# OPERATING SYSTEM – IMPORTANT QUESTIONS & ANSWERS

*(2-Hour Power Prep)*

---

## 1 Basics of Operating System

### Q1. What is an Operating System?

**Answer:**
An Operating System (OS) is system software that acts as an **interface between user and hardware**.
It manages **CPU, memory, storage, I/O devices, and processes**.

---

### Q2. Main functions of an OS?

**Answer:**

- Process Management
- Memory Management
- File System Management
- Device Management
- Security & Protection
- Resource Allocation

---

### Q3. Types of Operating Systems?

**Answer:**

- Batch OS
- Time Sharing OS
- Real-Time OS (RTOS)
- Distributed OS
- Network OS
- Embedded OS

---

### Q4. Difference between OS and Kernel?

**Answer:**

- **Kernel**: Core part of OS, interacts directly with hardware
- **OS**: Kernel + utilities + system programs

---

## 2 Process Management

### Q5. What is a Process?

**Answer:**
A process is a **program in execution** along with its current state and resources.

---

### Q6. Process states?

**Answer:**

- New
- Ready

- Running
- Waiting (Blocked)
- Terminated

---

## Q7. What is PCB (Process Control Block)?

**Answer:**
PCB stores all information about a process:

- Process ID
- Program Counter
- CPU Registers
- Scheduling info
- Memory info

---

## Q8. Difference between Process and Thread?

**Answer:**

| Process | Thread |
|---|---|
| Heavyweight | Lightweight |
| Separate memory | Shared memory |
| Slower context switch | Faster |

---

### Q9. What is Context Switching?

**Answer:**
Saving the state of one process and loading the state of another process so CPU can switch execution.

---

## 3CPU Scheduling

### Q10. What is CPU Scheduling?

**Answer:**
The method by which OS decides **which process gets CPU time**.

---

### Q11. Scheduling criteria?

**Answer:**

- CPU Utilization
- Throughput
- Turnaround Time
- Waiting Time
- Response Time

---

### Q12. FCFS Scheduling?

**Answer:**
First Come First Serve – non-preemptive, simple but causes **convoy effect**.

---

### Q13. Shortest Job First (SJF)?

**Answer:**
CPU assigned to process with **smallest burst time**.
Optimal but difficult to predict burst time.

---

### Q14. Round Robin Scheduling?

**Answer:**
Each process gets fixed **time quantum** in cyclic order.
Used in **time-sharing systems**.

---

### Q15. Preemptive vs Non-Preemptive?

**Answer:**

- **Preemptive**: OS can take CPU forcefully
- **Non-Preemptive**: Process releases CPU voluntarily

---

## 4 Deadlocks

### Q16. What is Deadlock?

**Answer:**
A situation where processes wait indefinitely for resources held by each other.

---

### Q17. Necessary conditions for Deadlock?

**Answer:** *(All must hold)*

1. Mutual Exclusion
2. Hold and Wait

3. No Preemption
4. Circular Wait

---

### Q18. Deadlock prevention?

**Answer:**
Break any one deadlock condition (e.g., allow preemption).

---

### Q19. Deadlock avoidance?

**Answer:**
Avoid unsafe states using algorithms like **Banker’s Algorithm**.

---

### Q20. Deadlock detection?

**Answer:**
Detect deadlocks using **resource allocation graph** and recover.

---

## 5 Memory Management

### Q21. What is Memory Management?

**Answer:**
Managing primary memory (RAM) efficiently for multiple processes.

---

### Q22. Logical vs Physical Address?

**Answer:**

- Logical Address: Generated by CPU
- Physical Address: Actual memory location

---

### Q23. What is Paging?

**Answer:**
Memory is divided into **fixed-size pages** to avoid external fragmentation.

---

### Q24. What is Segmentation?

**Answer:**
Memory divided into **logical segments** like code, stack, data.

---

### Q25. Paging vs Segmentation?

**Answer:**

- Paging → fixed size, no fragmentation
- Segmentation → variable size, logical view

---

### Q26. What is Virtual Memory?

**Answer:**
Technique that allows execution of programs larger than physical memory using disk as backup.

---

### Q27. Page Fault?

**Answer:**
Occurs when a required page is **not present in RAM**.

---

## 6 File System

### Q28. What is a File?

**Answer:**
A collection of related data stored on secondary storage.

---

### Q29. File access methods?

**Answer:**

- Sequential
- Direct
- Indexed

---

### Q30. File allocation methods?

**Answer:**

- Contiguous
- Linked
- Indexed

---

### Q31. What is Directory?

**Answer:**
A structure that stores information about files (name, size, location).

---

## 7 I/O & Disk Management

### Q32. What is Spooling?

**Answer:**
Technique where data is temporarily stored so I/O devices can process it later.

---

### Q33. Disk Scheduling Algorithms?

**Answer:**

- FCFS
- SSTF
- SCAN
- C-SCAN
- LOOK

---

### Q34. Purpose of Disk Scheduling?

**Answer:**
Reduce **seek time** and improve performance.

---

## 8 Synchronization

### Q35. What is Critical Section?

**Answer:**
Part of program where shared resources are accessed.

---

### Q36. What is Race Condition?

**Answer:**
Occurs when multiple processes access shared data simultaneously and outcome depends on execution order.

---

### Q37. What is Semaphore?

**Answer:**
A synchronization tool using integer value to control access to shared resources.

---

### Q38. Binary vs Counting Semaphore?

**Answer:**

- Binary → 0 or 1
- Counting → Multiple resource instances

---

## 9Security & Protection

### Q39. Difference between Authentication and Authorization?

**Answer:**

- Authentication → Who you are
- Authorization → What you can access

---

### Q40. What is Access Control?

**Answer:**
Restricting unauthorized access to system resources.

## AI. Playlist (Pro Version)

### 📍Section 1: Foundations of Deep Learning (5 videos)

1. What is Deep Learning? And Why It Matters Today
2. Neural Network Intuition – Like a Brain but Faster
3. Forward Propagation Explained Simply – With Code (From Scratch in NumPy)
4. Backward Propagation (Backprop) – The Learning Magic
5. Activation Functions – ReLU, Sigmoid, Softmax Visually Explained

### 📍Section 2: Building Deep Learning Models with Keras (4 videos)

1. Getting Started with Keras – Build a Neural Network in Minutes
2. MNIST Digit Recognition – First Hands-on DL Model
3. Overfitting and Dropout – Save Your Models from Failing
4. Custom Loss Functions and Optimizers

### 📍Section 3: CNNs for Computer Vision (4 videos)

1. CNN Intuition – Why They Work So Well with Images
2. Building CNNs with Keras – Image Classifier from Scratch
3. Transfer Learning with ResNet/MobileNet – Speed Up Training
4. Build a Real-Time Image Classifier (Camera/Webcam Feed)

### 📍Section 4: RNNs, LSTM & Sequence Models (4 videos)

1. What is a Sequence Model? RNNs Explained Practically
2. LSTMs vs GRUs – When and Why to Use Them
3. Text Generation with LSTM – Write Like Shakespeare (Fun Project)
4. Time-Series Forecasting – Stock Price Prediction with RNN

### 📍Section 5: Transformers – Stepping into LLM World (6 videos)

1. The Attention Mechanism – Explained with Simple Examples
2. What is a Transformer? How It Changed the Game
3. Build a Mini-Transformer from Scratch (with code)
4. HuggingFace Transformers 101 – Use BERT, GPT-2 in Minutes
5. Text Classification Using BERT (Fine-tuning with real data)
6. Summarization and Question Answering using Pretrained Models

### 📍Section 6: LLMs and Their Applications (5 videos)

1. What is an LLM? From GPT to Claude to Gemini – Intuitive View
2. How Tokenization Works (BPE, WordPiece, etc.)
3. Prompt Engineering Basics – How to Talk to LLMs
4. Building an LLM-Powered App (Streamlit + OpenAI API)
5. LangChain Basics – Your First AI Agent App
6. Finetuning a Small LLM (Like LLaMA or GPT-Neo) – Intro Only

### 📍Section 7: Deployment & Career Projects (2 videos – Optional)

1. Deploy Your DL Model on Web with Streamlit/Gradio
2. GitHub + Resume Tips for DL/LLM Projects

### 📍Section 8: Building Smarter LLM Apps with RAG (4 videos)

1. What is RAG? Why LLMs Need Retrieval for Better Responses
2. RAG Architecture Explained (Vector DB + LLM)
3. Implement RAG with LangChain + FAISS/Chroma + OpenAI/BERT
4. Build a Custom Chatbot on Your Notes, Books, or Docs

### 🌟 Optional Bonus

1. RAG with Open Source LLMs (LLamaIndex + GPT4All)

## Cybersecurity Playlist (Pro Version)

### 📍 Phase 1: Cybersecurity & Networking Foundations (Weeks 1–2)

*Goal: Build the hacker's brain—concepts, mindset, and how data really flows.*

1. **What is Cybersecurity?**
   – Career Map & Fields (Blue Team, Red Team, DFIR, etc.)
2. **The Hacker Mindset**
   – How real hackers think, ethical considerations
3. **CIA Triad + Core Security Concepts**
   – Authentication, Authorization, Confidentiality, Integrity, Non-Repudiation
4. **Computer Networking Basics**
   – IP addressing, DNS, MAC, routers, ports
5. **OSI Model + TCP/IP Model**
   – Real-life analogies for each layer
6. **Common Protocols & Abuse**
   – HTTP/S, FTP, SSH, SMTP, DNS → how attackers exploit them
7. **VPN, Proxy, Firewall**
   – Shields vs. evasion techniques
8. **How the Internet Works**
   – Domains, hosting, subdomains, DNS records
9. **Cryptography Essentials**
   – Symmetric vs. asymmetric encryption (AES, RSA), hashing (SHA-256, MD5), TLS/SSL, digital signatures

### 📍 Phase 2: Linux & Command‑Line Mastery (Week 3)

*Goal: Achieve full comfort in Linux (plus essential Windows intro)—your primary hacker platform.*

10. **Why Linux for Hackers?**

– Setting up Kali Linux / Parrot OS; brief Windows vs. Linux comparison

11. **Linux Filesystem & Key Directories**

– /etc, /var, /tmp, /home, permissions model

12. **50+ Essential Linux Commands**

– Navigation, file ops, text processing, networking tools

13. **User Management & SUDO Privileges**

– Users/groups, chmod/chown, sudoers

14. **Bash Scripting Basics**

– Automate recon, parsing, simple tool creation

15. **Networking Commands**

– ifconfig/ip, netstat/ss, traceroute, ping, arp, nmap basics

### 📍 Phase 3: Advanced Foundations (Week 4)

*Goal: Stand up your lab, peek under the hood, and get a taste of reverse engineering.*

16. **Virtual Lab Setup**

– VirtualBox/VMware, Kali + Metasploitable/OWASP Juice Shop, snapshots

17. **System Internals & Monitoring**

– Process management, memory (top/htop), logs (syslog, dmesg, Event Viewer)

18. **Python for Security**

– Sockets, file parsing, quick scanners, integrating with tools

19. **Basic Assembly & Reverse Engineering**

– CPU registers, memory/stack basics, simple GDB walkthrough

20. **Putting It All Together**

## 📍 Phase 4: Practical Networking & Info Gathering (Week 4)

**Goal**: Become pro in reconnaissance and information gathering.

**5-Step Hacking Methodology Introduction** (Every real hack follows these 5 steps.)

1. **Reconnaissance** (Information Gathering)
2. **Scanning** (Finding Live Systems, Open Ports)
3. **Exploitation** (Taking Control)
4. **Post-Exploitation** (Maintaining Access)
5. **Reporting** (Documenting Everything)
6. **Active vs Passive Reconnaissance** Explained
7. **WHOIS, DNS Reconnaissance, Subdomain Enumeration** (Tools: whois, dig, nslookup)
8. **Google Hacking** (Dorking) – Finding Secrets Publicly
9. **Scanning Basics** – Nmap, Masscan Practical
10. **Banner Grabbing and Service Version Detection**

## 📍 Phase 4: Vulnerability Analysis and Exploitation Basics (Week 5)

**Goal**: Learn to find weak spots and exploit them.

20. **What are Vulnerabilities?** CVEs and Exploits
21. **Basics of Password Cracking** (John The Ripper, Hydra, Hashcat Overview)
22. **Vulnerability Scanning** using Nikto, OpenVAS
23. **Introduction to Metasploit Framework**
24. **Exploit Development Basics** (Very light introduction)

## 📍 Phase 5: Web Application Hacking (Week 6–7)

**Goal**: Hacking websites and APIs like a pro.
25. **How Websites Work** (Client, Server, Database Architecture)
26. **OWASP Top 10** (SQL Injection, XSS, CSRF, IDOR, SSRF, etc.)
27. **Burp Suite Basics**: Setup + Live Usage
28. **SQL Injection Practical** (with Demo)
29. **XSS (Cross-Site Scripting)** Practical (with Demo)
30. **File Upload Vulnerabilities & Path Traversal Practical**
31. **Bypassing Authentication and Business Logic Bugs**

## 📍 Phase 6: Wireless and System Hacking (Week 8)

**Goal**: Learn Wi-Fi Hacking + Windows/Linux System Attacks.
32. **Introduction to Wireless Hacking**
33. **WPA/WPA2 Cracking using Aircrack-ng**
34. **Evil Twin Attacks** (Rogue Access Points)
35. **Introduction to Privilege Escalation** (Linux + Windows)
36. **Windows Enumeration Tools** (Enum4Linux, smbclient basics)
37. **Linux Privilege Escalation Techniques** (local exploits, file permissions)

## 📍 Phase 7: Post Exploitation and Maintaining Access (Week 9)

**Goal**: After hack, how attackers stay hidden.
38. **What is Post Exploitation?** Goals After Breaking In
39. **Creating Backdoors and Reverse Shells** (Netcat, msfvenom)
40. **Pivoting and Lateral Movement Basics**
41. **Clearing Logs and Anti-Forensics Basics**
42. **Linux Persistence Mechanisms** (init.d, cronjobs, SSH keys, etc.)

## 📍 Phase 8: Malware Analysis and Social Engineering Basics (Week 10)

**Goal**: Broaden skills beyond just hacking.
43. **Basics of Malware** (Trojan, Virus, Worm, Ransomware)
44. **Introduction to Static and Dynamic Malware Analysis**
45. **Introduction to Social Engineering** – Human Hacking
46. **Phishing Attacks and Defense Techniques

## 📍 Phase 9: Mobile Application Hacking (Week 11)

**Goal**: Learn to hack mobile applications and understand the risks involved.
47. **Introduction to Mobile Application Security** (Android & iOS Overview)
48. **OWASP Mobile Top 10** (Including Insecure Data Storage, Insecure Communication, etc.)
49. **Reverse Engineering APKs** – Decompiling and Analyzing Mobile Apps
50. **Mobile Application Vulnerability Testing** using Burp Suite and MobSF
51. **Exploiting Mobile App Vulnerabilities** – SQL Injection, Insecure Data Storage, etc.
52. **Bypassing Mobile App Authentication Mechanisms**

## 📍 Phase 10: Reporting and Cyber Defense Basics (Week 12)

**Goal**: Document hacks properly like real pentesters.
53. **How to Write a Penetration Test Report** (Structure, Examples)
54. **Basics of Defensive Cybersecurity** (SIEM, IDS, IPS)
55. **Introduction to Blue Teaming**: Incident Response

## 📍 Part 1: Capture The Flag (CTF) Competitions – Crash Course

➔ **Goal**: Learn how to approach real hacking challenges smartly.

1. What is CTF? Types (Jeopardy vs Attack-Defense) – Overview
2. CTF Categories Explained:

    - Web Exploitation
    - Binary Exploitation (Pwn)
    - Reverse Engineering
    - Forensics
    - Cryptography
    - OSINT
3. Setting up CTF Practice Lab (TryHackMe / HackTheBox / PicoCTF)
4. Tools of the Trade for CTFs:
    - BurpSuite
    - Wireshark
    - Ghidra
    - IDA Free
    - CyberChef
5. Basic CTF Strategies:
    - Enumeration First
    - Thinking like a Puzzle Solver
    - Time Management in CTFs
6. First Mini Challenge: Solve 5 Beginner CTFs (Practical)

## 📍 Part 3: Bug Bounty Hunting – Crash to Intermediate

➔ **Goal**: Start Web App Hacking on real programs.

1. Introduction to Bug Bounties: How Platforms Work (HackerOne, Bugcrowd)
2. Bug Hunting Methodology:
    - Recon (Subdomain Enumeration, DNS)
    - Information Gathering (Tools like Amass, Subfinder)
    - Vulnerability Finding (XSS, SQLi, SSRF, IDOR)
    - Reporting Bugs (Professional Writeups)
3. Essential Tools for Bug Bounty:
    - Burp Suite Pro (or Community)
    - Nmap
    - ffuf
    - httpx
    - nuclei
4. Deep Dive into Common Vulnerabilities:
    - Broken Authentication
    - Insecure Direct Object Reference (IDOR)
    - Server-Side Request Forgery (SSRF)
    - Business Logic Bugs
5. Real World Project:
    - Pick 1 Public Program → Recon + Find + Report (even if you don't find a real bug at first)

## 📍 Part 4: Cloud Security Basics (AWS Hacking)

➔ **Goal**: Enter the future of hacking: Cloud Systems.

1. Cloud Computing Basics (AWS, Azure, GCP intro)
2. Understanding AWS Core Services:
    - EC2 (Servers)

    - S3 (Storage)
    - IAM (Identity and Access Management)
3. Setting Up Free AWS Account for Practice (careful of billing)
4. Common AWS Vulnerabilities:
    - Misconfigured S3 Buckets
    - IAM Privilege Escalation
    - SSRF to Metadata API attacks
5. Cloud Pentesting Tools:
    - ScoutSuite
    - Pacu
    - S3Scanner
6. Mini Cloud Project:
    - Deploy vulnerable EC2 + S3.
    - Try to escalate privileges or leak data.

## 🧨 Phase 2: Hacker Elite Training Map

(After you finish CTFs, Labs, Bug Bounty Basics, Cloud Basics)

### 📍 Week 1–2: Advanced Pentesting & Exploit Development Start

- Manual Buffer Overflow Attack (Windows/Linux)
- Stack-based Buffer Overflows (build your first exploits)
- Introduction to Return Oriented Programming (ROP Chains)
- Tools: Immunity Debugger, Mona.py, GDB

**Goal:** Understand memory corruption and craft your first working exploits manually.

### 📍 Week 3–4: Deep Dive into Advanced Web Hacking

- Server-Side Request Forgery (SSRF) Deep
- Server-Side Template Injection (SSTI) Deep
- Web Cache Deception & Poisoning
- Business Logic Exploitation Techniques
- Race Condition Exploits (bug bounty goldmine)

**Goal:** Find those rare, high-severity web bugs companies miss.

### 📍 Week 5–6: Active Directory (AD) Hacking Basics

- Windows Domain Basics
- BloodHound for AD Mapping
- Pass-the-Hash, Pass-the-Ticket Attacks
- Mimikatz Usage
- Privilege Escalation in Domain

**Goal:** Become deadly inside corporate networks.

### 📍 Week 7–8: Advanced Cloud Hacking (AWS, Azure, GCP)

- AWS Misconfigurations (IAM, EC2, S3 abuse)
- Azure Enumeration and Attacks
- GCP Basics for Pentesters

- Kubernetes Basics + Cluster Attacks

**Goal:** Hunt real-world cloud vulnerabilities.

### 📍 Week 9–10: Red Teaming Fundamentals

- Red Team Operations Basics
- C2 Frameworks (like Cobalt Strike / Mythic / Sliver)
- Evasion Techniques (bypass AV/EDR, Payload Obfuscation)
- Basic Malware Development Concepts (shellcode runners)

**Goal:** Simulate real-world attacker behavior stealthily.

### 📍 Week 11–12: Building Tools & Scripts

- Python for Pentesters
- Build your own Port Scanner
- Build your own Directory/Content Bruteforcer
- Automation of Reconnaissance

**Goal:** Stop depending on tools — start building your own mini-tools!

### 🌟 After Phase 2: Specialization Paths

(Choose based on passion and career goals)

- **Bug Bounty Focus:** Master Web + Mobile App Hacking
- **Red Team Focus:** Go full AD, Malware Development, C2 Frameworks
- **Cloud Focus:** Specialize in AWS/Azure Hacking and get top-paying cloud security jobs
- **Malware Analysis:** Learn Reverse Engineering (Assembly + Malware Research)
- **Blockchain Focus:** Smart Contract Hacking (Ethereum, Solana)

### Mobile Application Hacking (Android + iOS Apps)

### Blockchain and Smart Contract Hacking (future $$$)

### AI/ML Model Attacks (Adversarial ML security)

### Firmware/IoT Hacking (low-level reverse engineering)

### 🚀 Final Mindset Reminder

- **It’s a marathon, not a sprint.**
- **1% improvement daily = 37x growth in a year.**
- **Focus on deep understanding, not shallow certifications.**
- **Document everything (build your personal wiki/notion).**
- **Teach what you learn → It makes you a master.**