Voyage à travers les virus MS-DOS

Damiano Mazza
CNRS, UMR 7030, Laboratoire d'Informatique de Paris Nord
Université Paris 13, Sorbonne Paris Cité

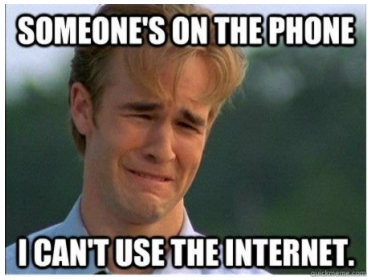
Séminaire Codes Sources
Paris, 19 mars 2015

La Préhistoire

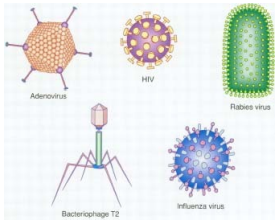
— 1974 : Intel 8080 + Digital Research CP/M
— 1978 : Intel 8086
— 1981 : Microsoft PC-DOS 1.0 (FAT 12)
— 1983 : PC-DOS 2.0 (disques durs, répertoires)
— 1986 : Brain (premier virus pour DOS)
— 1987 : PC-DOS 3.3 (disquettes 1.44Mo)
— 1987 : Jerusalem (fichiers EXE), Stoned (secteur démarrage)
— 1988 : Cascade (crypté)
— 1989 : Eddie (infection sur ouverture), Frodo (furtif)

L'Age d'Or

— 1990 : Chameleon (polymorphe), Flip (*multipartite*), Micro-128
— 1991 : MS-DOS 5.0, l'industrie des anti-virus décolle
— 1991 : Tequila (*stealth*, polymorphe, *armored*, *multipartite*)
— 1992 : Mutation Engine (polygen), Michelangelo (très médiatisé)
— 1993 : MS-DOS 6.0, utilisé par 75% des PC dans le monde
— 1993 : Tremor, Uruguay, Neuroquila (parmi les plus sophistiqués)
— 1994 : le nombre de virus MS-DOS est estimé entre 4500 et 7500
— 1995 : Windows 95



Zoologie des virus



— Modalités de fonctionnement : **resident** / **non-resident**.

— Modalités d'infection :
  — **boot sector :** infecte le secteur de démarrage des disques ;
  — **file infector :** infecte les fichiers exécutables ;
  — **multipartite :** capable des deux types d'infection.

— Caractéristiques particulières :
  — **stealth :** essaie de cacher sa présence ;
  — **polymorphic :** change à chaque réplication (≪ mutant ≫) ;
  — **armoured :** emploie des techniques d'obfuscation ;
  — **anti-anti-virus :** attaque les logiciels anti-virus.

Resident vs. non-resident

— Exécution typique d'un virus non-résident :
  1. recherche de nouveux hôtes et infection ;
  2. passage de contrôle à l'hôte et terminaison de l'action virale.

— Exécution typique d'un virus résident :
  1. test de résidence ; si négatif, installation en mémoire ;
  2. passage de contrôle à l'hôte ; le virus reste actif en mémoire et monitore l'activité de l'ordinateur. Il peut ainsi :
     — trouver et infecter des nouveaux hôtes ;
     — réaliser des fonctionnalités *stealth* ou anti-anti-virales.

— Les virus non-résidents ont un comportement très simple et leur capacité de diffusion est moindre. Leur intérêt est donc limité.

Un peu d'architecture 8086

— Architecture de base à 16 bit, adressage à 20 bit : mémoire limitée à $2^{20} = 1$Mo. C'est la *mémoire conventionnelle* de MS-DOS.

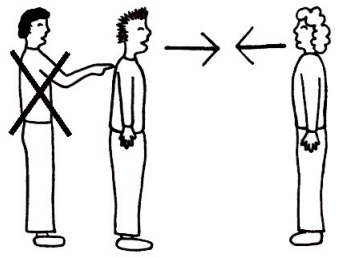
— *Segmentation* pour compatibilité avec 8080 (64Ko de mémoire) :

$$\texttt{segm:offs} \quad \longrightarrow \quad \text{adresse absolue} = \texttt{segm} \times \texttt{10h} + \texttt{offs}$$

— Registres (16 bit, c.-à-d. 2 octets) :

`AX,BX,CX,DX`	génériques (`X` = `H+L`, 8 + 8 bit)
`SI,DI`	index : source, destination
`IP,BP,SP`	pointeurs : instruction, base, stack
`CS,DS,ES,SS`	segment : code, data, extra, stack
`status`	flags : carry, zero, direction, interrupt. . .

Interruption



— L'architecture x86 prévoit des *software interrupts* :

```
int N
```

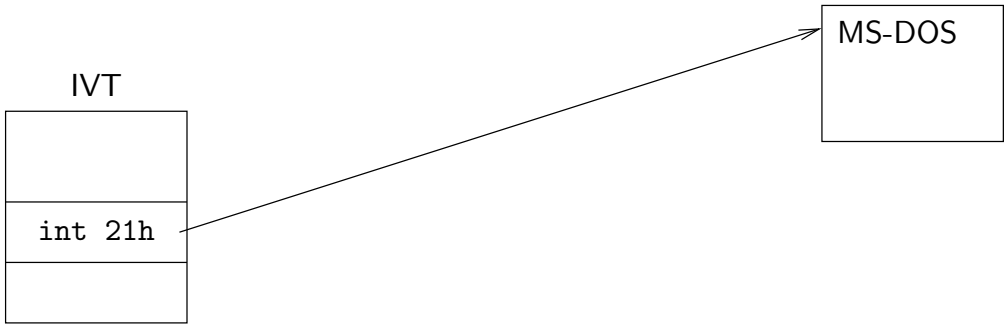
où `N` est un nombre entre 00h et FFh (0 et 255 en hexadécimal).

— Les addresses des services d'interruption se trouvent dans un tableau (IVT) aillant de `0000:0000` à `0000:03FFh` (4 octets par interrupt).

— Ces services accèdent aux péripheriques et aux fichiers :

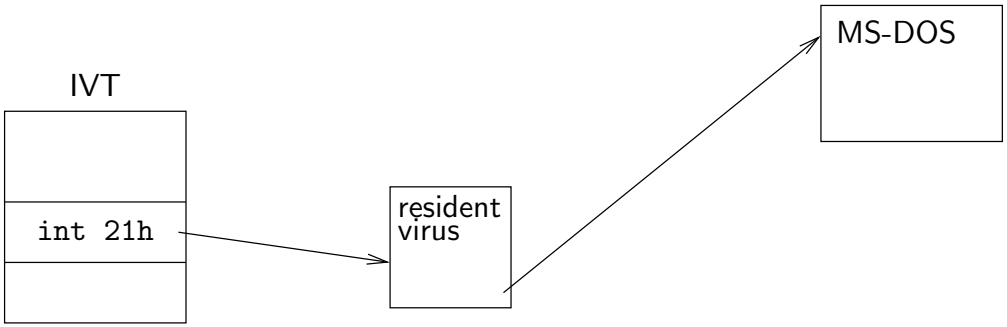
`N`	service
`10h`	vidéo (BIOS)
`13h`	accès aux disques (BIOS)
`21h`	appels système (MS-DOS)

Interrupt hooking



— La ligne 21h de l'IVT normalement contient l'adresse des services de base de MS-DOS.

Interrupt hooking



— La ligne 21h de l'IVT normalement contient l'adresse des services de base de MS-DOS.
— En modifiant l'IVT, un virus peut intercepter tous les appels qui lui intéressent et, une fois terminée son action, rédiriger l'appel à MS-DOS.

Interrupt hooking

```
Hook_Int_21:    xor ax,ax
                mov es,ax                           ; ES = 0
                mov ax,offset Int_21_Handler        ; AX = Handler Offset
                mov bx,cs                           ; BX = Handler Segment
                cli                                 ; clear interrupts
                xchg ax,word ptr es:[84h]           ; set AX = old handler offset
                                                    ; and set new offset.
                xchg bx,word ptr es:[86h]           ; set BX = old handler segment
                                                    ; and set new segment.
                mov word ptr [Int_21_Offset],ax
                mov word ptr [Int_21_Segment],bx
                sti                                 ; restore interrupts
                mov dx,VSIZE
                int 27h                             ; terminate and stay resident

Int_21_Handler: cmp ah,4bh                          ; Check for activation
```

Interrupt hooking

```
                je execute_a_program        ; conditions by looking
                cmp ah,3dh                  ; at the function numbers
                je open_a_file              ; of Int 21 that you wish
                                            ; to intercept. Make sure
                                            ; to save any registers that
                                            ; you change inside the
                                            ; various handlers!!!

Go_Int_21:      db EAh                      ; This simulates a far jump
Int_21_Offset   dw ?                        ; to the old interrupt handler.
Int_21_Segment  dw ?                        ; (EAh is code for a far jmp.)
```

Virus de type boot sector

— Le secteur de démarrage est le premier secteur du disque. Il ne comporte que 512 octets (comme tous les secteurs). Exécuté après le BIOS au démarrage de la machine, il charge en mémoire les fichiers du système d'exploitation et il leur passe le contrôle.

— Idée : le virus substitue son code au secteur de démarrage, en gardant ailleurs une copie de l'original. Le virus prend ainsi le contrôle avant tout autre logiciel (en particulier anti-virus). Il infecte toute disquette insérée dans le lecteur.

— Transmission efficace : les virus les plus répandus étaient de ce type. Toutefois, moins virulents avec le déclin du démarrage par disquette.

File infectors

- Deux types de fichiers exécutables sous MS-DOS :
  - COM : rétrocompatibilité avec CP/M, taille limitée à 64Ko ; infection simple mais peu d'hôtes (e.g. COMMAND.COM) ;
  - EXE : segments multiples (code, données, pile), taille arbitraire ; infection plus complexe mais beaucoup plus d'hôtes potentiels.
- Idée : si un fichier contaminé est lancé par l'utilisateur, le virus s'installe en mémoire. Ensuite, il intercepte les appels à l'INT 21h et infecte les fichiers exécutables lorsque :
  - ils sont exécutés par l'utilisateur ;
  - ils sont listés (e.g. commande `dir`) ;
  - ils sont ouverts (e.g. contrôle anti-virus).
- Moins efficace que boot sector, mais possibilité de diffusion d'une machine à l'autre sans démarrage disquette.

File infectors

Structure typique :

```
start:   goto loader

         *** host code (first bytes overwritten by goto instruction) ***

loader:  are-you-there?
         if yes, goto done
         make room in memory for virus code (perhaps use this same copy)
         hook interrupts (typically INT 21h)
         make sure virus code will stay resident
done:    restore host code
         goto start

handler: this will be executed whenever a hooked interrupt is invoked
         typically INT 21h function 4Bh (execute file):
         open file; is it infected? if yes, goto jmpout
         append virus code (taken from memory) to file and modify entry point
         (saving original for restoring it later); close file
jmpout:  goto old interrupt handler
```

Brain (1986)

— Premier virus MS-DOS, écrit par deux frères pakistanais. Resident boot sector mais seulement disquettes. Taille : environ 3Ko.

— Brain sauvegarde le secteur de démarrage original et le marque comme « bad » (ainsi que le reste du corps du virus). Il ne fait rien à part changer l'étiquette de la disquette en `(c)Brain`. La copie du secteur original permet une fonctionnalité *stealth* :

```
int13h: ; ... intercept floppy drive read request (AH = 02h, bit 7 of DL != 1)
fdread: cmp   cx,1                    ; cylinder 0, sector 1?
        jne   old13h                  ; if not, proceed as usual
        cmp   dh,0h                   ; head 0?
        jne   old13h                  ; if not, proceed as usual
        mov   cx,7                    ; redirect to sector 7
        pushf                         ; save flags to simulate int instruction
        call  dword ptr cs:[old13h]   ; invoke old INT 13h
        mov   cx,1                    ; restore CX not to arise suspicion
        retf  2                       ; simulate iret instruction
        ; ... rest of the handler ...
old13h: jmp   dword ptr cs:[old13h]   ; jump to old INT 13h
```

Stoned (1987)

— Écrit (paraît-il) par un étudiant néozelandais. Resident boot sector, infecte aussi les disques durs.

— C'est le virus le plus réussi de l'époque pré-Internet. En 1992, on estime que près de 25% des infections sont des variantes de Stoned. La variante originale n'est pas malveillante :

```
            test  bite ptr es:[TIMER],7   ; message displayed 1 in 8 times
            jnz   MSG_DONE                ; lucky, no message
            mov   si,offset STONED_MSG1   ; play the message
            push  cs
            pop   ds                      ; DS = CS
MSG_LOOP:   lodsb                         ; get a byte to AL
            or    al,al                   ; AL=0?
            je    MSG_DONE                ; yes, all done
            mov   ah,0eh                  ; display byte using BIOS
            mov   bh,0
            int   10h
            jmp   MSG_LOOP                ; and go get another
MSG_DONE:   ; ... remainder of boot sector code
STONED_MSG1 db    7,"Your PC is now Stoned!",7,0dh,0ah,0
STONED_MSG2 db    "LEGALIZE MARIJUANA!"
```

— Variante destructive : le 6 mars (1992), Michelangelo efface les 100 premiers secteurs du disque C.

Micro-128 ($\sim$1990)

- C'était le plus petit infecteur de fichiers COM résident en mémoire connu : il ne fait que 128 octets de taille (moins qu'un Tweet !).
- Pour achever une telle minimalité, il est très rusé :
  - méthode atypique de résidence en mémoire (s'installe dans l'espace inutilisé de l'IVT) ;
  - les instructions sont soigneusement choisies pour être les plus petites réalisant la tâche souhaitée ;
  - certains morceaux de code ont 2, voire 3 fonctionnalités.
- Cela vient à un prix :
  - instabilité potentielle (on touche à l'IVT) ;
  - le virus n'infecte que les fichiers COM ;
  - aucun test d'infection : le virus re-infecte les fichiers infectés ;
  - aucune capacité de dissimulation (*stealth*, polymorphisme. . .).

Un avant-goût de polymorphisme

— Un exemple simple de déchiffreur :

```
; Group 1  Prolog Instructions
mov     ax,0E9B   ; set key 1
mov     di,012A   ; offset of Start
mov     cx,0571   ; this many bytes - key 2

; Group 2  Decryption Instructions
Decrypt:
xor     [di],cx   ; decrypt first word with key 2
xor     [di],ax   ; decrypt first word with key 1

; Group 3  Decryption Instructions
inc     di        ; next byte
inc     ax        ; slide key 1
loop    Decrypt   ; until all bytes are decrypted  slide key 2

Start:
;     Encrypted/decrypted virus body
```

— Insertion d'instructions aléatoires inutiles : forme de polymorphisme.

Un avant-goût de polymorphisme

Déchiffreur possible de Chameleon (infecteur EXE non-résident, 1990) :

```
; Group 1  Prolog Instructions
inc     si          ; optional, variable junk
mov     ax,0E9B     ; set key 1
clc                 ; optional, variable junk
mov     di,012A     ; offset of Start
nop                 ; optional, variable junk
mov     cx,0571     ; this many bytes - key 2

; Group 2  Decryption Instructions
Decrypt:
xor     [di],cx     ; decrypt first word with key 2
sub     bx,dx       ; optional, variable junk
xor     bx,cx       ; optional, variable junk
sub     bx,ax       ; optional, variable junk
sub     bx,cx       ; optional, variable junk
nop                 ; non-optional junk
xor     dx,cx       ; optional, variable junk
xor     [di],ax     ; decrypt first word with key 1
```

```
; Group 3  Decryption Instructions
inc     di          ; next byte
nop                 ; non-optional junk
clc                 ; optional, variable junk
inc     ax          ; slide key 1
; loop
loop    Decrypt     ; until all bytes are decrypted  slide key 2
; random padding up to 39 bytes

Start:
;     Encrypted/decrypted virus body
```