INTERNALS SESSIONS 02: Syscalls Journey in Windows

Abolfazl Kazemi

Mahsan
Your Support
In Digital World

mahsan.co
info@mahsan.co

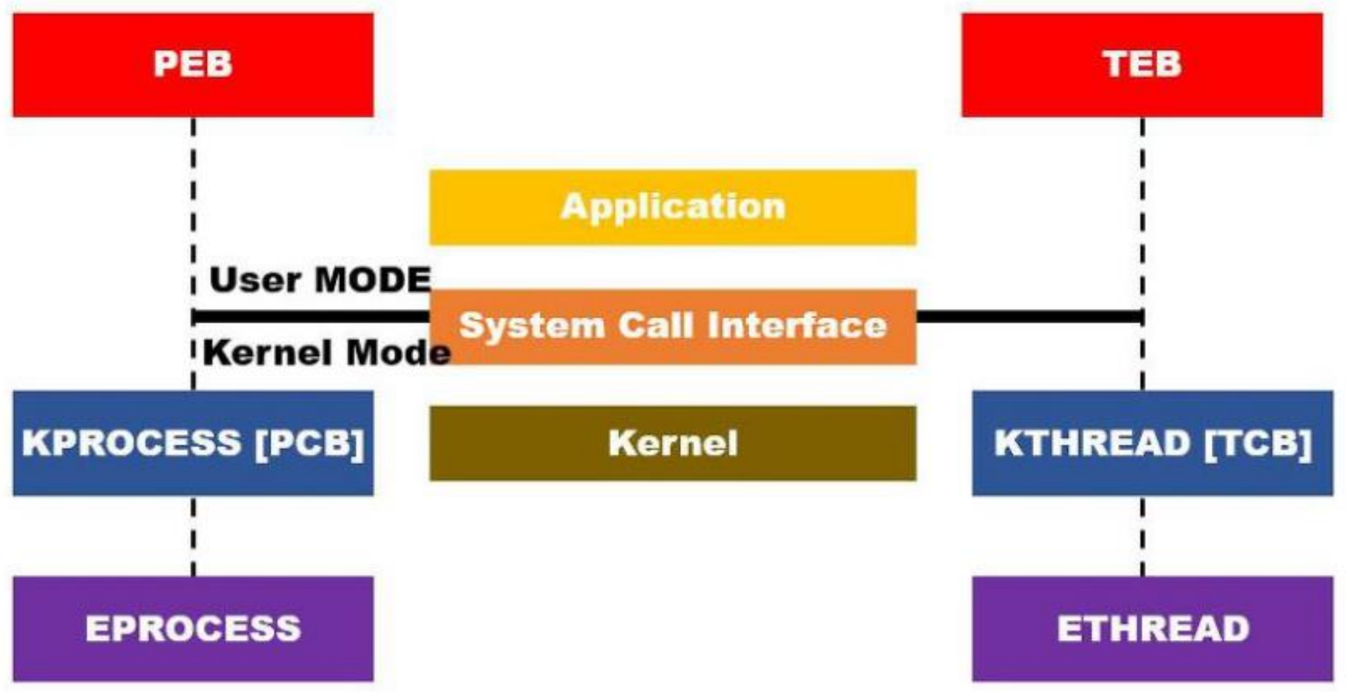
Agenda

مهسان
تکیه گاه شما
در دنیای هوشمند

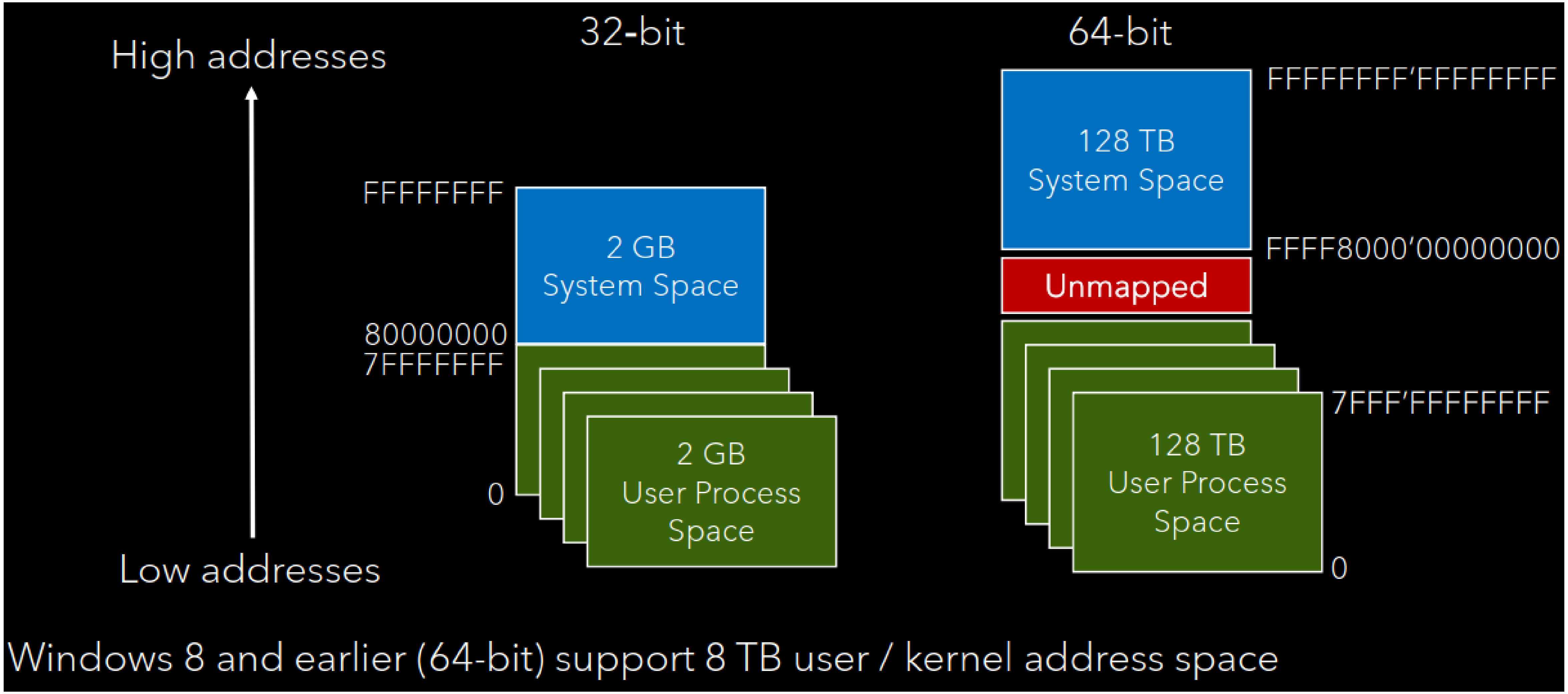
User vs Kernel Space

- **User mode**
  - Allows access to non-operating system code & data only
  - No access to the hardware
  - Protects user applications from crashing the system

- **Kernel mode**
  - Privileged mode for use by the kernel and device drivers only
  - Allows access to all system resources
  - Can potentially crash the system



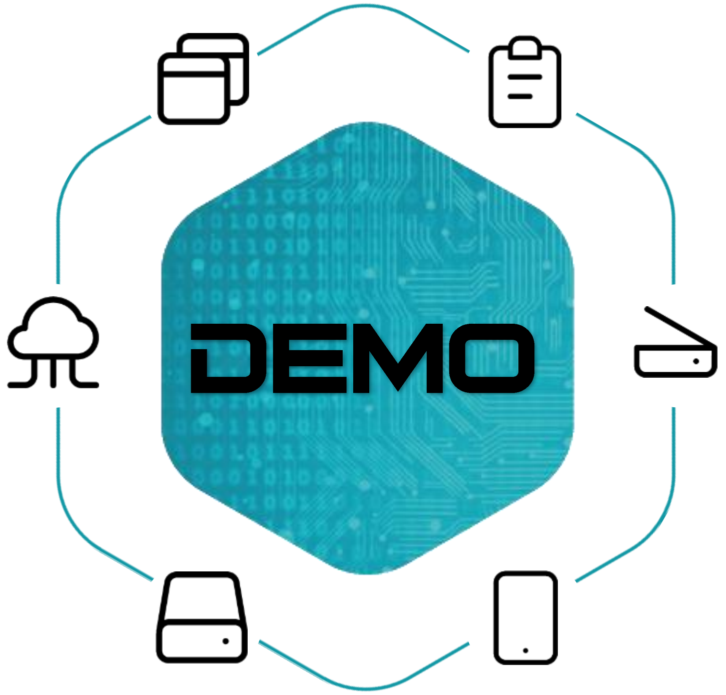
Virtual Memory Layout



Windows 8 and earlier (64-bit) support 8 TB user / kernel address space

Tools of the Trade

- Built-in Windows tools
  - Task manager, Performance Monitor, …
- *SysInternals* (www.sysinternals.com)
  - Process Explorer, Process Monitor, Debug View, …
- Debugging Tools for Windows (Part of the Windows SDK)
  - WinDbg, cdb, ntsd, kd, gflags, …
- Pavel's tools (github.com/zodiacon/AllTools)
  - Total PE, PoolmonX, System Explorer, GflagsX, …



Working with ProcessExplorer/ProcessMonitor

Introduction to WinDbg

- Part of the "Debugging Tools for Windows" package
- Package contains four debuggers
  - Cdb, Ntsd, Kd, WinDbg
  - All based on the same engine (*DbgEng.dll*)
- New **WinDbg Preview** can be downloaded from the Microsoft Store
  - Requires Windows 10 version 1607 or later to run
- *WinDbg* is a standalone GUI debugger
  - Used by Microsoft to debug Windows itself
  - User mode or kernel mode debugger
- UI windows
  - Command – most important window
  - Call Stack, Processes & Threads, Source, Locals, Watch, Registers, others
- Command window can do anything
  - Some shortcuts available through the GUI

WinDbg Commands

- Regular Commands
  - Intrinsic to the debugger engine
  - Have no prefix
  - Work on the debugged target
- Meta Commands
  - Work on the debugger itself or the environment of debugging
  - Prefixed with a dot (.)
- Extension Commands (“bang” commands)
  - Supplied by extension (custom) DLLs
  - Prefixed with an exclamation mark (!)
  - Some extension DLLs are loaded automatically

Basic WinDbg Commands

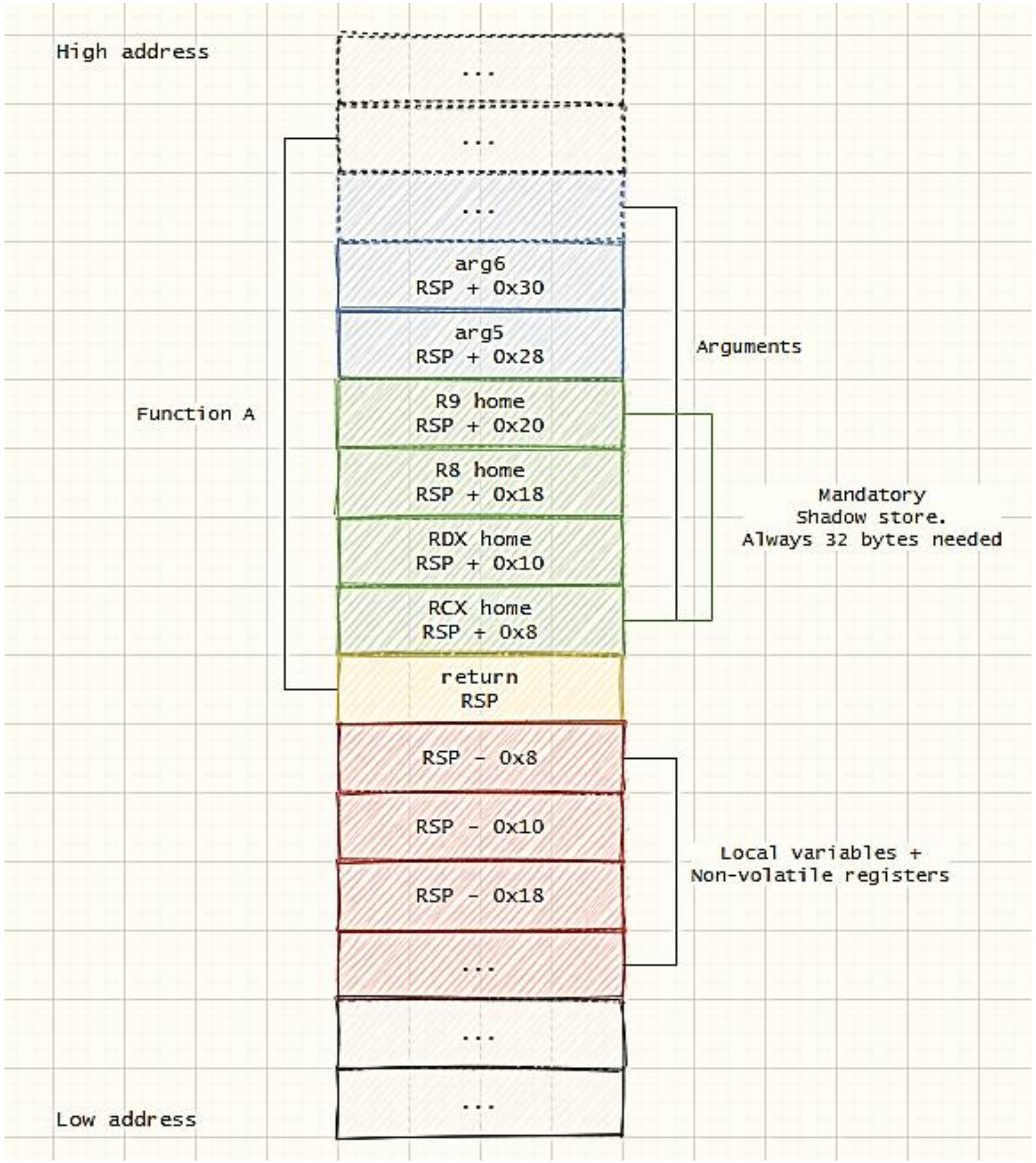
Command(s)	Description
`~`	Show threads in process
`k`	Display call stack of current thread
`~`*n*`k`	Display call stack of thread *n*
`~`*n*`s`	Switch to thread *n*
`bp [module!]function`	Set a breakpoint based on a symbol
`bl, bc, bd, be`	Breakpoint list, delete (clear), disable, enable
`dt [module!]type [address]`	Display type information (and values if *address* is specified)
`db, du, dd, dq, dp`	Display as byte, UTF-16 string, DWORD (32 bit), Quad-word (64 bit), Pointer-size (4 bytes on 32-bit, 8 byte on 64-bit)
`lm`	Display loaded modules (with module symbol status)
`lmvm` *`modulename`*	Display detailed information for a module
`.reload`	Reload symbols (if configuration changed)
`.reload [/f]` *`module.ext`*	Reload (forced) symbols for a specific module

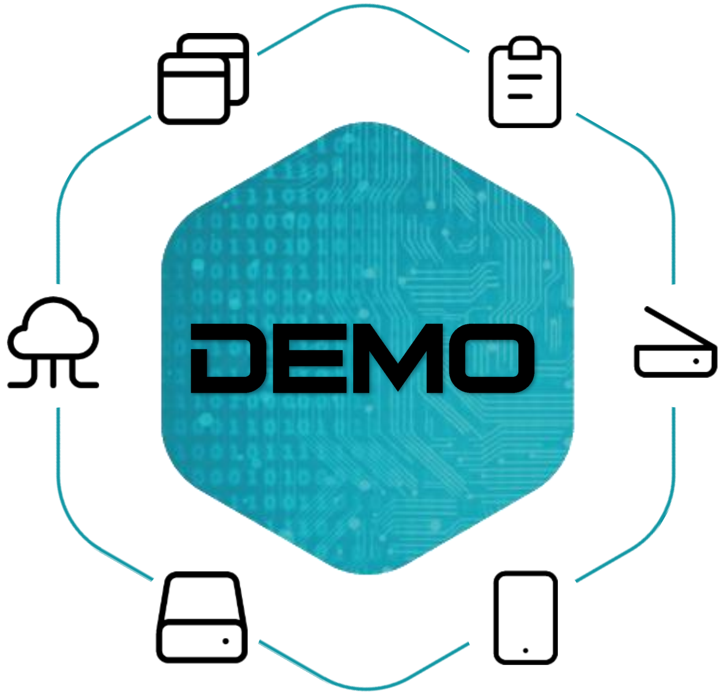
Configuring Symbols

- Debugging symbols come in several flavors
  - Full program database (PDB files)
  - Public symbols only (PDB files)
  - Exported symbols only (in the DLL itself)
- Select File->Symbol File Path…
  - Add search folders as appropriate
- Automatically uses the _NT_SYMBOL_PATH environment variable
- To get the symbols of the OS DLLs automatically, add the following string
  SRV*C:\Symbols*http://msdl.microsoft.com/download/symbols

x64 Stack Frame

- Integer arguments are passed in registers RCX, RDX, R8, and R9
- The rest of the parameters are stored on the stack.

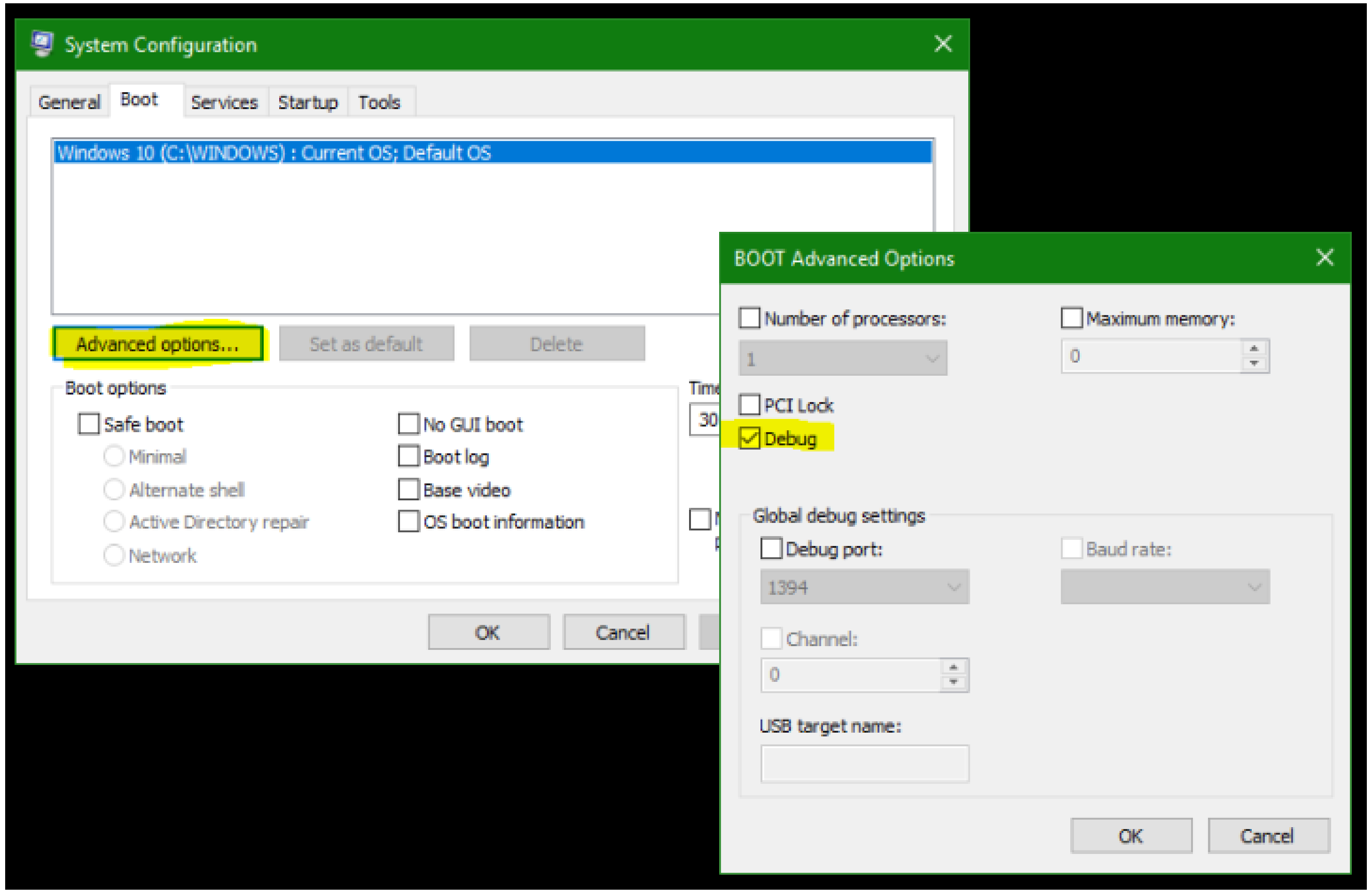




Symbol files and user mode debugging with WinDbg

Local Kernel Debugging

- Configure machine for kernel debugging
  - Run *MsConfig.exe* (GUI) or *bcdedit.exe* from an elevated command window
    - bcdedit –debug on
  - Reboot
- Launch *WinDbg* elevated
  - File | Kernel Debug… | Local


System Configuration

General | Boot | Services | Startup | Tools

Windows 10 (C:\WINDOWS) : Current OS; Default OS

Advanced options… | Set as default | Delete

Boot options
- [ ] Safe boot
  - ○ Minimal
  - ○ Alternate shell
  - ○ Active Directory repair
  - ○ Network
- [ ] No GUI boot
- [ ] Boot log
- [ ] Base video
- [ ] OS boot information

Time 30

OK | Cancel

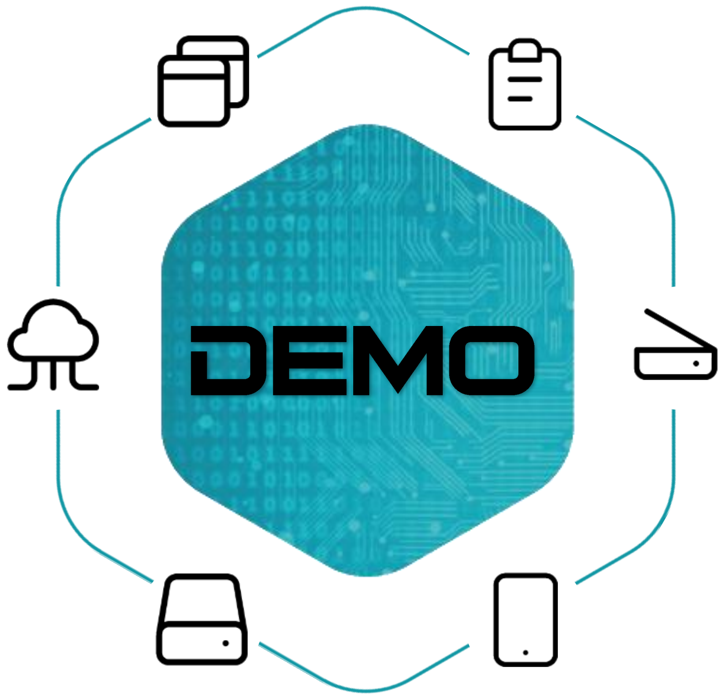
BOOT Advanced Options
- [ ] Number of processors: 1
- [ ] Maximum memory: 0
- [ ] PCI Lock
- [✓] Debug

Global debug settings
- [ ] Debug port: 1394
- [ ] Baud rate:
- [ ] Channel: 0
- USB target name:

OK | Cancel

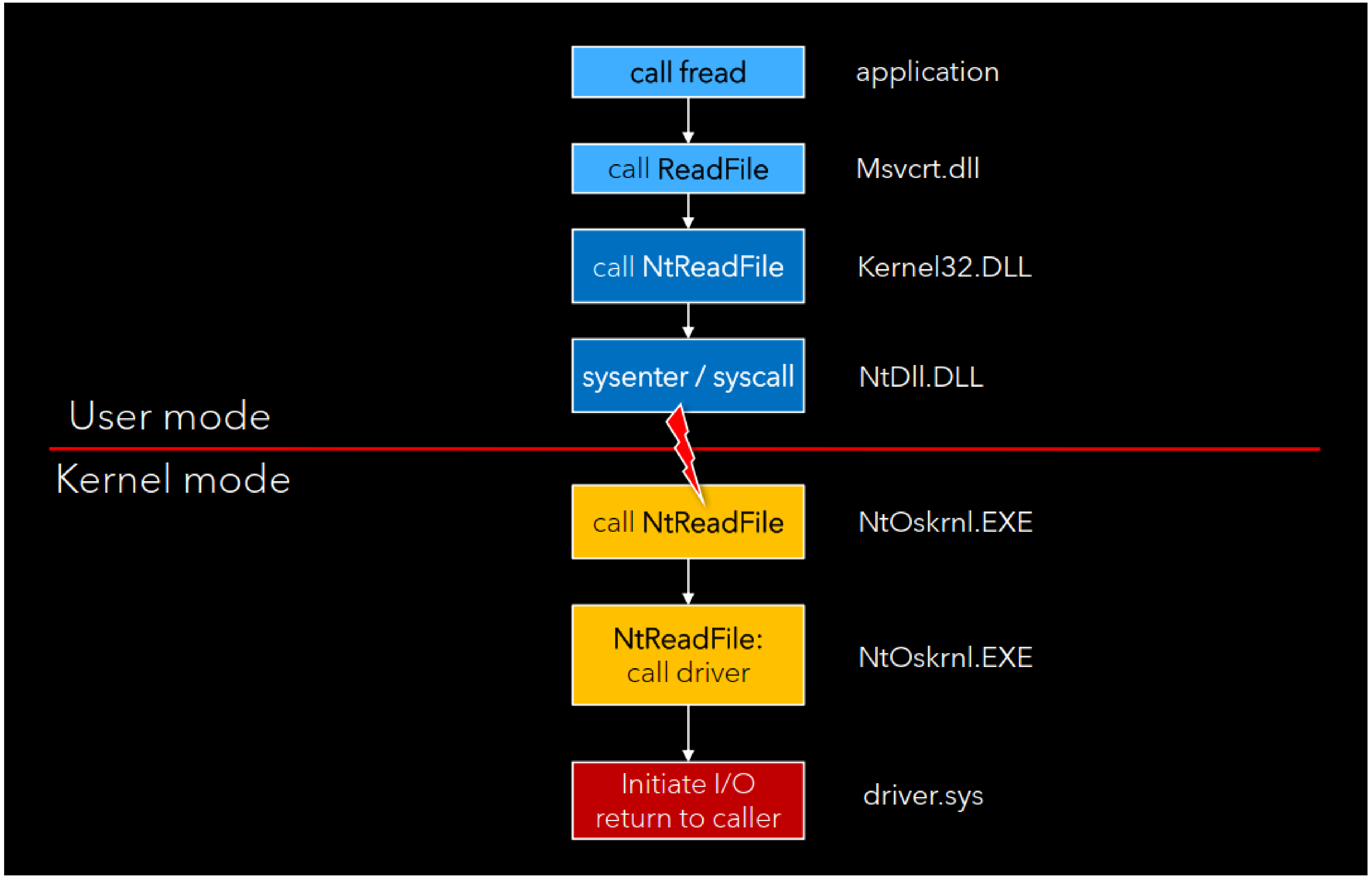
Remote Kernel Debugging

- Target machine
  - Configure for debugging as before
  - Select a communication medium
    - msconfig.exe or bcdedit.exe /dbgsettings
    - Serial, USB, Network (Windows 8+)
- Host machine
  - File | Kernel Debug…
  - Select configured communication medium
- If target is a virtual machine
  - Can expose a VM COM port as a host named pipe
  - Or use network if target is Windows 8+

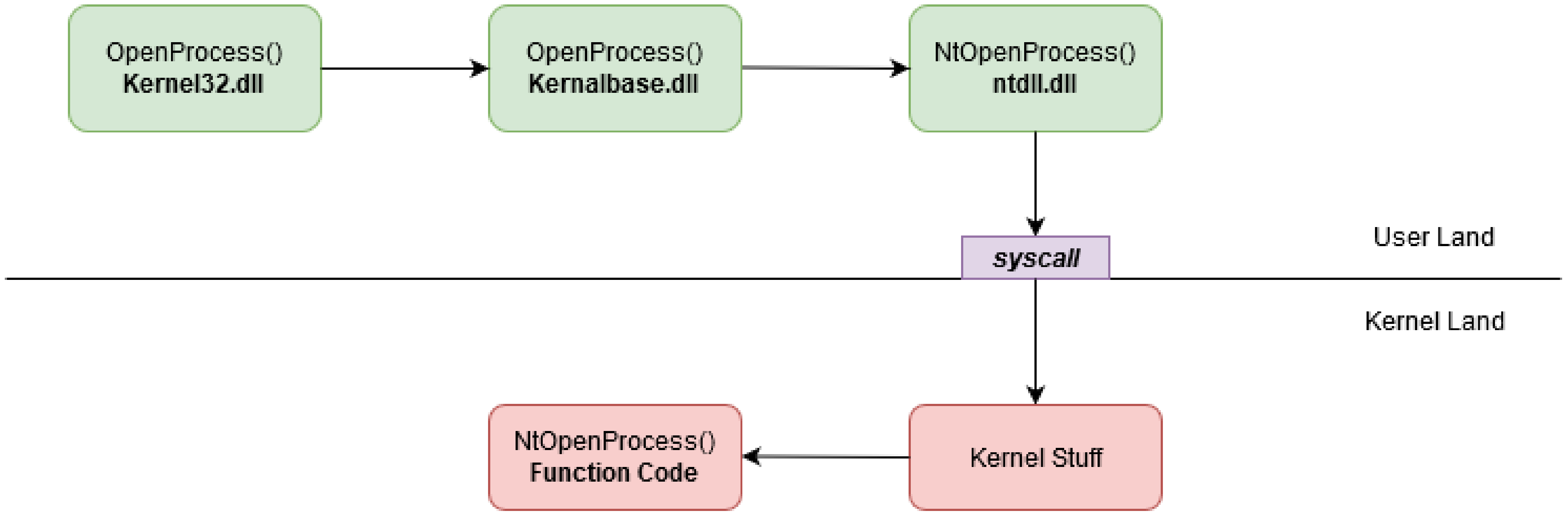


Local and Remote Kernel Debugging

Function Call Flow

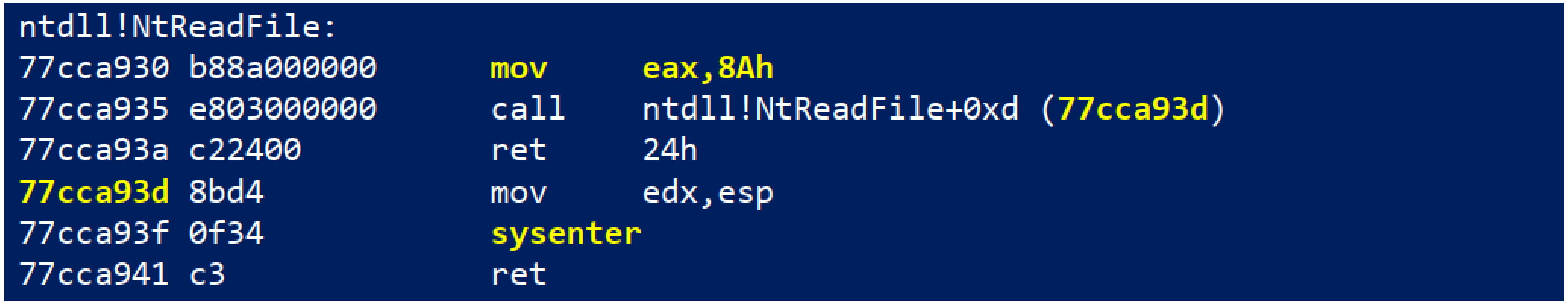


Function Call Flow



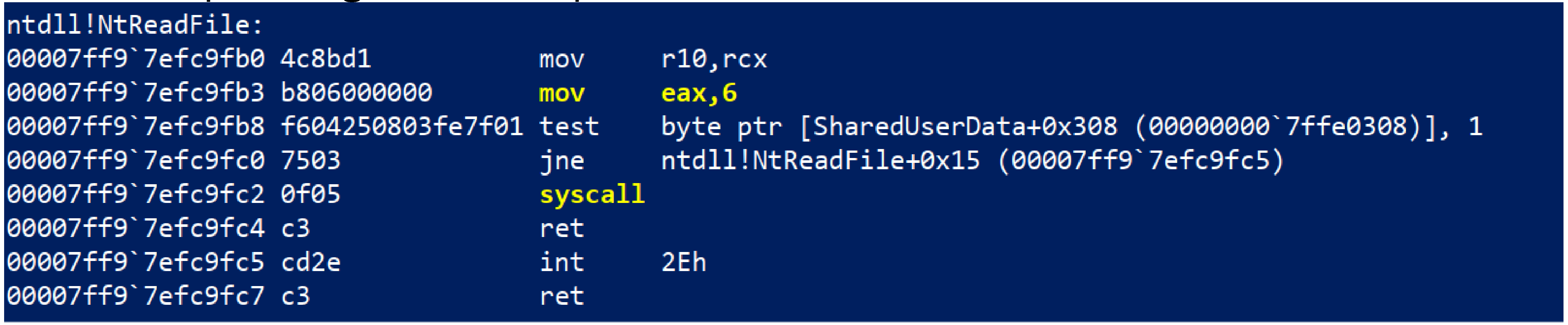
NTDLL.DLL Kernel Gate

- 32-bit dispatching code example (Windows 8.1)



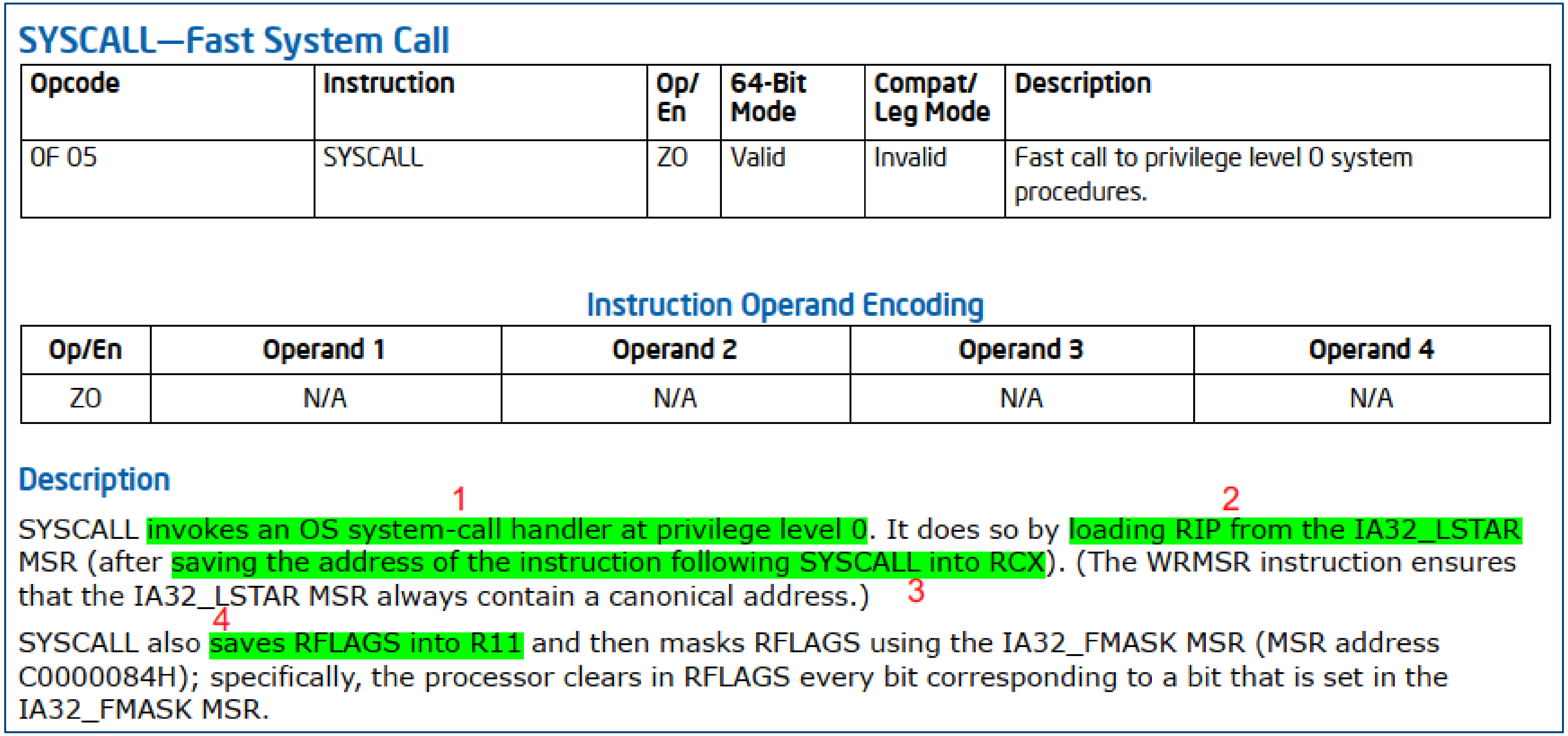
```
ntdll!NtReadFile:
77cca930 b88a000000      mov     eax,8Ah
77cca935 e803000000      call    ntdll!NtReadFile+0xd (77cca93d)
77cca93a c22400          ret     24h
77cca93d 8bd4            mov     edx,esp
77cca93f 0f34            sysenter
77cca941 c3              ret
```

- 64-bit dispatching code example (Windows 10)



```
ntdll!NtReadFile:
00007ff9`7efc9fb0 4c8bd1           mov     r10,rcx
00007ff9`7efc9fb3 b806000000       mov     eax,6
00007ff9`7efc9fb8 f604250803fe7f01 test    byte ptr [SharedUserData+0x308 (00000000`7ffe0308)], 1
00007ff9`7efc9fc0 7503             jne     ntdll!NtReadFile+0x15 (00007ff9`7efc9fc5)
00007ff9`7efc9fc2 0f05             syscall
00007ff9`7efc9fc4 c3               ret
00007ff9`7efc9fc5 cd2e             int     2Eh
00007ff9`7efc9fc7 c3               ret
```

syscall instruction



SYSCALL—Fast System Call

Opcode	Instruction	Op/ En	64-Bit Mode	Compat/ Leg Mode	Description
0F 05	SYSCALL	ZO	Valid	Invalid	Fast call to privilege level 0 system procedures.

Instruction Operand Encoding

Op/En	Operand 1	Operand 2	Operand 3	Operand 4
ZO	N/A	N/A	N/A	N/A

Description

1 2

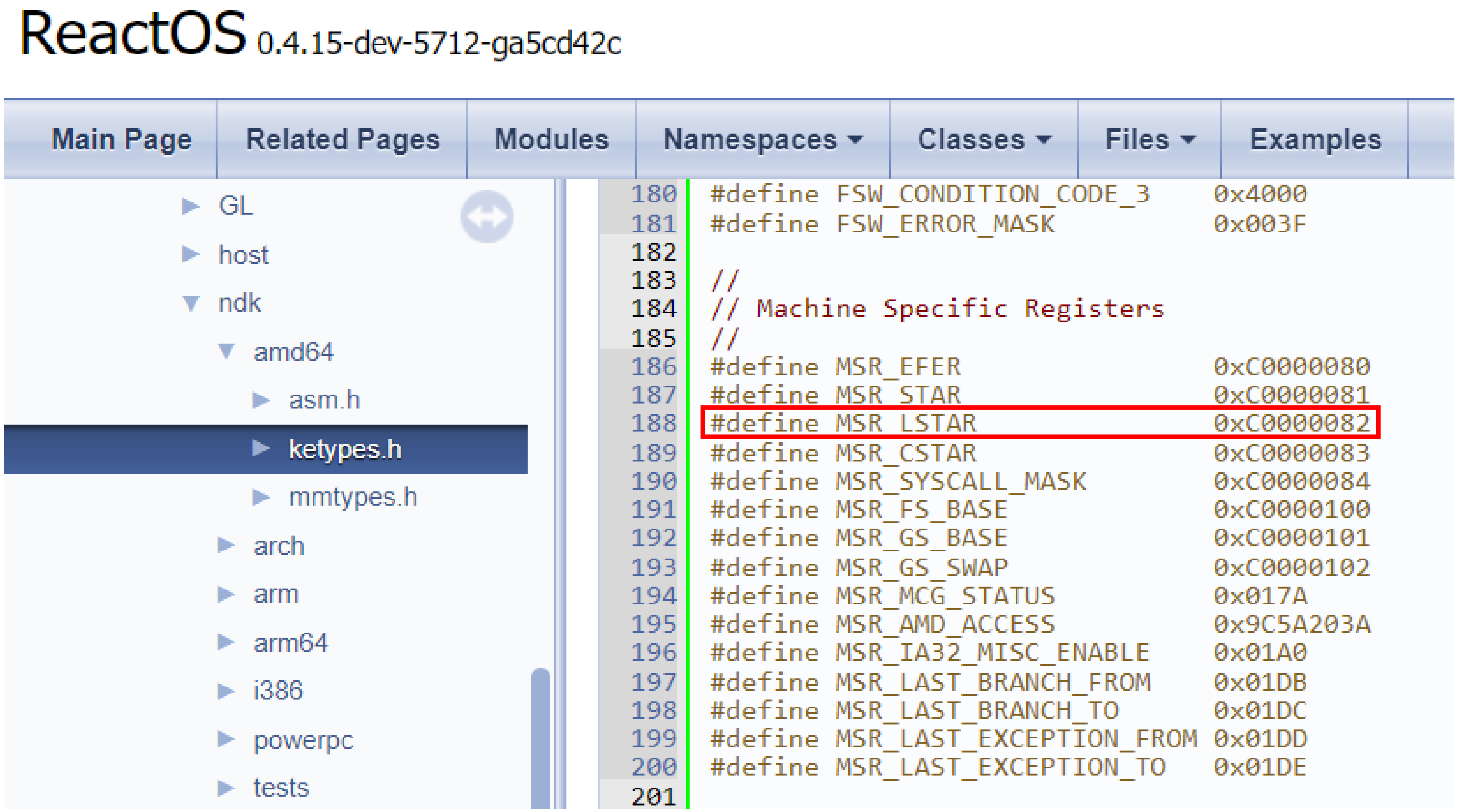
SYSCALL invokes an OS system-call handler at privilege level 0. It does so by loading RIP from the IA32_LSTAR MSR (after saving the address of the instruction following SYSCALL into RCX). (The WRMSR instruction ensures that the IA32_LSTAR MSR always contain a canonical address.) 3

4

SYSCALL also saves RFLAGS into R11 and then masks RFLAGS using the IA32_FMASK MSR (MSR address C0000084H); specifically, the processor clears in RFLAGS every bit corresponding to a bit that is set in the IA32_FMASK MSR.

Source: Intel Software Developer’s Manual

MSR_LSTAR Register

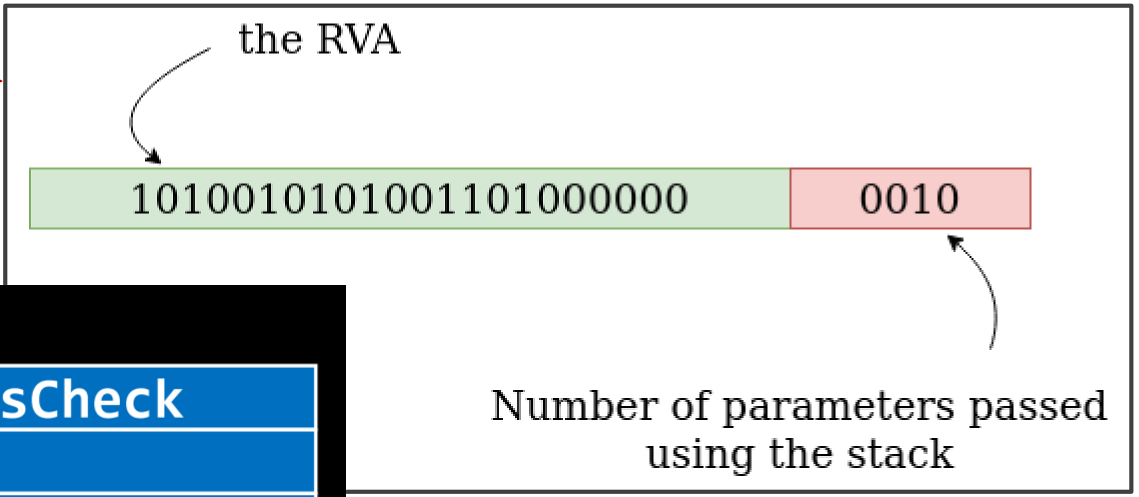


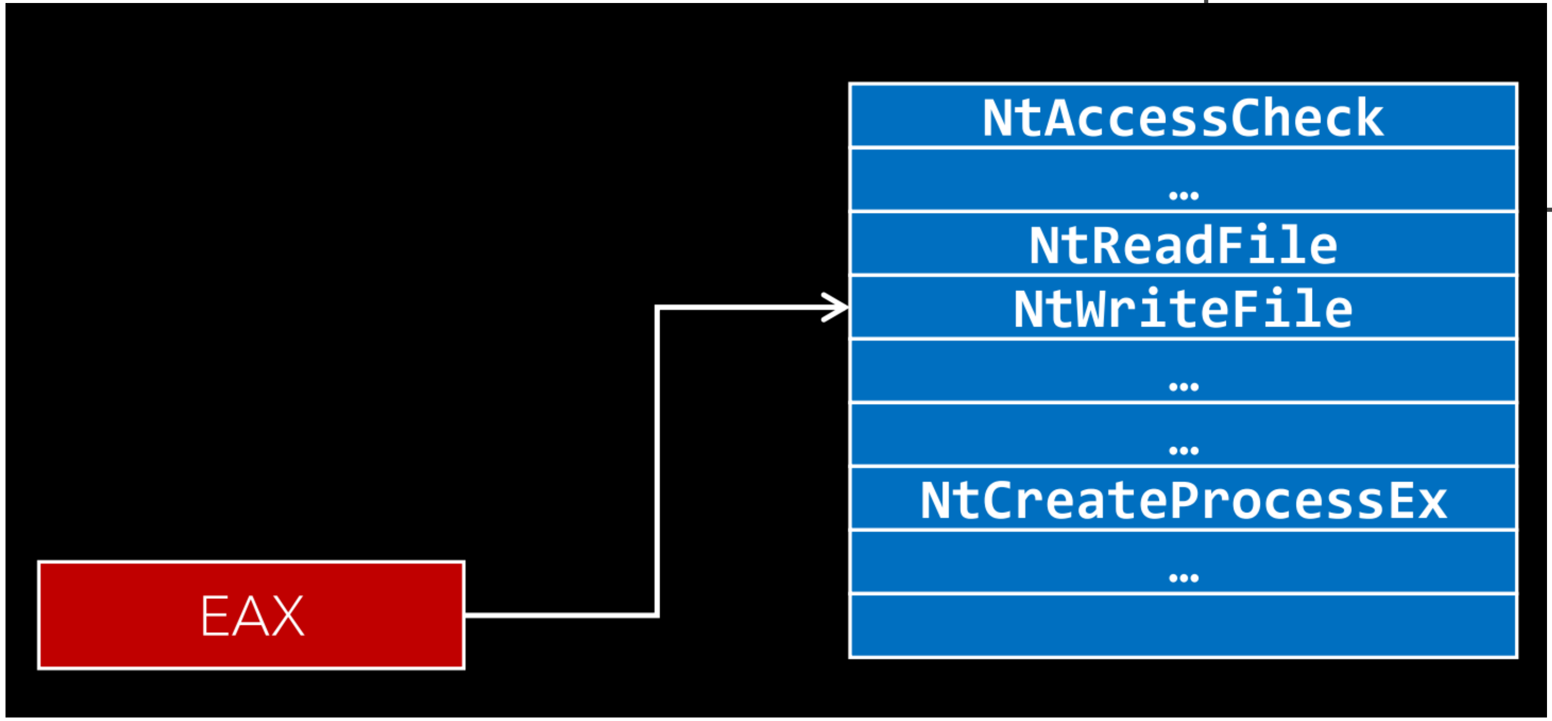
```
#define FSW_CONDITION_CODE_3     0x4000
#define FSW_ERROR_MASK           0x003F

//
// Machine Specific Registers
//
#define MSR_EFER                 0xC0000080
#define MSR_STAR                 0xC0000081
#define MSR_LSTAR                0xC0000082
#define MSR_CSTAR                0xC0000083
#define MSR_SYSCALL_MASK         0xC0000084
#define MSR_FS_BASE              0xC0000100
#define MSR_GS_BASE              0xC0000101
#define MSR_GS_SWAP              0xC0000102
#define MSR_MCG_STATUS           0x017A
#define MSR_AMD_ACCESS           0x9C5A203A
#define MSR_IA32_MISC_ENABLE     0x01A0
#define MSR_LAST_BRANCH_FROM     0x01DB
#define MSR_LAST_BRANCH_TO       0x01DC
#define MSR_LAST_EXCEPTION_FROM  0x01DD
#define MSR_LAST_EXCEPTION_TO    0x01DE
```

System Service Table

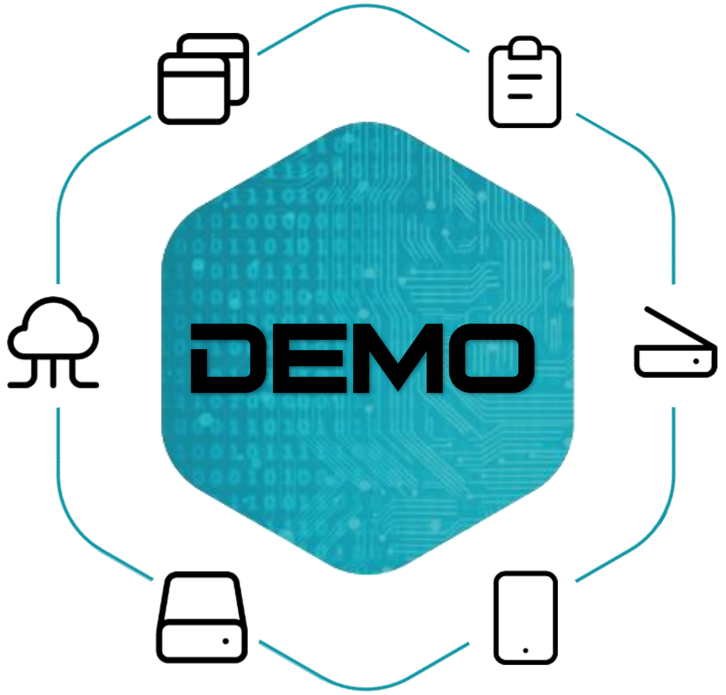
- On Intel/AMD CPUs, EAX serves as an index to the system service required
  - 32-bit Windows holds the actual addresses
  - 64-bit Windows holds offsets (32 bit)
    - Lower 4 bits used as argument count
    - Must be masked off to get actual offset



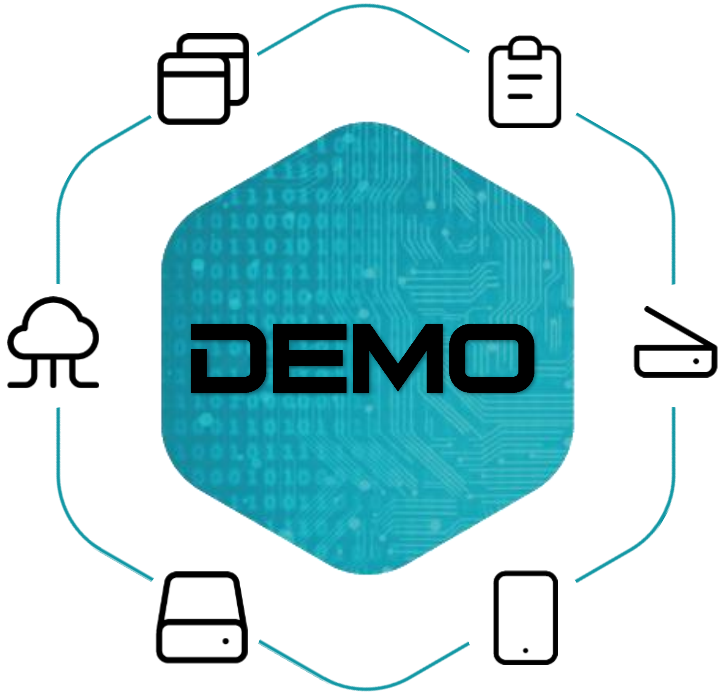


Default System Tables

- KeServiceDescriptorTable
  - Contains only kernel related entries
  - Value is KiServiceTable
- KeServiceDescriptorTableShadow
  - Contains both kernel entries and USER and GDI entries
- When a thread is first created, it uses KeServiceDescriptorTable
- The first time it makes any GDI or USER call, it starts using KeServiceDescriptorTableShadow



Examining Windows System Service Table



Direct Syscall Example in Windows

Resources

- ❑ Pavel Yosifovich Slides
- ❑ https://alice.climent-pommeret.red/posts

Thanks

Mahsan

Your Support In Digital World