**PRESENTATION**

**Name:Srinu Peddavarapu**

**Student:** **S02073306**

**Course:** **Introduction to Cybersecurity**

1.**What is Cryptography**

Definition in slide

Explain this example

Think of when you shop on Amazon. When you enter your credit card details to make a purchase, cryptography ensures that your sensitive information is securely encrypted. This means your card details are turned into a coded format that hackers can't read, even if they intercept the data. Only Amazon's secure systems can decode it to process your payment safely.

* AES

AES stands for Advanced Encryption Standard, a cryptographic algorithm that protects electronic data. It's a symmetric block cipher that uses the same key to encrypt and decrypt data.

**Tools Required**

* **Python**:
  + Install Python 3 from the official Python website.
* **Cryptography Library**:
  + Install via the terminal with the following command:
  + pip install cryptography
* **IDE**:
  + Python IDE like PyCharm or Jupyter Notebooks for running scripts.
* **What is Encryption?**
  + Converts plaintext into ciphertext (unreadable format) using a key.
* **What is Decryption?**
  + Converts ciphertext back to readable data (plaintext) using the same key.

**Demo Plan Explanation of Results**

* **Generated Key**:
  + A randomly generated key that must be kept secure.
* **Encrypted Text**:
  + The unreadable ciphertext.
* **Decrypted Text**:
  + The original message successfully retrieved after decryption.

**Python Script for AES Encryption and Decryption**

**pip install cryptography**

from cryptography.fernet import Fernet

# Step 1: Generate a Key

key = Fernet.generate\_key()

cipher\_suite = Fernet(key)

print("Generated Key (Keep it Secret):", key)

# Step 2: Encrypt Data

plaintext = b"This is a secret message."

ciphertext = cipher\_suite.encrypt(plaintext)

print("\nEncrypted Text:", ciphertext)

# Step 3: Decrypt Data

decrypted\_text = cipher\_suite.decrypt(ciphertext)

print("\nDecrypted Text:", decrypted\_text.decode())

**Mitigation Strategies**

* **Key Management**:
  + Secure key storage using hardware security modules (HSMs).
* **Strong Keys**:
  + Use complex, random keys to prevent brute-force attacks.
* **Secure Communication Channels**:
  + Use HTTPS to protect data in transit.

**Best Practices**

* **Use a Unique Key for Each Session**:
  + Avoid reusing encryption keys to enhance security.
* **Regularly Rotate Encryption Keys**:
  + Change keys periodically to reduce the risk of exposure.
* **Ensure Proper Access Control**:
  + Limit access to encryption keys and sensitive data.