

# Distributed Systems: A Complete Learning Roadmap

## Part I: Foundations

### Chapter 1: Introduction to Distributed Systems

- What is a Distributed System?
- Why Distributed Systems? (Scalability, Reliability, Performance)
- Challenges in Distributed Systems
- Examples of Distributed Systems (Web services, Cloud computing, P2P networks)
- Distributed vs Centralized vs Decentralized Systems

### Chapter 2: Network Fundamentals

- Network Models (OSI, TCP/IP)
- Communication Protocols (TCP, UDP, HTTP, gRPC)
- Network Latency and Bandwidth
- Remote Procedure Calls (RPC)
- Message Passing vs Shared Memory
- Synchronous vs Asynchronous Communication

### Chapter 3: System Models

- Architecture Models (Client-Server, Peer-to-Peer, Hybrid)
- Interaction Models (Synchronous, Asynchronous)
- Failure Models (Crash failures, Omission failures, Byzantine failures)
- Security Models
- Timing Models (Synchronous, Asynchronous, Partially Synchronous)

## Part II: Core Concepts

### Chapter 4: Time and Ordering

- Physical Clocks and Clock Synchronization
- Cristian's Algorithm
- Berkeley Algorithm
- Network Time Protocol (NTP)
- Logical Clocks (Lamport Timestamps)

- Vector Clocks
- Happened-Before Relationship
- Causal Ordering

## **Chapter 5: Mutual Exclusion and Coordination**

- The Mutual Exclusion Problem
- Centralized Algorithms
- Distributed Algorithms (Ricart-Agrawala, Maekawa's Algorithm)
- Token-Based Algorithms (Token Ring)
- Election Algorithms (Bully Algorithm, Ring Algorithm)
- Distributed Coordination Services (ZooKeeper, etcd)

## **Chapter 6: Consistency Models**

- Strong Consistency
- Eventual Consistency
- Causal Consistency
- Sequential Consistency
- Linearizability
- Read Your Writes Consistency
- Monotonic Reads and Writes
- CAP Theorem Introduction

## **Part III: Fault Tolerance and Reliability**

### **Chapter 7: Fault Tolerance Fundamentals**

- Types of Failures
- Failure Detection (Heartbeat, Ping-Ack)
- Redundancy Techniques
- Process Resilience (Process Groups, Replication)
- Checkpointing and Recovery
- Forward and Backward Error Recovery

### **Chapter 8: Reliable Communication**

- Point-to-Point Communication

- Reliable Multicast
- Atomic Multicast
- Ordered Multicast (FIFO, Causal, Total)
- Gossip Protocols
- Epidemic Algorithms

## **Chapter 9: Replication**

- Why Replication?
- Data-Centric vs Client-Centric Replication
- Replication Strategies (Active, Passive, State Machine)
- Primary-Backup Replication
- Chain Replication
- Quorum-Based Replication
- Consistency vs Availability Trade-offs

## **Chapter 10: Consensus and Agreement**

- The Consensus Problem
- Two-Phase Commit (2PC)
- Three-Phase Commit (3PC)
- Paxos Algorithm (Basic Paxos, Multi-Paxos)
- Raft Consensus Algorithm
- Byzantine Agreement
- Byzantine Fault Tolerance (PBFT)
- Practical Applications of Consensus

## **Part IV: Distributed Data Management**

### **Chapter 11: Distributed Transactions**

- ACID Properties
- Distributed Transaction Processing
- Concurrency Control (Locking, Timestamp Ordering)
- Two-Phase Locking (2PL)
- Optimistic Concurrency Control

- Deadlock Detection and Prevention
- Distributed Deadlock Algorithms

## **Chapter 12: CAP Theorem and BASE**

- CAP Theorem Deep Dive
- Consistency, Availability, Partition Tolerance
- Real-World Trade-offs
- BASE Properties (Basically Available, Soft state, Eventual consistency)
- PACELC Theorem
- Choosing Consistency Models

## **Chapter 13: Distributed Databases**

- Distributed Database Architecture
- Data Partitioning/Sharding Strategies (Range, Hash, Directory-based)
- Data Placement and Replication
- Query Processing in Distributed Databases
- Distributed Joins
- NoSQL Databases (Key-Value, Document, Column-Family, Graph)
- NewSQL Databases

## **Chapter 14: Distributed File Systems**

- File System Fundamentals
- Network File System (NFS)
- Google File System (GFS)
- Hadoop Distributed File System (HDFS)
- Object Storage Systems (Amazon S3)
- Caching and Replication in File Systems

## **Part V: Advanced Topics**

### **Chapter 15: Distributed Computing Frameworks**

- MapReduce Programming Model
- Hadoop Ecosystem
- Apache Spark

- Stream Processing (Apache Kafka, Apache Flink)
- Batch vs Stream Processing
- Data Flow Models

## **Chapter 16: Scalability and Performance**

- Horizontal vs Vertical Scaling
- Load Balancing Strategies
- Consistent Hashing
- Caching Strategies (Local, Distributed, CDN)
- Content Delivery Networks
- Database Indexing and Sharding
- Performance Metrics and Monitoring

## **Chapter 17: Distributed Hash Tables (DHT)**

- DHT Fundamentals
- Chord Protocol
- Kademlia
- Pastry
- CAN (Content Addressable Network)
- Applications in P2P Systems

## **Chapter 18: Microservices Architecture**

- Monolithic vs Microservices
- Service Discovery
- API Gateways
- Service Mesh (Istio, Linkerd)
- Inter-Service Communication
- Circuit Breakers and Bulkheads
- Distributed Tracing
- Container Orchestration (Kubernetes)

## **Chapter 19: Security in Distributed Systems**

- Authentication and Authorization

- Public Key Infrastructure (PKI)
- Secure Communication (TLS/SSL)
- Key Distribution and Management
- Access Control in Distributed Systems
- Byzantine Fault Tolerance Security
- Blockchain Basics

## **Chapter 20: Cloud Computing and Distributed Systems**

- Cloud Service Models (IaaS, PaaS, SaaS)
- Virtualization and Containerization
- Cloud Storage Systems
- Serverless Computing
- Multi-Tenancy
- Elasticity and Auto-Scaling
- Cloud-Native Design Patterns

## **Part VI: Cutting-Edge Topics**

### **Chapter 21: Edge Computing and IoT**

- Edge vs Cloud Computing
- Fog Computing
- IoT Architecture
- Data Processing at the Edge
- 5G and Distributed Systems
- Edge Analytics

### **Chapter 22: Blockchain and Distributed Ledgers**

- Blockchain Fundamentals
- Distributed Consensus in Blockchain
- Proof of Work vs Proof of Stake
- Smart Contracts
- Distributed Ledger Technology
- Cryptocurrency Systems

## **Chapter 23: Machine Learning in Distributed Systems**

- Distributed Training
- Parameter Servers
- Federated Learning
- Data Parallelism vs Model Parallelism
- ML Pipeline Distribution
- Distributed Inference

## **Chapter 24: Eventual Consistency Patterns**

- CRDTs (Conflict-Free Replicated Data Types)
- Operational Transformation
- Anti-Entropy Protocols
- Read Repair and Hinted Handoff
- Vector Clocks in Practice
- Real-World Eventual Consistency

## **Part VII: Practical Implementation**

### **Chapter 25: Design Patterns and Best Practices**

- Saga Pattern
- Event Sourcing
- CQRS (Command Query Responsibility Segregation)
- Bulkhead Pattern
- Retry and Timeout Patterns
- Idempotency
- Backward and Forward Compatibility

### **Chapter 26: Monitoring and Observability**

- Metrics Collection
- Distributed Logging
- Distributed Tracing (Zipkin, Jaeger)
- Health Checks
- Alerting Strategies

- Chaos Engineering
- SLIs, SLOs, and SLAs

## Chapter 27: Case Studies

- Amazon DynamoDB
  - Google Spanner
  - Apache Cassandra
  - Netflix Architecture
  - WhatsApp Infrastructure
  - Uber's Microservices
  - Twitter's Timeline Architecture
  - Lessons from Production Systems
- 

## Learning Path Recommendations

**For Beginners:** Start with Chapters 1-6 to build strong foundations.

**For Intermediate Learners:** Focus on Chapters 7-14 to understand fault tolerance and data management.

**For Advanced Practitioners:** Study Chapters 15-24 for specialized knowledge.

**For Implementation:** Review Chapters 25-27 for practical patterns and real-world examples.

## Recommended Hands-On Projects

1. Build a distributed key-value store
2. Implement a consensus algorithm (Raft)
3. Create a load balancer with consistent hashing
4. Design a simple distributed file system
5. Build a microservices application with service discovery
6. Implement a gossip protocol
7. Create a distributed cache system