Lab Report 3

Due: Nov 14, 2024

Adrian Nelson Aidan Chin

Code

Code Snippets:

Python

```python
# ./chat_client.py

# make keypairs
public, private = generate_keypair(p, q)

message = "public_key: %d %d private_key: %d %d" % (
       public[0],
       public[1],
       private[0],
       private[1],
)  # send over private and public keys

# encrypt the message, make string so code.encode works
message_encoded = [str(encrypt(public, char)) for char in message]

# ./chat_server.py

keys = re.search(r"public_key: (\d+) (\d+) private_key: (\d+) (\d+)", data)
# use regex to find message of key
public_key_e = int(keys.group(1))
public_key_n = int(keys.group(2)) # extract private and public keys
private_key = (int(keys.group(3)), int(keys.group(4)))

data_decoded = decrypt(private_key, cipher) # decrypt text

# ./image_client.py

public, private = generate_keypair(p, q)  # make keypairs

message = "public_key: %d %d private_key: %d %" % (
       public[0],
       public[1],
       private[0],
       private[1],
)  # send over private and public keys
```

```
des_encoded = []
for char in des_key:  # encode each char in des_key
	encrypted_char = str(encrypt(public, char))
	des_encoded.append(encrypted_char)

# code taken from ECE371/lab2/main.py
coder = des.des()
r = coder.encrypt(des_key, data, cbc=False)  # encrypted image
# write the encrypted image into file
r_byte = bytearray()
for x in r:
	r_byte += bytes([ord(x)])

# ./image_server.py

# code taken from ./chat_server.py
	keys = re.search(
		r"public_key: (\d+) (\d+) private_key: (\d+) (\d+)", data
	)  # use regex to find message of key
	public_key_e = int(keys.group(1))
	public_key_n = int(keys.group(2))  # extract private and public keys
	private_key = (int(keys.group(3)), int(keys.group(4)))

for _ in range(8):
		(data, addr) = mySocket.recvfrom(SIZE)
		encrypted_byte = int(
		data.decode()
		)  # collect part of key and convert to int
		decrypted_byte = decrypt(private_key, encrypted_byte)  # decrypt
		des_key += decrypted_byte  # append to des_key

# code taken from ECE371/lab2/main.py and modified with des_key_decoded =
des_key and r = data
	decoder = des.des()  # set decoder to des.des()
	des_key_decoded_str = ""
	for i in des_key:  # convert des_key to string
		des_key_decoded_str = des_key_decoded_str + str(i)
	rr = decoder.decrypt(  # decrypt without cipher block chaining
		des_key, data, cbc=False
	)  # this is in string  format, must convert to byte format
	rr_byte = bytearray()
	for x in rr:  # convert to ASCII then to bytes
		rr_byte += bytes([ord(x)])
	# write to file to make sure it is okay
```

```
file2 = open(r"penguin_decrypted.jpg", "wb")
file2.write(bytes(rr_byte))
file2.close()
```

chat_client.py

Make keypairs: using the code from lab2 we just utilize the function to create the keys for us

Send over public keys: make a string with the public and private keys to send over network, you cannot decode without the private key so we needed to send that too

Encrypt the message: for each character in the message, we encrypt using the function built before and convert the whole thing to a string to be ready to be sent over the port

chat_server.py

Use regex to find message of key: we collect the message sent over from the other alterra and use regex to extract the keys from the message

Decrypt text: simply run the text through the decrypt function we made in lab 2

image_client.py

Make keypairs and send: same as in chat_client.py

Encode each char: for each char in the des_key we encrypt using rsa to be prepared to be sent over

code taken from ECE371/lab2/main.py: set coder to des.des, encrypt the image and store in a byte array to be sent over

image_server.py

code taken from ./chat_server.py: see use regex in chat_server.py section

code taken from ECE371/lab2/main.py and modified with des_key_decoded = des_key and r = data: using the code from ECE371/lab2/main.py, we repurpose it to decode the sent over image, decrypt without cipher block chaining, and since it is in string format, we convert to byte format, then we convert to ascii then into bytes to be read as a .jpeg file
We write this into the penguin_decrypted.jpeg file

Images ← in album so we can send over a video of it working on 2 alterras