# Foundation concepts and terms

- Windows API
- Services, functions, and routines
- Processes
- Threads
- Jobs
- Virtual memory
- Kernel mode vs. user mode
- Hypervisor
- Firmware
- Terminal Services and multiple sessions
- Objects and handles
- Security
- Registry
- Unicode

7eRoM

# Windows API

## Windows Application Programming Interface

- the user-mode system programming interface
- 16-bit, 32-bit, 64-bit windows APIs

## Component Object Model (COM)

- clients communicate with objects (sometimes called COM server objects) through interfaces
- component implementation is loaded dynamically rather than being statically linked to the client

7eRoM

# Windows API

## Windows Runtime (WinRT)

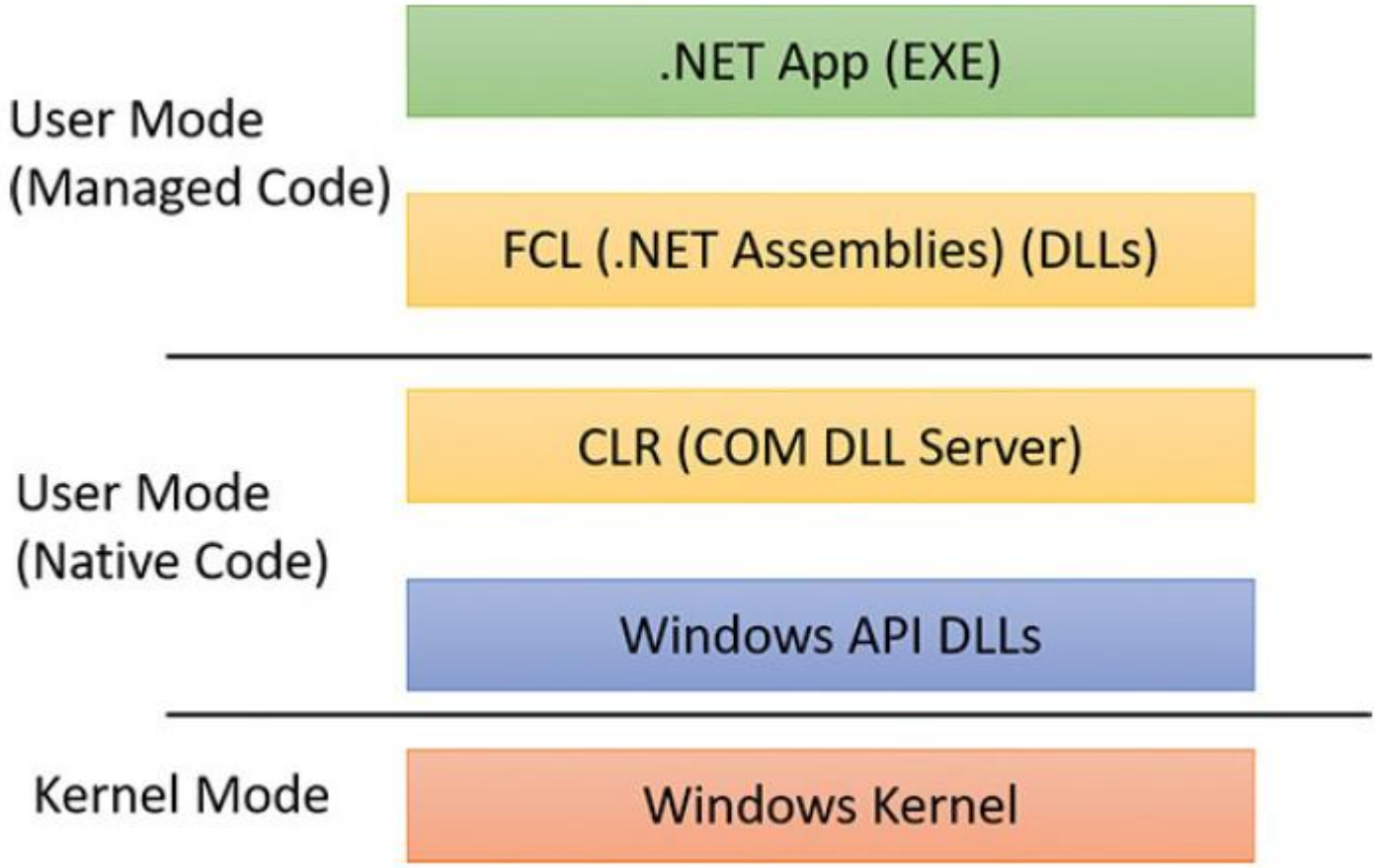
- a new API consists of platform services aimed particularly at app developers for the so-called Windows Apps
- is built on top of COM

## .NET Framework

- The Common Language Runtime (CLR)
- The .NET Framework Class Library (FCL)

7eRoM

# Windows API




7eRoM

# Services, functions, and routines

Windows API

Native system services (or system calls)

Kernel support functions (or routines)

Windows services

Dynamic link libraries (DLLs)

7eRoM

# Processes

A Windows process comprises the following:

- A private virtual address space
- An executable program
- A list of open handles
- A security context
- A process ID
- At least one thread of execution

7eRoM

# Threads

- A thread includes the following essential components:
  - The contents of a set of CPU registers
  - Two stacks
  - A private storage area called thread-local storage (TLS)
  - A unique identifier called a thread ID
- Context:
  - volatile registers, stacks, and private storage area
  - `GetThreadContext` function provides access to `CONTEXT` block

7eRoM

# Threads

- Fibers:
  - lightweight threads
  - an application to schedule its own threads
  - Important APIs:
    - `ConvertThreadToFiber`
    - `CreateFiber`
    - `SwitchToFiber`
  - not a good idea:
    - shared thread local storage (TLS)
    - I/O-bound
    - concurrency

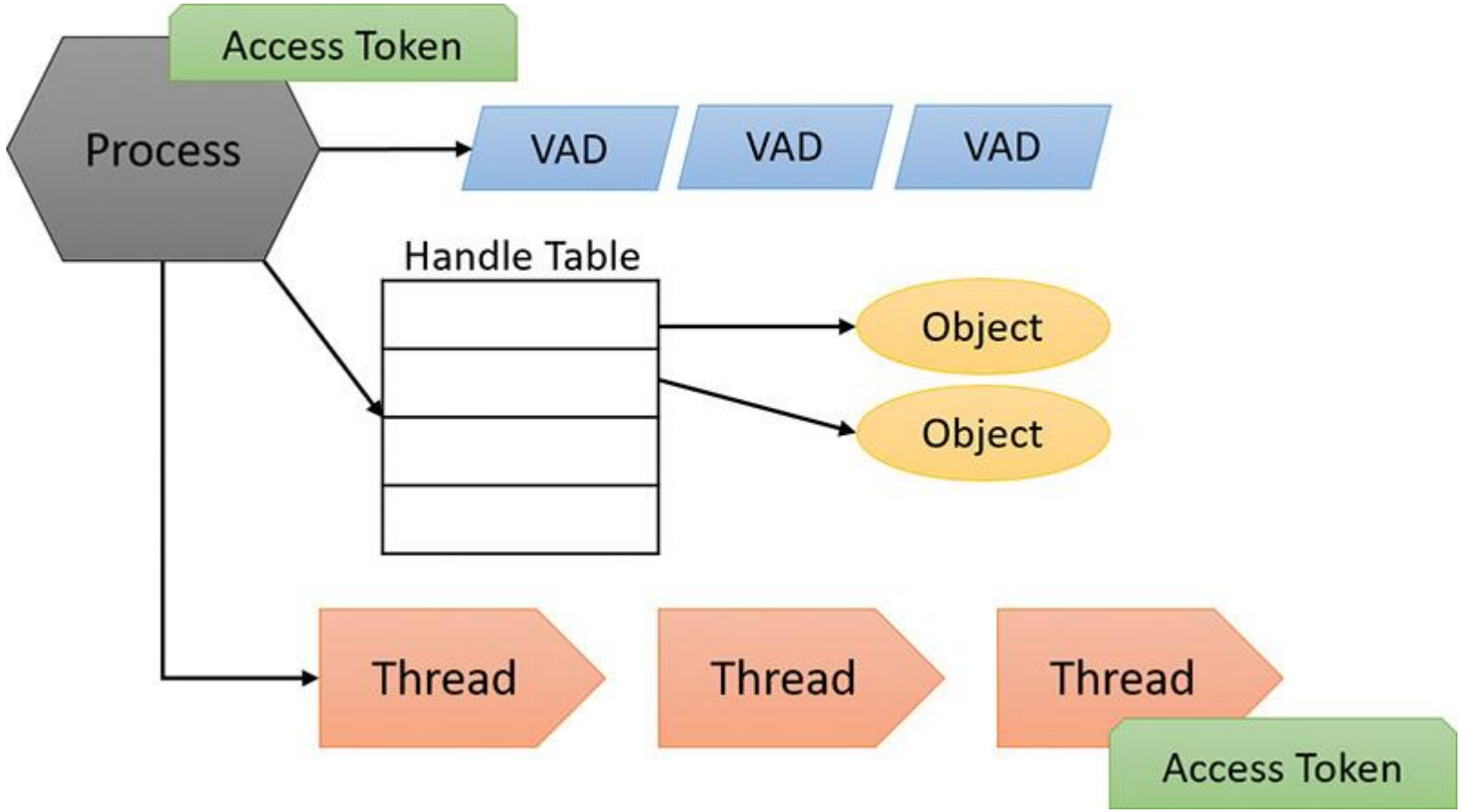
7eRoM

# Threads

- User-mode scheduling (UMS):
  - Provide the same basic advantages as fibers
  - Visible to the kernel
  - Allows multiple UMS threads to issue blocking system calls and share and contend on resources
  - Scheduling in user-mode
  - Cannot run on multiple processors

7eRoM

# Threads



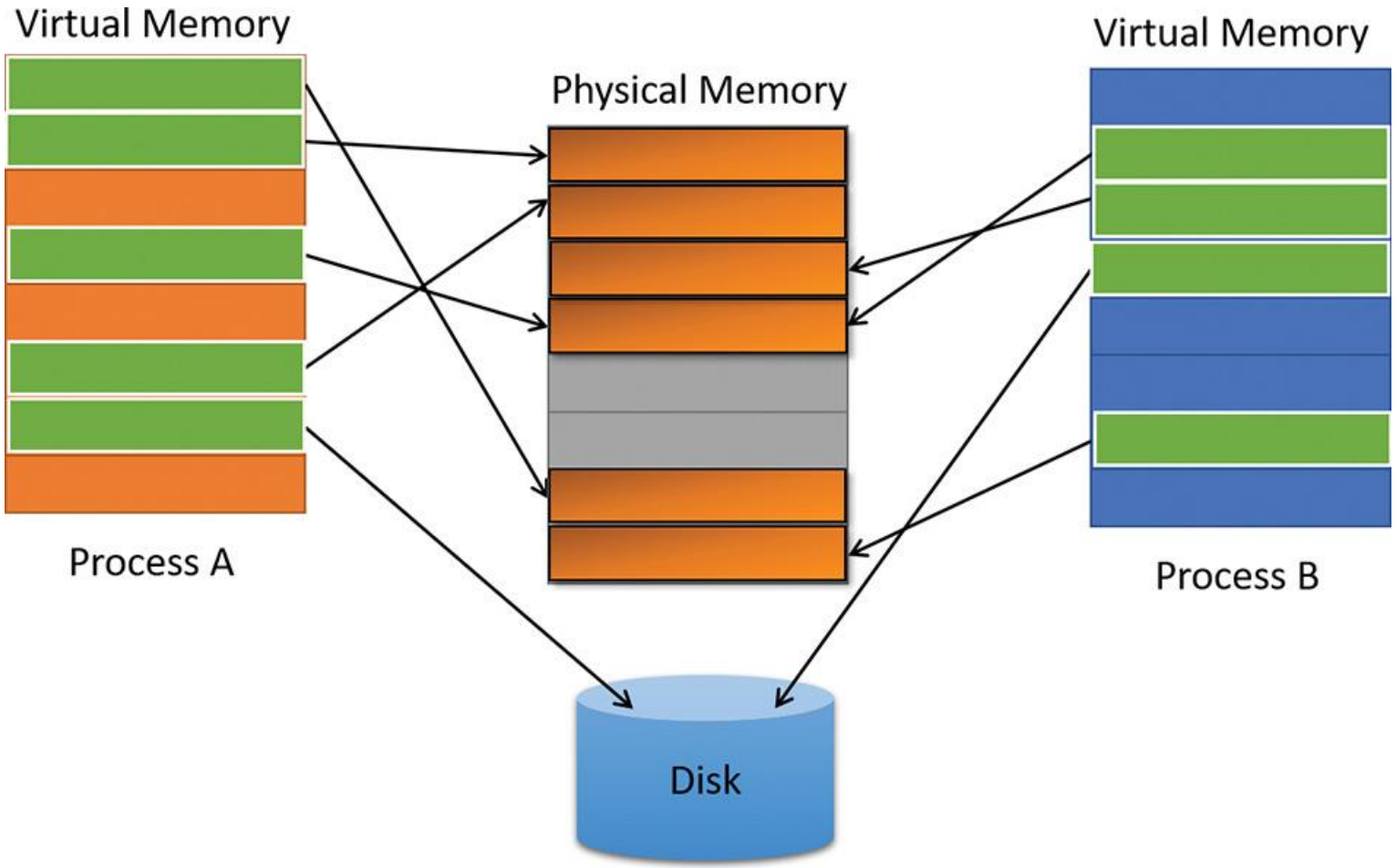

7eRoM

# Jobs

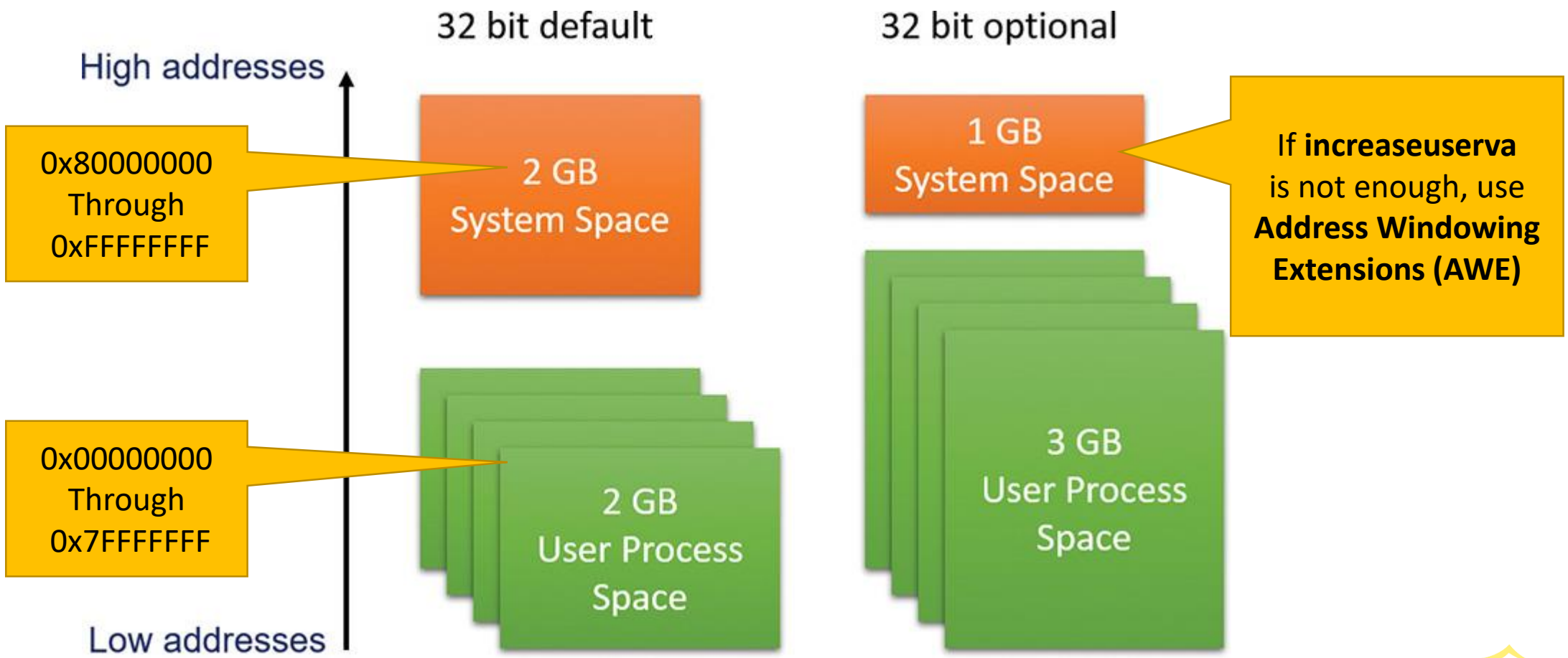
- management and manipulation of groups of processes as a unit
- control of certain attributes and provides limits for the process(es)
- records basic accounting information for all running or terminated processes associated with the job

7eRoM

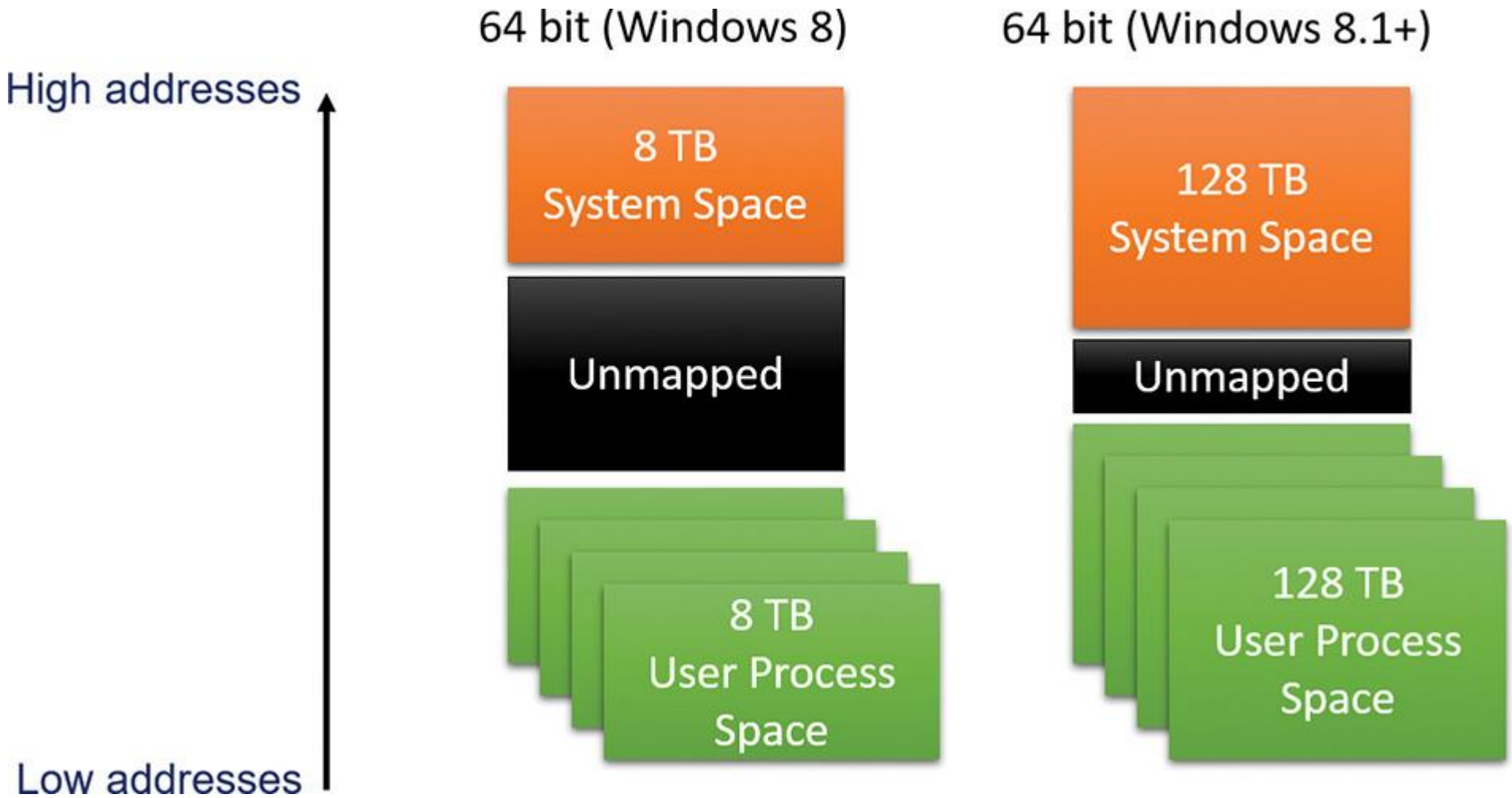
# Virtual memory



# Virtual memory



# Virtual memory




7eRoM

# Kernel mode vs. user mode

- To protect user applications from accessing and/or modifying critical OS data.
- User application code runs in user mode.
- OS code (such as system services and device drivers) runs in kernel mode
- Some processors differentiate between such modes by using the term code privilege level or ring level, while others use terms such as supervisor mode and application mode.
- Although each Windows process has its own private memory space, the kernel-mode OS and device-driver code share a single virtual address space.
- Pages in system space can be accessed only from kernel mode, whereas all pages in the user address space are accessible from user mode and kernel mode.

7eRoM

# Kernel mode vs. user mode

| Object: Counter | Function |
|---|---|
| Processor: % Privileged Time | Percentage of time that an individual CPU (or all CPUs) has run in kernel mode during a specified interval |
| Processor: % User Time | Percentage of time that an individual CPU (or all CPUs) has run in user mode during a specified interval |
| Process: % Privileged Time | Percentage of time that the threads in a process have run in kernel mode during a specified interval |
| Process: % User Time | Percentage of time that the threads in a process have run in user mode during a specified interval |
| Thread: % Privileged Time | Percentage of time that a thread has run in kernel mode during a specified interval |
| Thread: % User Time | Percentage of time that a thread has run in user mode during a specified interval |

**Mode-related performance counters**

7eRoM

# Hypervisor

- Is a specialized and highly privileged component that allows for the virtualization and isolation of all resources on the machine, from virtual to physical memory, to device interrupts, and even to PCI and USB devices.
- Microsoft leverages the Hyper-V hypervisor to provide a new set of services known as virtualization-based security (VBS):
  - Device Guard
  - Hyper Guard
  - Credential Guard
  - Application Guard
  - Host Guardian and Shielded Fabric
- Virtual Trust Levels (VTLs)

7eRoM

# Firmware

- Load Windows components securely and can authenticate their contents.

- Provides a root chain of trust that can guarantee an unencumbered boot process

7eRoM

# Terminal Services and multiple sessions

- The support in Windows for multiple interactive user sessions on a single system.
- The first session is considered the services session, or session zero, and contains system service hosting processes.
- The first login session at the physical console of the machine is session one.
- Windows client editions permit a single remote user to connect to the machine.
- Windows server systems support two simultaneous remote connections.

7eRoM

# Objects and handles

- A kernel object is a single, run-time instance of a statically defined object type.
- An object type comprises a system-defined data type, functions that operate on instances of the data type, and a set of object attributes.
- An object attribute is a field of data in an object that partially defines the object's state.
- Object methods, the means for manipulating objects, usually read or change object attributes.
- The most fundamental difference between an object and an ordinary data structure is that the internal structure of an object is opaque.
- Not all data structures in the Windows OS are objects. Only data that needs to be shared, protected, named, or made visible to user-mode programs (via system services) is placed in objects.

7eRoM

# Objects and handles

- Objects, through the help of a kernel component called the object manager, provide a convenient means for accomplishing the following four important OS tasks:
  - Providing human-readable names for system resources.
  - Sharing resources and data among processes.
  - Protecting resources from unauthorized access.
  - Reference tracking, which allows the system to recognize when an object is no longer in use so that it can be automatically deallocated.

7eRoM

# Security

- The core security capabilities of Windows include:
  - Discretionary (need-to-know) and mandatory protection for all shareable system objects, such as files, directories, processes, threads, and so forth.
  - Security auditing for accountability of subjects, or users, and the actions they initiate.
  - User authentication at logon.
  - The prevention of one user from accessing uninitialized resources, such as free memory or disk space, that another user has deallocated.

7eRoM

# Security

- Windows has three forms of access control over objects:
  - Discretionary access control
  - Privileged access control
  - Mandatory integrity control

- AppContainers:
  - Provides isolation
  - Code in AppContainers can communicate with brokers
  - AppContainer model forces a significant shift in traditional programming paradigms, moving from the traditional multithreaded single process application implementation to a multi-process one.

- The Windows subsystem implements object-based security in the same way the OS does: protecting shared Windows objects from unauthorized access by placing Windows security descriptors on them.

7eRoM

# Registry

- The system database that contains the information required to boot and configure the system

- System-wide software settings that control the operation of Windows

7eRoM

# Unicode

- Unicode is an international character set standard that defines unique values for most of the world's known character sets, and provides 8, 16, and even 32-bit encodings for each character.
- Many Windows functions that accept string parameters have two entry points: a Unicode (wide, 16-bit) version and an ANSI (narrow, 8-bit) version.
- APIs:
  - Windows APIs: two functions for ANSI and Unicode. For example, `CreateFile` is not a function at all; instead, it's a macro that expands to one of two functions: `CreateFileA` (ANSI) or `CreateFileW` (Unicode, where `W` stands for wide).
  - COM-based APIs: Typically use Unicode strings, sometimes typed as `BSTR`.
  - Windows Runtime APIs: use Unicode strings only, typed as `HSTRING`.

7eRoM