Subverting Windows Embedded CE 6 Kernel

petr@research.coseinc.com

www.coseinc.com

22nd june 2k8

Speaker

Petr Matousek

- Before also known under the handle Ratter, member of 29A
- Author of WinCE.Dust, proof of concept WinCE PE EXE files infector
- Co-author of Hxdef, one of the most used rootkits in Windows NT world
- Team member of AML (advanced malware lab, known for Blue Pill virtualised rootkit) of COSEINC (www.coseinc.com)

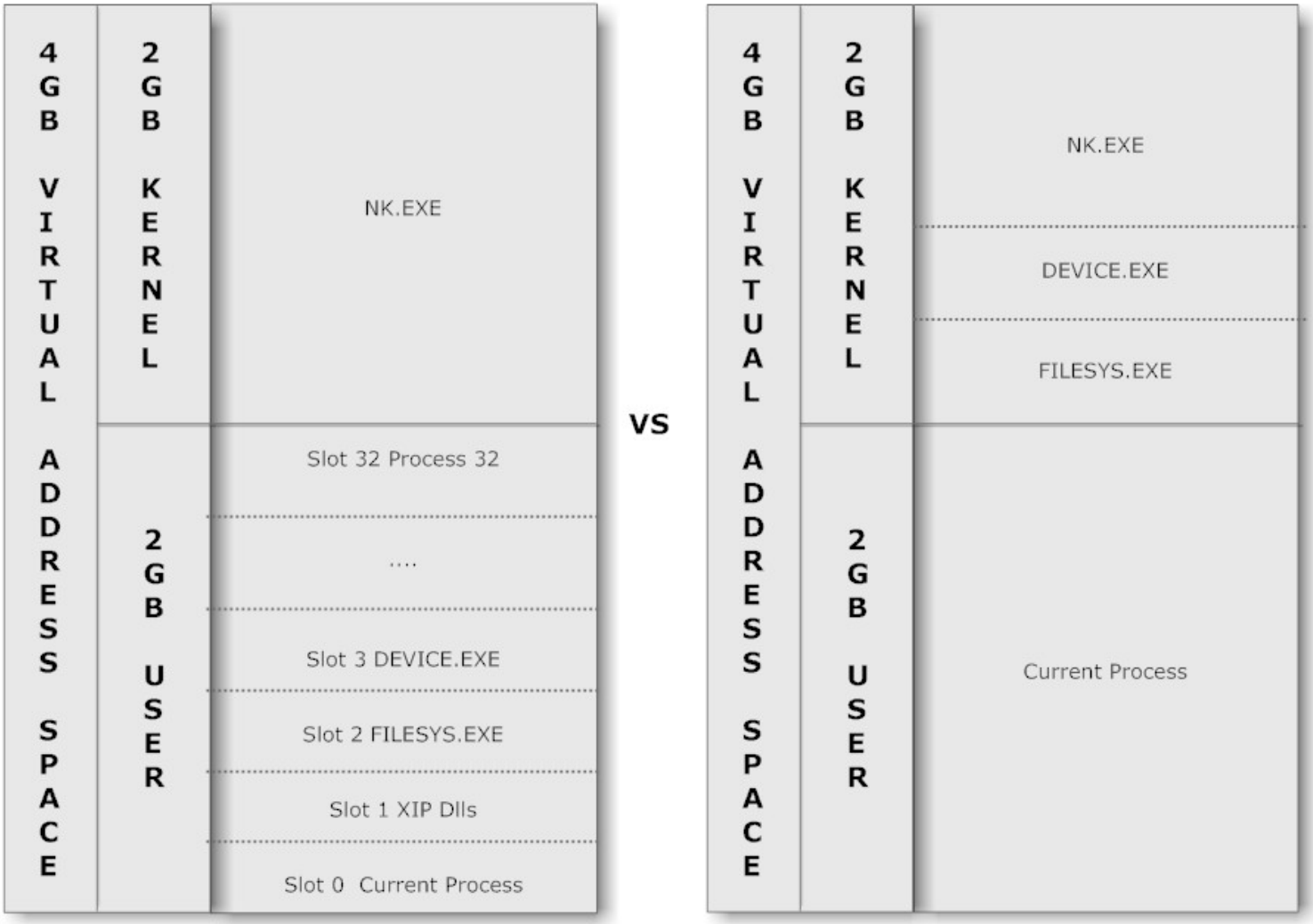
Agenda

- Basis for Windows Mobile 7
- Windows Mobile for Automotives
- ARM accounts 75% of embedded 32-bit RISC CPUs
- ARM as main platform Windows Mobile 7
- Techniques presented work on any platform
- Rootkit for Windows CE 5 already exists

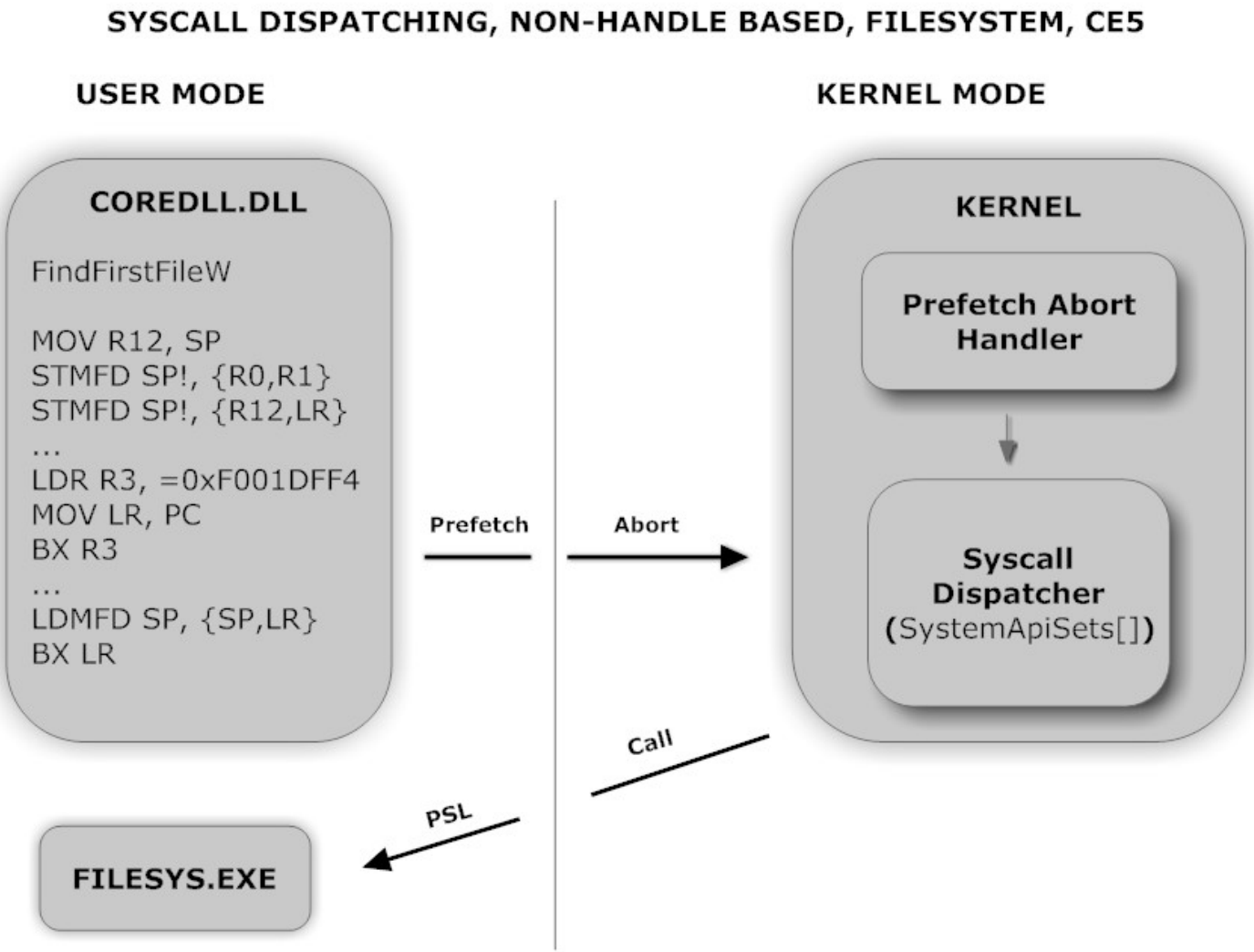
- Kit that keeps you Root
- But we're in embedded world ...
- Kit that let's you do dirty things INVISIBLY
  - Hiding Files
  - Hiding Processes
  - Hiding Network connections
  - Hiding OS Database Entries

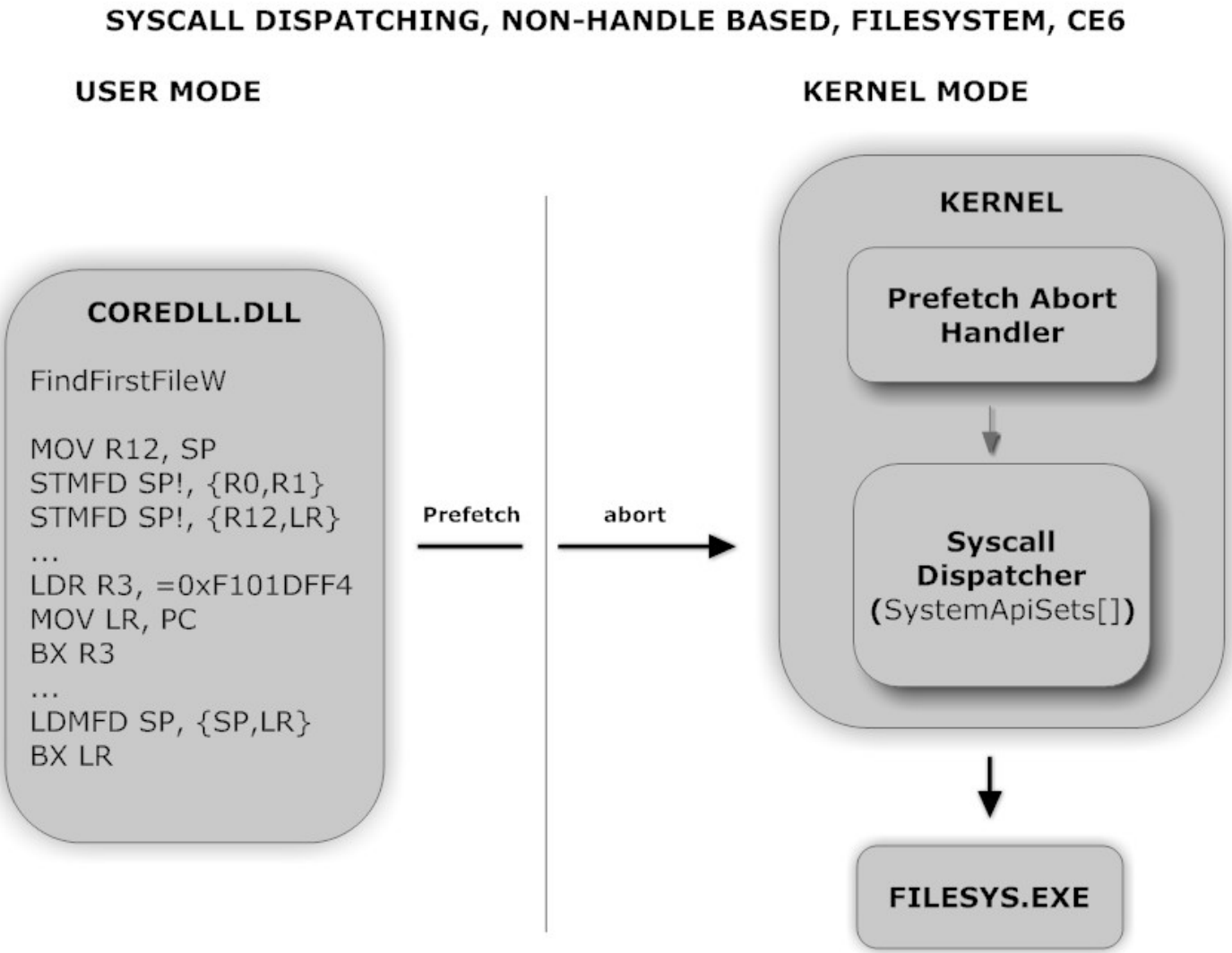
- Way to explore OS system design
- Rootkit techniques can be used in debugging
- Every system can be rootkited
- Every rootkit can be detected, but the detection method must be already there
- Because I've already written a virus -)

Virtual Memory Layout: Windows CE 5.0 vs. Windows Embedded CE 6.0



- 32MB Address Space VS 2GB Address Space
- 32 Processes VS 32K Processes
- Device.exe, filesys.exe, GWES.exe in User Mode
  VS
  Device.exe, filesys.exe, GWES.exe in Kernel Mode
- Better parameter validation during syscalls
- Per-process page and handle tables
- Code based security in the loader (signing)


SYSCALL DISPATCHING, NON-HANDLE BASED, FILESYSTEM, CE5
USER MODE
KERNEL MODE
COREDLL.DLL
FindFirstFileW
MOV R12, SP
STMFD SP!, {R0,R1}
STMFD SP!, {R12,LR}
...
LDR R3, =0xF001DFF4
MOV LR, PC
BX R3
...
LDMFD SP, {SP,LR}
BX LR
Prefetch
Abort
KERNEL
Prefetch Abort Handler
Syscall Dispatcher (SystemApiSets[])
Call
PSL
FILESYS.EXE


SYSCALL DISPATCHING, NON-HANDLE BASED, FILESYSTEM, CE6
USER MODE
KERNEL MODE
COREDLL.DLL
FindFirstFileW
MOV R12, SP
STMFD SP!, {R0,R1}
STMFD SP!, {R12,LR}
...
LDR R3, =0xF101DFF4
MOV LR, PC
BX R3
...
LDMFD SP, {SP,LR}
BX LR
Prefetch
abort
KERNEL
Prefetch Abort Handler
Syscall Dispatcher (SystemApiSets[])
FILESYS.EXE

- APIS in the kernel described by APISET and CINFO structs

```
typedef struct APISet {
    CINFO   cinfo; /* description of the API set */
    int     iReg;       /* registered API set index (-1 if not registered) */
} APISET;
typedef APISET *PAPISET;

typedef struct _CINFO {
    char            acName[4];       /* 00: object type ID string */
    uchar           disp;            /* 04: type of dispatch */
    uchar           type;            /* 05: api handle type */
    ushort          cMethods;        /* 06: # of methods in dispatch t..
    const PFNVOID   *ppfnExtMethods; /* 08: ptr to array of methods  ...
    const PFNVOID   *ppfnIntMethods; /* 0C: ptr to array of methods  ...
    const ULONGLONG *pu64Sig;        /* 10: ptr to array of method si...
    DWORD           dwServerId;      /* 14: server process id */
    PHDATA          phdApiSet;       /* 18: HDATA of API set */
    PFNAPIERRHANDLER         pfnErrorHandler; /* 1C: ptr to the API s...
} CINFO;
typedef CINFO *PCINFO;
```

- SystemApiSets[] is a global array of CINFO structs

```
[APISET index=0] CINFO [Wn32] disp=3,type=0,cMethods=82,extmethods=80098c70,
                               intmethods=80098e78,sigs=80099080,dwServerId=0
[APISET index=1] CINFO [THRD] disp=2,type=1,cMethods=10,extmethods=800a3678,
                               intmethods=800a3638,sigs=800a36b8,dwServerId=0
...
[APISET index=7] CINFO [W32H] disp=2,type=7,cMethods=10,extmethods=0,intmethods=0,
                               sigs=0,dwServerId=ffffffff
[APISET index=8] CINFO [FFEX] disp=2,type=8,cMethods=3,extmethods=0,intmethods=0,
                               sigs=0,dwServerId=ffffffff
```

- OS and components can register their own API sets
- There are ”prototype” (HANDLE-based) API sets

- Coredll.dll acts as a proxy

```
.text:10040120 FindNextFileW           ; CODE XREF: EnumUILanguagesW+D
.text:10040120                         ; DATA XREF: .pdata:10101278
.text:10040120
...
.text:10040120                 MOV     R12, SP
.text:10040124                 STMFD   SP!, {R0,R1}
.text:10040128                 STMFD   SP!, {R12,LR}
...
.text:1004013C                 LDR     R3, =0xF101DFF8
.text:10040140                 MOV     LR, PC
.text:10040144                 BX      R3
...
.text:1004015C                 LDMFD   SP, {SP,LR}
.text:10040160                 BX      LR
.text:10040160 ; End of function FindNextFileW
```

- Branching to 0xF101DFF8 causes prefetch abort handler (permission fault, not translation fault) to fire up

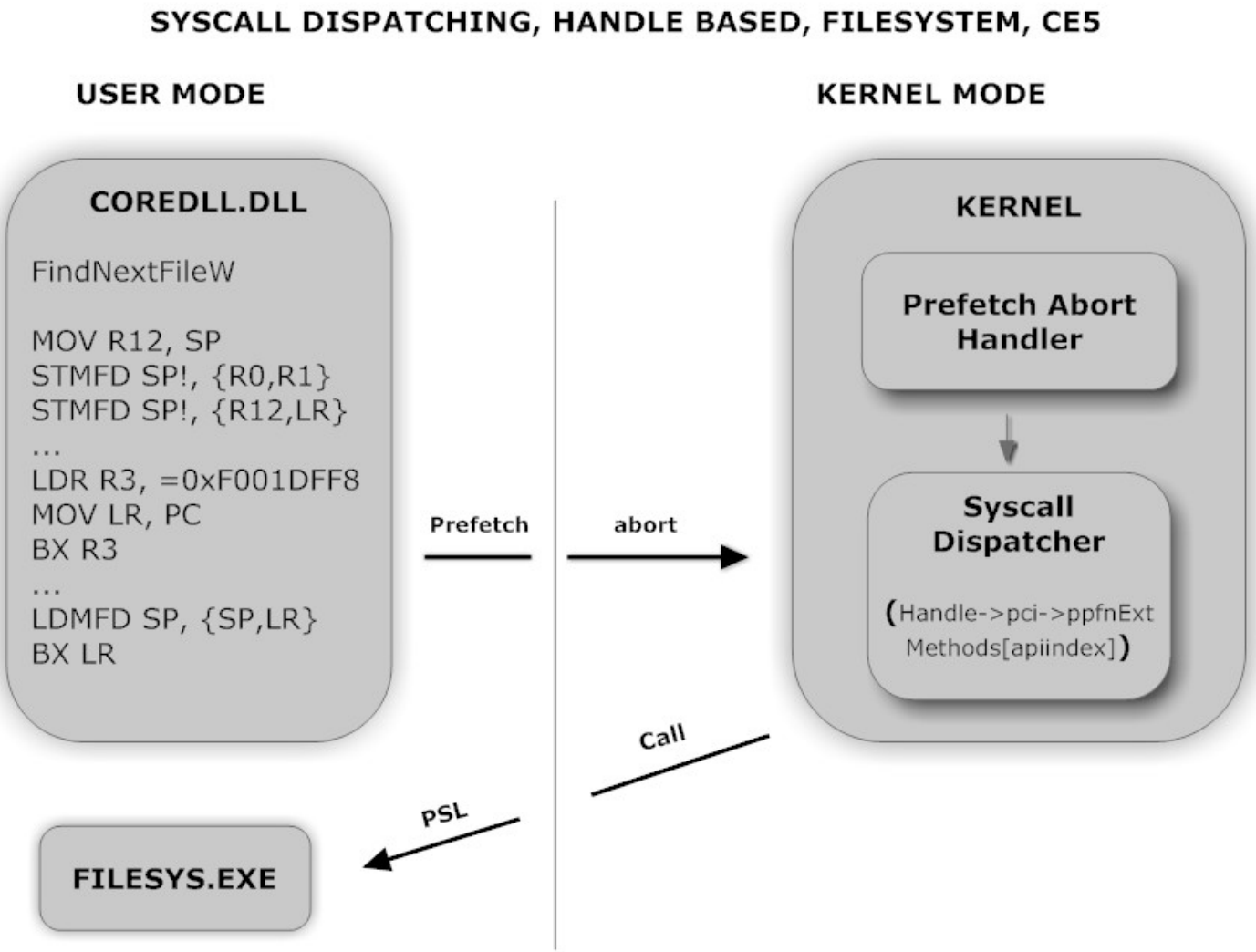
```
pte: 1e2; virtual address [f1000000] : physical address [0] - READONLY, DOMAIN - 15
```

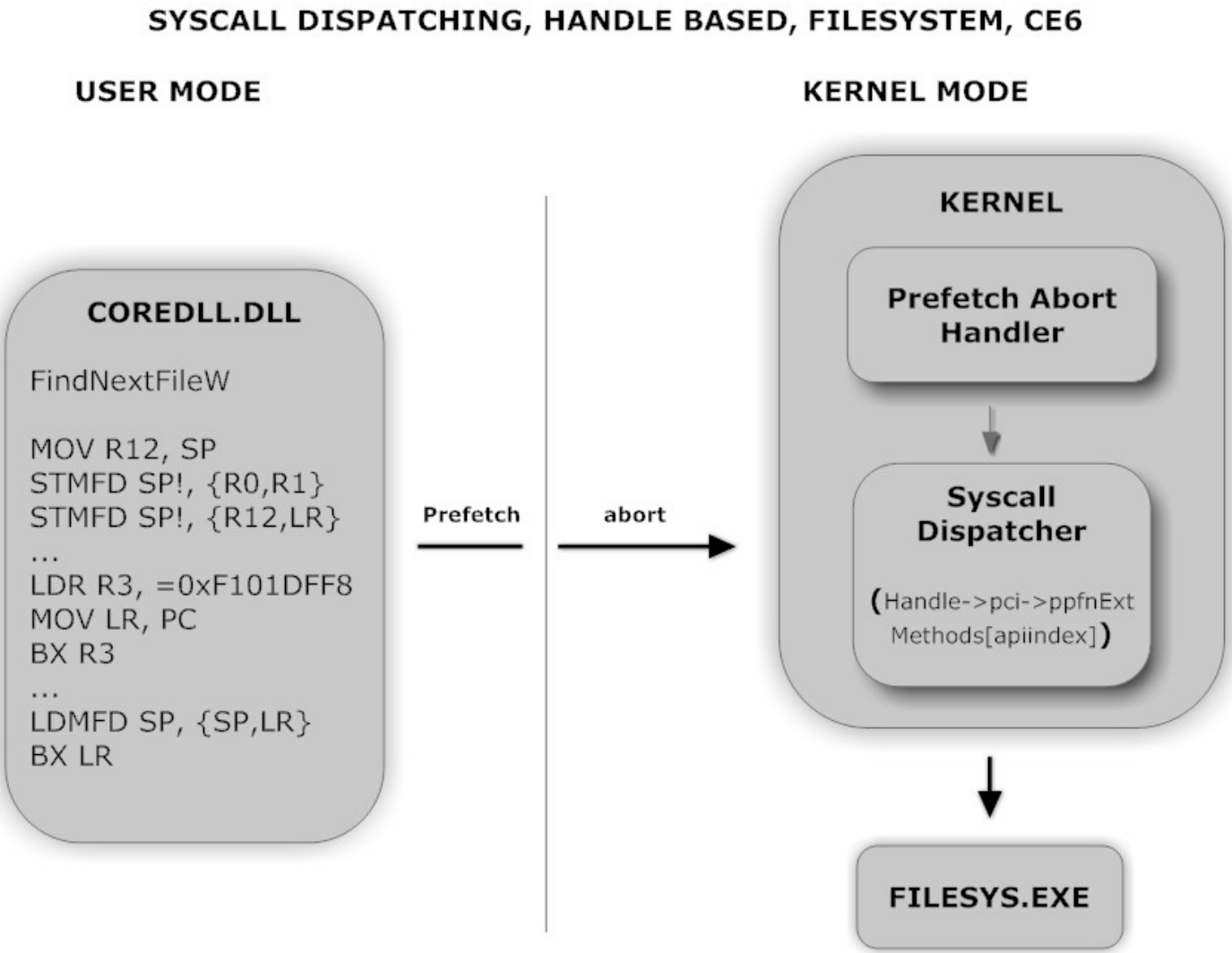
- Control transfer to armtrap.s[PrefetchAbort()]
- Decomposing to API set and API number and call objdisp.c[ObjectCall()]
- For non-HANDLE-based apis SystemApiSets[] is used
- API function (pci->ppfnExtMethods[apiindex]) is called

- HANDLE Struct

```
// HDATA object (one per object)
struct _HDATA {
DLIST           dl;                 // doubly linked list.
PCCINFO         pci;                // handle server information
LPVOID          pvObj;              // pointer to the real object
DWORD           dwRefCnt;           // total ref count
DWORD           dwData;             // per-object data
PNAME           pName;              // Name of the object
PNAME           psd;                // security descriptor
};
```

- For HANDLE-based apis handle is looked up in the handle table and proper API set is retrieved ((PHDATA *)->pci)
- ((PHDATA *)->pci->ppfnExtMethods[apiindex]) is called with the handle parameter replaced by (PHDATA *)->pvObj


SYSCALL DISPATCHING, HANDLE BASED, FILESYSTEM, CE5
USER MODE
KERNEL MODE
COREDLL.DLL
FindNextFileW
MOV R12, SP
STMFD SP!, {R0,R1}
STMFD SP!, {R12,LR}
...
LDR R3, =0xF001DFF8
MOV LR, PC
BX R3
...
LDMFD SP, {SP,LR}
BX LR
Prefetch
abort
KERNEL
Prefetch Abort Handler
Syscall Dispatcher
(Handle->pci->ppfnExt Methods[apiindex])
Call
PSL
FILESYS.EXE


SYSCALL DISPATCHING, HANDLE BASED, FILESYSTEM, CE6
USER MODE
KERNEL MODE
COREDLL.DLL
FindNextFileW
MOV R12, SP
STMFD SP!, {R0,R1}
STMFD SP!, {R12,LR}
...
LDR R3, =0xF101DFF8
MOV LR, PC
BX R3
...
LDMFD SP, {SP,LR}
BX LR
Prefetch
abort
KERNEL
Prefetch Abort Handler
Syscall Dispatcher
(Handle->pci->ppfnExt Methods[apiindex])
FILESYS.EXE

Kernel-Mode Hooks

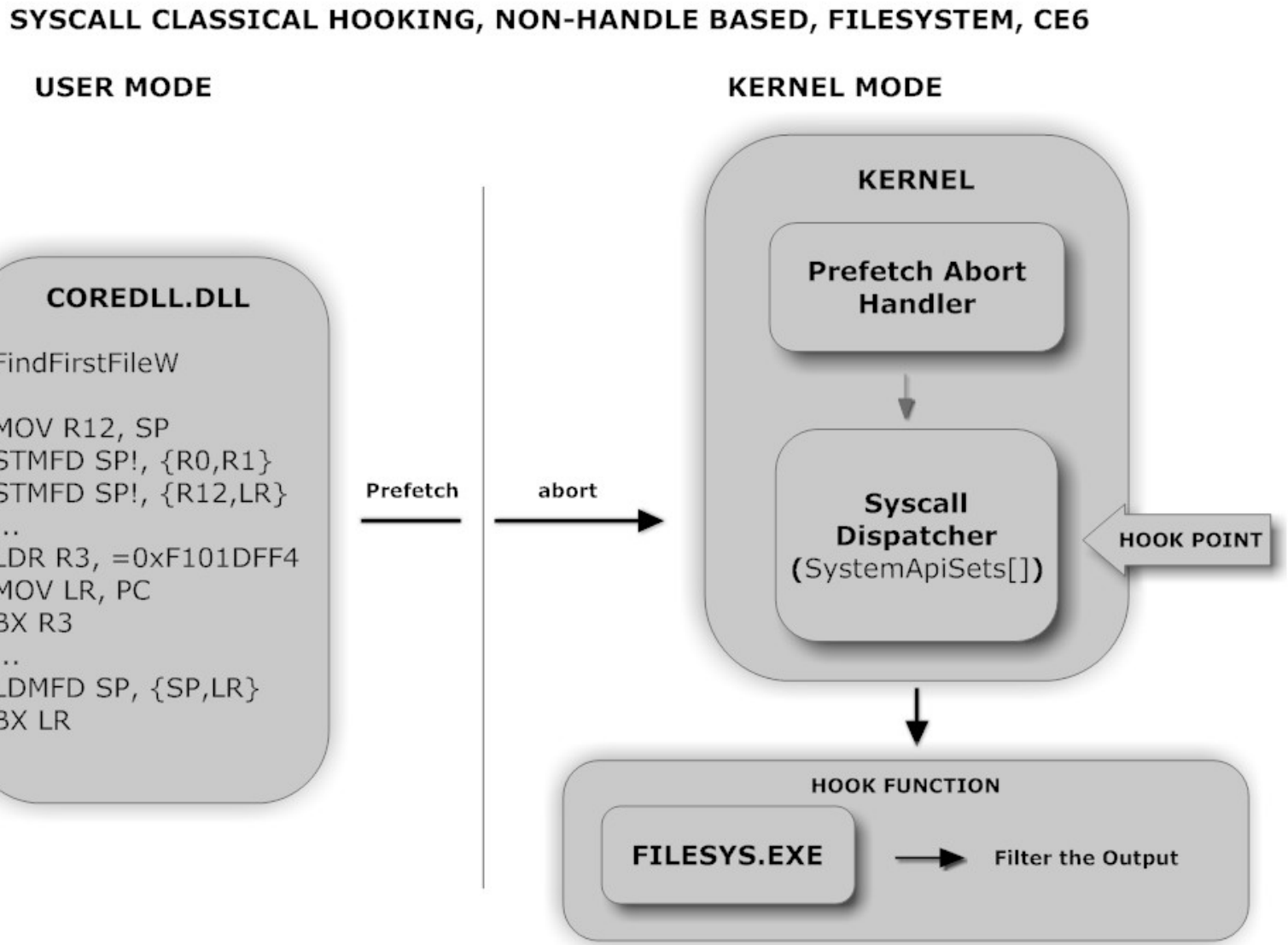
- Classic API Set Functions Hook
- Prefetch Abort Hook

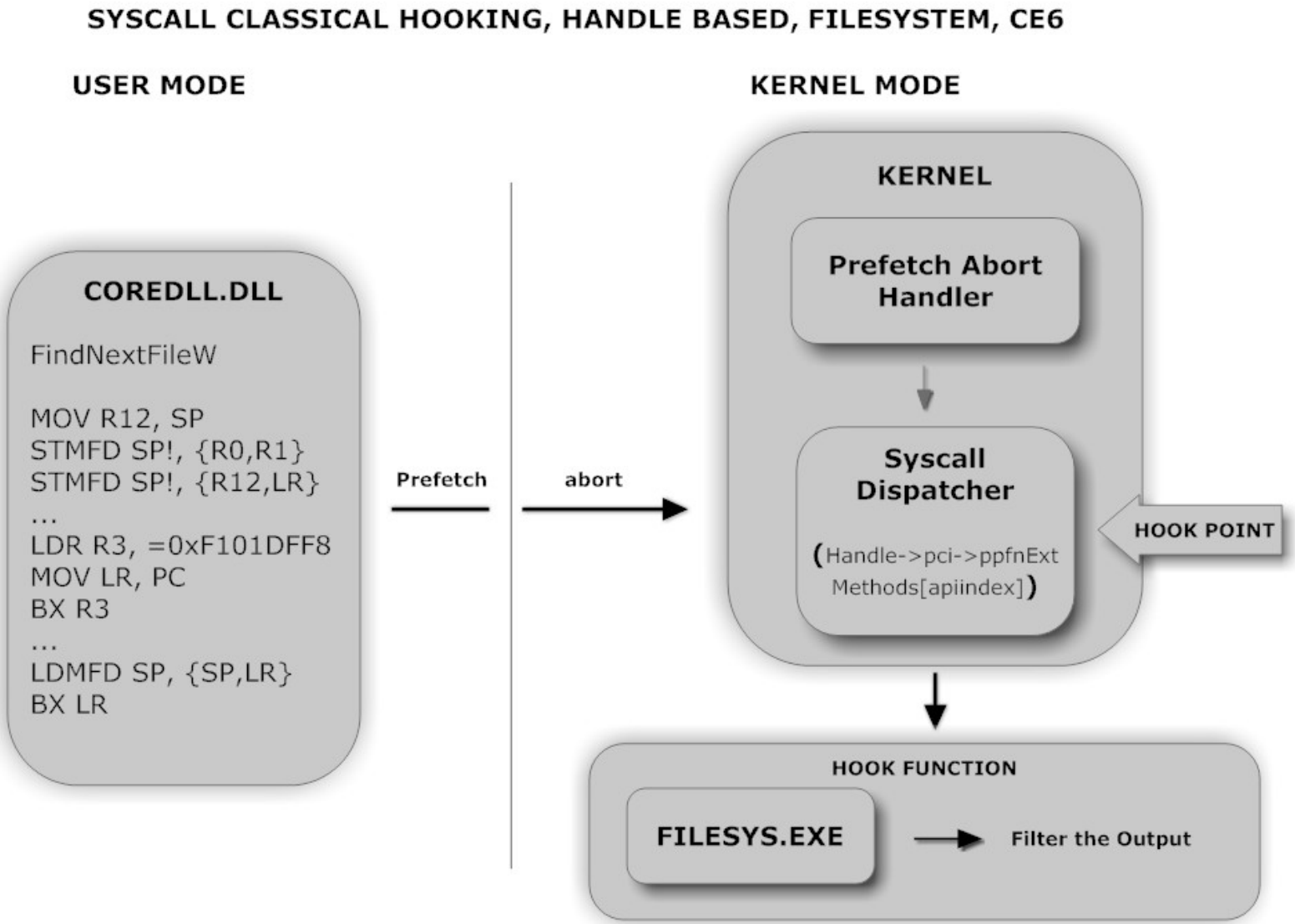
User-Mode Hooks

- Shim Dlls
- Source Code Hook
- Other ”Hook”

Classic API Set Functions Hook

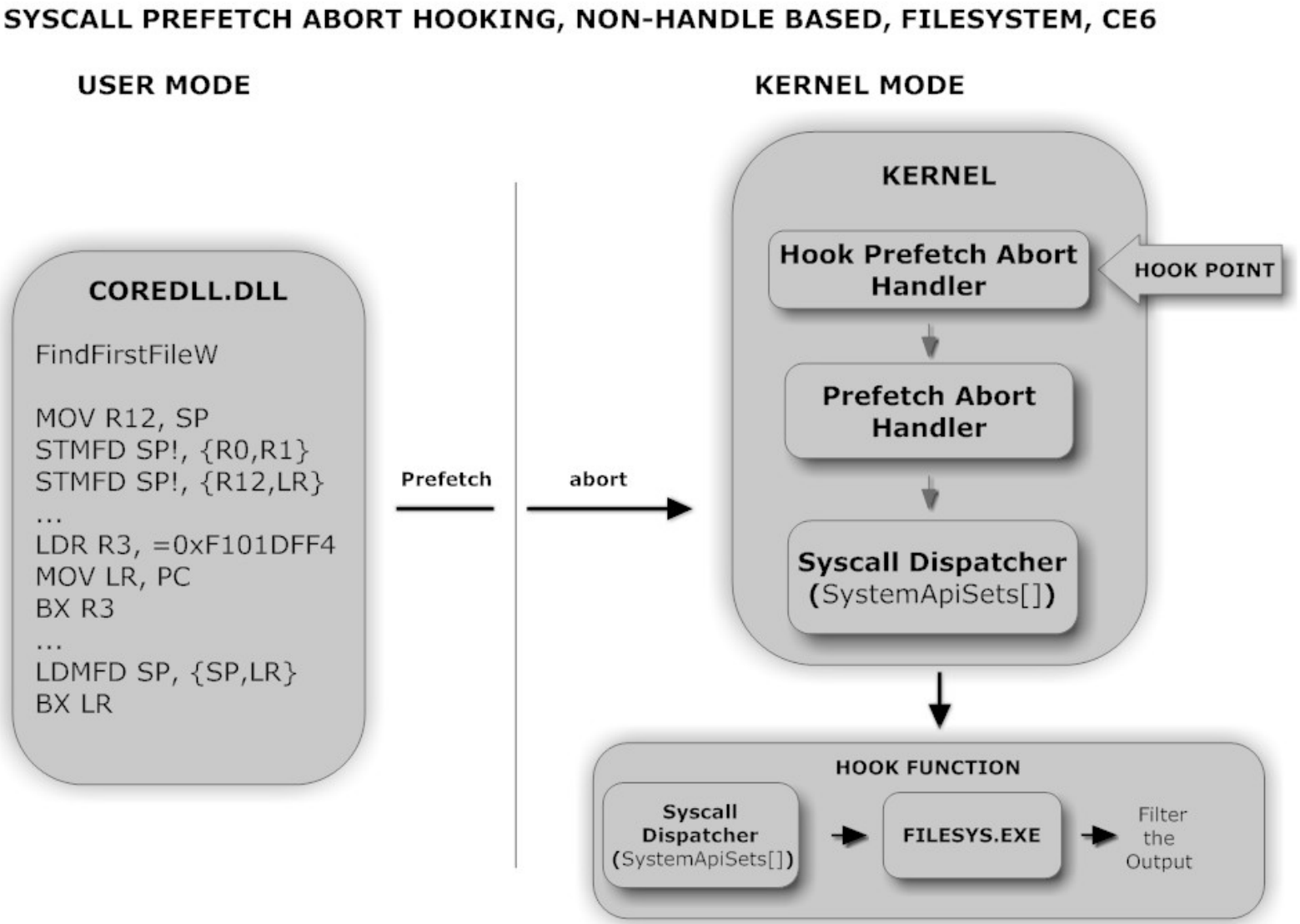
- Replace function pointers in CINFO struct
- Filesys.exe is in kernel so no more code injecting
- Easily detectable

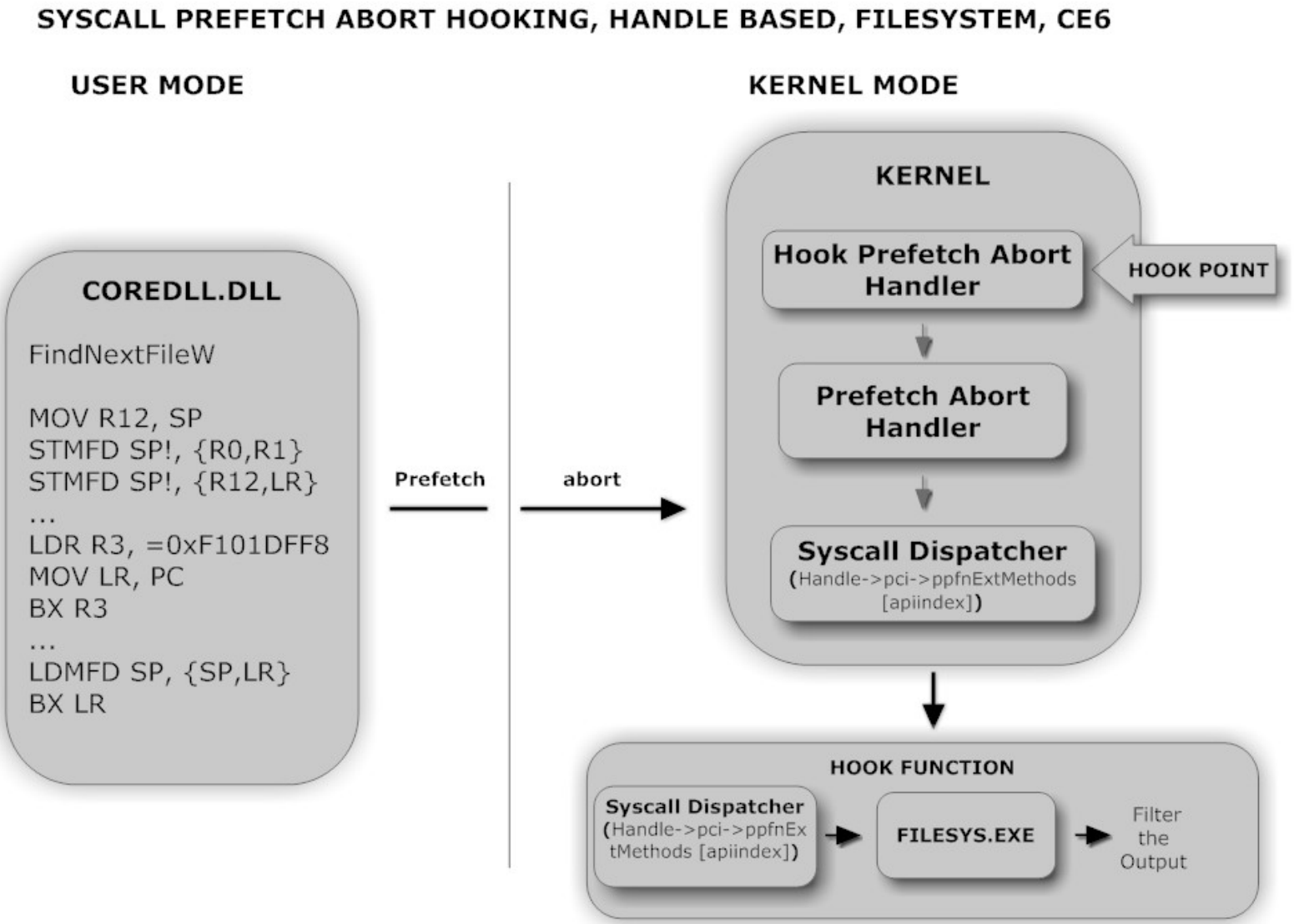

SYSCALL CLASSICAL HOOKING, NON-HANDLE BASED, FILESYSTEM, CE6
USER MODE
KERNEL MODE
COREDLL.DLL
FindFirstFileW
MOV R12, SP
STMFD SP!, {R0,R1}
STMFD SP!, {R12,LR}
...
LDR R3, =0xF101DFF4
MOV LR, PC
BX R3
...
LDMFD SP, {SP,LR}
BX LR
Prefetch
abort
KERNEL
Prefetch Abort Handler
Syscall Dispatcher (SystemApiSets[])
HOOK POINT
HOOK FUNCTION
FILESYS.EXE
Filter the Output


SYSCALL CLASSICAL HOOKING, HANDLE BASED, FILESYSTEM, CE6
USER MODE
KERNEL MODE
COREDLL.DLL
FindNextFileW
MOV R12, SP
STMFD SP!, {R0,R1}
STMFD SP!, {R12,LR}
...
LDR R3, =0xF101DFF8
MOV LR, PC
BX R3
...
LDMFD SP, {SP,LR}
BX LR
Prefetch
abort
KERNEL
Prefetch Abort Handler
Syscall Dispatcher
(Handle->pci->ppfnExt Methods[apiindex])
HOOK POINT
HOOK FUNCTION
FILESYS.EXE
Filter the Output

Prefetch Abort Hook

- Exploits the syscall implementation itselves
- Creates own API Set with ”hooking” functions
- Installs prefetch abort hook that redirects hooked APIs to ”hooking” functions by changing the prefetch abort address


SYSCALL PREFETCH ABORT HOOKING, NON-HANDLE BASED, FILESYSTEM, CE6
USER MODE
KERNEL MODE
COREDLL.DLL
FindFirstFileW
MOV R12, SP
STMFD SP!, {R0,R1}
STMFD SP!, {R12,LR}
...
LDR R3, =0xF101DFF4
MOV LR, PC
BX R3
...
LDMFD SP, {SP,LR}
BX LR
Prefetch
abort
KERNEL
Hook Prefetch Abort Handler
HOOK POINT
Prefetch Abort Handler
Syscall Dispatcher (SystemApiSets[])
HOOK FUNCTION
Syscall Dispatcher (SystemApiSets[])
FILESYS.EXE
Filter the Output


SYSCALL PREFETCH ABORT HOOKING, HANDLE BASED, FILESYSTEM, CE6
USER MODE
KERNEL MODE
COREDLL.DLL
FindNextFileW
MOV R12, SP
STMFD SP!, {R0,R1}
STMFD SP!, {R12,LR}
...
LDR R3, =0xF101DFF8
MOV LR, PC
BX R3
...
LDMFD SP, {SP,LR}
BX LR
Prefetch
abort
KERNEL
Hook Prefetch Abort Handler
HOOK POINT
Prefetch Abort Handler
Syscall Dispatcher
(Handle->pci->ppfnExtMethods [apiindex])
HOOK FUNCTION
Syscall Dispatcher
(Handle->pci->ppfnExtMethods [apiindex])
FILESYS.EXE
Filter the Output

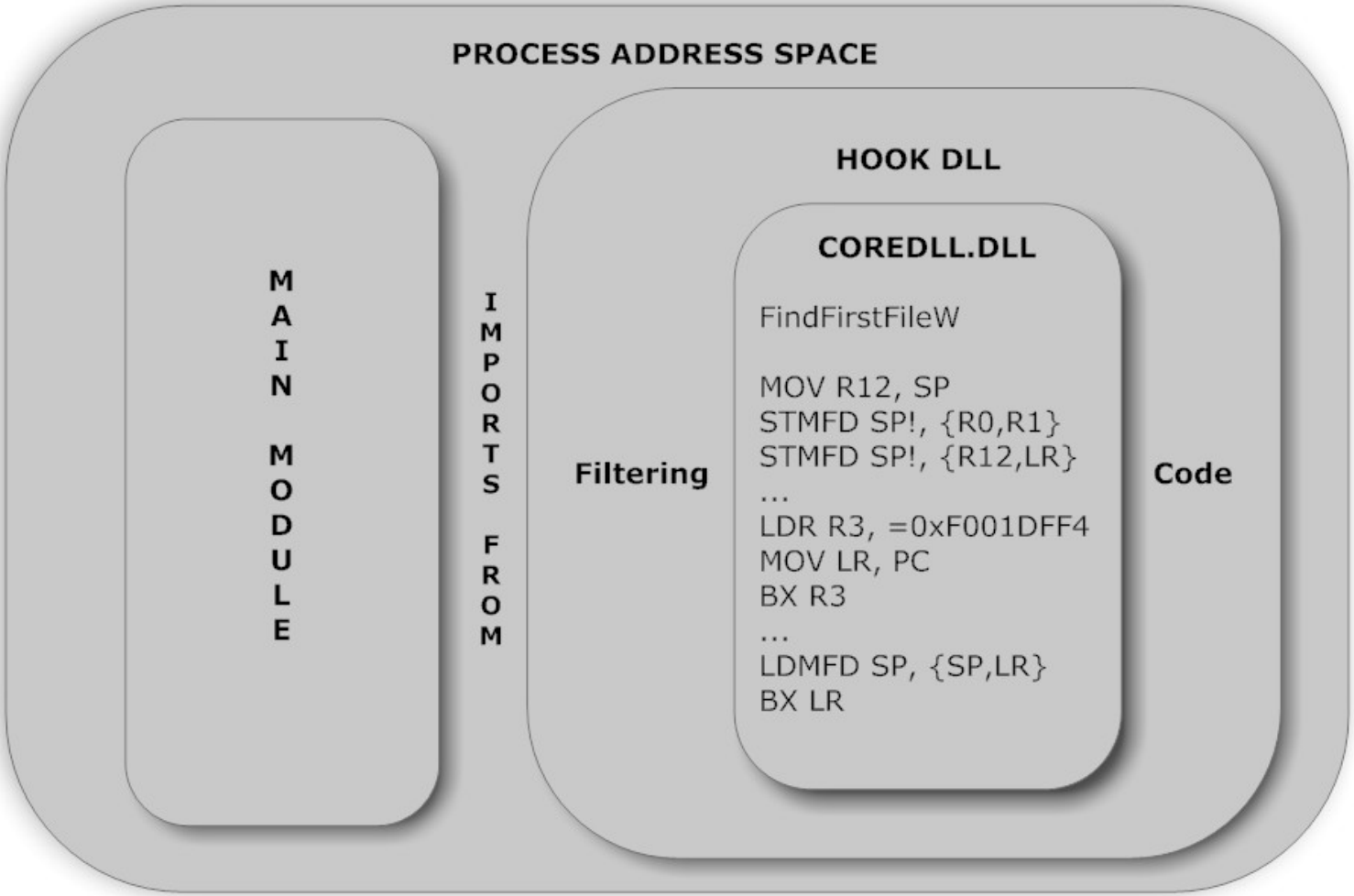
Shim Dlls

- Hooking method supported by the OS itselves
- Loader loads ”shim modules” for every loaded module

```
iesample.exe:5da0022    OpenKey HKLM\ShimEngine SUCCESS  Key:d0151980
iesample.exe:5da0022    QueryValue HKLM\ShimEngine\GlobalEnable    SUCCESS 0x1
iesample.exe:5da0022    CloseKey HKLM\ShimEngine SUCCESS  Key:d0151980
iesample.exe:5da0022    OpenKey HKLM\ShimEngine\LPCRT.dll       NOTFOUND
iesample.exe:5da0022    OpenKey HKLM\ShimEngine\iesample.exe    NOTFOUND
iesample.exe:5da0022    OpenKey HKLM\ShimEngine\{all}.dll       NOTFOUND
iesample.exe:5da0022    OpenKey HKLM\ShimEngine SUCCESS  Key:d0151980
```

- Integrated facility for ”hooking” exports

SYSCALL HOOKING, SHIM MODULES, CE6



Hiding Files

- We're interested in SH_FILESYS_APIS && HT_FIND API Sets

```
[SH_FILESYS_APIS]
fc=8,sig=0xc53 - FindFirstFile
  (ARG_I_WSTR, ARG_IO_PTR, ARG_DW)
```

```
[HT_FIND]
fc=2,sig=0x803 - FindNextFile
  (ARG_DW, ARG_O_PTR, ARG_DW)
```

- HT_FIND API Set is HANDLE based

Hiding Files

- FindFirstFile Hook

```
HANDLE my_FindFirstFile(LPCTSTR lpFileName, LPWIN32_FIND_DATA lpFindFileData, DWORD fsz)
{
  HANDLE r = call_old_FindFirstFile();

  if(is_current_process_root_process() || r == INVALID_HANDLE_VALUE)
    return r;

  if(returned_file_name_contains_hide_mask())
  {
    if(!call_old_FindNextFile())
    {
      FindClose(r);
      return INVALID_HANDLE_VALUE;
    }
  }

  return r;
}
```

Hiding Files

- FindNextFile Hook

```
BOOL my_FindNextFile(HANDLE hFindFile, LPWIN32_FIND_DATA lpFindFileData, DWORD fsz)
{
  BOOL r;

  get_hdata_from_handle();
  get_old_FindNextFile_from_handle_pci();
  pvobj = get_ctx_from_hdata();

  if(is_current_process_root_process())
    return call_old_FindNextFile();

  for(r = call_old_FindNextFile();
      r && returned_file_name_contains_hide_mask();
      r = call_old_FindNextFile();

  return r;
}
```

Hiding Registry Items

- We're interested in SH_FILESYS_APIS API Set

```
[SH_FILESYS_API]
fc=21,sig=0x908dd90800a - RegEnumValue
  (ARG_DW, ARG_DW, ARG_O_PTR, ARG_DW, ARG_O_PDW,
   ARG_IO_PDW, ARG_IO_PDW, ARG_O_PTR, ARG_DW, ARG_O_PDW)

fc=22,sig=0x9908d90800a - RegEnumKeyEx
  (ARG_DW, ARG_DW, ARG_O_PTR, ARG_DW, ARG_O_PDW,
   ARG_IO_PDW, ARG_O_PTR, ARG_DW, ARG_O_PDW, ARG_O_PDW)
```

Hiding Registry Items

- RegEnumValue Hook

```
LONG my_RegEnumValue(HKEY hKey, DWORD dwIndex, LPWSTR lpValueName...
{
  LONG r;

  if(is_current_process_root_process())
    return call_old_RegEnumValue();

  // adjust index because hidden entries can be before the current index
  adjust_index_by_hidden_entries();

  for(;;)
  {
    r = call_old_RegEnumValue();
    if(r)
      return r;

    if(returned_value_name_contains_hide_mask())
      return r;

    dwIndex++;
  }
}
```

Hiding Registry Items

- RegEnumKeyEx Hook

```
LONG my_RegEnumKeyEx(HKEY hKey, DWORD dwIndex, LPWSTR lpName, DWORD cbNameIn...
{
  LONG r;

  if(is_current_process_root_process())
    return call_old_RegEnumKeyEx();

  // adjust index because hidden entries can be before the current index
  adjust_index_by_hidden_entries();

  for(;;)
  {
    r = call_old_RegEnumKeyEx();
    if(r)
      return r;

    if(returned_value_name_contains_hide_mask())
      return r;

    dwIndex++;
  }
}
```

Hiding Open Ports

- We're interested in AFD API Set

```
[AFD]
fc=3,sig=0x90804007 - Afd_Unknown -)
  (ARG_DW, ARG_DW, ARG_I_PTR, ARG_DW, ARG_O_PTR, ARG_DW, ARG_O_PDW)
```

Hiding Open Ports

- AFD_Unknown Hook

```
DWORD my_Afd_Unknown(DWORD u1, DWORD u2, DWORD u3, DWORD u4, DWORD u5, DWORD u6, DWORD u7)
{
  DWORD r;

  if(is_current_process_root_process())
    return old_Afd_Unknown();
  r = old_Afd_Unknown();

  if(size_of_input_structure_doesnt_match())
    return r;

  switch(call_reason())
  {
  case MIB_TCP_TABLE:
  { // hide mib_tcptable }
  case MIB_UDPTABLE:
  { // hide mib_udptable }
  default: return r;
  }
  return r;
}
```

Hiding Processes

- Processes are linked in global double linked list

```
void hide_process()
{
  for(every_process_in_process_list())
  {
    if(process_is_nk())
      break;

    if(process_name_contains_hide_mask())
    {
      remove_from_double_linked_list();
    }
  }
}
```

Hiding Modules

- Modules are linked in per-process double linked list

```
void hide_our_module_from_nk()
{
  for(every_module_in_process_module_list())
  {
    if(module_name_contains_hide_mask())
    {
      remove_from_double_linked_list();
    }
  }
}
```

Demo

- Detects API Set hooks by checking ”where” API Set function pointers point to
- Detects process unlinking by comparing process handles in kernel handle table with global process list
- Not able to detect prefetch abort handler hooking method

- Works similar to Rootkit Revealer by microsoft
- Diffs the raw SD Card on-disk data with API output
- Contains built-in FAT filesystem code
- If file is on SD Card but not on API output it is hidden

- As Win32 counterparts FileMon and RegMon show accessed registry and filesystem entries
- Work by hooking common registry and filesystem APIs
- Rootkit-like behaviour for good purpose

- Windows Embedded CE 6 Dumping Utility

Dumps

- KDataStruct Table
- Kernel Memory Info
- API Sets (signatures included)
- Active Process (modules, threads, open handles, VirtualAddresses List)
- Page Tables

Demo

- Every system can be rootkited.
- Every rootkit can be detected.
- There are improvements in Windows CE Embedded 6 that inherently ease the rootkit development.
- Rootkits and Detectors have been for Windows CE Embedded 6 have been presented.
- Aditional debugging software that can help finding rootkits have been presented.

Any questions?

petr@research.coseinc.com

Interested in Windows Mobile Rootkit and Malware Training? -)

Interested to be my colleague? We are HIRING ...

Contact my Boss - thomas@coseinc.com

Thank you.