštite

Literatura

- The Art of Assembly Language, Randall Hyde
  - http://www.plantation-productions.com/Webster/www.artofasm.com/index.html
  - pogledati stariju verziju, javno dostupan PDF
- Assembly Language for x86 Processors, 7th edition, Kip Irvine



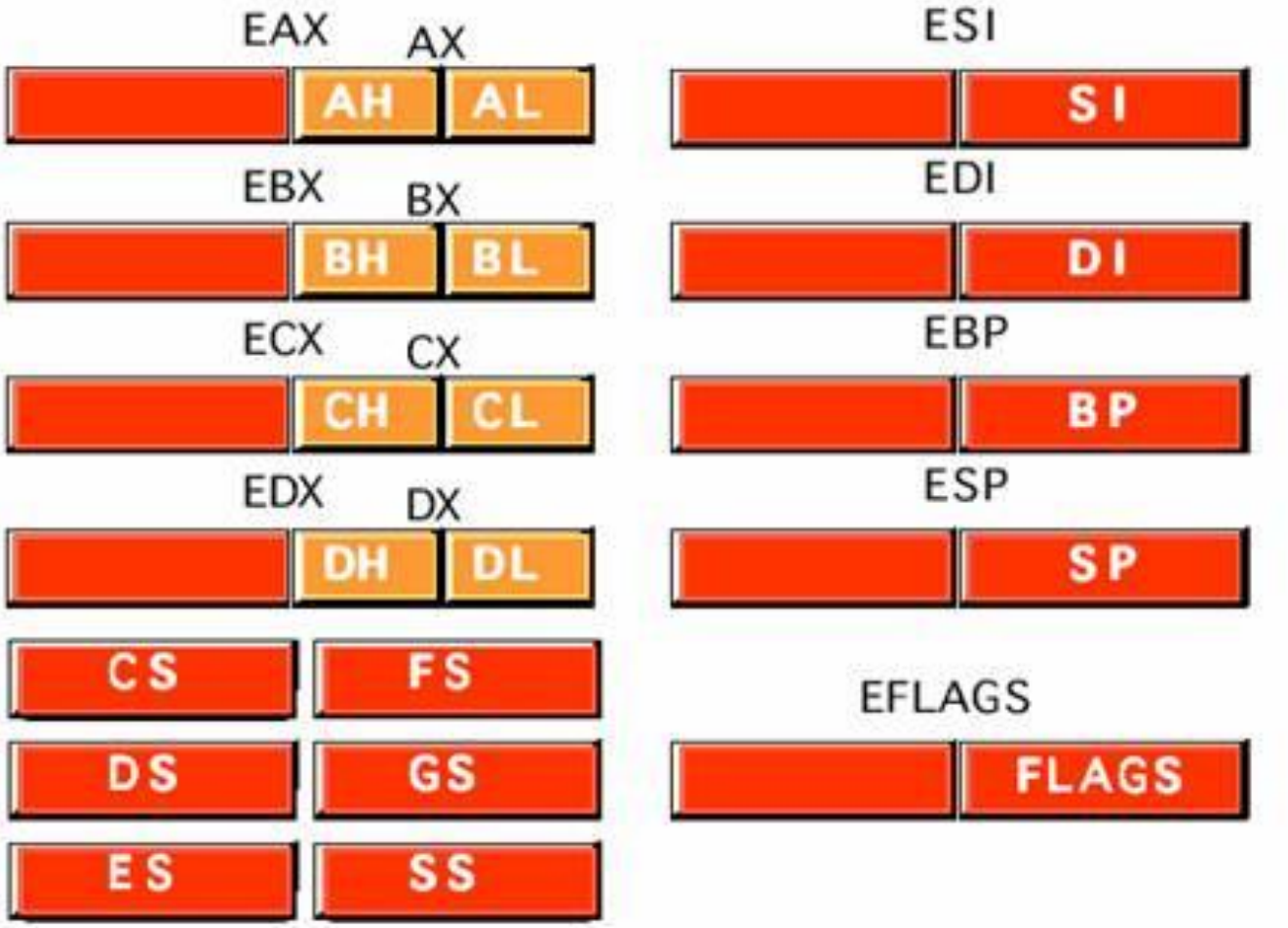
Arhitektura x86 procesora

- 1972. – Intel 8008, 8-bit
- 1978. – Intel 8086, 16-bit
- 1985. – Intel 80386, 32-bit
- CISC (ali RISC mikrokod)
- pojava proširenja
  - x87, MMX, SSE/SSE2/..., AVX/AVX2/...
  - VT-x / AMD-V
- ≈ 900 – 2000 instrukcija, max duljine 15 bajtova
  - pomicanje podataka
  - aritmetika / logika
  - upravljanje tokom (control flow)

Arhitektura x86 procesora

- 8 registara opće namjene
  - akumulator, brojač, podatkovni, bazni
  - pokazivač stoga, bazni pokazivač, izvorišni i odredišni pokazivač
- 6 segmentnih registara (širine 16 bita)
  - kodni, podatkovni, stogovni, extra, F (slijedi E), G (slijedi F)
- instrukcijski pokazivač
- registar sa zastavicama
  - prijenos, preljev, predznak, nula

Arhitektura x86 procesora



Arhitektura x86 procesora

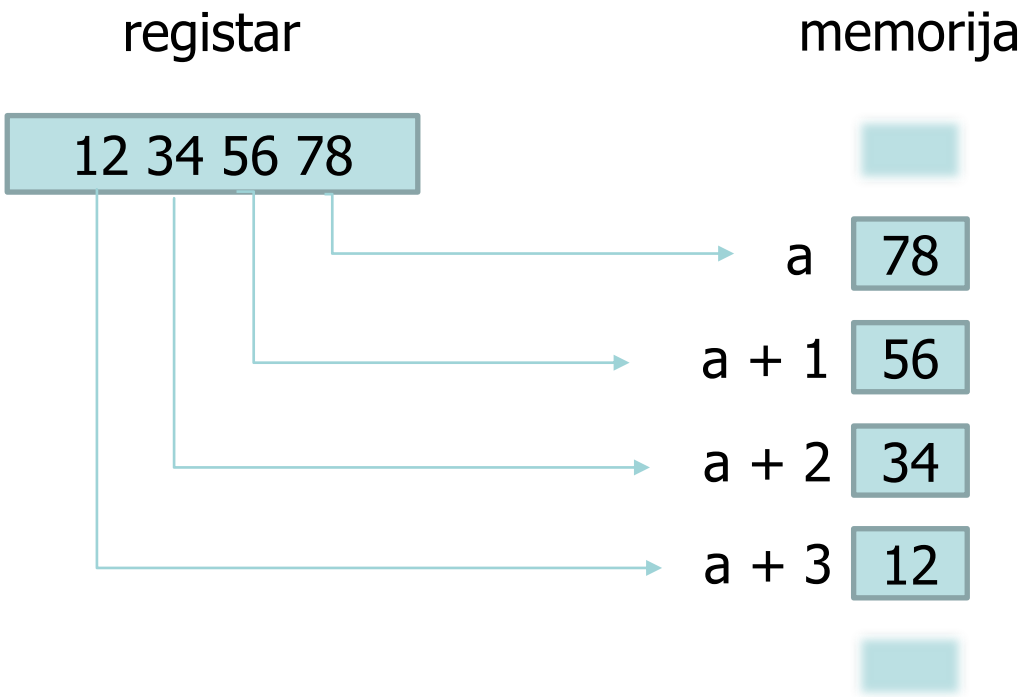
- dijelovi registara
  - EAX = 12345678h
  - AX = 5678h
  - AH = 56h
  - AL = 78h

EAX				**12 34 56 78**			
---		AX		12 34		56 78	
-	-	AH	AL	12	34	56	78

  - x86_64 -> RAX (- / EAX)

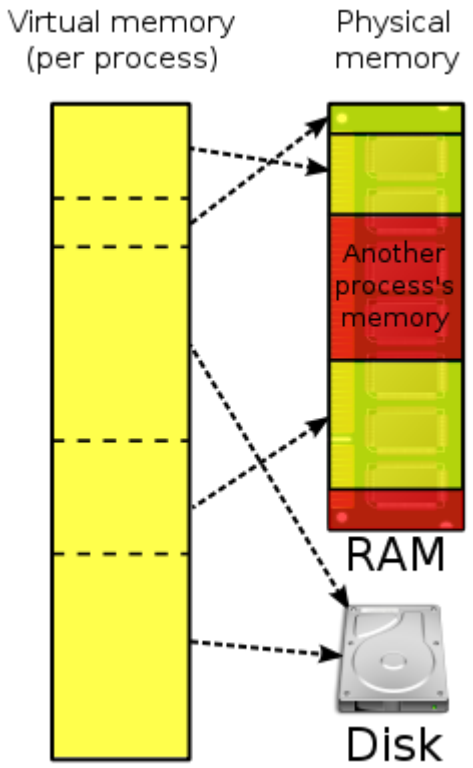
Arhitektura x86 procesora

- little endian (malokrajna) arhitektura
  - najmanje značajan bajt (LSB) sprema se prvi

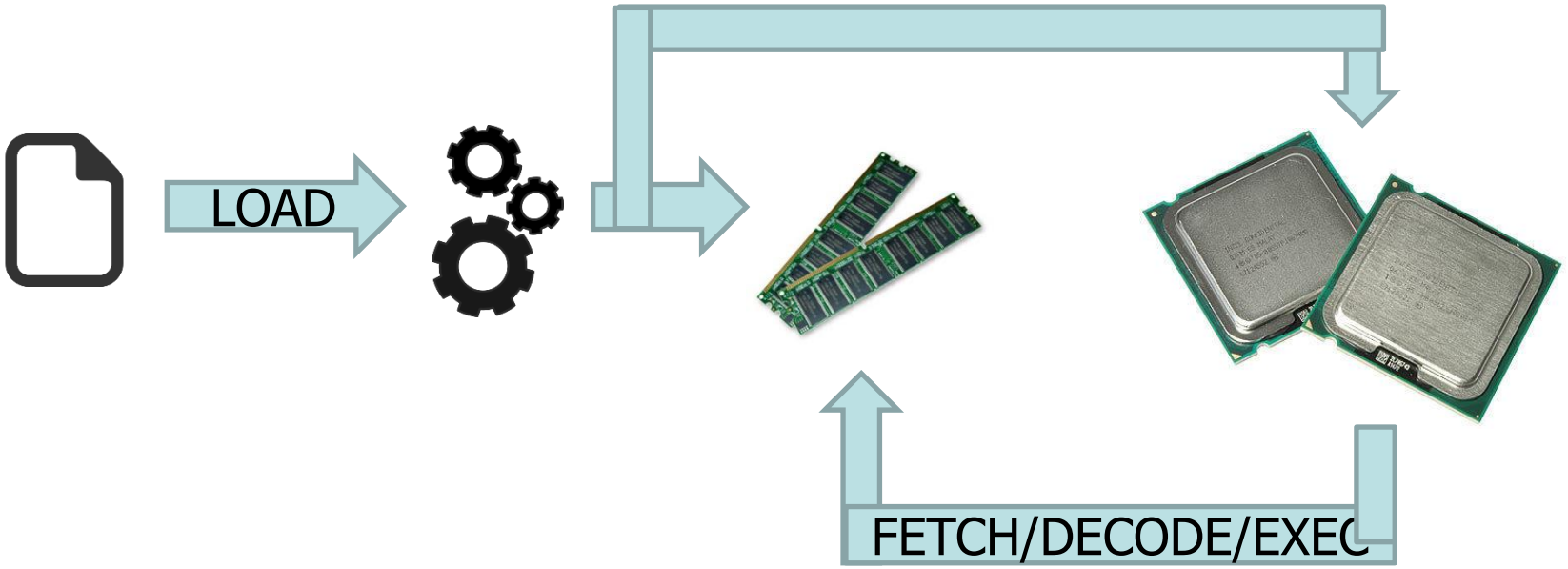


Arhitektura x86 procesora

- fizička memorija
  - pristup ima samo operativni sustav
  - upravitelj memorije dodjeljuje memoriju procesima
- virtualna memorija
  - tehnika upravljanja memorijom
  - svaki proces ima svoj „pogled” na memoriju
- memorijski modeli
  - segmentacija
  - straničenje



Arhitektura x86 procesora



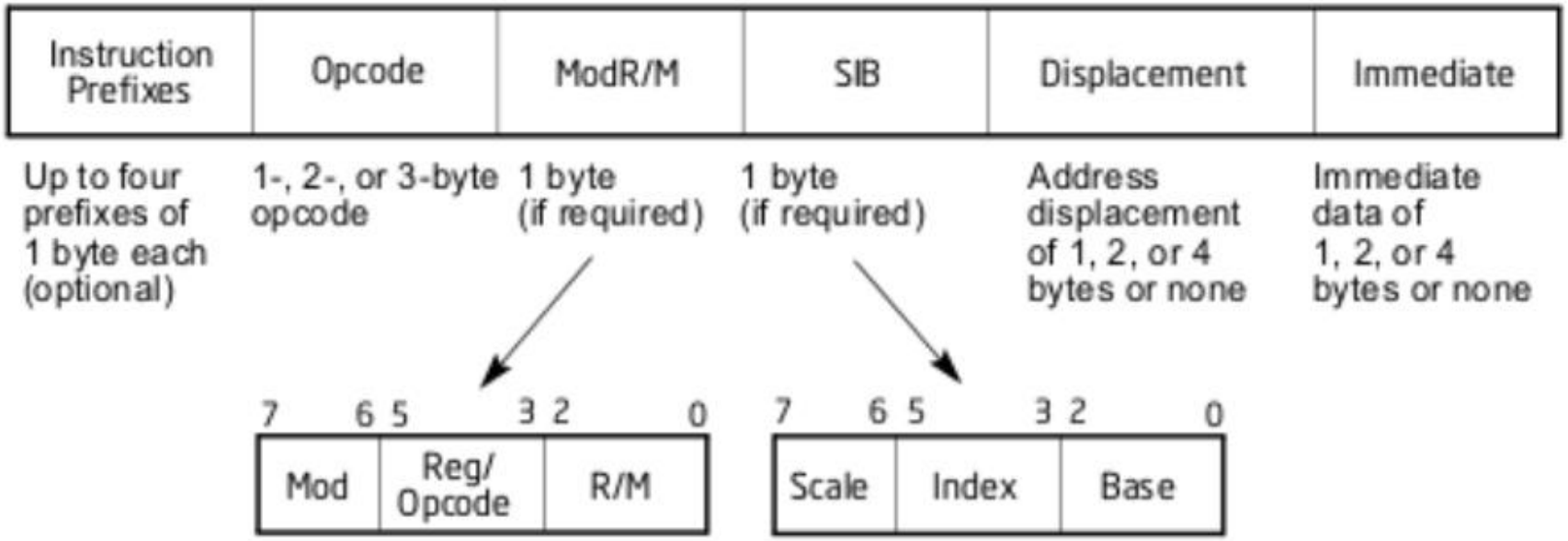
DISASSEMBLE

E8 00 00 00 00 5D B8 01 00 00 00 …

```
00401000    E8 00000000     CALL 6_except.00401005
00401005    5D              POP EBP
00401006    B8 01000000     MOV EAX,1
0040100B    0100            ADD DWORD PTR DS:[EAX],EAX
0040100D    FA              CLI
```

Arhitektura x86 procesora

- enkodiranje x86 naredbi



Intel 64 and IA-32 Architectures Software Developer Manuals

Arhitektura x86 procesora

- 0x8B 0x44 0x8B 0x24

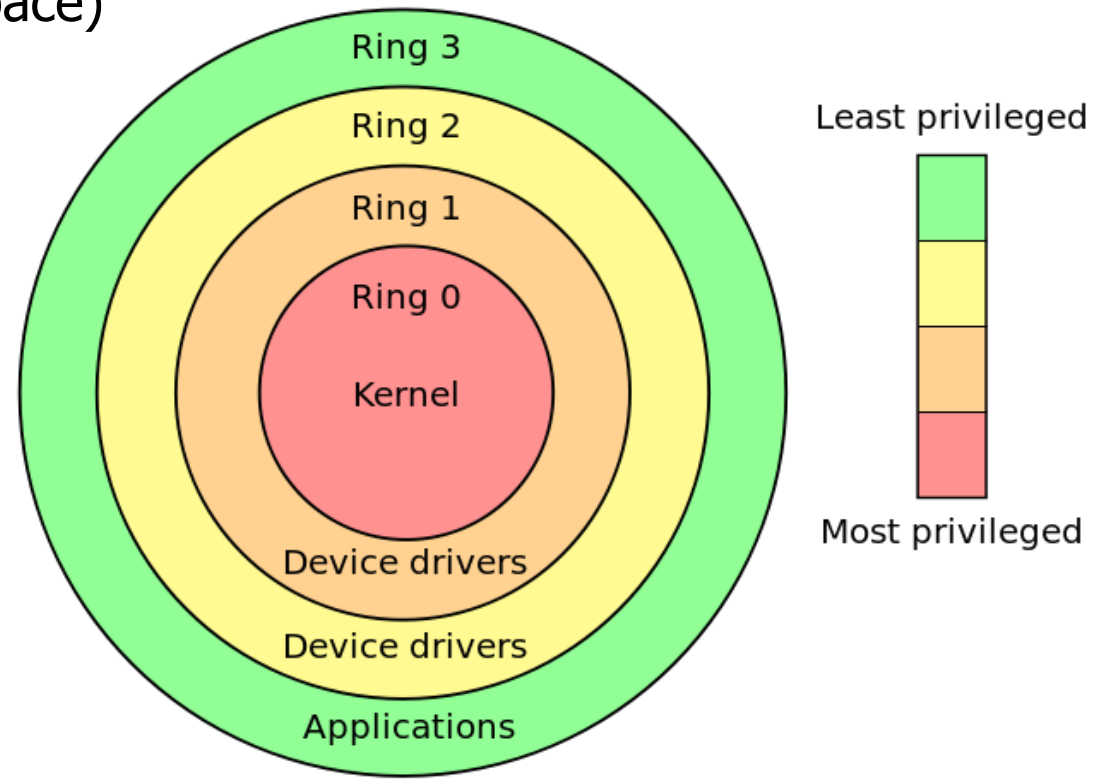
- 0x8B = mov r32, m/r32
- 0x44 0x8B = 0100 0100 1000 1011
- Mod = 01 (disp8), SS (scale) = 10 (*4)
- Reg = 000 (EAX), Index = 001 (ECX)
- RM = 100 ([][]), Base = 011 (EBX)

- mov eax, dword ptr [ebx + ecx * 4 + 24h]

Intel 64 and IA-32 Architecture Instruction Set Reference, 2-6 i 2-7

Arhitektura x86 procesora

- zaštićeni način (engl. protected mode)
  - prstenovi (rings) – slojevi zaštite
  - uglavnom se bavimo „ring 3” aplikacijama
  - korisnički prostor (user space)



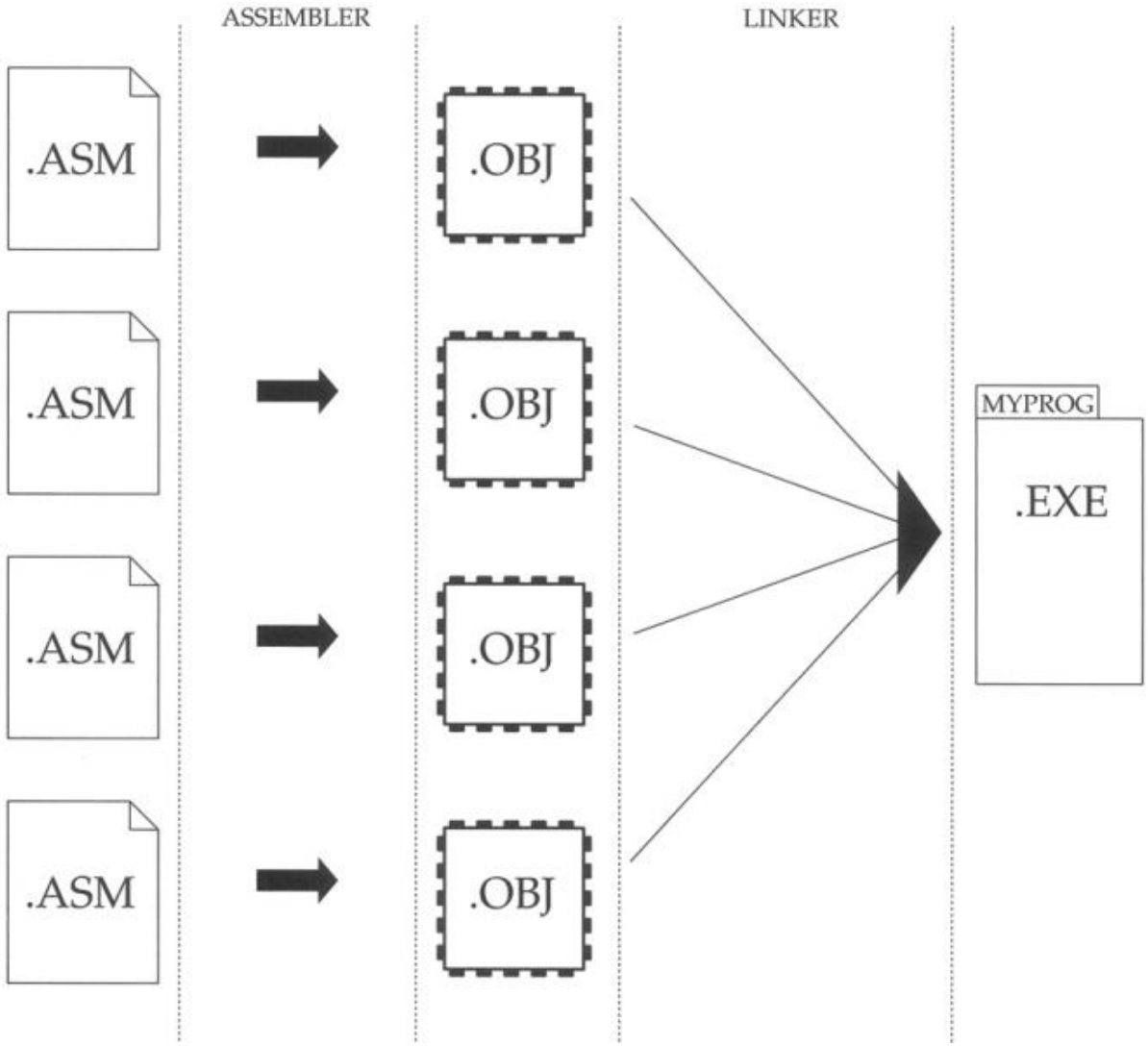
Arhitektura x86 procesora

- ... zečja rupa je dublja ☺
- ring -1
  - bare-metal hypervisor i virtualizacija (VT-x)
- ring -2
  - system management mode

Programiranje u x86 zbirnom jeziku

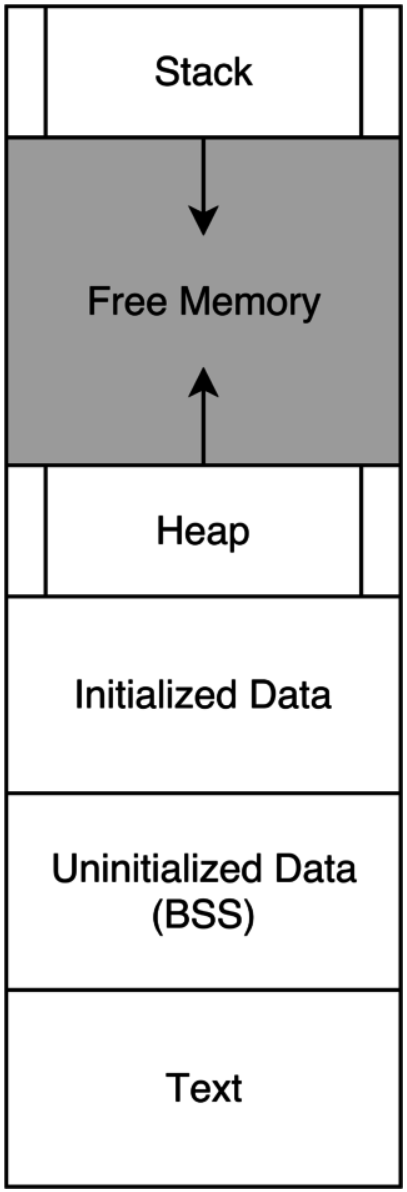
- 2 glavne vrste sintakse
  - Intel (ljepotica)
    - mov eax, DWORD PTR [ebx+ecx*4+0x24]
  - AT&T (zvijer)
    - mov 0x24(%ebx,%ecx,4), %eax
- brojni asembleri
  - MASM, NASM, FASM, gas, ...
- cjeline i raspored izvornog koda
  - asemblerske direktive (upute) i includeovi
  - deklaracija/definicija podataka
  - asemblerski kod
- tanka je granica između podataka i koda
  - mi dajemo značenje bajtovima (međusobno zamjenjivi)

Programiranje u x86 zbirnom jeziku



Programiranje u x86 zbirnom jeziku

- kod
- podaci
  - inicijalizirani - .data
  - neinicijalizirani - .bss
  - gomila
  - stog



Programiranje u x86 zbirnom jeziku

- prvi primjer – „hello world”
- komentari počinju s ‘;’
  - mogu biti bilo gdje u liniji, komentar je sve do novog retka
- ml /c /coff [/Cp] <assembly>.asm
  - /c – samo asembliraj, bez linkanja
  - /Cp – „preserve case of identifiers” – zadrži velika i mala slova za javne simbole (koji drugi mogu koristiti)
  - /coff - .obj izlazna datoteka će biti u Common Object File Formatu
- link /subsystem:console <assembly>.obj
  - /subsystem:console – napravi konzolnu aplikaciju (ne DOS)
  - /subsystem:windows – nemoj alocirati konzolni prozor

Programiranje u x86 zbirnom jeziku

- .386 – „dovoljna” (minimalna) arhitektura procesora
- .model flat [, stdcall]
  - memorijski model u kojem se kod i podaci nalaze u istom prostoru
  - tiny, small, compact, huge, ... 16-bit aplikacije
- option casemap :none
  - ne obaziri se na velika/mala slova u mnemonicima
- include/includelib – korištenje drugih funkcija
- .data – inicijalizirani podaci
- .data? – neinicijalizirani podaci
- .code
  - _start: / end _start
  - start: / end start (ako piše stdcall)

Programiranje u x86 zbirnom jeziku

- položaj sekcija u .obj datotekama
- položaj sekcija u .exe datotekama
- analiza
  - hex editor
  - file format viewer
- strings
  - aplikacija, ako ju nemate, instalirajte
  - strings v2.53, Mark Russinovich

Osnove debugginga

- OllyDbg, ImmunityDbg, x64dbg, windbg, gdb
- 4 glavna prozora (pogleda)
  - prozor s kodom
  - prozor s registrima
  - prozor sa stogom
  - prozor s memorijom
- breakpoint – prekidna točka
  - softverski
  - memorijski
  - hardverski

Osnove debugginga

- step over/into – prijeđi preko/uđi u
- popis znakovnih nizova
- pretraga
- dodatci
  - ASM help
  - Windows API help
  - anti-debugging
- asembliraj u kodu
  - multiline ASM plugin
- učitani moduli
  - uključeni APIji

Osnove debugginga

- iznimke
  - dijeljenje s nulom
  - pogreška memorijskog pristupa (access violation)
  - neispravne / privilegirane instrukcije
  - INT3 – softverski breakpoint
  - …

Instrukcijski skup

- mov dest, src
- „adresiranje”
  - register: mov eax, eax
  - immediate: mov eax, 1h
  - base-index: mov eax, [ebx + ecx*1]
  - register indirect: mov eax, [eax]
  - direct offset addressing: mov eax, [ebx + 2h]
  - direct memory addressing: mov eax, [401000h]
  - base-index with displacement: mov eax, [ebx + ecx*2 + 4]

Instrukcijski skup

- aritmetičke i logičke instrukcije
- add\adc\sub\sbb dest, src
  - sub -> dest = dest - src
- and\or\xor dest, src
- cmp\test dest, src
  - cmp je isto kao i sub, ali bez spremanja u dest
  - test je isto kao i and, ali bez spremanja u dest
- div\mul src
  - mul -> edx:eax = eax * src
  - div -> eax = edx:eax / src ; edx = edx:eax % src
  - idiv \ imul – predznačno dijeljenje / množenje

Instrukcijski skup

- shr\shl\sar\sal\ror\rol dest, src
  - shr \ shl su logički pomaci (nepredznačeni brojevi)
  - sar \ sal su aritmetički pomaci (predznačeni brojevi)
  - ror \ rol su rotacije
- not\neg dest
  - not – logička negacija (prvi komplement)
  - neg – dvojni komplement
- inc\dec dest
- lea dest, src
  - load effective address
  - „trik” za spremanje izračunate adrese, umjesto sadržaja na koji ta adresa pokazuje
  - zgodno za adresiranje i indeksiranje polja

Instrukcijski skup

- instrukcije imaju i „nuspojave” (side effects)
- aritmetika i zastavice
  - nula (Z) – ako je rezultat == 0x0, Z = 1
  - predznak (S) – ako je najznačajniji bit 1, S = 1
  - preljev (O) – aritmetika predznačenih brojeva
    - 0x7FFFFFF + 0x1, O = 1
  - prijenos (C) – aritmetika nepredznačenih brojeva
    - 0xFFFFFFFF + 0x2, C = 1
    - 0x00000000 – 0x2, C = 1 (posudba)
  - parnost, paritet (P) – broj postavljenih bitova u LSB
    - 0x01, P = 0
    - 0x101, P = 0
    - 0x203, P = 1

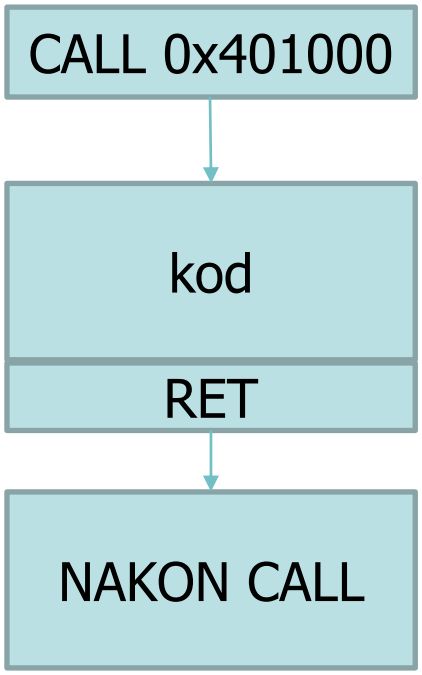
Instrukcijski skup

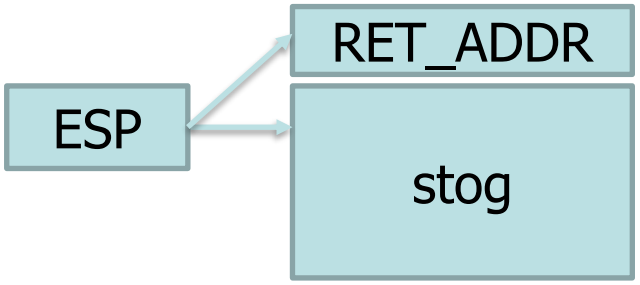
- aritmetika i zastavice
  - prilagodba (A) – prijenos / posudba na 4 najmanje značajna bita
    - 0x0F + 0x01, A = 1
  - trap flag – pojedinačno izvršavanje instrukcije (single stepping)
  - zastavica smjera – koristi se za kopiranje i rad sa znakovnim nizovima

Instrukcijski skup

- grananje
  - bezuvjetno: jmp
  - uvjetovano: jnz \ jz \ je \ jne \ jc \ jnc \ jo \ jp \ jg \ ja \...
- poziv procedure i povratak
  - call \ ret

Instrukcijski skup





Instrukcijski skup

- petlje
  - loop
    - umanji brojač (ECX \ CX)
    - ponavljaj dok brojač != 0
  - loope\loopne
    - umanji brojač (ECX \ CX)
    - ponavljaj dok brojač != 0 i ZF (!)= 1
  - jcxz
    - korisno prije ulaska u petlju
    - loop prvo umanji brojač, a onda provjerava njegovo stanje
    - nezgodno ako je brojač u početku == 0

Instrukcijski skup

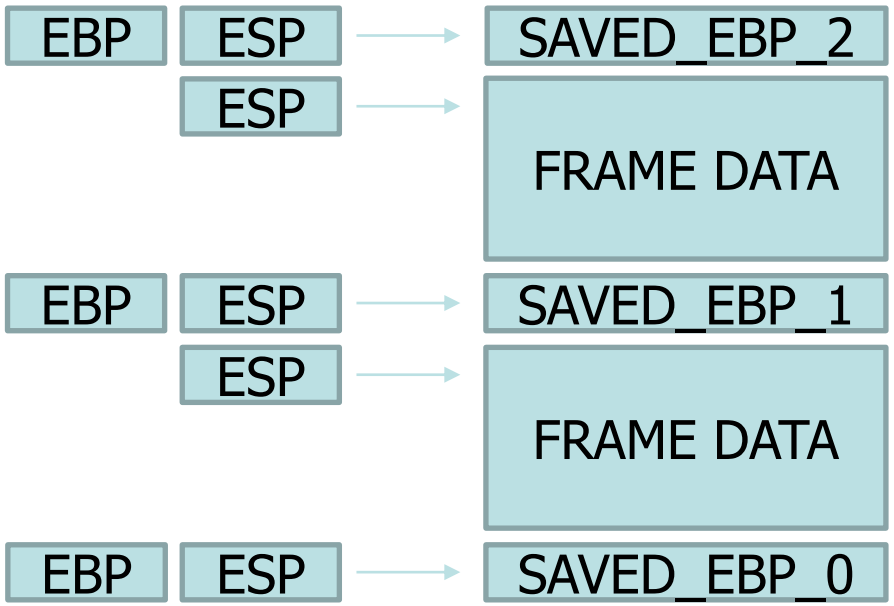
- druge instrukcije i neki prefiksi
  - nop – no operation
  - scas – scan string for byte
  - lods \ stos src – load \ store string
  - movs – move byte from memory to memory
  - int - interrupt
  - lock prefix – atomic execution
  - rep \ repe \ repne prefix – repeat following instruction
  - enter \ leave – shorthand for function prologue \ epilogue

Instrukcijski skup

- operacije sa stogom
- push \ pop dest
- pushad \ popad \ pushfd \ popfd
  - sprema stanje svih registara \ zastavica

Instrukcijski skup

- stogovni okvir
- lokalne vs. globalne varijable
- pokazivači
  - na bazu stoga (EBP)
  - pokazivač na vrh stoga (ESP)
- **ulančana lista**



Instrukcijski skup - digresija

- pozivni dogovori (konvencije)
  - cdecl, syscall, pascal, **stdcall**, thiscall, ...
- stdcall
  - argumenti se predaju stogom, s desna na lijevo
  - onaj koga se zove (callee) čisti stog
    - RETN <veličina>
  - povratna vrijednost u EAX registru

Instrukcijski skup

- drugi primjer
- višebajtna XOR enkripcija i dekripcija
  - prolazak petlje po podacima koje šifriramo (plaintext)
  - indeks prolazi i kroz XOR ključ
  - dva načina
  - pisanje funkcije za dekripciju
- prvi zadatak
  - napraviti Cezarovu enkripciju i dekripciju za string “zebra”
  - enkripciju ostvariti petljama, dekripciju funkcijom
  - paziti na wrap-around!

Sistemski pozivi i API

- servisi koje nam pruža operativni sustav
  - kontrola procesa
  - upravljanje datotekama
  - upravljanje uređajima
  - upravljanje informacijama i stanjem
  - komunikacija
- sistemskih poziva ima malo
  - oko 500 na Windows OS
- nad njima se grade API-ji
  - enkapsuliraju i nadograđuju limitirani set sistemskih poziva
  - oko 50.000 na Windows OS

Sistemski pozivi i API

- treći primjer
- skoro pa pravi „hello world”
  - ispisati poruku pomoću MessageBox API-ja

Sistemski pozivi i API

- važni API-ji
  - kontrola i stvaranje procesa i dretvi
    - CreateProcess, CreateThread, CreateMutex
  - upravljanje i rad s datotekama
    - CreateFile, ReadFile, WriteFile, CloseHandle
  - upravljanje i rad s memorijom
    - VirtualAlloc, VirtualProtect, VirtualFree, HeapAlloc, HeapFree
  - dinamičko učitavanje dodatnih modula
    - LoadLibrary, GetProcAddress

Sistemski pozivi i API

- četvrti primjer
- jednostavna poslužiteljska aplikacija
  - poslužitelj "sluša" na portu 1234
  - primljenu naredbu treba parsirati i izvršiti ovisno o parametrima naredbe
  - naredbe:
    - e/echo <neka_poruka>
    - r/run<neki_proces.exe>
    - k/kill <PID>

Zaobilaženje jednostavne zaštite

- patching („krpanje”)
  - zamjena instrukcija vlastitim instrukcijama
  - uklanjanje dosadnih („nag”) prozora, provjere autorizacije, CD-provjere
- „pecanje” serijskih ključeva / brojeva
- pisanje generatora serijskih ključeva
  - tema za idući tjedan ☺