ציון: 89
(הציון הכי גבוה במועד)

מעבדת לתכנות מערכת 2021/2

גרסא ג

ענו על כל השאלות, **במקום המוקצב לכך בלבד**. חומר עזר: מותר.

בסעיף "אמריקאי" (Multiple choice) סמנו **בצורה ברורה** עיגול סביב האות הנכונה, 5 נקודות לתשובה נכונה, ועל תשובה לא נכונה 1- נקודות. על תשובה חסרה תקבלו 0 נקודות. בכל מקרה הניקוד של שאלה שלמה לא יהיה קטן מ-0. הניחו שהכל רץ על LINUX במוד של 32 סיביות אלא אם נאמר אחרת.

שאלה 1: נושאי SYSTEM, קלט-פלט, ותהליך יצירת קובץ ההרצה. (25%)

סעיף 1: מה דרוש ל-Linker בקבצי OBJECT כדי שיוכל ליצר קובץ EXEC לתכנית משתמש?

א) המיקום הסופי של התוכנית בדפי הזיכרון הראשי (הפיסי).
ב) SYMBOL TABLE
ג) RELOCATION TABLE
ד) PROGRAM HEADER
ה) כל התשובות הקודמות נכונות.
ו) תשובות ב' ו-ג' נכונות.

סעיף 2: ידוע כי cat משרשר מספר קבצים ומדפיס אותם ל stdout. מה יודפס אחרי ביצוע השורות הבאות?

```
cd ~
cat /bin/ls  /bin/ls  /bin/ls > cthulu
chmod u+x cthulu
cthulu
```

א) segmentation fault
ב) רשימת הקבצים ב-directory הנוכחי שלוש פעמים.
ג) רשימת הקבצים ב-directory ב-HOME DIRECTORY של המשתמש הנוכחי.
ד) רשימת הקבצים ב-directory של /bin/ls/.
ה) /bin/ls /bin/ls /bin/ls/
ו) יודפס רק cthulu

סעיף 3: נתונה התוכנית הבאה, המופעלת עם command line argument של myfile, שהוא שם קובץ שכבר קיים וניתן לקריאה וכתיבה:

```
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char** argv){
        FILE * file1 = fopen (argv[1],"r");
        int i;
        printf("hello");
        fputs(" there\n",file1);
        while((i=fgetc(file1))!=EOF) printf("%c",i);
        fclose(file1);
}
```

הפלט שיראה על המסך הוא:

a) hello there
b) hello segmentation fault
c) hello
d) hello there there
e) sementation fault
f) other (fill in):

1) **סעיף 1:** ה- Linker בין היתר קובע את הכתובות הסופיות ולכן גם לא א'. הלוא ב-א' שלנו עם ה' שלנו. ה- Program Headers אינם חובה (אנחנו אין באלו) ולכן גם ב' נפסל. ברור שה- Linker בקוק ל- symbol table לכן ג' נכון, ראה לו שגם ג' נכון ולכן ד' נכון...

סעיף 2: סעיפים ה' ו-ד' אינם אופלו ראויים להתייחסות. כל העסק סובב סביב הטיפה... לדעתי אם שהיו משאירה אותנו בתיקייה הנוכחית או שמעבירה אותנו ל- Home Directory של ה- user הנוכחי. אם נשארנו עם ה' או ג'. נשים לב ש ד' לא נכון כי בעצם שאר נו אנו נמצאים לא בתיקיה הנוכחית בה אנו נמצאים ולא לפי התיקיה בה היו הולכים. אבל רגע: ה' אומר שנלכים שוב פעמים?! מעניין לא... רק פעם אחת, ולכן הטיפה מובילה ל- Home Directory ... :)

סעיף 3: קודם כל הדפסנו "hello" ל- output הסטנדרטי לכן סעיף ה' נפסל. הוות ונתון כי שלנו השורה קריאה וכתיבה ל- myfile אז בשורה fputs(...) לא תיכשל. אבל! אנו מנסים את השורה there לקבל myfile ולא ל- output הסטנדרטי (אנו מתבוננים רק בשלב הזה), ולכן ד' שני (נסיון בלבד). כמו שהוסבר קודם לכן ב' שגוי. אבל myfile אינו רק קיים לכן גם ג' לא נכון. אבל מה בדיוק? הבעיה היא שאנו לא בודקים אם fputs מוסיף מחרוזת הסף לה ההבדל בין הקבל או מקבל את תכולת הקבל במחרוזת. מה? כי מה שקורה כאן אי ל-ד'. אבל בגלל אחורה. נשים לב שאנו משנה שקבל myfile קוראים לך הסף או קוראים מהר... קריאה מכתחילה לקריאה יש כשלת קריאה וכתיבה, בגלל שפתחת אותו אפילו שפתחנו כלבד (שורה fopen(...,"w")). נסיון כתיבה אליו יחד עם ההסבר מקודם אותו כ- עוצם לפשוט ונכנס ל- seg fault. נקבל: hello output הסטנדרטי (מסך) ולאחריו segmentation fault ... אבל סעיף ב' אינו נכון ... ניתוח לפחות שמחה לא לה בדמן המבחן נגו מאוחרי. :)

סעיף 4: נתון הקובץ hello.c הבא:

```
#include<stdio.h>
int main( ) { printf("Hello world\n"); }
```

עתה מבצעים את סדרת שורות הפקודה:

```
gcc –o blah hello.c
chmod  477 hello.c
./blah > hello.c
```

התוצאה של שורת הפקודה האחרונה היא:

א) יודפס Hello world למסך.

ב) יודפס Hello world לקובץ בשם hello.c

ג) תתקבל שגיאה: permission denied על קובץ hello.c

ד) תתקבל שגיאה: segmentation fault

ה) תתקבל שגיאה: cannot execute על קובץ blah

סעיף 5:

קובץ ה makefile הבא הוכן עבור קבצים task1.c,task1.h, task2.c,task2.h:

```
Run: task1.o task2.o
        gcc -m32 -o Run task1.o task2.o
task1.o:  task1.c  task1.h
        gcc -c -m32 -ansi -Wall task1.c
task2.o: task2.c
        gcc -c -m32 -ansi -Wall task2.c
```

לאחר הרצת make, בהנחה שלא היו בעיות קומפילציה וקישור בקבצים, מה מהבאים נכון:

1) מאחר ו task2.h לא מקומפל, לא יווצר קובץ task2.o ולכן גם לא יווצר קובץ executable

2) מאחר וה linker לא מקשר את task1.h ו task2.h, לא יווצר קובץ executable

3) יווצר קובץ executable בשם Run.

4) task2.o לא יווצר, אלא אם כן הפונקציה main( ) נמצאת ב task1.c

שאלה 2: תכנות ב-C וקריאות מערכת (45%)

(נתונים לסעיפים 1, 2) שמוליק הסטודנט המוכשר ממעבדה 4, החליט להריץ את שורת הפקודה: test > 3

כאשר תוכנית test נוצרה מתוכנית C הבאה:

```
#define BUF_SIZE 1024
#define DT_REG 8
struct linux_dirent {
      unsigned long  d_ino;     /* Inode number */
      unsigned long  d_off;     /* Offset to next linux_dirent */
      unsigned short d_reclen;  /* Length of this linux_dirent */
      char           d_name[];  /* Filename (null-terminated) */
/*    char           pad;      // Zero padding byte
      char           d_type;   // File type (only since Linux 2.6.4);*/};

int main(int argc, char *argv[])
{
   int fd, nread;   unsigned char buf[BUF_SIZE];
   fd = open(".", O_RDONLY | O_DIRECTORY);
   nread=syscall(SYS_getdents,fd,buf,BUF_SIZE); /*getdents system call*/
   write(1,buf,nread);
   return 0;
}
```

סעיף 4: אותו הומצא את hello.c בקבץ blah. אח"כ אנו אוסרים גישה אל ה- רשת הנוכחי אל hello.c, למרות קריאה בפנים. אשאר המשתמש נותנת גישה מחדש הפעלנו את blah ועד כה הכל סבבה... אבל wait! אנו מצליחים להכניס את הקבץ שלך hello.c כך כה אסור! הנה הרי עוד read עבורנו (ה- רשת הנוכחי). לכן אז כ-ה גישות. ד כ-ה אור הגורם. אבל נשארנו עם ג' (שהוא גם הגוני ותשות מהספר בקבץ).

סעיף 5: אי לא נכון כי בשום מקום לא כתוב כי יש - הכרת מקומי גם את task2.h. נכון שהוא בפל שם אבל בכבוכל זאת, אך אין הכרת ופאו שבנרו. כ' שאו כי ה- Linker מחשר את הבאי ה: o.* נלו h.* , ד' בשוב לאו קשור ולכן ג' נכון (שנה ג'ב').

2) **סעיף 1:** sys_getdents היא הסקודה ה- 141. (תלבורת).

פלט HEXEDIT של קובץ 3 שהתקבל הוא:

```
000000  24 02 82 00 01 00 00 00 10 00 2E 00 00 00 00 04 $...............
000010  FB 51 DB 01 02 00 00 00 10 00 2E 2E 00 7E 79 04 .Q............~y.
000020  68 2B 04 01 03 00 00 00 10 00 38 00 47 4E 55 0A h+........8.GNU.
000030  64 2B 04 01 04 00 00 00 10 00 38 2E 63 00 CA 08 d+........8.c.¨.
000040  66 2B 04 01 05 00 00 00 18 00 65 6C 5F 71 75 73 f+........el_qus
000050  65 69 72 00 87 F0 96 08 67 2B 04 01 06 00 00 00 eir.....g+......
000060  10 00 38 2E 38 00 CA 08 69 2B 04 01 07 00 00 00 ..8.8.¨.i+......
000070  18 00 38 2E 38 2E 38 2E 38 2E 38 00 00 00 00 08 ..8.8.8.8.8.....
000080  6A 2B 04 01 08 00 00 00 10 00 74 65 73 74 00 08 j+........test..
```

סעיף 1: כמה קבצים בסה"כ היו ב- CURRENT DIRECTORY בזמן שהפקודה הורצה?

א) 6 ב) 7 ג) 8 ד) 9 ה) 10 ו) 11

סעיף 2: כמה קבצים "רגילים" היו ב- CURRENT DIRECTORY בזמן שהפקודה הורצה?

א) 3 ב) 4 ג) 5 ד) 7 ה) 8 ו) 12

סעיף 3: נתונה התוכנית הבאה:

```
main( ) {  if(fork( ) && (fork( ) | fork( ) )
                printf("Bingo!\n");  }
```

תיקון: && ,|| &

הפלט שיופיע על המסך יהיה:

א) Bingo!
ב) Bingo!
Bingo!
ג) 3 פעמים Bingo!
ד) 4 פעמים Bingo!

סעיף 4: מה יודפס בעת ביצוע הקוד הבא?

```
char * str="0124";
printf("%c%d", str[str[3]-str[2]+str[4]], str[str[3]-str[1]] );
```

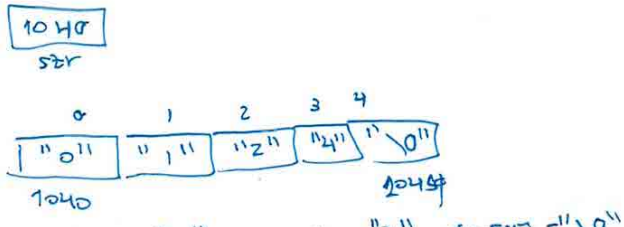
א) 324
ב) 252
ג) 31
ד) 32
ה) Segmentation fault

סעיף 5: במימוש ה- loader במעבדה 9, התבקשתם למפות את קוד ההרצה של הקובץ הנטען לכתובת בזיכרון. נכנה בשם length את מס' הבתים הנדרשים למיפוי קוד ההרצה, בהתאם לשדה המתאים בקובץ ה- ELF. בהנחה כי כתובת היעד במיפוי אינה כפולה של page size, כיצד יתבצע המיפוי בפועל?

1. כמות הבתים שיטען ה- loader גדולה מ- length. ריפוד (padding) יוכנס לפני הקוד.
2. כמות הבתים שיטען ה- loader גדולה מ- length. ריפוד (padding) יוכנס אחרי הקוד.
3. כמות הבתים שיטען ה- loader קטנה מ- length. הקוד ייחתך בהתחלתו.
4. כמות הבתים שיטען ה- loader קטנה מ- length. הקוד ייחתך בסופו.
5. כמות הבתים שיטען ה- loader שווה ל- length. הקוד יוסט לכפולה מתאימה של page size.

סעיף ב: אחרי ה-fork הראשון יהיו שני אב ובנו, אבל גם ה-לוק יהיה אפס ובה שקול ל- false. נשים לב שאחרי התקין משמש &&(ו&) & יחיד, משמע אם יש בצד false&& אז הביטוי לא יחושב ויוחזר ישר false. מסקנה: הבן לא נכנס ל-if. עכשיו לאבא נגלה בן שני. אבל האבא ה-לוק>0, ואבל הבן 0=לוק. ברור שהאבא ייכנס ל-if, אבל שפני כן הוא יבצע fork שוב! אז הן האב והבן השלישי (שנוצר עתה) "פנסן ל-if. הבן השני יבצע fork() ואז גם הוא ייכנס. לאמא, הנכד לא נכנס. כמות הכניסות ל-if = כמות ההפסות לפחות 2 נכונה.

סעיף 4: char* str="0124"



str[3] = "4", str[2] = "2", str[4] = "\0"

str[52 - 50 + 0] = str[2] = "2" = 50

אז התוצאה "נדפס "2" אין "seg fault"
ולכן א', ב', ג' שווים.

str[3] - str[2] = "4" - "2" = 52-49=3

אבל str[3] = "4" = 52

ולכן כי הוא הנכון. (סוף סוף א' ב')

סעיף 5: כי ו-ג' לא ביוניות, ד' אולי...
אבל עדיין אי משמע יותר מדי...
ה' הוא דומה ל-א' אבל לא ביוניכ"כ,
אז כנראה לא גם לא ה'. נשארנו עם
א' ו-ד'. ד' לא ביוני כי אז אפשר
נוסף להריץ את התכנית, א' לא
ביוני כי הוא בכתבני אף יותר
מ-ה' ונשאתו כבר הפלנו כל
בכתבנות. אוקי... לפי תומר הקרטוה
המעבדה 9 פה א' אוב'. ב' הרי
לא כ"כ ביוני ולכן נפחר ב-א'...

סעיף 6: הורצה תוכנית שקוד המקור שלה ב- C הוא:

```
#include <stdio.h>
int main( ) {
        FILE* src = fopen("source","r");
        FILE* dst = fopen("destination","w");
        char c = (char) fgetc(src) ;
        while ( c  != EOF ) {  fprintf(dst,"%c",c); c = (char) fgetc(src) ;}
        fclose(src);      fclose(dst);      return 0;
}
```

נניח כי הקבצים source ו- destination נפתחו, נקראו, נכתבו ונסגרו בהצלחה, ללא בעיות הרשאה.
מהי הטענה הנכונה ביותר מבין הבאות:

א. תוכן destination יהיה תמיד זהה לחלוטין לתוכן source.
ב. תוכן destination יהיה תמיד שונה לגמרי מתוכן source.
ג. אורך הקבצים בסיום הריצה יהיה תמיד זהה.
ד. אף אחת מהתשובות הקודמות אינה נכונה

סעיף 7: מה יהיה הפלט של התוכנית הבאה:

```
void main() {
   int x1;
   int* x2 = (int*)malloc(sizeof(int));
   int def1 = (int)(&x1) - (int)(&x2);
   int def2 =  (int)(&x1) - (int)(x2);
   if (def1 > 0)
      printf ("a");
   else
      printf ("b");
   if (def2 < 0)
      printf ("c");
   else
      printf ("d");
   free(x2);
}
```

1. ac
2. ad
3. bc
4. bd
5. segmentation fault

סעיף 8: מה יהיה הפלט של התוכנית הבאה (הפונקציה printf מחזירה את מספר התווים שנפלטו)?

```
void main()
{ int p =    ( printf("a") & printf("b")  )  ||
             ( printf("c") && printf("d")  );
}
```

1. a
2. ab
3. abc
4. abcd

סעיף ו: אני לא מבין איך fprintf מוסיפה אז מקום בכה...
אם מתייחס אל לה ב' ואחרת בהנחה א' ולא ב' ו"הכי
נכון" מגיע לה א'... אבל רגע! אנו לא יודעים אם ב- stdout היה תכן
קודם לכן, אם היה תכן ("אנוך", תמיד שתכן, אפילו בתכן הריק)
אז א' ו-ב' שווים גם אם fprintf מוסיפה! אבל לא וודאי וכן גם
א' ו-ב' ובכן ב' נכון.

סעיף ז: הייתי אומר "לא" (סדר הגדרת המבנים, אולי). **אבל** ל- x:
variable x has not been initialized
חכם הייתי אומר שלה בכלל לא יתקמפל... הכי קרוב לה
seg fault.

סעיף ח: & ישיר גם אם אצל אחד הוא false (כאן לא משנה מה), נפיס "ה" ואח"כ "ל" ואח"כ יש לנו **בשני** || & אבל
עדיר || מספיק אצל אחד שהוא true ולכן נקבל שעברנו כאן והתרך
של ק יהיה 1.

סעיף 9: נתונה התכנית:

```
#include <stdio.h>

char x[] = {0, 1, 2, 3, 4, 5};
void bar(char *foo) {
  int i;
  for (i=0; i<sizeof(foo); i++)    printf("%d",foo[i]);
}
void main( ) {   bar(x); }
```

הפלט הצפוי מהרצתה הוא:

א) 0 ב) 01 ג) 0123 ד) 01234 ה) 012345

שאלה 3: מבנה קבצי ELF (30%)

בכל סעיפי שאלה זו יש להתייחס לקובץ ELF לפי HEXEDIT בדף המצורף. טבלת הסימבולים מתחילה ב-OFFSET של 0x0238, וטבלת שמות ה-sections (ה-.shstrtab) מתחילה ב-0x0096 בקובץ, ה- ENTRY POINT הוא: 0x8048066, וה- SECTION HEADER TABLE מתחיל ב-OFFSET של 208 (האחרון נתון ב-DECIMAL).

סעיף 1: איזו מהתשובות הבאות מכילות רק שמות סימבולים שאינם SECTIONS מקובץ זה?

א) Hamas Tahrir Iran
ב) Mursi Egypt Syria
ג) Mursi Lupus Sisi
ד) ELF Argo Hizbala

-

סעיף 2: באיזה section נמצאת מחרוזת התווים Tahrir?

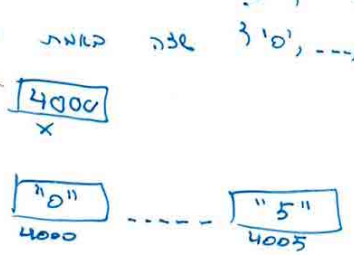
א) Egypt
ב) Tahrir
ג) Hizbala
ד) .text

סעיף 3: מה ערכו ההתחלתי של המשתנה Mursi? (לא ערך הסימבול!)

א) 0
ב) 0x8048066
ג) 0x34
ד) 0xffffffff, כלומר 1-

סעיף ב: foo הוא char* ולכן קוראים לו. קודם כל השאלה אם מדפיסה (בצדק) כי {5,...,50} זה מערך של int ולא של char והתווים שמתאימים ל-5,...,0 הם כל מיני סימנים שלא מדפיסים ולא נראים בכלל בקורס! אז אני מניח שהכוונה הייתה לו: {'5',...,'0'} שזה באמת מערך char-ים.

אז:



דבר ש: sizeof(foo) זה sizeof(int), וזה 4.

אוקי, אז עכשיו אני מודה בי... אכן הכוונה היתה ל- {5,...,0} ולא ל- {'5',...,'0'} כי אנו מדפיסים אבי לא, ואז מהות התווים כבר לא תשובה, לכן יוצגו המספרי ארבעת התווים הראשונים שזה 1234.

ג) **סעיף 1:** Tahrir הוא section לכן לא א'. Egypt כנ"ל - גם ב'.

ג: ה-ELF אי קשור. אז ב'. (שגם).

סעיף 3: כי לא קשור... נסך בכתובת 0x804805C שם נשמר ערך המשתנה Mursi וזה הרי offset של 5C וזה 00000000 ולכן א'.

סעיף 2: ה-section השני שם Egypt והוא מתחת ל-offset 0x0054 שזה המקום של Tahrir...

סעיף 4: מה צריך להיות כתוב במקום ה- XX XX XX XX בקובץ? (סימנתי סעיף ב' ואז א')

א) 66 80 04 08
ב) D0 00 00 00
ג) 94 80 04 08
ד) 94 00 00 00

סעיף 5: לאיזו כתובת וירטואלית לטעינת התוכנית מצוינת ב -PROGRAM HEADER הראשון?

א) 0
ב) 0x00000034
ג) 0x08048000
ד) 0x08048066

סעיף 6: כמה SECTION HEADERS יש בקובץ?

א) 1 ב) 3 ג) 7 ד) 9

בהצלחה!

סעיף 4: 0x80 48066 ולכן בריבוי הוא כתוב 66 80 04 08
וכן אפשרות ג' לא!!! התבלבלתי ותפסתי שאנחנו מדברים!!!
האמת היא שאנחנו אותנו ה- section header offset, שהוא

כי $(208)_{10} = (D0)_{16}$

$D = 13$

ואכן $13 \cdot 16 = 160 + 48 = 208$

ולכן כ' נכון.. *phew* *sigh* , כמעט הפסדתי 6 נק'...

סעיף 5: לפי הקבל .. אין מה להסביר

סעיף 5: סתם עידוד כללי: $(34)_{16} = (48+4)_{10} = 52$

ניגש ל-offset של $(34)_{16}$ שם שמת הפרוגרם הדרס...
אבל כל שורה $(20)_{16} = (32)_{10}$
והשדה השני הוא ה- vaddr ...

```
00000000   7F 45 4C 46  01 01 01 00  00 00 00 00  00 00 00 00  .ELF............
00000010   02 00 03 00  01 00 00 00  66 80 04 08  34 00 00 00  ........f...4...
00000020   XX XX XX XX  00 00 00 00  34 00 20 00  01 00 28 00  ........4. ...(.
00000030   09 00 06 00  01 00 00 00  00 00 00 00  00 80 04 08  ................
00000040   00 80 04 08  96 00 00 00  96 00 00 00  04 00 00 00  ................
00000050   00 10 00 00  54 61 68 72  69 72 0A 00  00 00 00 00  ....Tahrir......
00000060   01 00 00 00  90 90 B8 04  00 00 00 BB  01 00 00 00  ................
00000070   B9 54 80 04  08 BA 08 00  00 00 CD 80  E8 01 00 00  .T..............
00000080   00 90 A1 8E  80 04 08 BB  00 00 00 00  CD 80 01 00  ................
00000090   00 00 00 00  00 00 00 2E  73 79 6D 74  61 62 00 2E  ........symtab..
000000A0   73 74 72 74  61 62 00 2E  73 68 73 74  72 74 61 62  strtab..shstrtab
000000B0   00 45 67 79  70 74 00 41  72 67 6F 00  54 61 68 72  .Egypt.Argo.Tahr
000000C0   69 72 00 53  79 72 69 61  00 48 61 6D  61 73 00 00  ir.Syria.Hamas..
000000D0   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  ................
000000E0   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  ................
000000F0   00 00 00 00  00 00 00 00  1B 00 00 00  01 00 00 00  ................
00000100   02 00 00 00  54 80 04 08  54 00 00 00  10 00 00 00  ....T...T.......
00000110   00 00 00 00  00 00 00 00  01 00 00 00  00 00 00 00  ................
00000120   21 00 00 00  01 00 00 00  02 00 00 00  64 80 04 08  !...........d...
00000130   64 00 00 00  1D 00 00 00  00 00 00 00  00 00 00 00  d...............
00000140   01 00 00 00  00 00 00 00  26 00 00 00  01 00 00 00  ........&.......
00000150   02 00 00 00  81 80 04 08  81 00 00 00  0D 00 00 00  ................
00000160   00 00 00 00  00 00 00 00  01 00 00 00  00 00 00 00  ................
00000170   2D 00 00 00  01 00 00 00  02 00 00 00  8E 80 04 08  -...............
00000180   8E 00 00 00  04 00 00 00  00 00 00 00  00 00 00 00  ................
00000190   01 00 00 00  00 00 00 00  33 00 00 00  01 00 00 00  ........3.......
000001A0   02 00 00 00  92 80 04 08  92 00 00 00  04 00 00 00  ................
000001B0   00 00 00 00  00 00 00 00  01 00 00 00  00 00 00 00  ................
000001C0   11 00 00 00  03 00 00 00  00 00 00 00  00 00 00 00  ................
000001D0   96 00 00 00  39 00 00 00  00 00 00 00  00 00 00 00  ....9...........
000001E0   01 00 00 00  00 00 00 00  01 00 00 00  02 00 00 00  ................
000001F0   00 00 00 00  00 00 00 00  38 02 00 00  10 01 00 00  ........8.......
00000200   08 00 00 00  0D 00 00 00  04 00 00 00  10 00 00 00  ................
00000210   09 00 00 00  03 00 00 00  00 00 00 00  00 00 00 00  ................
00000220   48 03 00 00  49 00 00 00  00 00 00 00  00 00 00 00  H...I...........
00000230   01 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  ................
00000240   00 00 00 00  00 00 00 00  00 00 00 00  54 80 04 08  ............T...
00000250   00 00 00 00  03 00 01 00  00 00 00 00  64 80 04 08  ............d...
00000260   00 00 00 00  03 00 02 00  00 00 00 00  81 80 04 08  ................
00000270   00 00 00 00  03 00 03 00  00 00 00 00  8E 80 04 08  ................
00000280   00 00 00 00  03 00 04 00  00 00 00 00  92 80 04 08  ................
00000290   00 00 00 00  03 00 05 00  01 00 00 00  00 00 00 00  ................
000002A0   00 00 00 00  04 00 F1 FF  07 00 00 00  54 80 04 08  ............T...
000002B0   00 00 00 00  00 00 01 00  0C 00 00 00  5C 80 04 08  ............\...
000002C0   00 00 00 00  00 00 01 00  12 00 00 00  60 80 04 08  ............`...
000002D0   00 00 00 00  00 00 01 00  17 00 00 00  82 80 04 08  ................
000002E0   00 00 00 00  00 00 03 00  1D 00 00 00  8E 80 04 08  ................
000002F0   00 00 00 00  00 00 04 00  22 00 00 00  92 80 04 08  ........".......
00000300   00 00 00 00  00 00 05 00  2A 00 00 00  66 80 04 08  ........*...f...
00000310   00 00 00 00  10 00 02 00  31 00 00 00  96 90 04 08  ........1.......
00000320   00 00 00 00  10 00 F1 FF  3D 00 00 00  96 90 04 08  ........=.......
00000330   00 00 00 00  10 00 F1 FF  44 00 00 00  98 90 04 08  ........D.......
00000340   00 00 00 00  10 00 F1 FF  00 65 33 62  2E 73 00 4E  .........e3b.s.N
00000350   69 6C 65 00  4D 75 72 73  69 00 53 69  73 69 00 4C  ile.Mursi.Sisi.L
00000360   75 70 75 73  00 49 72 61  6E 00 48 69  7A 62 61 6C  upus.Iran.Hizbal
00000370   61 00 5F 73  74 61 72 74  00 5F 5F 62  73 73 5F 73  a._start.__bss_s
00000380   74 61 72 74  00 5F 65 64  61 74 61 00  5F 65 6E 64  tart._edata._end
00000390   00
```