University of Portsmouth
BSc (Hons) Computer Science
Second Year

Operating Systems and Internetworking (OSINT)

M30233
September 2023 - January 2024
20 Credits

Thomas Boxall
up2108121@myport.ac.uk

Contents

Theme I

Operating Systems

Page 1

Lecture - Introduction

2023-09-26 13:00 Tamer

1.1 Operating Systems

The *Operating System* is a special type of software which controls the hardware. It is not the desktop, as the desktop, start screen and any other GUI software is provided by a suite of *application level software* which exists at a higher level than that of the operating system. The operating system is only accessible by application programs, not directly from the user. The user cannot interact with the hardware directly.

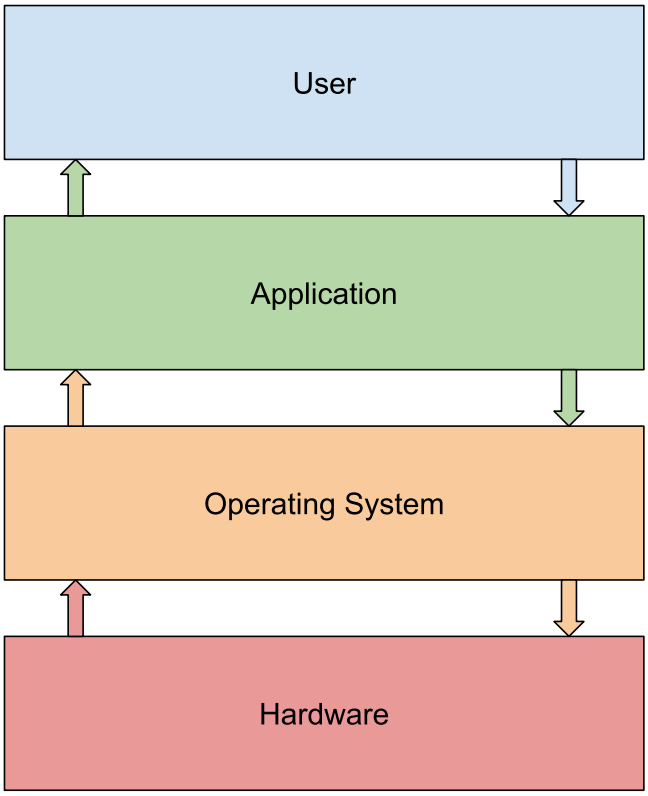

Figure 1.1: Location of the operating system in relation to the user, applications and hardware

1.1.1 What does it do?

The operating system is a piece of systems software that manages the computer's hardware, resources and control processing. It allows multiple computational processes and users to share a processor

simultaneously, protect data from unauthorised access and keep independent input / output (I/O) devices operating correctly.

The operating system provides common services for application software, making developers lives easier as the hardware interfacing has already been done for them. Users cannot run any software application without it.

1.1.2 Characteristics of the Operating Systems

There are two key characteristics of the Operating System.

Extended Machine

The part of the Operating System which behaves as an *extended machine* deals with the Input / Output devices which involves reading and writing control registers, handling interrupts etc. If a mistake is made, it will crash the entire computer. The Operating system provides a cleaner, safer, higher level set of operations for doing these - thus making developers lives easier as they are less worried about the 'nitty gritty' of hardware handling.

Resource Manager

The part of the operating system which behaves as a *resource manager* deals with sharing the resources between the many different processes which are happening simultaneously. The OS arbitrates between the requests these processes make to make to I/O subsystems, memory, etc to ensure smooth functioning of the system.

1.2 Software Types

There are two key types of software - systems software and applications software.

1.2.1 System Software

Systems software is the software that controls the computer system and ultimately allows you to use the computer. This includes the operating system and utility programs. They allow tasks to be performed such as:

- enabling the boot process
- launching applications
- transferring files
- controlling hardware configuration
- managing files on the hard drive
- and protecting the machine from unauthorised use.

1.2.2 Application Software

Application software is software which allows the user to perform a specific task on the computer. They allow tasks to be performed such as:

- Word processing
- Playing games
- browsing the web
- listening to music

1.3 Central Processing Unit

The *Central Processing Unit* (or CPU for short), is the heart of the computer. It is sometimes also referred to as the processor, microprocessor or processing unit. The CPU's primary purpose is to interpret processes and execute instructions.

1.3.1 CPU Organisation

Modern CPUs are complex, containing many different components. All CPUs will contain a: control unit, Arithmetic Logic Unit, cache memory, and memory management unit. The inner workings of the CPU will be discussed futher in a later lecture.

The CPU processes a sequence of machine instructions. A single instruction might perform simple arithmetic on data values - typically individual words; or more data between memory and / or registers.

1.3.2 Registers

Registers are very fast storage built into the CPU. They are typically big enough to store one word of data. Nowerdays, this will usually be 64-bits however in the past, 32-bit words were common. Registers are small amounts of high-speed memory contained within the CPU which are used by the processor to store small amounts of data that is needed during processing. This could include: the address of the next instruction to be executed or the current instruction being decoded.

A CPU has many registers as they are commonly single purpose; they also play a key role in OS design because they form part of state of a computation. Most computer architecture provides a small set of General Purpose Registers (GPR). The program status word register is responsible for setting which mode the CPU is operating in.

Name	**Use**	**Description**
EAX	Accumulator	The default register for many additions and multiple instructions.
EBX	Base	Stores the base address during memory addressing.
EXC	Count	The default counter for repeat (REP) prefix instructions and LOOP instructions.
EDX	Data	Used for multiple and divide operations
ESI	Source Index	Store source index
EDI	Destination Index	Stores destination index
EBP	Base Pointer	Mainly helps in referencing the parameter variables passed to a subroutine
ESP	Stack Pointer	Provides the offset value within the program stack.

Table 1.1: GPR Registers and their purposes

1.4 Classification of Programming Languages

Higher level languages cannot interact directly with the hardware. High level language source code is *translated* into a series of low level languages - ultimately ending up with machine code that can interface with the hardware directly.

1.4.1 Assembly Language

Assembly Language is a symbolic form of machine code used by system programmers. It will generally have the same instructions as machine code but rather than the instructions being represented in binary or hexadecimal format - assembly language uses mnemonics, making it easier to read, write and understand the code.

The following two lines of code copy the contents of the `EAX` register to the `EBX` register then increases the value in the `EXB` register by 4. In a high level language, this would look something like: `b = a + 4`.

```
MOV EBX, EAX
ADD EBX, 4
```

The Intel assembler instruction set also includes the ability to access the content within a memory address. This is done by putting square brackets (`[]`) around the register containing the memory address to look in.

```
MOV ESI, 105672
MOV EAX, [ESI]
```

An I/O device, like a hard disk, will have an associated set of *ports* through which the device is controlled and data transferred. A range of ports will be associated with each device. The instruction `IN` and `OUT` are used to read or write to ports.

1.5 User and Kernel Modes

Typical CPUs support different modes of operation controlled by a register called the *Program Status Word* (recent X86 processors actually use bit 0 of the Control Register (CR0), when its set - we are in *User Mode* or *Protected Mode*).

When machine code executes while the CPU is in *user mode*, it can only use limited instructions, for example not the `IN` and `OUT` instructions.

When machine code executes while the CPU is in *kernel mode*, it can use privileged instructions - for example `IN` and `OUT`.

The Operating System will always run in Kernel Mode. Thus, enabling all I/O operations to be performed by the OS on behalf of application programs. This has multiple benefits: the OS keeps control over whats done with those I/O operations and it makes it easier for software developers as they don't have to worry about interfacing directly with hardware.

1.6 Interrupts

When an I/O controller (i.e. on a disk card) has requested data available, it must gain the attention of the CPU. This is because the CPU can't be focusing on just waiting for the disk as it has other processes it needs to service. Gaining attention of the CPU is done through asserting an electrical signal called an *interrupt*. When the CPU receives an interrupt - it must abandon the program its currently executing and instead execute specialised code to deal with the new event. Specialised

code takes form of *interrupt handlers* which are typically installed at boot time and run in kernel mode.

Interrupt handlers have a wide significance in operating systems - beyond their original role in processing data received from I/O controllers. They have a role in process scheduling and in the implementation of system calls - these two topics will be covered in later lectures. Inn some sense, the whole operating system is driven by variations on the theme of "interrupt handler".

Theme II

Internetworking

Page 2

Lecture - Networking Services: DNS, DHCP, etc

2023-09-25 09:00 Thanos

Follow up material for lectures will be posted on Moodle. This will commonly include LinkedIn Learning courses. Do them. Answers to Lab Sessions should be uploaded to our individual Wiki sections for each theme as pdf files. They will not be assessed but we may be asked to show them to Lab staff at some point.

2.1 Dynamic Host Configuration Protocol

Dynamic Host Configuration Protocol (DHCP) provides a set of important configuration parameters for devices which are connected to a network. These parameters include: IP address (this is required for any device to be able to talk on a network); router address (the address of the device which your communications has to go through to be passed onto the right place); subnet mask; and DNS server address.

DHCP was introduced in 1993, before DHCP - IP addresses were manually assigned to each device on the network. Whilst, this was a viable option and can still be done to this day - it makes network administrators lives much more complicated. There was also the Bootstrap Protocol (BOOTP) as DHCP supports temporary leases of IP addresses to clients with minimal human interaction. DHCP servers are compatible with BOOTP clients.

For DHCP to work on a network, you require a DHCP server. This commonly is built into modern domestic routers however in larger organisations - a separate (virtual) server will be used.

When a client is shut down or it terminates its connection to the internet - it releases it's IP address. This IP address is returned to the IP pool which means it is then available for another client to use. IP address leases are automatically renewed when 50% of the lease time is used. This works by a request to the original DHCP server. If its not available then the request is broadcast to all available DHCP servers. The IP address lease gets renewed as it prevents the need for a new IP address to be assigned.

We use DHCP for a number of reasons: it saves the network administrator from a lot of manual configuration; it allows devices to move from one network to another and gain instant connectivity (there may be conflicting devices if static IPs were used); it allows for more efficient utilisation of available IP addresses (whereby inactive clients do not obtain IP addresses).

There are, however, a number of disadvantages to using DHCP: DHCP packets are UDP packets which means they are unreliable and insecure; there is a potential for unauthorised clients obtaining IP addresses which would then make them appear legitimate (this can be avoided by using MAC address filtering); and there is potential for malicious DHCP clients and server which could lead to

incorrect configuration parameters being supplied to clients and / or the IP pool being exhausted.

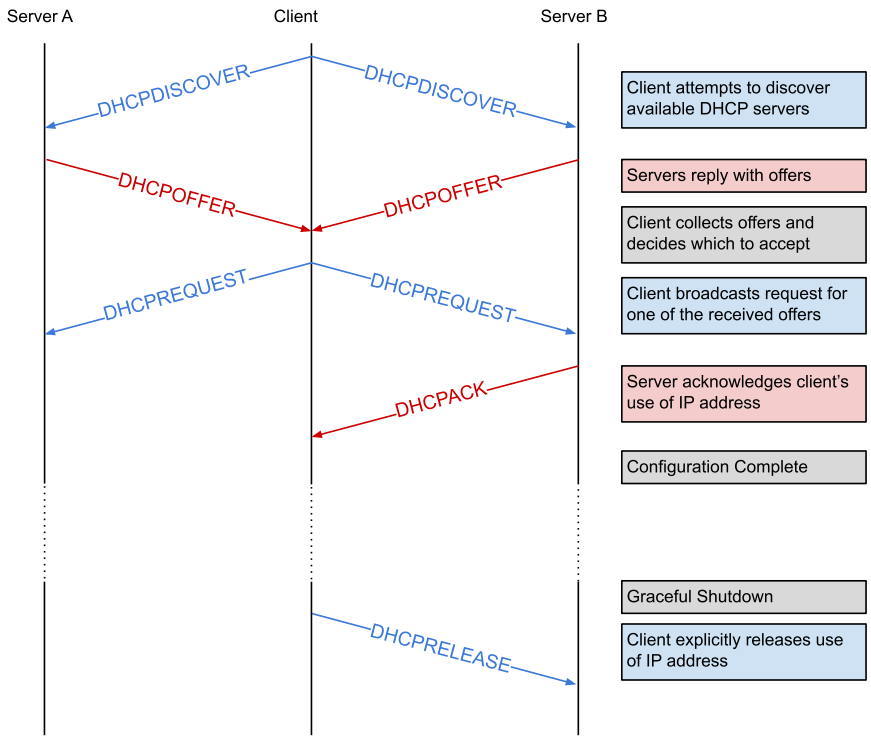

Figure 2.1: DHCP Initial Message Flow

2.1.1 Terminology

DHCP Packet DHCP Message

DHCP Client Client

DHCP Server Server

Lease Length of time a DHCP client can use a specified IP address

2.2 Domain Name System

Domain Name System (DNS) is the mechanism by which Internet Software translates names to attributes such as IP addresses. Architecturally, DNS is a globally distributed, scalable and reliable database which is comprised of three components: a *namespace*, *servers* (makes the namespace available) and *resolvers* (clients - these query the servers about the namespace).

DNS exists to make users' use of the internet easier. Users generally prefer names (`thomasboxall.net`) to numbers however computers usually prefer numbers (`145.14.152.146`) to names. DNS provides the mapping between the *domain names* and *IP addresses of servers*.

DNS is distributed globally throughout many different devices. No single computer holds all the DNS data, however some remote DNS data is locally cached to improve performance. DNS lookups can be performed by any device. DNS lookups can be performed by any device. On UNIX systems, the

command `dig` provides this utility.

The DNS database is always internally consistent. This is achieved by each version of a subset of the database (a zone) having a serial number which is incremented on every database change. Changes to the master copy of the database are replicated according to timing set by the zone administrator, generally this is quite frequent. Cached data expires according to a timeout set by a zone administrator. While there is no limit to the size of the DNS database, common sense dictates that its not a good idea to store 200,000,000 domain names in the same database as there is no limit to the number of queries. This can lead to 10,000+ queries being sent each second which are handled easily. Queries are distributed among primary and secondary DNS servers as well as caches. The `nslookup` command will tell you where it has obtained the DNS information from.

Due to DNS data being replicated from the primary to multiple secondary servers, there is high levels of reliability. Clients will typically query local caches first, and if they do not contain the data requested then the queries will be passed to either the primary server or any secondary server. DNS uses both UDP and TCP (port 53) for different things: TCP is used for intra-server communications and UDP is used for communications between clients and servers.

The DNS database can be updated dynamically. This includes teh addition, deletion or modification of any record. However, it is only the primary server which can be dynamically updated. The modification of the primary database triggers replication to all the secondary databases.

2.3 Domain Names

A domain name is the sequence of labels from a node to the root, separated by dots (`.`) which is read from left to right. The namespace has a maximum depth of 127 levels and domain names are limited to 255 characters in length. A nodes domain name identifies its position in the namespace.

One domain is a subdomain of another if its domain name ends in the other's domain name.
Name servers store information about the namespace in units called *zones*. The nameservers that serve a complete zone are said to *have authority* or *be authoritative for* the zone. More than one name server can be authoritative for the same zone, ensuring redundancy and load spreading. Also, a single name server may be authoritative for many zones. There are two types of Name Servers: *authoritative* which maintains the data (has subtypes of primary and secondary) and *non-authoritative* which caches the authoritative server. No special hardware is needed for a name server.

Name resolution is the process by which local resolvers and the nameservers cooperate to find data in the namespace. Upon receiving a query from a resolver, a name server:

1. looks for the answer in its authoritative data and its cache.
2. if step 1 fails, the answer must be looked up through other servers (this can either be done recursively or iteratively).