XNU (X is Not Unix)

Notebook:	Raytheon Waiting Access		
Created:	6/23/2020 3:47 PM	**Updated:**	6/23/2020 4:35 PM
Author:	anthonykluu@gmail.com		

Hybrid Kernel

- combination of monolithic kernel and microkernel
- Monolithic Kernel
    - Entire OS works in kernel space
    - Note: Kernel space has highest privilege level, user space has lowest privilege level
    - Pros:
        - CPU scheduling, memory management, etc through system calls
        - Single address space
        - Static binary file
        - Simpler to create
        - Fast execution
    - Cons:
        - If any service fails, whole system will fail
        - Possibly less memory protection
        - New user service requires modification of entire OS
- Microkernel
    - Only essentials run in kernel mode, everything else runs in user space
    - Interrupts instead of direct system calls
    - Pros:
        - User services and kernel services are isolated so if user service fails, kernel can continue to operate
        - Easy to add new services to end of user address space and does not affect kernel space
        - More secure (memory protection)
    - Cons:
        - Slower execution because user applications have to use interrupts instead of direct system calls
        - More complicated to implement

Component #1: Mach microkernel

- Base for XNU kernel
- BSD functions built into Mach in attempts to reduce context switching overhead (the time it takes to switch from user mode to kernel mode and vice versa)
- Provides set of abstractions for dealing with memory management, ,interprocess (and interprocessor) communication with IPC
- Manages CPU usage and memory, handles scheduling, provides memory protection, and provides messaging-centered infrastructure to rest of OS layers
- Handles:
    - Interprocess Communication (IPC)
    - Remote Procedure Calls (RPC)
    - Symmetric multiprocessing (SMP)
    - support for real-time services
    - Virtual memory support
    - support for pagers (paging?)
    -

Component #2: BSD (Berkeley Software Distribution) - specifically FreeBSD

- Provides XNU with:
    - the Portable Operating System Interface (POSIX) API
    - Unix process model atop Mach tasks

    - basic security policies
    - user and group ids
    - permissions
    - network protocol stack
    - virtual file system
    - local file systems such as:
        - Hierarchical File System (HFS, HFS Plus)
        - Apple File System (APFS)
    - Network File System (NFS) client and server
    - Cryptographic framework
    - UNIX System V inter-process communication (IPC)
        - UNIX System V - UNIX OS
        - IPC - mechanism that allows process to manage shared data
    - Audit Subsystem
    - Mandatory Access Control
    - UNIX security model
    - syscall support
    - BSD process model, process IDs and signals
    - FreeBSD kernel APIs
    - kernel support for pthreads

K32/K64 (Probably not important)

- XNU in Mac OS X Snow Leopard
    - K32 - 32 bit
    - K64 - 64 bit
- Can manage more than 32 GB of RAM
- K64 was faster than K32

Component #3: I/O Kit

- Written in subset of C++ based on Embedded C++
- Object oriented
- Multi-threaded, symmetric multiprocessing safe
- Allows for hot-pluggable devices (plug and play)
- Dynamic device management

Useful Links:
https://developer.apple.com/library/archive/documentation/Darwin/Conceptual/KernelProgramming/Architecture/Architecture.html