Networks and Systems Security
Week 02
Foundations of Network Security

Aims of the Seminar

Welcome to the Foundations of Network Security. The aim is to build on the lecture material of giving you a practical understanding of security needs and technologies.

By the end of this workshop, you will be able to:

- Discuss computer security concepts in practical contexts
- Apply the **RASA algorithm** to analyse security scenarios

Workshop Outline:

1. RSA Algorithm
2. Socket library
3. Assesment work

Feel free to discuss your work with peers, or with any member of the teaching staff.

Reminder

We encourage you to discuss the content of the workshop with the delivery team and any findings you gather from the session.

Workshops are not isolated, if you have questions from previous weeks, or lecture content, please come and talk to us.

Exercises herein represent an example of what to do; feel free to expand upon this.

Helpful Resources

Cryptography library
https://cryptography.io/en/latest/

Socket library

https://realpython.com/python-sockets/

RSA Algorithm

https://www.geeksforgeeks.org/computer-networks/rsa-algorithm-cryptography/

3. Setting up

Install the cryptography library

```
pip install cryptography
```

Create 3 python scripts with the following scripts:

Step 1: Key Generation (Run Once)

We'll generate an RSA key pair and save it (simulate key exchange):

```
# generate_keys.py
from cryptography.hazmat.primitives import serialization
from cryptography.hazmat.primitives.asymmetric import rsa

# Generate private key
private_key = rsa.generate_private_key(public_exponent=65537, key_size=2048)

# Save private key
with open("private_key.pem", "wb") as f:
    f.write(
        private_key.private_bytes(
            encoding=serialization.Encoding.PEM,
            format=serialization.PrivateFormat.PKCS8,
            encryption_algorithm=serialization.NoEncryption()
        )
    )

# Save public key
public_key = private_key.public_key()
with open("public_key.pem", "wb") as f:
    f.write(
        public_key.public_bytes(
            encoding=serialization.Encoding.PEM,
            format=serialization.PublicFormat.SubjectPublicKeyInfo
        )
    )

print("☑ Keys saved: private_key.pem, public_key.pem")
```

Step 2: Receiver (Server) – Receive & Decrypt

```python
# receiver.py
import socket
import pickle
from cryptography.hazmat.primitives import serialization, hashes
from cryptography.hazmat.primitives.asymmetric import padding
from cryptography.hazmat.primitives.ciphers import Cipher, algorithms, modes

# Load private key
with open("private_key.pem", "rb") as f:
    private_key = serialization.load_pem_private_key(f.read(), password=None)

# Start server
with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
    s.bind(("localhost", 65432))
    s.listen()
    print("👂 Waiting for connection...")
    conn, addr = s.accept()
    with conn:
        print(f"🔗 Connected by {addr}")
        data = b""
        while True:
            chunk = conn.recv(4096)
            if not chunk:
                break
            data += chunk

# Unpack payload
encrypted_key, iv, encrypted_message = pickle.loads(data)

# 1. Decrypt AES key with RSA private key
aes_key = private_key.decrypt(
    encrypted_key,
    padding.OAEP(
        mgf=padding.MGF1(algorithm=hashes.SHA256()),
        algorithm=hashes.SHA256(),
        label=None
    )
)
# 2. Decrypt message with AES
cipher = Cipher(algorithms.AES(aes_key), modes.CFB(iv))
decryptor = cipher.decryptor()
message = decryptor.update(encrypted_message) + decryptor.finalize()

print("🔓 Decrypted message:", message.decode())
```

Step 3: Sender (Client) – Encrypt & Send

```python
# sender.py
import socket
import os
from cryptography.hazmat.primitives import serialization, hashes
from cryptography.hazmat.primitives.asymmetric import padding
from cryptography.hazmat.primitives.ciphers import Cipher, algorithms, modes
import pickle

# Load recipient's public key
with open("public_key.pem", "rb") as f:
    public_key = serialization.load_pem_public_key(f.read())

# Message to send
message = b"Hello from the secure sender! This is confidential."

# 1. Generate random AES key and IV
aes_key = os.urandom(32)  # AES-256
iv = os.urandom(16)

# 2. Encrypt message with AES (CFB mode)
cipher = Cipher(algorithms.AES(aes_key), modes.CFB(iv))
encryptor = cipher.encryptor()
encrypted_message = encryptor.update(message) + encryptor.finalize()

# 3. Encrypt AES key with RSA (recipient's public key)
encrypted_key = public_key.encrypt(
    aes_key,
    padding.OAEP(
        mgf=padding.MGF1(algorithm=hashes.SHA256()),
        algorithm=hashes.SHA256(),
        label=None
    )
)

# 4. Package: (encrypted_key, iv, encrypted_message)
payload = pickle.dumps((encrypted_key, iv, encrypted_message))

# 5. Send via socket
with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
    s.connect(("localhost", 65432))
    s.sendall(payload)
    print("☑ Encrypted message sent!")
```

Group work:

Peer Discussion and Feedback

Form pairs with another student. Each pair will discuss their findings from last week assessment work, and help each other reflect on their career readiness.

Instructions:

1. **Exchange Summaries:**
    a. Share your three chosen job roles and company research with your partner.
    b. Briefly explain why you selected these roles and what you found most appealing or challenging.
2. **Compare Skills Gap Analyses:**
    a. Review each other's skills gap tables.
    b. Identify at least two skills you both need to develop and suggest practical ways to strengthen them (e.g., online courses, volunteering, student projects).
    c. Highlight one unique strength your partner has that could help them stand out to employers.
3. **Mutual Feedback:**
    a. Give each other constructive feedback on your readiness for the chosen roles.
    b. Reflect together on how university learning or extracurricular activities can bridge remaining gaps.
4. **Joint Reflection:**
    a. Write a short (around 150 words) joint summary discussing what you both learned from comparing your career paths and skills.
    b. Submit this reflection alongside your individual work.

Outcome:

This activity encourages collaboration, peer learning, and realistic self-assessment. By discussing your career goals and skill development plans with a partner, you will gain new perspectives and refine your own professional strategy.

The end 😊