CSC 116 — Cybersecurity: An Introduction to Security in Cyberspace

Lecture Notes and Concept Guide

11/12/2025

Chapter 1. Overview

1. Cryptography

Cryptography is like locking your diary so that no one else can read it.

- **Encryption** means locking the message.
- **Decryption** means unlocking it with the correct key.

The purpose is to keep data private, unchanged, and accessible only to the right person.
Think about how hospitals store patient information. If your medical record were sent without encryption, anyone could read it. But encryption ensures only your doctor can open it.
The three key ideas are:

- **Confidentiality** – like sealing a prescription in an envelope so only the pharmacist can open it.
- **Integrity** – making sure no one secretly changes the medicine name.
- **Authentication** – confirming the prescription really came from your doctor, not someone pretending to be them.

Healthcare data is very sensitive because it contains personal details such as your medical history, address, and Social Security number. If leaked, it could lead to identity theft or insurance fraud.

2. System Architecture

A system usually has two main sides:

- The **frontend**, which is what users see (buttons, forms, web pages).
- The **backend**, which processes requests and connects to the **database**.

Imagine you log into a hospital portal. The frontend is the website you interact with, the backend verifies your password, and the database stores your medical results.

3. Security and Privacy

Security is about protecting systems from harm or unauthorized access.
Privacy is about controlling how your personal data is used.
Here are examples to understand the difference:

- Installing a surveillance camera focuses on *security* (protecting the building).
- Refusing an app to track your location focuses on *privacy* (controlling your data).
- Encrypting your messages focuses on *security* (keeping data safe while traveling).
- Using an anonymous username online focuses on *privacy* (protecting identity).

4. Network Security Problems

Typical network issues include:

- **Data Privacy** – sensitive data might leak.
- **Network Delay** – slow connections reduce system quality.
- **Server Crash or Attack** – no service available, like when a hospital system goes down during an attack.

Example: When you visit the University of Miami website, it's safe because it's open to everyone. But your salary information is private and must be protected. Therefore, using a website to access your salary, the sensitive information needs to be protected.

5. HTTP vs HTTPS

HTTP is like sending a postcard — anyone can read it during delivery. HTTPS is like sending a sealed, encrypted letter.

Feature	HTTP	HTTPS

Security	Plain text	Encrypted with SSL/TLS
Certificate	Not required	Requires SSL/TLS certificate
Use Case	Testing, open info	Payments, logins, healthcare data

That's why banks, hospitals, and payroll systems must always use HTTPS.

6. Availability, Safety, and Liveness

- **Availability**: The system must respond and be ready (like a hospital website that's always online).
- **Safety**: It should avoid harmful actions (like preventing unauthorized access to patient data).
- **Liveness**: It should keep progressing (like ensuring the system eventually loads results, even if it's slow).

7. Internet of Things (IoT)

IoT means connecting everyday devices — cameras, cars, fridges — to the Internet. These devices collect and share data.

Examples of vulnerabilities:

- A hacker exploits a camera flaw to view someone's home video feed.
- An employee clicks a phishing email, infecting their laptop, and the malware spreads through the company.
- A smart door lock sends commands without encryption, letting attackers capture the unlock code.

So IoT security is essential, especially in hospitals where devices like heart monitors or insulin pumps are connected to the Internet.

8. Cloud Security

Cloud platforms like AWS host many systems. They must ensure data safety, privacy, and proper access control.

9. Identity and Access Management (IAM)

IAM ensures the right people have the right access at the right time for the right reasons.
Example: In a hospital system, doctors can access patient data, but receptionists cannot see diagnoses.

10. Access Control

Access control limits who can view or use specific resources.
For example, when you log into the UM system, your CaneID verifies your identity, and permissions decide what you can do. This protects sensitive data from unauthorized users and ensures accountability.

Chapter 2. Symmetric and Asymmetric Encryption

Symmetric encryption: Symmetric encryption is a method of securing information where the same secret key is used for both encryption (locking the message) and decryption (unlocking it).
It's like using the same physical key to both lock and unlock a door.

Asymmetric encryption, also called public-key cryptography, uses two different keys — a public key and a private key — for encryption and decryption.
A message encrypted with the public key can only be decrypted with the private key, and vice versa.

1. Ancient Origins of Encryption
The story of encryption began more than 2,000 years ago.
In ancient Greece, messages were encrypted using a tool called a Scytale — a wooden stick around which a strip of parchment was wrapped. When written across the wrapped strip, the letters looked meaningless. Only someone with a Scytale of the same diameter could unwrap it correctly to read the message.
Example:
If you wrote *"This is my first encrypted message"* on the strip, once unwrapped, it might look like *"Tfrehiysirpasstgiteesedmnmyce"* — unreadable unless the recipient had the matching

2. Caesar Cipher and Substitution
Around 2000 years ago, Julius Caesar used a simple substitution cipher to send military messages.
Each letter in the alphabet was shifted by a fixed number of positions.
Example:
If you shift each letter by three places backward,
Plaintext: THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG
Ciphertext: QEB NRFZH YOLTK CLU GRJMP LSBO QEB IXWV ALD
This method was easy to use but also easy to break. It was an important historical step toward more complex encryption.

3. Modern Symmetric Encryption
Modern symmetric encryption uses one shared key for both encryption and decryption — just like the ancient Scytale, but now implemented digitally.
DES (Data Encryption Standard)
- Created by IBM and adopted by the U.S. government in 1977.
- Uses a 56-bit key.
- Once standard, but now insecure because computers can guess the key through brute-force attacks.

AES (Advanced Encryption Standard)
- Designed by Joan Daemen and Vincent Rijmen.
- Key sizes: 128, 192, or 256 bits.
- AES is the most widely used symmetric encryption method today — used in Wi-Fi, banking apps, and disk encryption tools.

4. How Data Looks to a Computer
When you type *Hello*, your computer turns it into binary — zeros and ones.
For example, *Hello World!* becomes:
01001000 01100101 01101100 01101100 01101111
00100000 01010111 01101111 01110010 01101100 01100100 00100001
That's 96 bits or 12 bytes of data.

Encryption turns these bits into new patterns that appear random, but the correct key can restore them to the readable message.

5. Stream and Block Ciphers

Two main approaches to symmetric encryption:

- Stream Cipher: Encrypts data one bit or byte at a time (like ChaCha20).
- Block Cipher: Encrypts data in fixed chunks, such as AES, which encrypts 16 bytes (128 bits) per block.

Why AES is secure:
With a 256-bit key, there are about 1.1×10^{77} possible keys — even with modern GPUs, guessing the right one would take billions of years.

6. Asymmetric Encryption

- A public key (shared openly).
- A private key (kept secret).

Example:
If Alice wants to send Bob a secure message:

1. She encrypts it using Bob's public key.
2. Only Bob's private key can decrypt it.

This system ensures secure communication even if the public key is visible to everyone.
However, asymmetric algorithms like RSA or ECC are slower and suitable only for encrypting small pieces of data (like symmetric keys), not large files.

7. Combining Both: Hybrid Encryption
Most real systems use both methods together:

1. Asymmetric encryption to safely share a symmetric key.
2. Symmetric encryption to securely send large data.

Example: HTTPS (the padlock in your browser)
When you connect to a secure website:

- The browser uses asymmetric encryption to exchange a secret AES key.
- Then it switches to symmetric encryption for fast data transfer.

8. One-Time Pad and Post-Quantum Cryptography
The One-Time Pad is the only encryption method proven to be unbreakable — but it's impractical because it requires a completely random key as long as the message.
Modern encryption (like AES and RSA) is practical and secure — but quantum computing could threaten them in the future.
That's why researchers are developing Post-Quantum Cryptography (PQC) — new algorithms that even quantum computers can't crack.

9. Real-World Applications

- File Encryption: Protects sensitive documents.
- Database Security: Keeps medical or financial records safe.
- Secure Web (HTTPS): Protects your passwords during login.
  SSH or GitHub connections: Use encryption to securely link computers for software development.

Chapter 3. Authentication, Message Authentication Codes, and Digital Signatures

1. Introduction
In a digital world, every message, transaction, or document must answer three critical questions:

1. **Who** sent it?

2. **Was it changed along the way?**
3. **Can the sender later deny it?**

Authentication, Message Authentication Codes (MACs), and Digital Signatures work together to answer these questions. They are the foundation of **trust** in modern computer systems, online services, and secure communication.

2. Authentication: Proving Who You Are

Authentication is the process of verifying a person's identity before granting access. It ensures that the user trying to enter a system is indeed who they claim to be authentication-today

.

2.1. Three Factors of Authentication

1. **Something you have** – a phone, smart card, or campus ID card.
    - Example: Swiping a university card to enter a lab.
2. **Something you know** – a password, passcode, or answer to a security question.
    - Example: Typing a login password and then an SMS code.
3. **Something you are** – a biometric trait such as a fingerprint, facial pattern, or voice.
    - Example: Using Face ID to unlock your phone.

Most secure systems use **multi-factor authentication (MFA)** — combining two or more of these factors. For example, logging in to a hospital portal might require both a password and a one-time verification code sent to your email.

2.2. Tokens and Smart Cards

A **smart card** is a physical token that contains a tiny embedded chip storing secure data. When inserted into a reader, it acts as a digital key that unlocks specific systems. In universities, corporate offices, and hospitals, smart cards are widely used for both physical access and computer login authentication.

2.3. Biometric and AI-based Authentication

AI has expanded authentication into new areas:

- **Facial recognition** uses computer vision models to match faces.
- **Behavioral biometrics** monitor typing rhythm or walking patterns.
- **Emotion recognition** analyzes facial expressions to verify identity or detect anomalies. **Anomaly detection** powered by deep learning identifies suspicious login patterns (e.g., logging in from a new country at midnight).

Example:

Amazon Go's cashier-free stores use computer vision and RFID sensors to authenticate who picked up which item, charging customers automatically without manual checkout.

3. Hashing and Message Authentication Codes (MACs)

Even if authentication proves *who* sent a message, we still need to ensure *the message itself* has not been changed. That's where hashing and MACs come in

Digital Signature is an encrypt…

.

3.1. Hash Functions

A **hash function** is a mathematical tool that converts any input — such as a password, file, or medical record — into a fixed-length string of characters called a **hash value**.

Example:

SHA-256("Hello") =
2cf24dba5fb0a30e26e83b2ac5b9e29e1b161e5c1fa7425e73043362938b9824

Key properties of hash functions:

1. The same input always produces the same output.
2. The hash length is fixed regardless of input size.

3. It is nearly impossible to reverse-engineer the original message.
4. Even a tiny change in input (like changing one letter) causes a completely different hash.

Because of these properties, hashes are used to:

- Check file integrity.
- Store passwords safely instead of plain text.
- Detect tampering during data transmission.

3.2. Password Hashing and Attacks

Storing plain passwords is dangerous. Instead, systems store only their hashes.
When you log in, your entered password is hashed again — if the hashes match, access is granted.
However, hackers use **rainbow tables** — precomputed lists of hashes — to reverse weak passwords. To prevent this, strong passwords and **salted hashing** (adding random data before hashing) are used.

3.3. Message Authentication Code (MAC)

A **MAC** ensures both **integrity** and **authenticity** of a message.
It works like a digital seal attached to a message that only the sender and receiver can generate or verify.

Example:

1. Alice and Bob share a secret key K.
2. Alice sends message M = "Hello" with its MAC = HMAC(K, M).
3. Bob receives both. He recalculates HMAC(K, M).
    - If his result matches Alice's MAC, the message is authentic and unaltered.
    - If not, the message has been tampered with.

Why not use only encryption?

Because encryption alone ensures confidentiality — but not authenticity. A hacker could still send fake encrypted data pretending to be Alice. A MAC guarantees that the data really came from the expected sender.

Real-world example:

When your bank app confirms a transaction, it includes a hidden MAC value to ensure no one modified your payment details during transmission.

4. Digital Signatures

A **digital signature** is an encrypted hash that verifies who signed a document and guarantees it has not been changed
authentication-today
.

4.1. Concept

Just as you sign a paper contract with a pen, a digital signature proves your identity electronically.
It provides four important assurances:

1. **Authenticity:** Confirms the signer's identity.
2. **Integrity:** Ensures the document hasn't been changed after signing.
3. **Non-repudiation:** The signer cannot later deny having signed.
4. **Notarization:** The signature can be legally verified by a trusted authority.

4.2. How a Digital Signature Works

1. **Key Generation:**
   The signer creates a pair of keys — a **private key** (kept secret) and a **public key** (shared with others).
2. **Hashing the Document:**
   The system applies a hash function (e.g., SHA-256) to the document to create a unique "fingerprint."
   Signing:
   The signer encrypts this hash using their **private key**. The result is the **digital signature**.
   Verification:
   The recipient uses the signer's **public key** to decrypt the signature. If the decrypted hash matches the hash of the received document, the signature is valid.

Example:
If you sign a medical report digitally using your private key, the hospital's system can verify the signature using your public key, ensuring the report is authentic and unaltered.

4.3. Certificates

One challenge remains: how can others trust that your *public key* really belongs to you?
This problem is solved by **digital certificates**, issued by **Certificate Authorities (CAs)** such as DigiCert or Let's Encrypt.
These certificates bind your identity to your public key, proving to others that it is genuine.

Chapter 4: Database

1. Understanding Databases

A **database** is an organized collection of information that allows people and computers to store, manage, and retrieve data efficiently. It works like a digital filing system — instead of keeping hundreds of paper folders, all the information is stored electronically and can be searched or updated within seconds.
In daily life, databases are everywhere. Your **university system** keeps student names, courses, and grades in a database. **Hospitals** use databases to store patient information, medical history, and prescriptions. **Online stores** record customer orders, product details, and payment records in their databases. Without databases, all of these systems would collapse under the amount of data they handle.
To control and organize this data, we use a **Database Management System (DBMS)** such as MySQL, PostgreSQL, or Oracle. A DBMS acts as a bridge between the user and the data. It lets users perform four basic actions: **create**, **read**, **update**, and **delete** information — often called **CRUD operations**. For example, when a student changes their address in the university system, that is an update operation handled by the DBMS.
Most modern databases are **relational**, meaning the data is stored in tables that are linked together by **keys**. Each table has rows (records) and columns (fields).

For example, one table may store student information:

StudentID	Name	Major
101	Alice	Computer Sci

102	Bob	Engineering

Another table may store course enrollment:

CourseID	CourseName	StudentID
C01	Database	101
C02	Networking	102

Here, the **StudentID** is the key that connects both tables, showing which student is enrolled in which course.
The language used to communicate with databases is called **SQL** (Structured Query Language). SQL allows users to select, insert, and modify data easily.
For example:
SELECT Name FROM Students WHERE Major = 'Computer Sci';
This command retrieves all student names from the Computer Science major.

2. Importance and Real-World Impact
Databases are vital because they bring **order, reliability, and security** to data. They prevent duplication, ensure consistency, and allow multiple people to work on the same data without conflicts.
Consider a **banking system**: every time someone transfers money, deposits a check, or withdraws cash, all those transactions are stored instantly in the bank's database. This ensures that balances remain accurate even when millions of users are active at the same time.
In **healthcare**, databases ensure that doctors can access a patient's complete medical history at any hospital in the network.
In **e-commerce**, databases help track which items are in stock, which are sold out, and who bought them.
Unlike simple spreadsheets such as Excel, databases can handle huge amounts of information from multiple users at once while maintaining speed and integrity. They also support **backup and recovery**, so even if a server fails, data can be restored without loss.
In short, a database is not just a place to store information; it is the foundation that keeps modern digital systems organized and trustworthy. Whether managing a small business or running a global application like Netflix or Amazon, everything depends on databases working quietly in the background.

Chapter 5: Blockchain

1 What is Blockchain?
A **blockchain** is a decentralized digital ledger that records transactions across multiple computers so that the record cannot be altered retroactively. Each block in the chain contains:

- A **timestamp**,
- A **list of transactions**, and
- A **cryptographic hash** of the previous block.

This creates a secure and immutable chain of data — once added, data cannot be easily changed or deleted.

Unlike traditional databases controlled by a single organization, blockchain operates on **distributed consensus**: every participant (node) in the network keeps a synchronized copy of the ledger.

Example

Imagine a class attendance sheet:

- In a normal setting, only the teacher keeps it.
- In blockchain, *every student* has a copy.
  If someone tries to change the attendance record, everyone else's copy reveals the discrepancy.

2 Why Blockchain is Needed

Traditional systems suffer from:

- **Single Point of Failure (SPOF)** — if the main server fails, the whole system stops.
- **Central Authority Dependence** — users must trust a single entity to maintain and secure data.
- **Tampering Risk** — data in a centralized database can be edited or deleted by insiders.

Blockchain solves these issues by distributing data across many nodes.
Each node verifies and stores the same dataset — ensuring that even if some nodes fail, the system continues to work.

3. Core Features

Feature	Description	Example
Decentralization	No central control; all nodes maintain a copy of the ledger.	Bitcoin network
Transparency	All participants can view recorded transactions.	Public block explorers
Immutability	Once data is confirmed, it cannot be modified.	Prevents fraud in banking
Security	Uses cryptographic hashes and digital signatures.	Protects medical data
Consensus	All nodes agree on the validity and order of transactions.	Byzantine Fault Tolerance

4 Byzantine Fault Tolerance (BFT)

In real systems, some nodes might behave maliciously or fail unexpectedly.
BFT (Byzantine Fault Tolerance) ensures that the network can still reach agreement even if up to **⅓ of nodes are faulty or malicious**..

Consensus Phases

1. **Phase 0:** Leader (instructor) broadcasts the message.
2. **Phase 1:** All nodes (students) broadcast what they heard to others.
3. **Phase 2:** Each node confirms it has received matching messages from most nodes.

Only when a sufficient majority (≥2f+1 of total n nodes) agree, is the message accepted as *true*.
To tolerate f faulty nodes: n-f >= 2f+1, then we get: $n \geq 3f+1$
Example:

- 4 nodes → tolerate 1 faulty
- 7 nodes → tolerate 2 faulty

- 10 nodes → tolerate 3 faulty

Applications of BFT
- **Blockchain:** ensures all nodes share the same transaction history
- **Financial Systems:** guarantees the order of transactions is correct
- **Distributed Databases:** keeps replicas consistent even if one server fails

Advantages
- Prevents fraud and data loss
- Maintains service availability
- Ensures all nodes agree on the same data (total order)

Drawbacks
- Slower consensus (due to message broadcasting)
- High latency and scalability limits
- Can only tolerate ≤33% malicious nodes
- Complex maintenance

5 Different Types of Blockchains

Type	Description	Example Use
Public Blockchain	Open to everyone; fully decentralized.	Bitcoin, Ethereum
Private Blockchain	Controlled by a single organization; faster but less transparent.	Hyperledger Fabric
Consortium Blockchain	Managed by a group of organizations.	Banking networks
Hybrid Blockchain	Combines features of public and private blockchains.	Healthcare record sharing

Each type balances **security**, **privacy**, and **efficiency** differently depending on the application.

6 Applications of Blockchain
1. **Healthcare** —Hospitals use blockchain to ensure patient data integrity and privacy. Example: Patient consent forms and medical histories stored securely.
2. **Finance** —Transactions are verified without central banks, reducing fraud.
3. **Supply Chain** — Blockchain tracks product movement, ensuring authenticity (e.g., anti-counterfeit).
4. **Voting Systems** — Each vote is recorded immutably, improving trust and transparency.
5. **Education** — Diplomas and transcripts can be verified instantly across institutions.

7 Conclusion
Blockchain represents a paradigm shift from **centralized trust** to **distributed verification**. Its ability to tolerate faults, ensure consistency, and maintain transparency makes it a foundational technology for future systems — from digital currencies to secure healthcare networks.

Chapter 6: Advanced Cryptography and Privacy Technologies

1 Introduction

Modern privacy technologies protect data even while it is being used, not just when it is stored or sent. This chapter introduces several techniques that allow organizations to share and process data securely without revealing sensitive information.
The main topics include:

- **Homomorphic Encryption (HE)**
- **Secure Multi-Party Computation (SMPC)**
- **Zero-Knowledge Proofs (ZKP)**
- **Differential Privacy**
- **Watermarking and Data Ownership**

These tools are widely used in healthcare, finance, and artificial intelligence to ensure collaboration without losing privacy.

6.2 Homomorphic Encryption (HE)

Definition:
Homomorphic Encryption allows data to remain encrypted even while calculations are performed on it. The result, once decrypted, is the same as if the operations had been done on the original data.
Example:
A hospital stores patient data in encrypted form.
A researcher can calculate the average age or blood sugar level directly on the encrypted data.
When the result is decrypted, the researcher gets the correct answer — but never saw the original patient data.
Applications:

- Secure medical data analysis.
- Financial modeling without exposing client data.
- Cloud computing where the service provider never sees user information.

Limitations:

- Works mainly on numbers.
- Slower than normal computation.
  Limited to basic operations like addition and multiplication.

6.3 Secure Multi-Party Computation (SMPC)

Definition:
Secure Multi-Party Computation allows multiple people or organizations to jointly calculate a result without revealing their individual data.
Example:
Several hospitals want to know the average blood pressure of their combined patients, but cannot share private medical records.
Each hospital inputs encrypted numbers, and the system calculates the average securely.
The final result is correct, but no one learns the others' data.
How It Works:
Each participant's data is split into random pieces that look meaningless.
Only when all parts are combined can the correct result be produced.
No single participant can reconstruct others' data.
Trusted Environments:
Sometimes, SMPC runs inside secure computer hardware — known as a **Trusted Execution**

Environment (TEE) — so even the system administrator cannot view the data while it is being processed.

Applications:

- Secure healthcare collaboration
- Privacy-preserving voting
- Joint AI model training between companies
- Shared control of digital assets

6.4 Zero-Knowledge Proofs (ZKP)

Definition:

A Zero-Knowledge Proof allows someone to prove that a statement is true without revealing *why* it's true or any additional information.

Example:

You want to prove you are over 18 to access a medical form.
Instead of showing your ID (which reveals your name and address), you can use a Zero-Knowledge Proof to confirm that you meet the age requirement — without exposing anything else.

Applications:

- In **healthcare**, a hospital can verify patient eligibility for a study without viewing full records.
- In **finance**, ZKPs are used in private blockchain transactions to verify correctness without revealing amounts or participants (for example, in Zcash).
- In **digital identity**, users can prove who they are without sharing private details.

Why It's Powerful:

Zero-Knowledge Proofs build trust between parties that don't fully trust each other — useful wherever privacy and verification must coexist.

6.5 Differential Privacy

Definition:

Differential Privacy protects individuals' information when data is shared or analyzed.
It adds small random changes ("noise") to each data point so that overall statistics remain accurate, but no one can trace the result back to a specific person.

Example:

Hospitals add tiny random differences to each patient's age before calculating an average.
The average stays the same overall, but no one can identify a single patient's real age.

Uses:

Apple and Google use differential privacy when collecting user data, ensuring trends can be analyzed without identifying specific users.

6.6 Watermarking and Data Ownership

Definition: Watermarking hides identifying information inside a file (such as an image or document) to prove who owns or created it.

Example in Healthcare:

A hospital can insert a hidden watermark into MRI images that identifies the radiologist or institution.
If the image is stolen or reused without permission, the watermark can prove who owns it.

Blockchain Connection:

Blockchain records *who owns what* by storing ownership data in a public ledger, while watermarking embeds ownership *inside* the data itself.
Used together, they provide strong protection for intellectual property and data integrity.

6.7 Summary

Advanced privacy-preserving technologies allow secure collaboration in a world where data must stay private but still useful.

- **Homomorphic Encryption** enables computation on encrypted data.
- **Secure Multi-Party Computation** lets multiple organizations work together without revealing information.
- **Zero-Knowledge Proofs** prove authenticity without disclosure.
- **Differential Privacy** protects individuals in shared datasets.
  Watermarking ensures ownership and prevents misuse.

Together, these techniques are shaping the future of secure data sharing and ethical AI.

Chapter 7: Artificial Intelligence and Machine Learning Applications

7.1 Introduction to Artificial Intelligence

Artificial Intelligence (AI) aims to design machines that can think and learn like humans. The term *Artificial Intelligence* was first defined at a conference that explored whether machines could simulate human reasoning and learning.

This idea laid the foundation for many subfields, including **Machine Learning (ML)**, **Natural Language Processing (NLP)**, and **Computer Vision**.
AI systems are built around the idea of **models** — structures that take **inputs**, process them using rules or learned patterns, and produce **outputs**. For example, an AI model might predict whether a person will enjoy a movie based on their interests, country, ticket price, weather, and other factors. Each feature is assigned a **weight**, which determines its importance in the prediction.

Learning, in this context, means adjusting these weights until the predictions match real-world outcomes.

7.2 Early Models: From Logic to Learning

The earliest AI models were based on logical reasoning — “if A and B, then true” — but these symbolic approaches struggled with uncertainty and large data.
Later, **perceptrons** and **multi-layer perceptrons (MLPs)** emerged. These networks consist of interconnected nodes that can learn patterns through data, enabling them to classify complex problems such as images or medical diagnoses.
For example:

- A perceptron might determine whether a patient should be selected for a medical study based on two conditions: age and citizenship.
- When multiple perceptrons are combined (as in MLPs), they can model non-linear relationships, such as predicting whether a CT image shows a tumor.

Training such models involves:

1. Randomly initializing weights.
2. Measuring prediction errors using a **loss function**.

3. Adjusting weights through **gradient descent** to minimize that loss.
4. Repeating the process until performance stabilizes.

This process requires **well-labeled datasets** — for instance, images labeled as "benign" or "malignant", so the model will understand how to find the best weights to distinguish between them.

7.3 Machine Learning Models

Machine learning models fall broadly into two categories:

1. **Classification Models**
   These models predict *categories*.
   - Example: Distinguishing between benign and malignant breast tumors.
   - Example: Detecting whether a CT scan shows bleeding.
   - Example: Classifying diabetic complications as *low*, *medium*, or *high risk*.
2. **Regression Models**
   These models predict *numerical values*.
   - Example: Predicting how many days a patient will stay in the hospital.
   - Example: Forecasting changes in a Parkinson's patient's clinical score over time.

The performance of these models is measured using:

- **Training Accuracy**: How well the model fits the training data.
- **Testing Accuracy**: How well it performs on unseen data (reflects generalization).
- **AUC/ROC (Area Under the Curve)**: Measures how well the model distinguishes between classes; 1.0 is perfect, 0.5 is random guessing.

When models perform well in training but poorly in real-world cases, **fine-tuning** — retraining a pre-trained model on a specific dataset — can greatly improve performance.

7.4 Explainable AI (XAI)

Modern healthcare AI must not only be accurate but also **explainable**.
Tools like **SHAP (SHapley Additive exPlanations)** visualize which features influence a prediction. For example, if a Parkinson's model predicts "high risk," SHAP can show that "age," "sleep disorder," and "cognitive test score" were key contributors.
Explainability builds **trust** — especially when AI systems are used for clinical decisions.

7.5 AI in Healthcare Applications

AI is transforming hospitals in multiple ways:

- **Faster Diagnosis**: Automatically detects abnormalities in medical images.
- **Predictive Analytics**: Forecasts patient outcomes and readmission risks.
- **Personalized Treatment**: Suggests treatments based on genetics or medical history.
- **Virtual Assistants**: Provides patients with 24/7 help for reminders, or FAQs.
- **Abnormal Prescription Detection**: Identifies suspicious or risky medication patterns.

In cybersecurity, AI also helps hospitals detect phishing emails or ransomware threats before they cause harm.

7.6 Natural Language Processing (NLP) and Transformers

NLP allows computers to understand and generate human language.
Models like **BERT** and **GPT** process clinical text to:

- Extract symptoms, diagnoses, or medications from medical notes.
- Summarize patient messages and route them to the right department.
- Support clinical decision systems by matching patient data with guidelines.

Example:
A patient types, *"I feel dizzy after taking the medication."*
A keyword-based system might only catch the word *"dizzy."*
But BERT understands the entire context — realizing the dizziness is caused by medication — and alerts medical staff.
How Transformers Work:
Transformers use a **self-attention** mechanism to assign importance to words based on their relationships in a sentence.
This allows the model to "understand" meaning across context, not just by proximity.

7.7 Federated Learning

One of the biggest challenges in healthcare AI is **data privacy**.
Hospitals cannot simply pool patient data because of confidentiality rules.
Federated Learning (FL) solves this problem by allowing models to be trained across multiple hospitals **without moving the data**.
Each institution trains the model locally on its own data and only sends **model updates** (not patient records) to a central aggregator.
The aggregator combines updates from all hospitals to improve the shared model, ensuring collaboration without violating privacy laws.
Example:
Three hospitals want to build a model to predict Parkinson's disease progression. Each trains the model on its local dataset. Only the learned parameters are shared and aggregated. No sensitive patient data ever leaves the institution.
This approach protects privacy, reduces data transfer, and encourages multi-institution cooperation.

7.8 Challenges and Ethical Considerations

Despite its benefits, AI introduces new concerns:

- **Bias**: If the training data is unbalanced (e.g., skewed toward one demographic), predictions can be unfair.
- **Transparency**: Clinicians must understand *why* a model made a decision.
- **Security**: AI systems are attractive targets for cyberattacks.
- **Trust**: Users must be confident that models are reliable, especially in life-critical applications.

These issues underscore the importance of explainability, fairness, and accountability in AI systems.