درس ۵

دسترسی به دیسک و سی دی رام (مفاهیم)

اگر قرار است بای پسها (برنامه‌های بار کننده) را در این آموزش بگنجانم، اجـازه دهیـد آنـها را در اینجـا قـرار دهـم،آمـاده کـردن یـک بـار کننـده بـرای پشـت سـر گذاشـتن محـافظت [Mario Andre] گاهی اوقات به منظور نفوذ به یک محافظت خاص از بردارها استفاده می‌شود. در این موارد یک روش خوب، آماده‌سازی یک برنامه بار کننده است. این نوع کراک کردن می‌تواند در اینترنت (روی برخی پیکربندهای fire wall (یا دیواره آتش) که در درس ۹ توضیح داده شد مـورد استفاده قرار گیرد.

برای مثال اجازه دهید «مسابقه ماریو اندرتی» را در نظر بگیریـم، یـک بـازی احمقانـه کـه در آن از همان طرح محافظتی که شما امروز روی برخی روتینهای دسترسی به سرورهای نظامی می‌بینید. استفاده می‌شود. برای این منظور شما باید بارکننده‌ای روی خطوط ذیل آماده کنید.

```
loc   code          instruction       what's going on

-------------------------------------------------

:0100 EB44          JMP 0146
```

```
...

:0142 0000          <- storing for offset of INT_21

:0144 5887          <- storing for segment of INT_21

:0146 FA            CLI

:0147 0E            PUSH CS

:0148 1F            POP DS

:0149 BCB403        MOV SP,03B4

:014C FB            STI

:014D 8C1EA901      MOV [01A9],DS      <- save DS

:0151 8C1EAD01      MOV [01AD],DS         three

:0155 8C1EB101      MOV [01B1],DS         times

:0159 B82135        MOV AX,3521        <- get INT_21

:015C CD21          INT 21                in ES:BX

:015E 891E4201      MOV [0142],BX      <- store offset

:0162 8C064401      MOV [0144],ES      <- store segment

:0166 BA0201        MOV DX,0102

:0169 B82125        MOV AX,2521        <- set INT_21 to

:016C CD21          INT 21                DS:0102

:016E 0E            PUSH CS

:016F 07            POP ES             <- ES= current CS
```

```
:0170 BBB403         MOV BX,03B4

:0173 83C30F         ADD BX,+0F

:0176 B104           MOV CL,04

:0178 D3EB           SHR BX,CL          <- BX= 3C

:017A B8004A         MOV AX,4A00        <- Modify memory block

:017D CD21           INT 21                to 3C paragraphs

:017F BA9E01         MOV DX,019E        <- ds:dx=program name

:0182 BBA501         MOV BX,01A5        <- es:bx = param. block

:0185 B8004B         MOV AX,4B00        <- load ma.com

:0188 CD21           INT 21

:018A 2E8B164201     MOV DX,CS:[0142]   <- reset old int_21

:018F 2E8E1E4401     MOV DS,CS:[0144]

:0194 B82125         MOV AX,2521

:0197 CD21           INT 21

:0199 B8004C         MOV AX,4C00        <- terminate with return

:019C CD21           INT 21                code

:019E 6D612E636F6D00 "ma.com"

      0000           fence

:01A7 B2015887

:01AB B2015887

:O1AF B2015887
```

```
      0000           fence

let's now prepare a routine that hooks INT_21:

push all

CMP AX,2500    <- go on if INT_21 service 25

JNZ ret

CMP Word Ptr [0065], C00B <- go on if location 65 = C00B

JNZ ret

MOV  Byte Ptr [0060], EB  <- crack instructions


MOV  Byte Ptr [0061], 3C

MOV  Byte Ptr [0062], 40  <- INC AX

MOV  Byte Ptr [0063], 90  <- NOP

MOV  Byte Ptr [0064], 48  <- DEC AX

pop all

JMP  FAR CS:[0142]  <- JMP previous INT_21
```

از حالا به بعد، هربار که یک برنامه در آدرس [0065]، شامل یک دستور OR AX, AX (کـه در ma.com قرار دارد) وقفه ۲۱ را فراخوانی می‌کند یک بردار بدام افتاده می‌شود (یا hook) می‌شـود. در آن صورت برنامه مقصد در آدرس [0060] قرار خواهد داشت. دستور JMP 3C کار را به پیـش خواهد برد. علیرغم این واقعیت کـه آن برنامـه، دارای روتنیـهایی اسـت کـه توانـایی چـک کـردن خودشان را به منظور اطمینان از عدم هرگونه تغییری خواهند داشت.

مهمترین چیز، روتینهایی است که شما برای فراخوانی وقفه 21 (یا هر وقفه دیــگری) از ســرویس ۲۵ (یا هر سرویس دیگری) به منظور کراک کردن و تخریب برنامه می‌نویسید. من یکی دیــگر را به شما نشان خواهم داد. این یکی Reach. Com نامیده می‌شود.

```
push all

CMP  AH,3D      <- is it service 3D? (open file)

JNZ  ret        <- no, so ret

CMP  DX,13CE    <- you wanna open file at 13CE?


JNZ  ret        <- no, so ret

MOV  AX,[BP+04] <- in this case

MOV  DS,AX

CMP  Byte Ptr [B6DA],74 <- old instructions

JNZ  015B

CMP  Byte Ptr [B6DB],0F <- ditto

JNZ  015B

CMP  Byte Ptr [B6DC],80 <- ditto, now we now where we are

JNZ  015B

MOV  Byte Ptr [B6DA],EB <- crack

MOV  Byte Ptr [B697],40 <- camouflaged  no-opping

MOV  Byte Ptr [B698],48 <- cam          nop

MOV  Byte Ptr [B699],90 <- cam          nop
```

```
MOV  Byte Ptr [B69A],40 <- cam          nop

MOV  Byte Ptr [B69B],48 <- cam          nop

MOV  DX,CS:[0165]

MOV  DS,CS:[0167]

MOV  AX,2521  <- set hook

INT  21

POP  all

JMP  FAR CS:[0165]
```

حالا شما دستورالعمل 740f در دستور EB0f را تغییر داده و دستورالعملها را در B697-B69B noop خواهید کرد. خوب اه دقیقتر از noop کردن آنها با «۹۰» بایت، اینست که به جای آن، توالی Dec Ax, Inc Ax, Nop, Dec Ax,Inc Ax را انتخاب کنید! استفاده از توالی دستورهای working به جای "NOP" دلایل منطقی دارد . طرحهای محافظت اخیر، nopهای متصل به یکدیگر را در داخل برنامه میفهمد. و در صورتیکه این طرحها، بیشتر از سه تا NOP متوالی بیابند,همه چیز را خراب میکنند! شما هم همواره باید سعی کنید تا هنگام کراک کردن، راه حل ایجاد کننده مزاحمت کمتر و مخفیتر را انتخاب کنید!

همچنین میتوانید، این نوع کراک کردن را، با خطوط یکجور، برای برنامه‌های زیادی بکار ببرید که خودشان، عمل کراک کردن کد را اجام داده و بردارها را بدام می‌اندازند.

روش اصلی دسترسی دیسک

حال به سراغ موضوع اصلی این درس می‌رویم:

طبق معمول از اول شروع میکنیم. تاریخ همواره کلیدی است که فهم موضوعات کراکینگ در حال و آینده را امکانپذیر میسازد. از زمانیکه دیسکهای فلاپی سیاه با قطر 5 1/4 اینچ مورد استفاده قرار گرفتند، یکی از رایجترین متدها برای حفاظت یک برنامه، فرمت کردن دیسک اصلی (کلیدی) به روشی غیرعادی بود. این فلاپی قدیمی برای PC(کامپیوترهای شخصی) معمولاً در ۹ سکتور (قطاع) در هر شیار، 360k داده ذخیره میکرد.

برای برخی از شما که یک سری چیزها را نمیدانید، به منظور اجتناب از این نوع کراک کردن لازم است دو چیز اساسی را بدانید.

بلوک پارامتر دیسک فلاپی (FDPB) و روتینهای وقفه‌ای که به فرمت کردن یا خواندن دیسک (در اصل وقفه ۱۳) میپرازند ,اغلب طرحهای حفاظتی، یا عمل فرمت را با سایز سکتور یا تراکی بیش از حد استاندارد (یعنی ۵۱۲ بایت) انجام میدهند، یا اینکه به یکی از سکتورها پهنای ۲۱۱ بایت میدهد.یا اینکه تراک کامل از سکتورهای 15/9/8 را فرمت نمی‌کند.

مثلاً اگر همان نسخه خیلی قدیمی برنامه اصلی ویزی کالک (Visicalc) را داشته باشید، خواهید دید که سکتور یا قطاع ۸ روی شیار ۳۹ کاملاً از بین رفته است. تحقیق در مورد اسمبلی یا استفاده اختصاصی به شما میگوید که کدامیک از شماره سکتورها، سایز آنها بر حسب بایت، تغییر داده شده‌اند و آیا با خطای CRC فرمت شده‌اند.

پارامترهای دیسک فلاپی در BIOS ذخیره میشوند. بردار وقفه 1E شامل آدرس بلوک پارامتر دیسک فلاپی است. محتوای اصلی FDPD عبارت است از :

Offset	Function	crackworthy?	Example
0	Step rate & head unload	no	DF
1	head load time	no	02

2	Motor on delay	no	25
3	Number of bytes per sector	yes	02
4	Last sector number	yes	12
5	Gap length	yes	1B
6	Data track length	yes	FF
7	Format gap length	yes	54
8	Format byte	no	F6
9	Head settle time	no	0F
A	Motor start time	no	02

۰) افست 0# : "nybble" سمت چپ (یک رقمی) ایـن عـدد، زمـان سـرعت عمـل هـدگردانـنده دیسک می‌باشد. "nybble" سمت راست، زمان تخلیه بار هددیسک است. ارقام سمت چپ بهترند.

۱) افست 1# : مجدداً با این مقادیر وقت خود را تلف نکنید . nybble سمت چپ زمان بارگذاری هد دیسک و nybble سمت راست. انتخاب مد دستیابی مستقیم به حافظه است.

۲) زمان انتظار تا زمانیکه موتور خاموش شود، که معمولاً استفاده چندانی ندارد.

۳) مقدار بایتها در هر سکتور : اگر برای این مقدار «صفر» را در نظر بگیرید، PC انتظـار دارد کـه تمام سکتورها ۱۲۸ بایت طول داشته باشند. عدد ۱ یعنی اندازه سکتور. ۲۵٦ بایت است. «۲» یعنـی اندازه سکتورهای ۵۱۲ بایت (اندازه استاندارد DOS) و «۳» یعنی، ۱۰۲٤ بایت در هر سکتور

٤) بیشترین عدد سکتور روی شیار: این عدد برای فرمت کردن استفاده می‌شود و به DOS میگوید که در هر شیار چند تا سکتور وجود دارد.

۵) طول شکاف بـرای خوانـدن دیسـک : وقتـی کـه سـعی میکنیـد تـا سـکتور دارای انـدازه غـیر استانداردی را بخوانید، دچار خطاهای CRC می‌شود و اینجاست کـه وقـت خـود را بیخـود تلـف میکنید. شما می‌توانید فرمت کردن را با وسیله U-format انجام دهید، در غیر این صورت از اینکار صرف نظر کنید.

٦) طول داده: طول داده شامل تعداد بایتها در یک سکتور است، البته در صورتیکه شـماره بـایت در جدول 4#، 2,1,0 یا ۳ نباشد.

۷) شماره بایتها در شکاف بین سکتورها: این هم زمانی اسـتفاده می‌شـود کـه شـیارهای خـاصی را فرمت میکنید.

۸) اولین بایت فرمت: هنگام فرمت کردن، این همان اولین بـایتی اسـت کـه در تمـام سـکتورهای جدید قرار داده می‌شود.

۹) زمان استقرار هد: که چندان مهم نیست.

A) زمان استارت موتور: چندان روی این موضوع هم زوم نکنید.

به منظور تغییر تعداد شیارها روی یک دیسک مشخص و تعداد شماره سکتورها در هر شـیار شـما میتوانید همیشه با سوئیچ فرمان DOS "t:/ و n:/ " فرمت کنید.

Format /t:track /n:sector

اگر مایلید ببینید که پارامترهای موجود کدامند [Debug-exe] یـا [symdeb.exe] را اجـرا کنیـد و فرمانهای زیرا را بدهید:

```
-    d 0:78    l 4                <- get FDPB address

 0000:0070     22 05 00       <- debugger's likely response

-    d 0:522   l a                <- get 10 FDPB values

 0000:520 DF 02 25 02 12 1B FF...  <- see preceding table
```

بخاطر داشته باشید که تمام فرمت دیسک استاندارد،تحت سیستم DOS، سکتور دارای سایز ۵۱۲ بایتی را ساپورت (حمایت) می‌کند. بدین ترتیب اندازه سکتور برای فلاپیهای یک طرفه ۵/۲۵ اینچی:

```
40t*8s*512b=163.840 bytes (160Kb)

40t*9s*512b=184.320 bytes (180Kb)
```

و برای فلاپیهای دو طرفه ۵/۲۵ اینچی:

```
40t*8s*512b*2sides=327.680 bytes (320Kb)

40t*9s*512b*2sides=368.640 bytes (360Kb)
```

پس اگر با مدل dos ۳/۰ شروع کنیم، به فرمت کردن دیسک فلاپی جدید کمک کرده‌ایم.

IBM AT(80286 cpu) ، معرف ظرفیت بسیار بالای دیسک فلاپی با قطر ۵/۲۵ اینچ است که در ۱۵ سکتور در هر شیار، ۱/۲ مگابایت ذخیره می‌کند.

```
80t*15s*512b*2sides=1.228.800 bytes (1.2Mb)
```

بعدها فلاپی‌های ۳/۵ اینچی مورد استفاده قرار گرفتند که در کاست پلاستیکی سفت و کوچکی قرار داشتند:

۳/۵ اینچ دوطرفه/ دانسیته دو برابر 720 k

۳/۵ اینچ دو طرفه / دانسیته چهاربرابر 1440k (HD)

۳/۵ اینچ دو طرفه/ دانسیته بالا 2880 k

به منظور ایجاد طرحهای کلی خارق العاده. حمایت گرایان (محافظه‌کارها) از وقفه 13h، سرویس 18h استفاده کنند که برای روتینهای فرمت سازی، تعداد شیار و شماره سکتور در هر شیار را مشخص می‌کند.

* ثبت کننده های ورودی : AH=18h ; CH=N0 شیارها ; CL سکتورهای هر شیار;DL=شماره درایوها B=1,A=0 ,...C=2 ... بیت ۷ در صورتیکه استفاده می‌شود که درایو مربوط به هارد دیسک باشد)

* ثبت کننده ها دربر گشت : DI=ادرس افست ازجدول پارامتر ۱۱ بیتی ;ES=ادرس قطعه از جدول پارامتر ۱۱ بیتی.

برای اینکه آنها را بخوانیم، باید از وقفه ۱۳ (INT 13)، سرویس ۲، استفاده نموده و سکتورهای دیسک رادر طرح ذیل بخوانیم:

* ثبت کننده‌های ورودی: AL=N0, AH=2h سکتورها ؛ BX = آدرس افست بافر داده؛

CH = شیار، CL = سکتور،DH شماره هد، DL= شماره درایو ؛ ES= آدرس قطعه بافر داده .

* ثبت کننده‌ها در برگشت: AH = کد برگشت. اگر پرچم یا نشان رقی نقلی(carry flag is not set) قرار داده نشود. AH=0 بنابراین سکتور خارق العاده خوانده می‌شود و اگر پرچم رقم نقلی(carry flag is set) بر عکس قرار داده شود، AH بایت وضعیت را بصورت ذیل گزارش می‌کند.

```
76543210  HEX  DEC       Meaning

1         80h  128       Time out - drive crazy

 1        40h  064       Seek failure, could not move to track

  1       20h  032       Controller kaputt
```

```
  1      10h  016       Bad CRC on disk read

   1     09h  009       DMA error - 64K boundary crossed

   1     08h  008       DMA overrun

    1    04h  004       Bad sector - sector not found

     11  03h  003       Write protect!

      1  02h  002       Bad sector ID (address mark

       1 01h  001       Bad command

[Return code AH=9: DMA boundary error]
```

یکی از خطاهایی که ممکن است ایجاد شود و در برخی طرحهای محافظت استفاده شود:

یکی از خطاها: خطاهای مرزی DMA است(ah=9)، یعنی یک مرز غیرقانونی، هنگ‌امیکه اطلاعات درون RAM ریخته میشوند از آنجا رد شود. DMA (دستیابی مستقیم به حافظه) در روتینهای سرویس دیسک مورد استفاده قرار می‌گیرد تا اطلاعات درون RAM ریخته شود. اگر آدرس افست حافظه که در پایان به ۳ تا صفر ختم می‌شود (Es:1000, Es:2000)، در وسط ناحیه‌ای قرار گیرد که توسط یک سکتور، جایگ‌ذاری شود، این خطا بوجود می‌آید (رخ میدهد) .

دیگر وقفه حفاظتی ممکن، وقفه 13h، سرویس ٤، است بازبینی سکتورهای دیسک است.

بازبینی دیسک، روی دیسک انجام شده و نیازی به بازبینی داده‌های روی دیسک، بر خلاف داده‌های موجود در حافظه، شیت! این عمل،هیچ ویژگی بافری ندارد، و یک دیسک را نه میخواند و نه می‌نویسد:

این باعث می‌شود تا سیستم داده‌های موجود در سکتور یا سکتورهای انتخاب شده را بخواند و علیرغم کنترل داده‌های ذخیره شده روی دیسک، بررسی توازن چرخه‌ای محاسبه شده را کنترل کند (CRC) . به وقفه 13، ثبات های AH=2 و گزارش خطا، توجه کنید.

CRC، جمع ارقام یا بیتهاست که خطاهای یکی را مشخص می‌کند. وقتیکه سکتوری روی دیسک نوشته می‌شود، CRC اصلی. محاسبه شده و همراه با داده‌های سکتور نوشته می‌شود. سرویس بازبینی، سکتور را میخواند، مجدداً CRC را محاسبه نموده و این CRC، را با CRC اصلی مقایسه می‌کند .

دیدیم که در برخی از طرحهای محافظت، سعی شده تا فراخوانیهای وقفه تغییر یابد. این امر بخصوص در طرحهای حفاظتی دستیابی دیسک که در آنها از وقفه ۱۳ استفاده می‌شود بسیار معمول است. اگر سعی میکنید تا چنین برنامه‌هایی را کراک کنید، مسیر عمل، سرچ کردن در مورد رخدادهای "CD13" است که همانا زبان ماشین برای وقفه ۱۳ می‌باشد. در روش دیگر، طرح محافظت، از این وقفه برای کنترل سکتورهای مخصوص دیسک استفاده می‌کند. اگر برنامه را بررسی کرده باشید، برنامه‌هایی را خواهید دید که در کدماشینشان، CD۱۳ وجود دارد، اما برنامه‌هایی وجود دارد که دیسک اصلی را به خاطر سکتورهای غیرعادیش، کنترل می‌کند، اما چگونه؟

تکنیکهای مختلفی وجود دارد که میتوان از آنها برای مخفی نمودن طرح محافظت استفاده نمود. حال ۳ تا از تکنیکهای معروفتر را برایتان توضیح میدهم:

۱) بخش کد ذیل، برابر است با صدور فرمان INT 13 برای خواندن یک سکتور از درایو A، طرف 0، شیار 29h، سکتور ffh، و سپس کنترل کد وضعیت 10h.

```
cs:1000   MOV  AH,02      ;read operation
```

```
cs:1002   MOV  AL,01     ;1 sector to read

cs:1004   MOV  CH,29     ;track 29h

cs:1006   MOV  CL,FF     ;sector ffh

cs:1008   MOV  DX,0000   ;side 0, drive A

cs:100B   XOR  BX,BX     ;move 0...

cs:100D   MOV  DS,BX     ;...to DS register

cs:100F   PUSHF          ;pusha flags

cs:1010   PUSH CS        ;pusha CX

cs:1011   CALL 1100      ;push address for next

                         instruction onto stack and branch

cs:1014   COMP AH,10     ;check CRC error

cs:1017   ... rest of verification code


...

...

cs:1100   PUSHF          ;pusha flags

cs:1101   MOV  BX,004C   ;address of INT_13 vector

cs:1104   PUSH [BX+02]   ;push CS of INT_13 routine

cs:1107   PUSH [BX]      ;push IP of INT_13 routine

cs:1109   IRET           ;pop IP,CS and flags
```

دقت کنید در کد منبع، هیچ فرمان INT 13 وجود ندارد، پس اگر برای سرچ کردن "CD13" در کد ماشین از یک عیب‌یاب (اشکال زدا) استفاده می‌کنید، هرگز روتین محافظت را پیدا نخواهید کرد.

۲) تکنیک دیگر، قرار دادن در دستورالعمل جایگزینی وقفه است. مثل وقفه ۱۰، که تا حدی، بی ضرر بوده و برنامه ای وجود دارد که ۱۰ را به ۱۳ تغییر میدهد. (و سپس به ۱۰ برمی‌گرداند). سرچ کردن یا جستجوی "CD13" نتیجه‌ای نخواهد داشت.

۳) بهترین روش استتار (اختفاء) برای وقفه‌هایی که من کراک می‌کردم، (البته نه در INT13) پرش به بخش کد برنامه (پروگرام) بوده که کد وقفه را مجدداً ایجاد می‌کند.

بارگذاری سکتورهای کامل دیسک

ابزار قدیمی [debug.com] «چاقوی ارتش سوئیس» کراکر نامیده می‌شود. چیزهایی نظیر بارگذاری،خواندن، اصلاح و بازنویسی سکتورهای کامل دیسکها را ممکن می‌سازد. شمارش سکتور با اولین سکتور شیار صفر شروع می‌شود. سکتور بعدی شیار صفر است. اگر دو طرفه باشد به طرف اول، شیار ۱ برمی‌گردد. الی آخر . تا پایان دیسک، تا بیش از 80h (128) را می توان در یک زمان بارگذاری کرد . برای استفاده، هر بار میتوان بیش از 80h سکتور را بارگذاری کرد. برای استفاده شما باید آدرس شروع، درایو (۰=A , 1=B, etc)، سکتور شروع و شماره سکتورها را مشخص کنید. -1 100 0 10 20

این دستورالعمل، به DEBUG ،میگوید که بارگذاری کند. آدرس شروع در DS: 0100. درایو شروع A، سکتور 10h برای سکتورهای 20h خواهد بود. این عمل بازیابی داده‌های فرمت شده مخفی یا عجیب و غریب را امکانپذیر میسازد. اگر خطا کردید، محل حافظه مربوط به آن داده را

پیدا کنید. اکثر اوقات، بخشی از داده قبل از وقوع خطا، انتقال داده و باقیمانده آن را میتـوان از روی فهرست راهنما، مجدداً ثبت یا جمع آوری کرد.

یادگیری کراکهای ذیل را همواره به خاطر بسپارید. حال میخواهیم،یک برنامـه اولیـه و قدیمـی را کراک کنیم:

شبیه ساز یا سیمیلاتور MS flight (مدل قدیمی 2/12 سال ۱۹۸۵).

این برنامه قدیمی استفاده شده در سال ۱۹۸۵، طرح محافظتی زیبای ذیل را الگو قرار میدهد. روی دیس، فقط یک stub بنام FS.com دارید که دستورالعملهای زیر را اجرا می‌کند:

```
loc   code           instruction        what's going on
-------------------------------------------------------
:0100 FA             CLI                ;why not?
:0101 33C0           XOR AX,AX          ;ax=0
:0103 8ED0           MOV SS,AX          ;ss=0
:0105 BCB0C0         MOV SP,C0B0        ;SP=C0B0
:0108 8EC0           MOV ES,AX          ;ES=0
:010A 26C70678003001 MOV Wptr ES:[0078],0130 ;Wp 0:78=130
:0111 268C0E7A00     MOV ES:[007A],CS   ;0:7A=Segment
:0116 BB0010         MOV BX,1000        ;BX=1000
:0119 8EC3           MOV ES,BX          ;ES=1000
:011B 33DB           XOR BX,BX          ;BX=0
:011D B80102         MOV AX,0201        ;AH=2 AL=1 sector
:0120 BA0000         MOV DX,0000        ;head=0 drive=0
:0123 B96501         MOV CX,0165        ;track=1 sector=65 (!)
:0126 CD13           INT 13             ;INT 13/AH=2
:0128 B83412         MOV AX,1234        ;AX=1234
:012B EA00000010     JMP 1000:0000      ;JMP to data we just read
:0130 CF             IRET               ;Pavlovian, useless ret
```

میتوانید حدس بزنید که در این طرح محافظت قدیمی چه اتفاقی افتاده است؟ همان اتفـاقی کـه در بیشتر طرحهای امروزی می‌افتد:

جستجوی حفاظت برای سکتور فرمت شده غیرعادی و یا جستجوی داده‌های خـاص بـرای شـما، چندان مشکل نیست:فقط باید هرچیزی را بر عکس کنید، داده‌های عجیب و غریب را به هم وصـل نموده و بالاخره معجونی که دوست دارید، توضیحات بیشتر در مورد طرحهای قدیمی محافظت، از آنها بسادگی نگذرید. برخی از آنها عبارتند از :

```
-- CLEVER
-- STILL USED
-- DIFFICULT TO CRACK... I mean, this older DOS programs had
```

منظورم اینست کـه برنامـه‌های قدیمـی ترسـیم DOS حفاظتـهای خوبـی داشـته‌اند. کـراک کـردن برنامه‌های ویندوز که نیاز به شماره ثبت دارد، تا حدی آزاردهنده اسـت : همـانطور کـه در درس ۳ ملاحظه کردید، شما باید نام و شماره سریال انتخاب خود را، که عبارت است از "666666666" را تایپ کنید. آن را وارد برنامه WINICEکنید، سرچ کنید و برای ارزیابی بهتر،نـام خودتـان را سـرچ کنید، نقطه توقف خواندن حافظه را جایی مشخص کنید که شماره مورد نظر در آنجا وجـود داشـته و بدنبال کدی بگردید که ورودی شما را دستکاری می‌کند. همانگونه که [Chris] بدرستی اشـاره می‌کند، شما میتوانید، کد را مستقیماً از برنامه بیرون کشیده (خارج نموده) و ژنراتور (مولد) کلیـدی ایجاد کنید که کد معتبری ایجاد کند. این کدبرای هر نامی که شما در طرحهای حفاظت «دسـتکاری ریاضیات مطلق» تایپ میکنید کار خواهد کرد و یا متقابلاً پس از هـر نـامی کـه شـما در آن تـایپ کنید، طرحهای حفاظت «دستکاری بسته» (مانند mod4win بـه درسـهای ویندوز مراجعـه کنیـد). طرحهای خاصی خواهند بود . در چنین وضعیتی pseudo-random xoring (شبه تصادفی)

Stunning: ایده های جدید همیشه به ندرت به وقوع میپیوندند، و در این دنیای رنجش آور کند و آهسته، نادرتر و کمیابتر هم میشوند، برنامه نویسان ناتوان، با برنامه های ناقصی مثل ویندوز ۹۵ از ما حمایت میکنند. ... بله، مثل همیشه، با نظر کاملاً مخالف میگویم، که هیچ پیشرفتی وجود ندارد. یک قدم به عقب برگردیم.

یک مارتینی ـ و ودکای خوب بنوشید (بخاطر داشته باشید کـه فقـط قطعـات یـخ:، مـارتینی درای، و طعم لیموی Tuskany، زیتــون سـبز از Schweppes «تونیـک هنـدی» Wodka Moskovskaja مالتی.این نوشیدنی را کامل خواهند کرد) و با چشمانی پاک، از بالکن منزل خود، به تماشای شهر و مردم اطراف خود بپردازید. کارگرها که همه جا ساعت ۷/۵ صبـح از خانـه بـیرون مـی آینـد، و در تصاعدی از ماشینهای یکجور گم شده اند، مجبورند تا ساعتها تابلو آگهیها را تماشا کنند و یک ریـز به جار و جنجالهای آنها برای تبلیغات گوش کنند، در طول روز، به خاطر رفتن سر کار خوشـحالند به منظور اینکه ماشینهای دیگری تولید کنند، تا بتوانند روزی ماشین جدیدی با یک رنــگ متفـاوت بخرند (البته اگر شانس کار کردن در این اجتماع غیرمنصفانه را داشته باشند).

چرا مردم نسبت به ستاره‌ها بی توجهند، همدیگر را دوست ندارند، باد را حــس نمیکننـد، رفـت و آمد ماشینها را از محل زندگی و تغذیه خود حذف نمیکنند، و به رنگها بی اعتنایند... چرا بــه خـود برچسب بی مصرف میزنند؟ چرا شعر نمیخوانند؟

هیچ احساس شاعرانه‌ای در اجتماع یکنواخت و خسته کننده کارگران تبلیغاتی وجود ندارد...

احساس شاعرانه بزودی قدغن خواهد شد، چون نمیتوانید وقت خود را صرف خواندن اشعار کنید، و در نمایش مضحک چنین جامعه‌ای، در صرف وقت، محدویت دارید، این تنها چــیزی اسـت کـه آنها از شما میخواهند انجام دهید... پسرم،امروز، من جای برخی از بمبهای نوترونــی را گم کـرده‌ام، بمبهایی که تمام این آدمهای ماشینی بی مصرف را نابود میســازد و تنـها کتابـهای اصیـل و ودکـای دست نخورده را جا میگذارد. پس نمیتوان به دموکراسی اعتقاد داشت... اگر من رأی داده‌ام.... تمــام آدمهای ماشینی بی مصرف هم، متأســفانه، چنیـن کـاری را میکننـد و بـه «ظاهرهـای متبسـم» رأی میدهند، «به افراد ابلهی که قدیمی فکر میکنند، و همانگونه که واقعاًهستند عمل میکنند، و نسبت بـه هیچ چیزی بی تفاوت نیستند، به جز حس مسئولیت و از الگویــی بـی اهمیـت و کوتـه فکرانـه ای

حمایت میکنند. کارگران،افرادی را انتخاب میکنند که در تلویزیون دیده اند... همانگونه که مصریـها به فرعون هایشان رأی میدادند که در زیر شلاقهای تبلیغاتی به وجد می‌آمدند.

ببخشید، فراموش کردم، که هدف شما از مطالعه این برنامه، کراک کردن است و چیزهـایی کـه مـن می اندیشم، به شما مربوط نمیشود.

در صورتیکه با برخی از ترفندهای تجارتی مرا آگاه کنید (از طریق پست الکترونیکی)

من برایتان درسهای جا افتاده را خواهم فرستاد. نمیدانم آیا درس مرا فهمیده‌اید. البته اغلب میفـهمم، اما اگر جدید باشند، باید تمام واحدهای درسی را بگیرید، حتی اگر جدید هم نباشند. البته البته مـن معتقدم که شما با کار و فعالیت خود، یاد خواهید گرفت یا در صورتیکه واقعاً روی آنها کـار کننـد، تمام درسهای باقیمانده را برایتان خواهم فرستاد. من آماده شنیدن پیشنهادات و انتقادات شما هستم.

an526164@anon.penet.fi