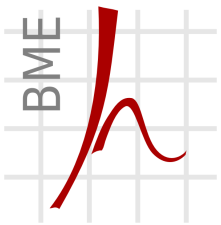
Bevezetés – Alapfogalmak – Jog

Kód visszafejtés.



Híradástechnikai Tanszék

Izsó Tamás

2016. szeptember 8.

Tartalom

Section 1

Bevezetés

Bemutatkozás

Izsó Tamás

- BME Villamosmérnöki és Informatikai Kar
  - Hálózati Rendszerek és Szolgáltatások Tanszék
- honlap: www.hit.bme.hu/∼izso
- moodle: https://moodle.hit.bme.hu/
- email: izso@hit.bme.hu
- szoba: IB124
- Tel: 06 1 463 3277

Követelmény

- max 4 hiányzás;
- 1 házi feladat;
- 3 kiszh.

Tematikáról

- ötödször kerül előadásra;
- a tematika tartalma és sorrendje változik;
- a tanulságok általánosak;
- a példák rendszerfüggőek;
  - Intel x86 proc;
  - Windows/XP Windows 8, Windows 10;
  - MS Visual Studio 2008 vagy későbbi verzió;
  - (ha szükséges VMWARE image biztosítva lesz).

Milyen tantárgyakat érint

1. Programozás alapjai I. II.
2. Digitális technika
3. Számítógép-architektúrák
4. Gépi nyelvek
5. Operációs rendszerek
6. Assembly programozás
7. stb.

Milyen tantárgyakat érint

1. Programozás alapjai I. II.
2. Digitális technika
3. Számítógép-architektúrák
4. Gépi nyelvek
5. Operációs rendszerek
6. Assembly programozás
7. stb.

- Csak az első pont ismeretét követeli meg a tananyag.

Milyen tantárgyakat érint

1. Programozás alapjai I. II.
2. Digitális technika
3. Számítógép-architektúrák
4. Gépi nyelvek
5. Operációs rendszerek
6. Assembly programozás
7. stb.

- Csak az első pont ismeretét követeli meg a tananyag.
- Vannak átfedések, de ezen részek részletesebben vannak kifejtve.

Tantárgy célkitűzése

- megismerteti a program végrehajtását;
- megalapozza a programok nyomkövetését (debuggolás);
- jártasságot ad a gépi kód megértésében (visszafejtését);
- ismeretet nyújt a programok visszafejtésének megnehezítésében;
- stb.

Ideális hallgató hozzáállása a tantárgyhoz

- Érdeklődő;
- kreatív;
- logikusan gondolkodik;

Section 2

Alapfogalmak

Alapfogalmak

Definíció: (Reverse engineering)

Azon eljárásokat, amelynek keretében a tárgykódból a szükséges információkat kinyerjük, és értelmezzük reverse engineeringnek nevezzük.

Alapfogalmak

Definíció: (Reverse engineering)

Azon eljárásokat, amelynek keretében a tárgykódból a szükséges információkat kinyerjük, és értelmezzük reverse engineeringnek nevezzük.

Definíció: (Disassembler)

A gépi kódból assembly utasításokat állít elő.

Alapfogalmak

Definíció: (Reverse engineering)

Azon eljárásokat, amelynek keretében a tárgykódból a szükséges információkat kinyerjük, és értelmezzük reverse engineeringnek nevezzük.

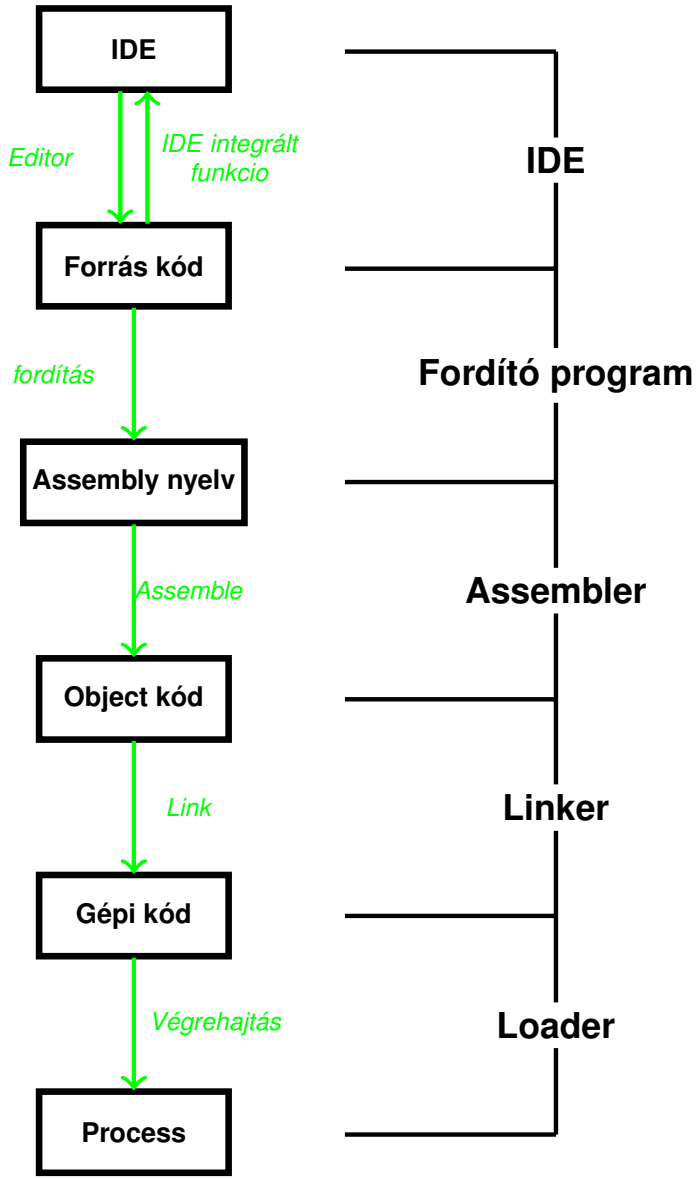
Definíció: (Disassembler)

A gépi kódból assembly utasításokat állít elő.

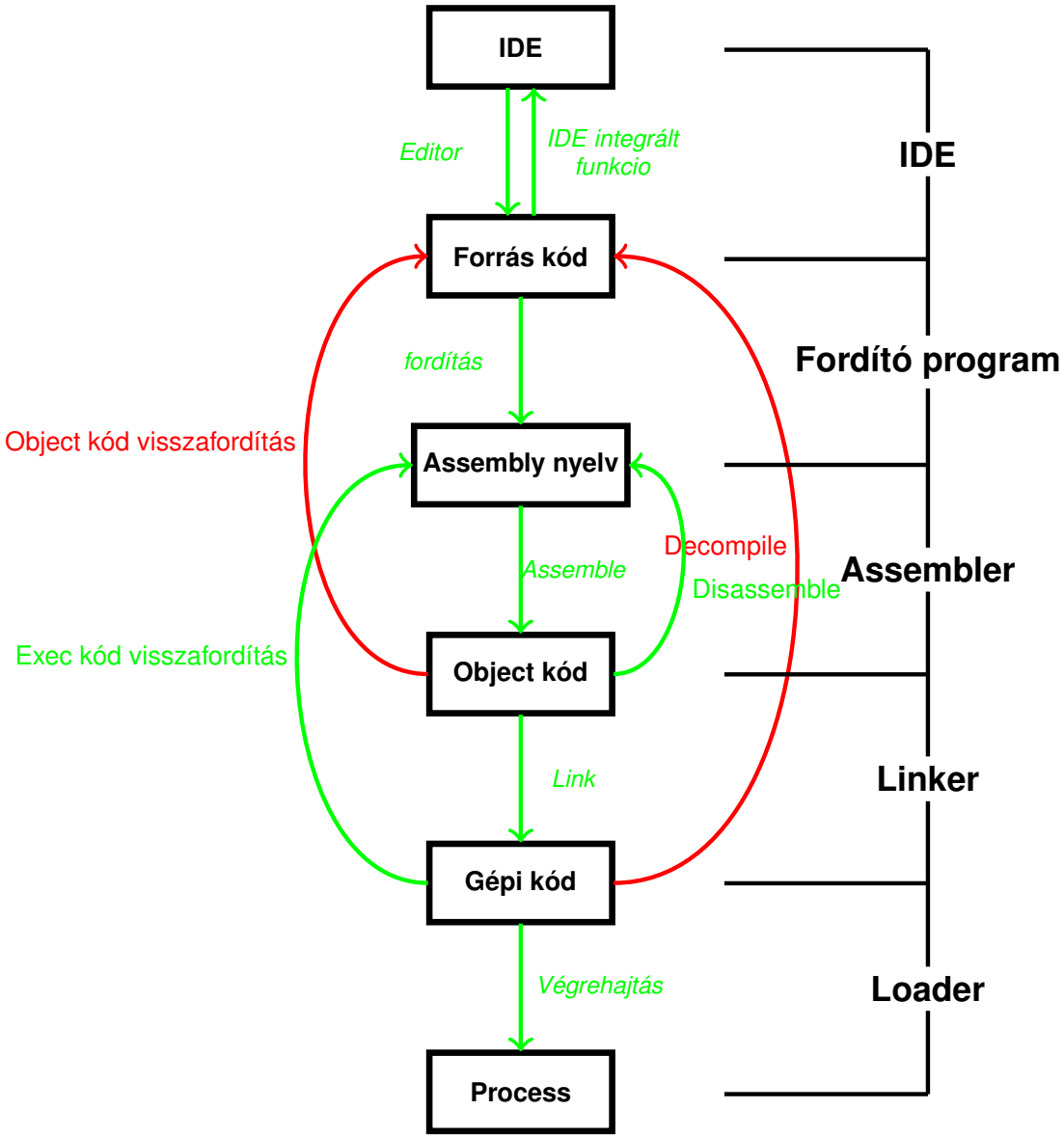
Definíció: (Decompiler)

A tárgykódból egy visszafordító program segítségével magasszintű (például C) forráskódot állít elő

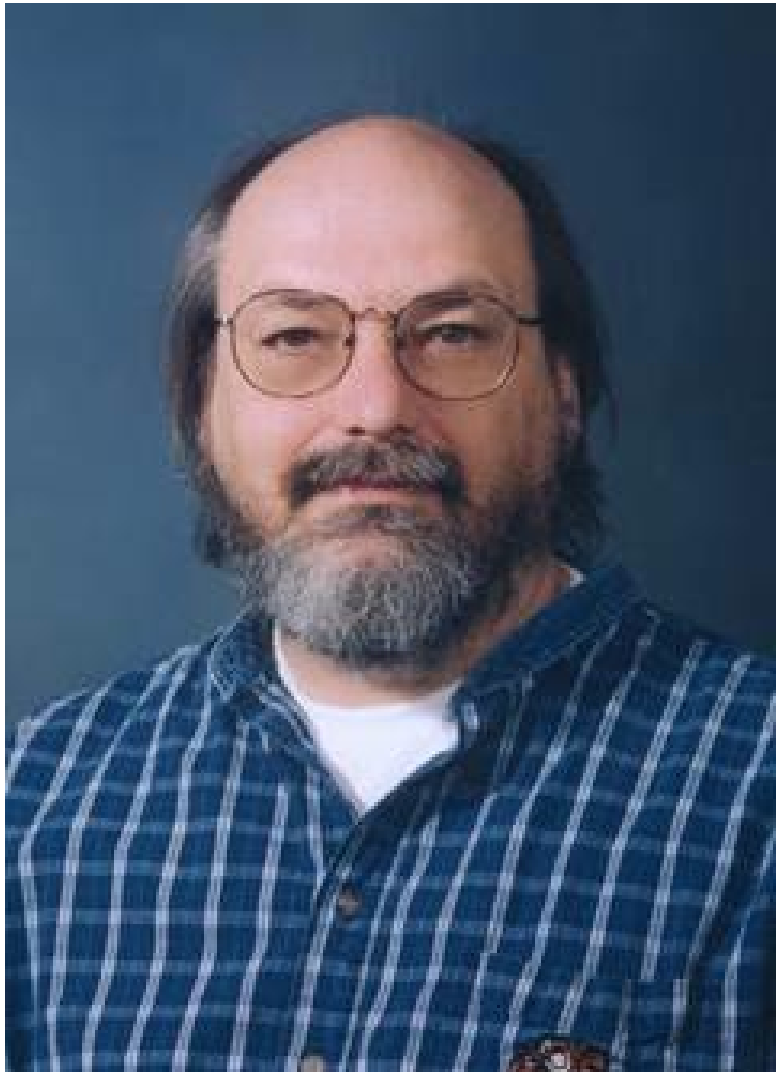
Forward engineering



Reverse engineering



Első "hacker"



Ken Thompson

- Főbb munkássága:
  - B Programming Language (Később Dennis Ritchie ez alapján készítette a C-t)
  - UNIX
  - Plan 9
  - reguláris kifejezések elterjesztése
  - Számítógépes sakk algoritmus (végjáték)
  - Rob Pike-vel UTF-8 kódolás (változó hosszúságú)
- 1983: Turing Award (Dennis Ritchie-vel) a UNIX operációs rendszerért
- 1999: US National Medal of Technology
- 1999: First IEEE Tsutomu Kanai Award

Analógia

```
char*f="char*f=%c%s%c;main() {printf(f,34,f,34,10);}%c";
main(){printf(f,34,f,34,10);}
```

Output:

Analógia

```
char*f="char*f=%c%s%c;main() {printf(f,34,f,34,10);}%c";
main(){printf(f,34,f,34,10);}
```

Output:

```
char*f=
```

Analógia

```
char*f="char*f=%c%s%c;main() {printf(f,34,f,34,10);}%c";
main(){printf(f,34,f,34,10);}
```

Output:

```
char*f="
```

Analógia

```
char*f="char*f=%c%s%c;main() {printf(f,34,f,34,10);}%c";
main(){printf(f,34,f,34,10);}
```

Output:

```
char*f="char*f=%c%s%c;main() {printf(f,34,f,34,10);}%c
```

Analógia

```
char*f="char*f=%c%s%c;main() {printf(f,34,f,34,10);}%c";
main(){printf(f,34,f,34,10);}
```

Output:

```
char*f="char*f=%c%s%c;main() {printf(f,34,f,34,10);}%c"
```

Analógia

```
char*f="char*f=%c%s%c;main() {printf(f,34,f,34,10);}%c";
main(){printf(f,34,f,34,10);}
```

Output:

```
char*f="char*f=%c%s%c;main() {printf(f,34,f,34,10);}%c";
main(){ printf (f,34,f,34,10);}
```

Analógia

```
char*f="char*f=%c%s%c;main() {printf(f,34,f,34,10);}%c";
main(){printf(f,34,f,34,10);}
```

Output:

```
char*f="char*f=%c%s%c;main() {printf(f,34,f,34,10);}%c";
main(){ printf (f,34,f,34,10);}
```

C fordító

- A C (cc, gcc, cl) fordítót C-ben írták.
- Speciális karaktereket csak úgynevezett *escape sequence* segítségével írhatunk le. Például '\n', '\\' .

```
c = next();
if(c != '\\')
    return c;
c = next();
if(c == '\\')
    return '\\';
if(c == 'n')
    return '\n'
. . .
```

C fordító

Adjunk hozzá új *escape sequence*-t.

- A C (cc, gcc, cl) fordítót C-ben írták.
- A '\v'-t még nem ismeri a régi verziójú fordító, ezért fordítása hiba lesz.

```
c = next();
if(c != '\\')
    return c;
c = next();
if(c == '\\')
    return '\\';
if(c == 'n')
    return '\n'
. . .
if(c == 'v')
    return '\v' //hiba
```

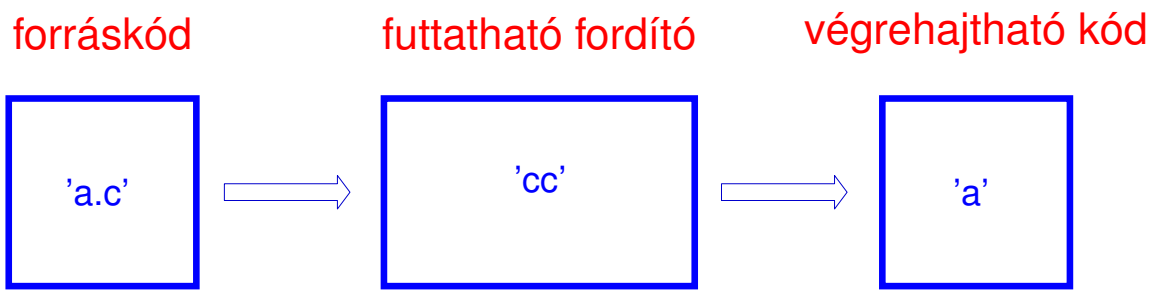
C fordító

Adjunk hozzá új *escape sequence*-t.

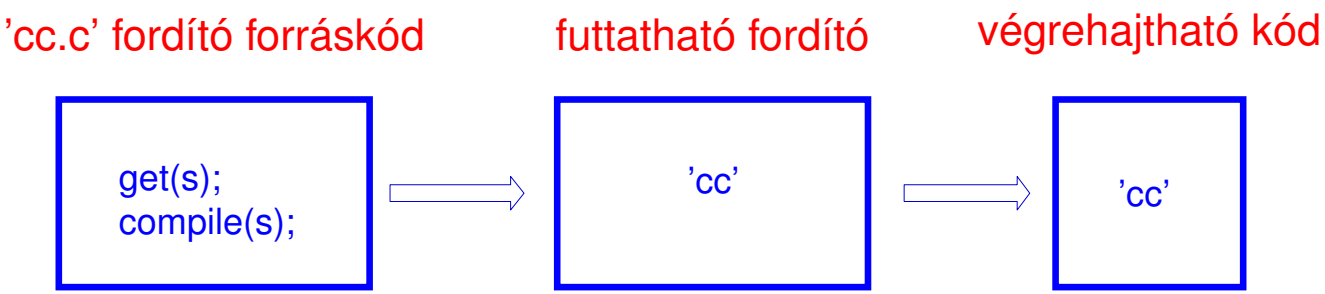
- A C (cc, gcc, cl) fordítót C-ben írták.
- A '\v'-t még nem ismeri a régi verziójú fordító, ezért fordítása hiba lesz.

```
c = next();
if(c != '\\')
    return c;
c = next();
if(c == '\\')
    return '\\';
if(c == 'n')
    return '\n'
...
if(c == 'v')
    return 11; // jo
```

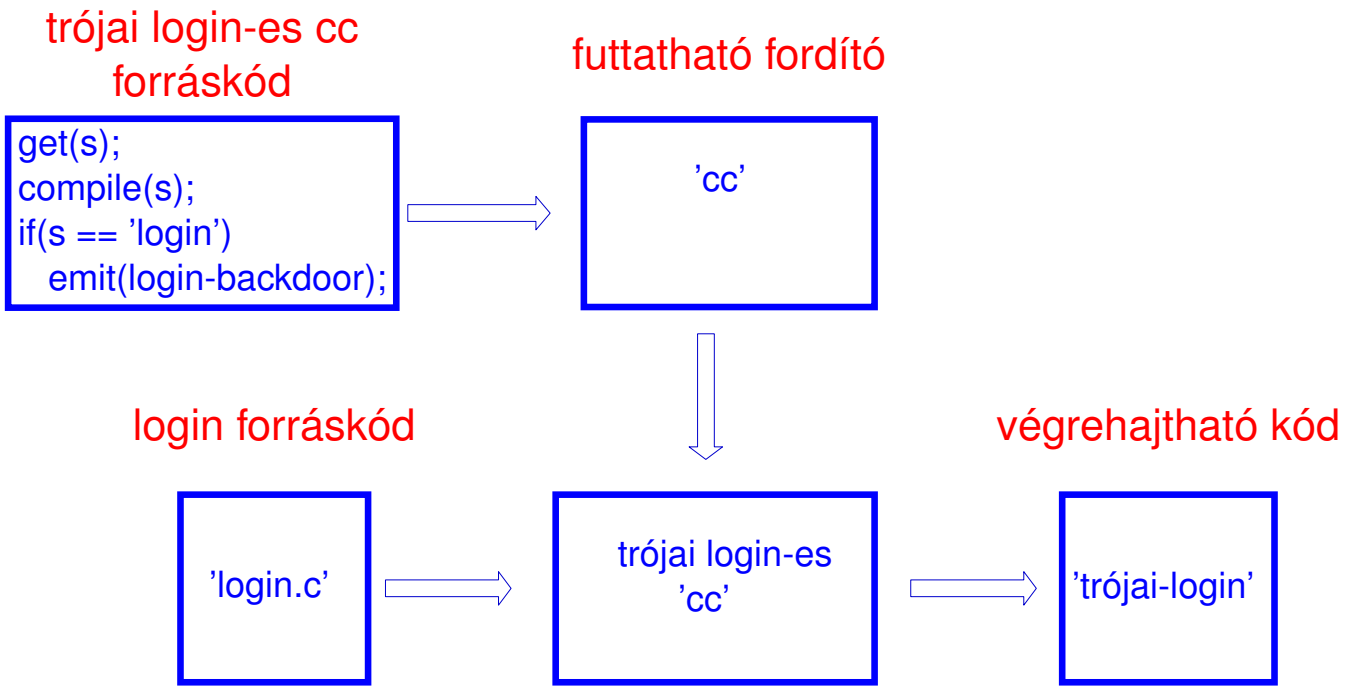
Fordító működése



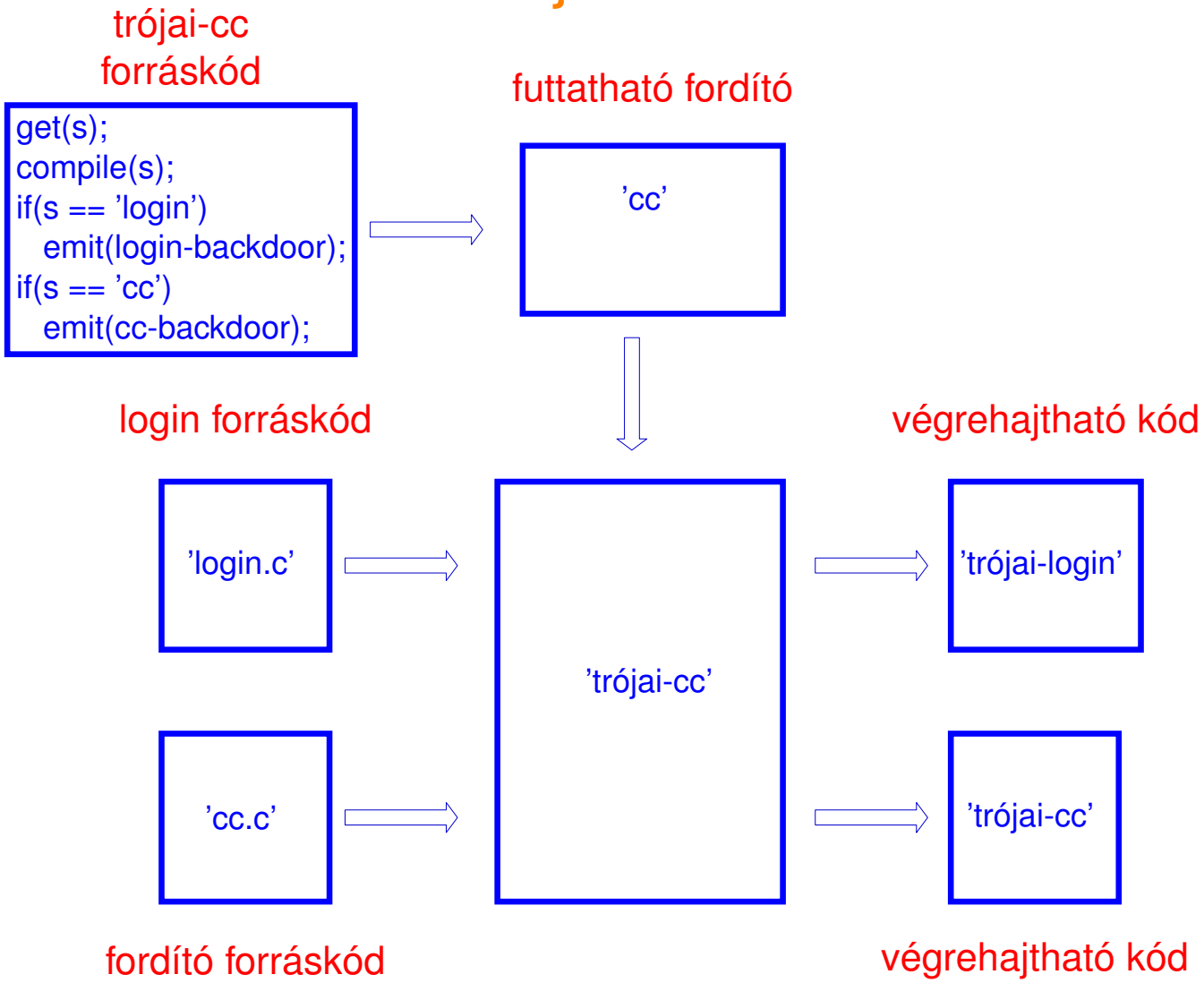
Fordító működése saját forrására



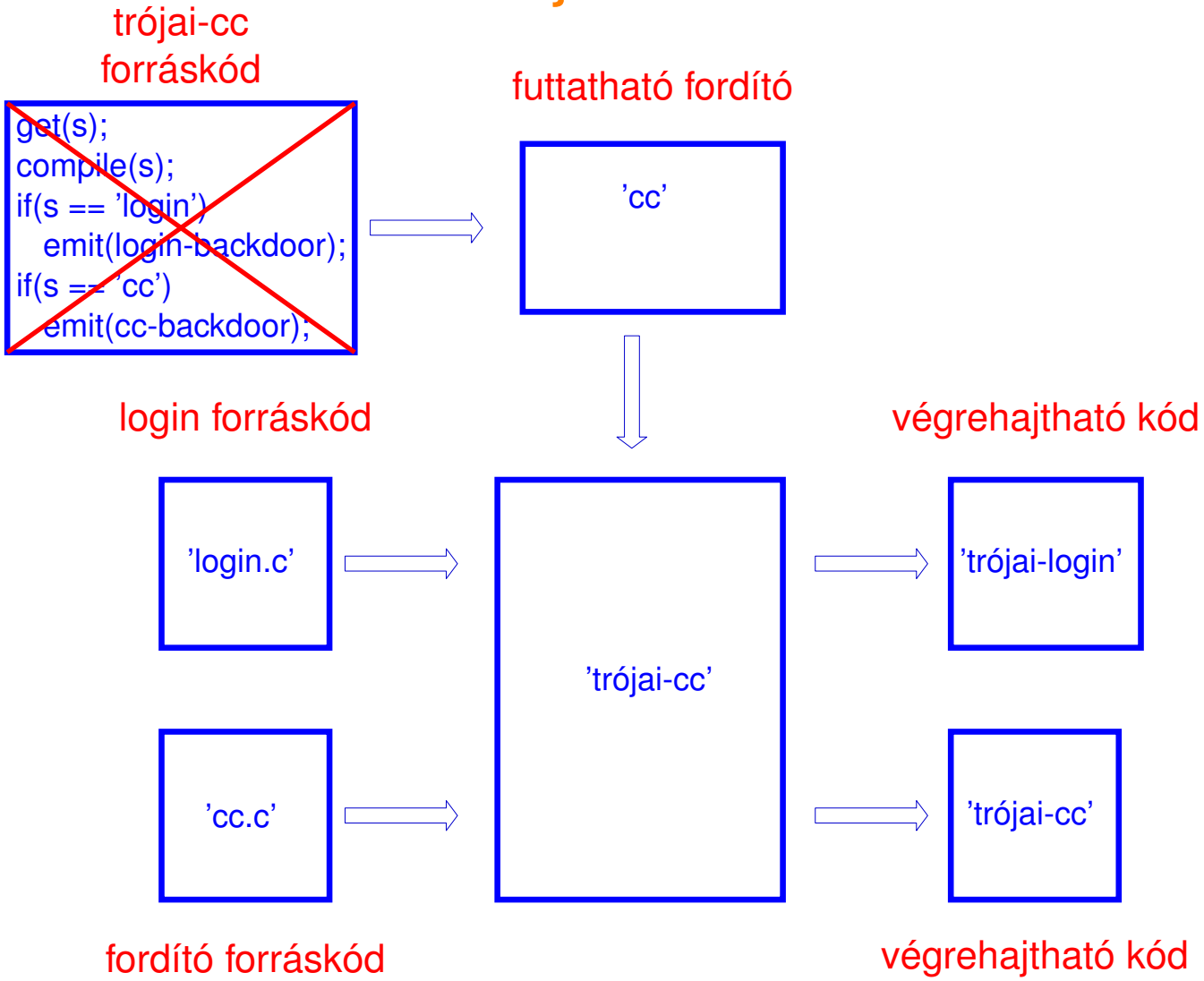
Trójai login



Trójai fordító



Trójai fordító



Miért érdemes a kódot visszafejteni?

- Programok közötti együttműködés biztosítása (nem készítettek jól dokumentált interface-t),
- létező, esetleg régen írt programrendszerbe fejlesztés,
- hiányos, vagy hiányzó dokumentáció,
- programanalízis (pl. a kód hatékonyságára),
- biztonsági ellenőrzések,
- víruskeresés,
- algoritmus megismerés *esetén*.

És még mikor?

- Katonai vagy kormányzati célú dobozos (Commercial Off-the-shelf) software-k felhasználása,
- inline assembly kódot tartalmazó C programban,
- különböző programozási nyelvben írt programok,
- programhoz linkelt könyvtárak hibában játszott szerepe,
- párhuzamos programozást (thread, openmp) használó programok,
- hibás fordító ellenőrzése,
- alacsony szintű hibák, például puffer túlcsordulás sebezhetőségének a megállapítása *esetén*.

Bináris programanalízis

Definíció: (Programanalízis)

Az a folyamat, amikor automatikusan a program viselkedéséből a program tulajdonságaira következtetünk.

Dinamikus analízis

Kiválasztott inputokra futtatjuk a programot, és a program konkrét futásából következtetünk a tulajdonságára. Hátrány, hogy nem biztos, hogy minden ágat bejárunk.

Statikus analízis

Futtatás nélkül, matematikai módszerekkel következtetünk a program tulajdonságára. A statikus analízis konzervatív, azaz amit meg tudunk állapítani, az minden input esetén igaz. Bizonyos dolgokra azonban így nem tudunk következtetni.

Technikák

Dinamikus analízis

- software tesztelés,
- teljesítmény analízis,
- program nyomkövetés,
- program instrumentáció,
- dinamikus program szeletelés.

Statikus analízis

- adatfolyam analízis,
- vezérlésfolyam analízis,
- programhelyesség bizonyítás,
- absztrakt interpretáció,
- szimbolikus végrehajtás,
- statikus program szeletelés.

Section 3

Motivációs példák

WYSINWYX

What You See Is Not What You eXecute[1]

[1] Gogul Balakrishnan doktori értekezésének a címe.

Kivételkezelő beláncolása

```
39  int main()
40  {
41
42      // Stack-be felépítjük a láncolást
43      __asm
44      {
45          sub     ESP, 8
46          mov     dword ptr pRegister,ESP
47          mov     EAX,FS:[0]
48          mov     dword ptr prev,EAX
49      }
50
51      pRegister->handler = my_except_handler;
52      pRegister->prev = prev;
53
54      // TIB-be regisztráljuk a hibakezelőnket
55      __asm
56      {
57          mov     EAX, dword ptr pRegister
58          mov     FS:[0],EAX
```

Optimalizáció hatása

```
1  #include <stdio.h>
2  #include <string.h>
3
4  #define ZeroMemory(Destination,Length) \
5      memset((Destination),0,(Length))
6
7  void DatabaseConnect(char *szDB) {
8    char szPwd[64];
9    if (GetPasswordFromUser(szPwd,sizeof(szPwd))) {
10       if (ConnectToDatabase(szDB, szPwd)) {
11           // Cool, we're connected
12           // Now do database stuff
13       }
14   }
15   ZeroMemory(szPwd,sizeof(szPwd));
16 }
```

A memória törlésével a password élettartamát szerették volna lerövidíteni, különben a program elszállásnál a dump file-ból ki lehet olvasni az értékét.

A lefordított program

```
> cl /Ox /c /Gs- passwordlivetime.c
> dumpbin /disasm passwordlivetime.obj
```

```
DatabaseConnect:
    sub     esp,40h
    lea     eax,[esp]
    push    40h
    push    eax
    call    _GetPasswordFromUser
    add     esp,8
    test    eax,eax
    je      00000026
    mov     edx,dword ptr [esp+44h]
    lea     ecx,[esp]
    push    ecx
    push    edx
    call    _ConnectToDatabase
    add     esp,8
    add     esp,40h
    ret
```

memset hívást kioptimalizálta a program!

Nem egyértelmű kifejezés

```c
#include <stdio.h>
int main() {
  int i=2;
  printf("%d %d ", ++i, ++i );
  return 0;
}
```

```
> cl /Ox /c /GS- wrongexpr.c
> dumpbin /disasm wrongexp.obj
```

```
_main:
   push        4
   push        4
   push        offset $SG2502
   call        _printf
   add         esp,0Ch
   xor         eax,eax
   ret
```

Bináris művelet operandusainak a kiegyenlítése

```
1  #include <stdio.h>
2
3  int main() {
4    int a = -5;
5    if (a < sizeof (int))
6      printf(" %d < %d\n", a, sizeof(int) );
7    else
8      printf(" %d >= %d\n", a, sizeof(int) );
9    return 0;
10 }
```

```
> cl /Ob2 /c /GS- conv.c
```

Bináris művelet operandusainak a kiegyenlítése

```
> dumpbin /disasm conv.obj

_main:
    push    ebp
    mov     ebp,esp
    push    ecx
    mov     dword ptr [ebp-4],0FFFFFFFBh
    cmp     dword ptr [ebp-4],4
    jae     00000026 ; előjel nélküli szám esetén
    push    4
    mov     eax,dword ptr [ebp-4]
    push    eax
    push    offset $SG2459
    call    _printf
    add     esp,0Ch
    jmp     00000039
    push    4
    mov     ecx,dword ptr [ebp-4]
    push    ecx
    push    offset $SG2462
    call    _printf
    add     esp,0Ch
    xor     eax,eax
    mov     esp,ebp
    pop     ebp
    ret
```

Biztonságos-e az algoritmus?

```
21 int authenticate(char *username, char *password) {
22    int authenticated;
23    char buffer[1024];
24    authenticated = verify_password(username, password);
25    if(authenticated == 0)   {
26       sprintf(buffer,
27          "password is incorrect for user %s\n",
28          username);
29       log("%s", buffer);
30    }
31    return authenticated;
32 }
```

```
> echo tomi csillag | authenticate.exe
> echo tomi alma | authenticate.exe
> a.exe 992 A | authenticate.exe
> a.exe 993 A | authenticate.exe
> a.exe 997 A 4 B   -a 77136cf0  -n 1 -f shellcode.bin
    ↪ 10 X | authenticate.exe
```

Section 4

Jog

Jog és fejlődés

Információhoz való jog

Habár én is kötve érzem magam ahhoz, hogy minden embernek biztosítsuk szerzői jogainak élvezetét, a tudományok fejlődése elé senki sem gördíthet akadályokat.

Lord Ellenborough

Forrás

- Dudás Ágnes "A szoftver szerzői jogi védelme 1." http://www.sztnh.gov.hu/hu/kiadv/ipsz/200504/01-dudas-agnes.html
- Dudás Ágnes "A szoftver szerzői jogi védelme 2." http://www.sztnh.gov.hu/hu/kiadv/ipsz/200506/01-dudas-agnes.html
- Mezei Péter "Mitől fair a fair? Szerzői művek felhasználása a Fair-Use teszt fényében" http://www.sztnh.gov.hu/kiadv/ipsz/200812-pdf/02-mezei.pdf

Mi is az a szoftver

Szoftver kategorizálása funkciójuk alapján

- operációs rendszerek;
- eszközmeghajtó programok;
- információt közlő;
- információt feldolgozó;
- tudományos számítás végző;
- szórakoztató;
- stb.

Szakirodalomban definiálva

- A szoftver a hardverre épített inteligencia.
- A szoftver magja, az operációs rendszer azoknak a vezérlő programoknak a gyűjteménye, amelyek a gép folyamatos üzemeltetését biztosító alaptevékenységeket látják el.
- A szoftver a számítógépi programok, eljárások, szabályok és az ezekre vonatkozó dokumentáció összessége. Olyan szellemi termék, mely a hardvert működteti.
- A rendszer minden olyan komponense, amely nem tartozik a hardverhez.

Szoftver részei jogi szemmel

1. *Számítógép program*, azaz: olyan parancsok (utasítások) sorozatát, amelyet egy gépi olvasásra alkalmas hordozóra átvéve elérhetjük, hogy egy - információ feldolgozásra képes- gép meghatározott műveletet, feladatot, eredményt jelezzen, kivitelezzen vagy végrehajtson (elérésre bírjon).
2. *Programozói dokumentáció*: ez egy eljárás átfogó ismertetése (szóban, sematikusan vagy egyéb módon) elegendő részletességgel ahhoz, hogy egy meghatározott számítógépi programot alkotó utasítássorozatot létrehozzunk.
3. *Kiegészítő leírások*: mindazon dokumentáció, amely a program megértését és alkalmazását segíti. (Értelemszerűen nem tartozik ide a program maga, sem annak leírása.)

Varró Dániel – Email

Hát el vagyok egészen andalodva,
és gyönge szívem, ímé, reszketeg,
mióta éjjelente, hajnalonta
veled titokban ímélezgetek.

```
void sort(char *array, size_t size, int (*cmp)(void*,void*), int begin, int end) {
  if (end > begin) {
    void *pivot = array + begin;
    int l = begin + size;
    int r = end;
    while(l < r) {
      if (cmp(array+l,pivot) <= 0) {
        l += size;
      } else {
        r -= size;
        swap(array+l, array+r, size);
      }
    }
    l -= size;
    swap(array+begin, array+l, size);
    sort(array, size, cmp, begin, l);
    sort(array, size, cmp, r, end);
  }
}
```

Irodalmi mű és a szoftver

BUE (1886) az irodalmi, művészeti, és tudományos művek védelméről szóló Berni Egyezmény

TRIPS-egyezmény (1994) a számítógépi programok, mindegy, hogy forráskódban vagy gépi kódban kerülnek kifejezésre, a Berni Egyezmény alapján irodalmi műként élveznek védelmet.

Nem esik szerzői jog alá

- ötlet;
- elv;
- elgondolás;
- eljárás;
- működési módszerek;
- matematikai módszerek;
- interface-k;

A számítógépes program kidolgozásának a részei

1. Az elektronikus számítógépi kezelésre alkalmas feladat feltárása.
2. A feladatnak az elektronikus számítógép által megkövetelt korrektséggel történő megfogalmazása (ún. szakmai modell készítése).
3. A feladat számítástechnikai (matematikai) modelljének a megkonstruálása.
4. A számítási algoritmus elkészítése (a gépi eljárás kidolgozása).
5. Az algoritmus alapján a számítási program elkészítése (valamely gépre orientáltan).
6. A számítási programhoz szükséges adatok biztosítása.
7. A gépi számítások elvégzése.
8. A számítás eredményeinek ellenőrzése

A számítógépes program részei

1-4 együttesen szellemi alkotás

2,3,4 önmagában szellemi alkotás

Szoftver és a zene jogi szabályozása

Mind a két esetben az értelemzéshez speciális közegre van szükség.

- átdolgozás joga a szerzőé;
- részek felhasználását a szerző szabályozhatja;
- a felhasználó kötelezheti a szerzőt a műve használhatóvá tételére;
- ha erre a szerző nem hajlandó akkor a felhasználó más úton megoldhatja a szoftver szükséges módosítását.

Információ szabadsága avagy a fair use teszt

1. a használat célja
   - kereskedelmi
   - nonprofit oktatási célú;
2. mű természete – van-e minimális művészeti, eszmei értéke ;
3. a felhasznált résznek a mértéke és lényegessége;
4. a felhasznált mű potenciális piacára vagy értékére kifejtett hatása.

Accolade vs Sega

- Sega Genesis játékkonzol (hardware);
- Accolade játékok a konzolra (software);
- Sega trademark security system, TMSS, a távol-keleti hamisítások megfékezésére;
- Accolade programja az újabb vason nem fut, ezért a TMSS-t visszafejtették;
- Sega bírósági pert kezdeményez;
- Accolade védekezése:
  - Sega programok visszafejtése nem tekinthető jogellenesnek;
  - fair use teszt.

Kereskedelmi szoftverek

- célja a haszonszerzés;
- vásárlónak be kell tartani a licenc szerződést;
- maximum 1 biztonsági másolatot készíthető;
- a program megrendelőjének joga van a forráskódba beletekinteni, de ez nem gyakorlat (megrendelt sw. szavatossági problémák miatt) ;
- lehetséges részek visszafejtése saját program együtműködése érdekében.

Speciális licenszelésű kategoriák kialakulása

1. amerikai kormány által finanszírozott szoftverfejlesztések (Public Domain);
2. programozó idealizmusa – majd felfigyelnek rám;
3. segédprogramok – ingyenes elérhetőségével a párhuzamos fejlesztéseket megsporolhatók;
4. reklám;
5. Berni Egyezmény előtti programok nem védettek.

Kategóriák

Freeware Kereskedelmi cég ingyenes szoftvere bocsájt ki a cég népszerűbbé tétele érdekében (Acrobat Reader).

Postcardware Ez is freeware, de itt egy levelezőlapot, vagy egyebet illik a szerzőnek küldeni. (népszerűség mérése, Guinness rekord felállítás)

Shareware Próbáld ki vásárlás előtt. A programban beépített korlátozások vannak.

Trialware Majdnem olyan mint a shareware, de valamilyen program vásárlása után adják, és szabadon nem terjeszthető.

Limited edition Legjobb részek ki vannak szedve. Általában csak azokat hagyják benne, amelyre létezik ingyenes alternatív megoldás.

Kategóriák

Patcware Ingyenes javítások a már megvásárolt programhoz.

Ad-powered Ingyen jut hozzá a felhasználó, de utána nézheti a sok reklámot.

Thankyouware Honlap látogatása fejében adnak egy programot.

Abandonware Régi programok, játékok ajándékba.

Free software

1. bármilyen céllal futtatható;
2. forráskód rendelkezésre áll;
3. másolható;
4. program tökéletesítése, módosítása az egész közösség javát szolgálja.

Free software licenc

- BSD licencek;
- Mozilla licencek (bizonyos részeket kereskedelmi szoftverbe is be lehet építeni);
- Artistic licencek korlátozza az átdolgozás jogát (perl).