LAMPIRAN

Lampiran 1 Surat Izin Penelitian

UMKT
Program Studi
Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Muhammadiyah Kalimantan Timur
Berkarakter | Berwawasan | Berkemajuan
Telp. 0541-748511 Fax.0541-766832
Website http://informatika.umkt.ac.id
email: informatika@umkt.ac.id

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

Nomor : 056-009/KET/FST.1/A/2024
Lampiran : -
Perihal : **Keterangan Melakukan Penelitian**

Assalamu'alaikum Warrahmatullahi Wabarrakatuh

Puji Syukur kepada Allah Subhanahu wa ta'ala yang senantiasa melimpahkan Rahmat-Nya kepada kita sekalian. Amin.

Dengan surat ini, kami menerangkan bahwa mahasiswa berikut:

No	Nama	NIM
1	Amelda Auniyah	1911102441019
2	Winda Amelia Pratiwi	1911102441105
3	Ima Sulistia Nopi Wulandari	1911102441173
4	Nadhiya Safira Arrahmah	1911102441112

Melakukan penelitian dengan citra digital menggunakan algoritma kriptografi kunci publik.

Demikian hal ini disampaikan, atas kerjasamanya kami ucapkan terima kasih.

Wassalamu'alaikum Warrahmatullahi Wabarrakatuh

Samarinda, 28 Dzulhijjah 1445 H
5 Juli 2024 M

Ketua Program Studi S1 Teknik Informatika

Arbansyah, S.Kom., M.TI
NIDN. 1118019203

Lampiran 2 Citra Digital

Lampiran 3 Kartu Kendali Bimbingan

KARTU KENDALI BIMBINGAN LAPORAN SKRIPSI

Nama Mahasiswa : Amelda Auniyah

NIM : 1911102441019

Nama Dosen Pembimbing : Sayekti Harits Suryawan, S.kom., M.Kom.

Judul Penelitian : PENERAPAN ALGORITMA *RSA* PADA CITRA DIGITAL

No	Tanggal	Uraian Pembimbingan	Paraf Dosen
1	7/03/2024	Membahas alur-alur penulisan pada bab1 dan metode yang digunakan	
2	24/04/2024	Membahas untuk latar belakang, rumusan masalah, tujuan penelitian, dan manfaat penelitian.	
3	05/05/2024	Memperbaiki di latar belakang bagian penelitian sebelumnya (Alfaozi, 2021	
4	07/05/2024	Membahas penjelasan pada isi dari bab 2 metode penelitian	
5	08/05/2024	Perbaikan diagram pada bab 2 dan menjelaskan isi diagram tersebut	
6	10/05/2024	Perbaikan proposal pada bab 2 pada penjelasan diagram	
7	13/05/2024	Revisi penjelasan dan pembahasan untuk memproses coding	
8	15/05/2024	Acc proposal pada bab1 dan bab 2 untuk riview desk	
9	05/06/2024	Menambahkan entropi pada bab 2 dan revisian proposal setelah selesai review desk	
10	26/06/2024	Pembahasan coding untuk bab 3 dan penyusunan	
11	28/06/2024	Memperbaiki coding untuk penerapan algoritma	

Dosen Pembim bing

(Sayekti Harits Suryawan, S.kom., M.Kom)
NIDN:1119048901

Ketua Program Studi

(Arbansyah, S.Kom., M.TI)
NIDN:1118019203

Lampiran 4 Seluruh Coding Python

```python
# Fungsi Extended Euclidean Algorithm
def extended_gcd(a, b):
    if a == 0:
        return b, 0, 1
    g, x1, y1 = extended_gcd(b % a, a)
    x = y1 - (b // a) * x1
    y = x1
    return g, x, y

# Fungsi untuk menghasilkan kunci publik dan privat RSA
def generate_rsa_keys():
    p = sympy.randprime(11, 20)
    q = sympy.randprime(11, 20)
    while p == q:
        q = sympy.randprime(11, 20)

    n = p * q
    phi = (p - 1) * (q - 1)
```

```python
    e = random.randrange(2, phi)
    g = sympy.gcd(e, phi)
    while g != 1:
        e = random.randrange(2, phi)
        g = sympy.gcd(e, phi)

    d = mod_inverse(e, phi)

    public_key = (e, n)
    private_key = (d, n)

    return public_key, private_key

# Fungsi untuk enkripsi pesan menggunakan RSA
def rsa_encrypt(message, public_key):
    e, n = public_key
    cipher = pow(message, e, n)
    return cipher
```

```python
# Fungsi untuk dekripsi pesan menggunakan RSA
def rsa_decrypt(cipher, private_key):
    d, n = private_key
    message = pow(cipher, d, n)
    return message

# Fungsi untuk menghitung entropi dari distribusi nilai piksel dalam gambar
def calculate_entropy(image):
    pixels = np.array(image)
    histogram, _ = np.histogram(pixels, bins=256, range=(0, 256))
    histogram = histogram.astype(float) / pixels.size
    histogram = histogram[histogram != 0]
    entropy = -np.sum(histogram * np.log2(histogram))
    return entropy
```

```python
# Fungsi untuk mengacak gambar
def scramble_image(image):
    random.seed(1234)  # Seed untuk hasil yang deterministik
    pixels = image.load()
    width, height = image.size
    scramble_map = {}

    for x in range(width):
        for y in range(height):
            r, g, b = pixels[x, y]
            r_scrambled = (r + random.randint(0, 255)) % 256
            g_scrambled = (g + random.randint(0, 255)) % 256
            b_scrambled = (b + random.randint(0, 255)) % 256
            pixels[x, y] = (r_scrambled, g_scrambled, b_scrambled)
            scramble_map[(x, y)] = (r_scrambled - r, g_scrambled - g, b_scrambled - b)

    return image, scramble_map
```

```python
# Fungsi untuk mengembalikan pengacakan gambar
def unscramble_image(image, scramble_map):
    pixels = image.load()
    width, height = image.size

    for x in range(width):
        for y in range(height):
            r, g, b = pixels[x, y]
            r_diff, g_diff, b_diff = scramble_map[(x, y)]
            r_unscrambled = (r - r_diff) % 256
            g_unscrambled = (g - g_diff) % 256
            b_unscrambled = (b - b_diff) % 256
            pixels[x, y] = (r_unscrambled, g_unscrambled, b_unscrambled)

    return image
```

```python
# Function for image data extraction
def extract_image_data(image):
    image_data = {
        "Ukuran": image.size,
        "Mode Warna": image.mode,
        "Format": image.format,
    }
    return image_data

# Fuction to upload image
def upload_and_process_image():
    upload = FileUpload(accept='image/*', multiple=False)
    container = VBox([upload])

    def on_upload_change(change):
        for filename, file_info in upload.value.items():
            img = Image.open(BytesIO(file_info['content']))
            display(img)

            # Ekstraksi data gambar
            image_data = extract_image_data(img)
            print("Data Gambar:", image_data)
```


```python
            # Ekstraksi data gambar
            image_data = extract_image_data(img)
            print("Data Gambar:", image_data)
```


```python
            # Generate RSA keys
            public_key, private_key = generate_rsa_keys()
            print('Public Key:', public_key)
            print('Private Key:', private_key)

            # Hitung entropi gambar asli
            original_entropy = calculate_entropy(img)
            print(f'Original Image Entropy: {original_entropy:.2f}')

            # Scramble image
            print('Scrambling image...')
            scrambled_img, scramble_map = scramble_image(img.copy())
            display(scrambled_img)

            # Hitung entropi gambar setelah diacak
            scrambled_entropy = calculate_entropy(scrambled_img)
            print(f'Scrambled Image Entropy: {scrambled_entropy:.2f}')
```

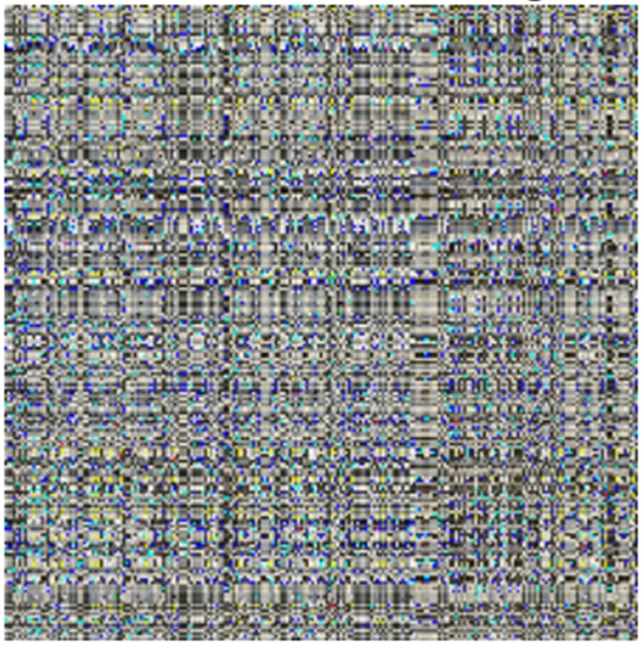

```python
            # Hitung entropi gambar setelah diacak
            scrambled_entropy = calculate_entropy(scrambled_img)
            print(f'Scrambled Image Entropy: {scrambled_entropy:.2f}')
```


```python
            # Encrypt scrambled image
            print('Encrypting image...')
            encrypted_img = scrambled_img.copy()
            pixels = encrypted_img.load()
            width, height = encrypted_img.size

            for x in range(width):
                for y in range(height):
                    r, g, b = pixels[x, y]
                    r_encrypted = rsa_encrypt(r, public_key) % 256
                    g_encrypted = rsa_encrypt(g, public_key) % 256
                    b_encrypted = rsa_encrypt(b, public_key) % 256
                    pixels[x, y] = (r_encrypted, g_encrypted, b_encrypted)

            display(encrypted_img)

            # Hitung entropi gambar setelah enkripsi
            encrypted_entropy = calculate_entropy(encrypted_img)
            print(f'Encrypted Image Entropy: {encrypted_entropy:.2f}')
```

```python
            # Decrypt encrypted image
            print('Decrypting image...')
            decrypted_img = encrypted_img.copy()
            pixels = decrypted_img.load()

            for x in range(width):
                for y in range(height):
                    r_encrypted, g_encrypted, b_encrypted = pixels[x, y]
                    r_decrypted = rsa_decrypt(r_encrypted, private_key) % 256
                    g_decrypted = rsa_decrypt(g_encrypted, private_key) % 256
                    b_decrypted = rsa_decrypt(b_encrypted, private_key) % 256
                    pixels[x, y] = (r_decrypted, g_decrypted, b_decrypted)

            display(decrypted_img)

            # Hitung entropi gambar setelah dekripsi
            decrypted_entropy = calculate_entropy(decrypted_img)
            print(f'Decrypted Image Entropy: {decrypted_entropy:.2f}')

            # Unscramble decrypted image
            print('Unscrambling image...')
            unscrambled_img = unscramble_image(decrypted_img.copy(), scramble_map)
            display(unscrambled_img)
```

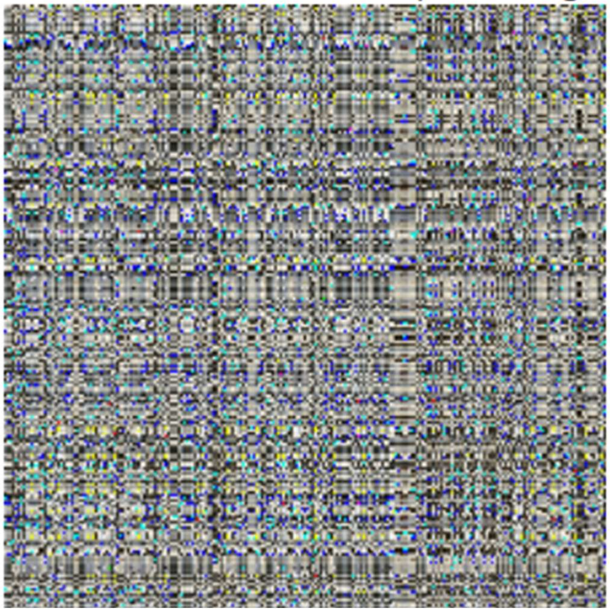
```python
            # Hitung entropi gambar setelah unscrambling
            unscrambled_entropy = calculate_entropy(unscrambled_img)
            print(f'Unscrambled Image Entropy: {unscrambled_entropy:.2f}')

            # Pastikan entropi setelah unscrambling sama dengan entropi asli
            if np.isclose(original_entropy, unscrambled_entropy, atol=0.01):
                print('Entropy check passed: The entropies are close enough.')
            else:
                print('Entropy check failed: The entropies differ.')

    upload.observe(on_upload_change, names='value')
    display(container)

print('Upload an image to scramble and encrypt using RSA:')
upload_and_process_image()
```

Gambar setelah scrambling:



Encrypting image...

Gambar setelah dekripsi dengan RSA:



Gambar setelah unscrambling: