Operating system Structure

اللَّهُمَّ صَلِّ عَلَى مُحَمَّدٍ وَعَلَى آلِ مُحَمَّدٍ، كَمَا صَلَّيْتَ عَلَى إِبْرَاهِيمَ، وَبَارِكْ عَلَى مُحَمَّدٍ وَعَلَى آلِ مُحَمَّد،

كَمَا بَارَكْتَ عَلَى آلِ إِبْرَاهِيمَ، فِي الْعَالَمِينَ، إِنَّكَ حَمِيدٌ مَجِيدٌ.

By: Mohamed Gamal Maklad

❖ Operating System Services:

- ***User interface*** Provide a **graphical user interface** (**GUI**) or a **command-line interface** (**CLI**).

  بتوفر واجهه اللي المستخدم هيتعامل معاها سواء رسوميه زي سطح المكتب او كتابيه زي التيرمينال

- ***Program execution:***
  - **Load programs into memory and run them**.
  - A program **must be able to end its execution**, either **normally or abnormally**.
- ***I/O operations*** :
  - User processes **cannot execute I/O operations directly**. علشان نضمن ميحصلش مشاكل بين تشغيل البرامج
  - **OS must** provide processes with **some means to perform I/O** **علشان** كده لازم نظام التشغيل هو اللي يتحكم فيهم
- ***File-system manipulation*:**
  - Allow user processes **to read, write, create, and delete files**. ب اختصار بتسمحله يعدل ع الفايلات
- ***Communications***
  - Enable user **processes to exchange information**. بتسمح لليوزرز انهم يتبادلوا المعلومات
  - Implemented (بيحصل/ بيتم عن طريق) via **shared memory** or **message passing**.
- ***Error detection***:
  - **Ensure correct computing by detecting errors in hardware or user programs**

بمعني ان لو في خطا معين بيطلعلي رساله تفهمني الخطأ ده بسبب ايه

❖ Additional Operating System Functions:

- ***Resource allocation*** :
  - **Allocate resources to the running processes**. بيحجز موارد لكل عمليه شغاله
- ***Logging***
  - **Recording which process** uses how much and **what kinds of resources**.بتستخدم لتسجيل كل حاجه بتحصل علي العمليه
  - This record may be used for **account billing or accumulating usage statistics**. امثله علي الاستخدام
- ***Protection and Security***:
  - ***Protection*** ensures that **all access to system resources is controlled**. يعني التأكد ان كل شيء بيحصل تحت تحكم
  - ***security*** of the system **from outsiders is also important**. زي عمل باسورد للجهاز فبيحمه من الأشخاص الغرباء

❖ System Calls:

- Provide **an interface to the services provided by the OS**.
- System calls are available as functions written in **C and C++,** low-level may be written using **assembly-language**
- **When a process executes a system call, the system traps (Make interrupt) to the OS.**
- The **OS provides application programmers with API**

بص الدنيا هنا قصده ايه لما تحب تعمل system call بدل ما تروح تعمل ب ايدك كل حاجه لا ي باشا هو بقي موفرلك function جهزه يدوب بس تباصي ليها ال parameter وهي هتتعامل بمعني بص لما اقولك اكتبلي كود بيطبع حاجه معينه ب بايثون انت بتروح تجري تستخدم print مهي دي برضه فانكشن جاهزه بتديها parameter وتطبع ع الشاشه

- **Note** **لما بيحصل interuupt ده بيحولنا ن user mode الي kernel mode**

❖ **Methods Of Passing Parameters to System Calls:**

- ***Method 1***: **Pass the parameters in CPU registers**. بياصي القيمه ل ريجيستر معين بس كده
- **Method 2**: **Store the parameters in a table in memory** (block) and **pass the address of the block as a parameter in a CPU register.** بيحط القيمه في بلوك في الميموري وبعدين بياصي العنوان بتاعها للريجيستر
- ***Method 3***: Push **the parameters onto the stack by the process**, and later **pop them of the stack by the operating system** **بيحط البراميتر في استاك وبعد كده نظام التشغيل بيجي ياخدها من الاستاك**

❖ **Types of System Calls:**

- ***Process control*:**
  - Create, load, execute, terminate, and abort.
  - Get process attributes and set process attributes.
  - Wait for time, wait event, signal even, allocate and free memory.
- ***File management*: بيتعامل مه أي حاجه خاصه بالفايلات**
  - **Create file and delete file,Open, close, read, write, reposition file**.
  - **Get file attributes and set file attributes**.
- ***Device management*: بيتحكم في الاجهزه**
  - **Request device, release device, read, write, reposition**.
  - **Get and set device attributes, logically attach or detach device**
- ***Information maintenance*: ده بيتحكم في كل حاجه حرفيا شامل كل حاجه**
  - **Get time/date, set time/date**
  - **Get system data, set system data**
  - **Get process, file, or device attributes, set process, file, or device attributes.**
- ***Communications*: بيتحم في الحاجات الخاصه بالتواصل عموما**
  - **Create, delete communication connection**, Send message, receive message, transfer status information
  - Attach remote device, detach remote device

❖ **System Boot:**

- ***System Boot***: The **process of starting a computer by loading the kernel**. فاكر دي كنا بنعملها ازاي ارجع للفايل اللي قبله
- **On most systems, the boot process proceeds as follows**: انت رجعت بجد تعالي الخطوات هنا اهي 😊
  1. **bootstrap** program **locates the kernel**. البوت استراب بيشوف كان الكيرنل
  2. **kernel is loaded into memory and started**. وبعدين يحملها في الميموري
  3. **kernel initializes hardware**. الكيرنل بيهيئ الهار وير يعني بيجهزه يعني
  4. **root file system is mounted**.

Operating System structure

ركز في الحته اللي جايه دي الاسئله كلها بتيجي منها أصلا ركز وان شاء الله الأمور تكون سهله

- **Monolithic Approach:**
  - **No structure at all**. النظام ده مش متقسم هو عباره عن كتله ع بعضه
  - **Difficult to implement and extend**. صعب التعديل عليه
  - **Communication within the kernel is fast** **الميزه اللي فيه ان التواصل مع الكيرنل سريع**
  - **Very little overhead** in the system-call interface. مفيش صعوبه او حمل انه يعمل سيستم كول
  - A **single, static binary file that runs in a single address space**. هو عباره عن فايل واحد مكتوب 0 و 1 وله عنوان واحد
  - **Tightly-coupled** as changes to one part can affect other parts. أي يغتير في جزي هيسمع في باقي الأجزاء
  - Several OSs use of this approach, such as **UNIX, Linux, and Windows**.
- **Layered Approach:**
  - Divides the OS **into a set of layers**. النظام ده بقي بيكون متقسم ل طبقات
  - **Makes it easier to implement** and extend the kernel. سهل انك تضيف كيرنل جديد لانه طبقات ف اشطا عادي
  - **Each layer** uses the functions of **only the lower-level layers**. كل طبقه مسموح تستخدم فانكشن من الطبقه اللي تحتها بس
  - Therefore, it is **difficult to define the functionality of each layer**.
  - **Loosely-coupled** as changes in one layer do not affect the others. أي تغير في طبقه مش هيأثر ع الباقي
  - Traversing multiple layers to call services results in **poor performance** الأداء ضعيف بسبب التنقل بين الطبقات الكتير
- **Micro-kernel Approach: هنا شاي أي حاجه مش مهم من الكرينل وحطها كسيستم بروجرام بحيث يصغر حجمها قدر المستطاع**
  - **Removes all nonessential components from the kernel and implements them as system programs**.
  - It is **easier to extend the OS**. ده خلي من السهل انك تزود او تمدد نظام التشغيل
    - **No need** to **modify of the kernel. من غير متكون محتاج تعدل علي الكيرنل**
    - **New services are added as system programs ولو في خدمات جديده حطها كسيستم بروجرام**
  - **It is easier to modify kernel if needed as it is a smaller kernel**.
  - It **is easier to port the OS from one hardware design to another**. ممكن نظام تشغيل واحد تشغله ع اكتر من هارد وير
  - It provides more security and reliability.
    - Most **services are running as user** rather than kernel processes. الخدمات بتشتغل في وضع اليوزر ف كده امان اكتر
    - If a service fails, the **rest of the operating system remains untouched. لو حاجه باظت بيفضل باقي النظام شغال**
- **Modules Approach: الكيرنبل هنا متقسمه ل موديول او ملفات**
  - The **kernel has a set of core components** and **can link in additional services via LKMs**(Loadable Kernel Module), either at boot time or during run time.
  - **Linking additional services dynamically is preferable to recompiling the kernel every time a change is made. بيقولك هنا ان يفضل ان الخدمات الاضافيه تتربط بطريقه ديناميكيه بيحيث لو جيت تجمعها مره تانيه لو حصل في الكيرنل تغير**
  - This type of design is common in modern implementations of **UNIX, such as Linux, macOS, and Solaris, as well as Windows**.
- **Layered approach And Modules Approach:**
  - ***Similarity***: each **kernel section has a defined and protected interface**.
  - ***Difference***: **Modules** Approach **more flexible, because any module can call any other module**

هنا ال Module اكتر مرونه علشان تقدر تعمل call لان moduel تاني مش بس اللي تحتك

- **Micro-kernel And Modules Approach:**
  - ***Similarity***: the **primary module has only core functions**. ليهم نواه واحده بس أساسه
  - ***Difference***: modules **do not need to use message passing to communicate**
- **Hybrid Approach: ده بقي مزيج من اللي فاتوا كلهم**
  - In practice, **very few OSs adopt a single, strictly defined structure**.
  - Instead, they combine different structures, resulting in hybrid systems that **address performance, security, and usability issues**.

بيقولك فكرة انك تعمل نظام تشغيل ب بنيه واحده من اللي فاتوا و يكون دقيق ده صعب جدا بيعملوا Hyper اللي هو نظام هجين منهم ف تكون الأمان فيه عالي والمشاكل قليله

  - For example, **Linux is both monolithic and modular**: يعني اللبنكس بيستخدم نوعين في نظام التشغيل بتاعه
    - ***Monolithic***: to provide **very efficient performance**. علشان يوفر أداء عالي لانه سريع زي ما قولنا
    - ***Modular***: **to add new functionality dynamically to the kernel وده علشان لو حب يضيف حاجه جديده زياده**

اللَّهُمَّ صَلِّ عَلَى مُحَمَّدٍ وَعَلَى آلِ مُحَمَّدٍ، كَمَا صَلَّيْتَ عَلَى إِبْرَاهِيمَ، وَبَارِكْ عَلَى مُحَمَّدٍ وَعَلَى آلِ مُحَمَّد،

كَمَا بَارَكْتَ عَلَى آلِ إِبْرَاهِيمَ، فِي الْعَالَمِينَ، إِنَّكَ حَمِيدٌ مَجِيدٌ.

By: Mohamed Gamal Maklad