Penambahan Lapisan Keamanan dengan Fitur Enkripsi Teks Menggunakan Algoritma RSA dalam Aplikasi Notion

Muhammad Helmi Hibatullah - 13520014
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail (gmail): 13520014@std.stei.itb.ac.id

***Abstrak*—Kini pengguna dapat menulis dan menyimpan teks secara digital pada aplikasi yang terhubung dengan layanan *cloud*. Salah satu contoh aplikasi pencatatan yang populer adalah Notion. Notion memiliki banyak fitur, seperti fitur kolaborasi, tabel basis data, kecerdasan buatan, dan lain-lain. Dibalik kepopulerannya, Notion memiliki celah keamanan, yaitu data pengguna tidak terenkripsi secara *end-to-end* yang berarti data pengguna dapat dilihat oleh karyawan Notion. Algoritma RSA dapat digunakan untuk menambahkan lapisan keamanan dengan enkripsi teks sehingga pengguna dapat menyimpan data yang sesensitif apapun pada aplikasi ini tanpa harus khawatir adanya pihak ketiga yang dapat melihat data tersebut.**

Keywords—Notion, Kerahasiaan, Algoritma Kunci Publik, Algoritma RSA.

I. PENDAHULUAN

Menyimpan teks secara digital kini sudah berkembang dengan pesat. Pengguna saat ini tidak hanya dapat menuliskan dan menyimpan teks secara digital pada penyimpanan lokal perangkat pengguna, namun pengguna juga dapat menuliskan dan menyimpannya secara langsung pada *cloud*. Dengan adanya *cloud*, teks yang ditulis oleh pengguna dapat langsung diakses oleh pengguna lain dari berbagai perangkat asalkan pengguna tersebut memberikan hak aksesnya ke pengguna lain. Bahkan, pengguna tersebut dapat membuka hak akses terhadap tulisannya sehingga memungkinkan kolaborasi antar pengguna dalam menulis.

Pada aplikasi pencatatan berbasis *cloud*, seperti Notion, pengguna dapat menuliskan apapun yang ia inginkan dalam suatu halaman dan tulisan tersebut akan langsung terkirim ke servernya Notion kemudian disimpan pada penyedia layanan *cloud* yang Notion gunakan, jika pengguna terkoneksi ke internet. Selain menulis, Notion juga memiliki banyak fitur lainnya yang dapat meningkatkan pengalaman pengguna dalam mencatat, seperti fitur tabel basis data, fitur kolaborasi, fitur kecerdasan buatan, fitur *embedded* URL, dan masih banyak lainnya. Semua fitur ini menjadikan Notion sebagai salah satu aplikasi terbaik dalam menyatukan segala pekerjaan di dalam satu tempat.

Notion juga sudah memiliki berbagai lapisan keamanan yang baik, seperti enkripsi data, infrastruktur yang aman serta handal, dan keamanan organisasi yang baik. Bahkan, Notion Labs, perusahaan pembuat Notion, sudah mematuhi SOC 2 tipe dua pada bulan Agustus 2021 lalu. Artinya, Notion Labs sudah melewati proses audit keamanan ketat yang memeriksa apakah perusahaan mengikuti proses kebersihan data dan kemanan internal yang baik. Berdasarkan artikel yang Notion Labs buat tentang praktik keamana yang mereka jalankan, data pengguna dienkripsi pada saat *at rest* dan *in transit*. Pada saat *at rest*, data pengguna di penyimpanan *cloud*, basis data, dan pencadangan pada aplikasi Notion telah dienkripsi menggunakan AES-256. Sedangkan pada saat *in transit*, data yang sedang ditransmisi telah dienkripsi menggunakan TLS 1.2 atau lebih.

Sayangnya, Notion Labs tidak menyebutkan bahwa data pengguna pada produknya telah dienkripsi secara *end-to-end*. Pengguna tidak memiliki tanggung jawab atas proses enkripsi ataupun dekripsi karena Notion Labs lah menyimpan kunci yang diperlukan untuk kedua hal tersebut pada data penggunanya. Secara umum, ini berarti bahwa pengguna tidak bisa menggunakan aplikasi Notion untuk mencatat kata sandi, rekening bank, kartu kredit, data kesehatan, dan data lain yang bersifat sangat personal. Bahkan, melalui fitur dukungan Notion, pengguna dapat diminta untuk memberikan akses ke dokumen pengguna untuk dapat menyediakan support. Ini berarti bahwa setiap pegawai di Notion Labs dapat melihat semua hal yang pengguna tulis di Notion. Jika Notion sudah mengenkripsi data pengguna secara *end-to-end*, hal ini tidak akan mungkin terjadi tanpa informasi yang dibutuhkan untuk mendekripsi data pengguna.

Untuk menambahkan lapisan kemanan pada aplikasi Notion, pengguna dapat memanfaatkan fitur *embedded* URL milik Notion itu sendiri. Dengan fitur ini, pengguna dapat menyematkan halaman web sederhana yang dapat mengenkripsi dan mendekripsi pesan yang pengguna masukkan. Beberapa algoritma kriptografi juga dapat memungkinkan pengguna menggunakan fitur kolaborasi pada Notion dengan tetap menjaga kerahasiaan datanya.

Gambar 1 Tampilan Sebuah Catatan pada Aplikasi Notion Android dengan Menyematkan URL Youtube.

II. TEORI DASAR

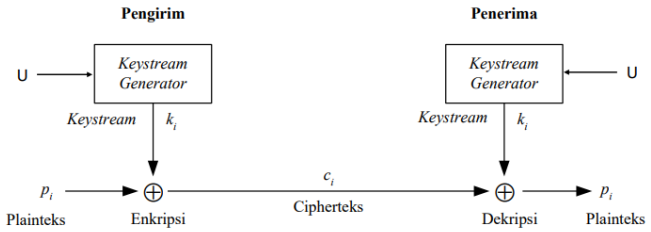
A. *Kriptografi Modern, Cipher Alir, dan Cipher Blok*

Kriptografi modern adalah kriptografi di era setelah penemuan komputer digital yang beroperasi dalam mode bit atau *byte*. Kriptografi modern, sama seperti kriptografi klasik, menggunakan dua teknik dasar, yaitu teknik substitusi (teknik mengganti karakter atau bit) dan teknik transposisi (teknik menggeser karakter atau bit). Selain itu, kriptografi modern juga menggunakan teknik lain, seperti rotasi, kompresi, ekspansi, penjumlahan modulo, dan lain-lain. Operasi-operasi yang ada pada kriptografi modern bersifat lebih kompleks daripada kriptografi klasik untuk menylitkan kriptanalisis.

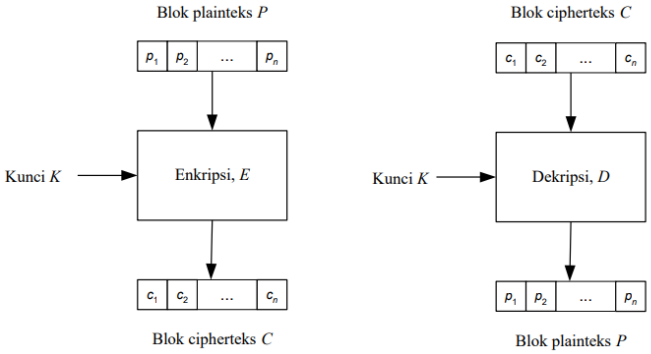
Kriptografi modern yang berbasis bit dapat dibagi menjadi dua kategori, yaitu cipher alir dan cipher blok. Cipher alir beroperasi pada bit atau *byte* tunggal dan mengenkripsi atau mendekripsi pesan dilakukan secara bit per bit atau *byte* per *byte*. Sedangkan, cipher blok beroperasi pada blok bit atau blok *byte* dan mengenkripsi atau mendekripsi pesan dilakukan secara blok per blok.

Pada cipher alir, bit-bit aliran kunci untuk enkripsi dan dekripsi disebut sebagai *keystream*. Aliran kunci ini dibangkitkan oleh sebuah *keystream generator*. Enkripsi dan dekripsi menggunakan cipher alir ini komputasinya sederahana dan juga cepat. Untuk mendapatkan bit-bit cipherteks c1, c2, …, dst, *keystream* k1, k2, …, dst di-XOR-kan dengan aliran bit-bit plainteks p1, p2, …, dst. Sedangkan untuk mendapatkan plainteks pada saat dekripsi, operasinya tinggal dibalik. Keamanan dari cipher alir ini sangat bergantung pada *keystream* yang dihasilkan oleh *generator*-nya. Semakin acak keluaran yang dihasilkan oleh *keystream generator*, semakin sulit juga kriptanalis memecahkan cipherteks.

$$c_i = p_i \oplus \mathrm{k_i} \quad (1)$$
$$p_i = c_i \oplus \mathrm{k_i} \quad (2)$$



Gambar 2 Diagram Cipher Alir.



Gambar 3 Diagram Cipher Blok Secara Umum.

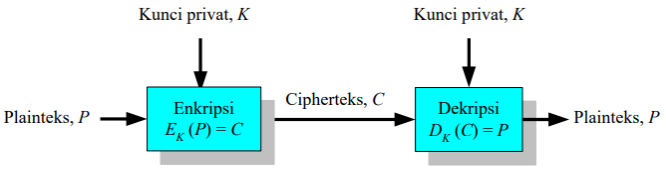
Pada cipher blok, pemrosesan plainteks terbagi menjadi blok-blok bit dengan panjang sama, biasanya 64 bit, 128 bit, 256 bit, dan lain sebagainya. Tidak seperti cipher alir, panjang kunci yang dimasukkan tidak harus sama dengan panjang blok plainteks. E dan D pada Gambar 3 merupakan cipher yang digunakan masing-masing untuk enkripsi dan dekripsi.

$$C = E(P) \quad (3)$$
$$P = D(C) \quad (4)$$

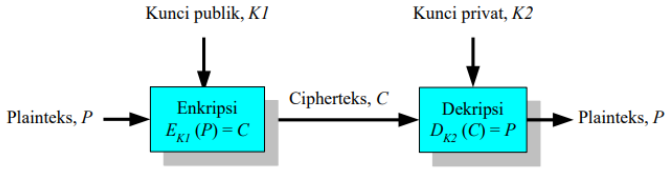
Algoritma yang dapat digunakan untuk E dan D didesain sangat rumit prosesnya agar sulit untuk dipecahkan. Beberapa contoh algoritma blok cipher yang cukup terkenal adalah DES, Triple DES, GOST, RC5, RC6, Blowfish, Twofish, Mars, dan Serpent.

B. *Kriptografi Kunci-Simetri dan Kunci-Publik*

Kriptografi kunci-simetri adalah sistem kriptografi yang proses enkripsi dan dekripsinya menggunakan kunci yang sama. Kriptografi kunci-publik bisa disebut juga kriptografi kunci-nirsimetri karena kunci yang digunakan untuk enkripsi dan dekripsi berbeda. Istilah "publik" ini muncul karena kunci untuk melakukan enkripsi tidak rahasia dan dapat dibuka ke publik. Sedangkan, kunci untuk melakukan dekripsi dirahasiakan oleh pemilik kunci sehingga disebut juga kunci privat.



Gambar 4 Sistem Kriptografi Kunci-Simetri.



Gambar 5 Sistem Kriptografi Kunci-Nirsimetri.

Kriptografi kunci-simetri memiliki beberapa kelebihan sebagai berikut.

1. Proses enkripsi dan dekripsi yang lebih cepat.
2. Ukuran kunci yang lebih pendek.
3. Pengirim pesan langsung diketahui dari cipherteksnya karena hanya pengirim dan penerima saja yang mengetahui kunci aslinya.

Sedangkan kelemahan dari sistem kriptografi ini adalah sebagai berikut.

1. Kunci simetri harus dikirim melalui saluran yang aman dan tidak sama dengan saluran pengiriman pesan.
2. Kedua pihak penerima dan pengirim harus saling menjaga kerahasiaan kunci.
3. Kunci harus sering diubah agar tetap aman.

Berikut adalah beberapa lebeihan dari kriptografi kunci-publik.

1. Pihak yang berkomunikasi hanya perlu menjaga kerahasiaan kunci privat.
2. Pasangan kunci tidak perlu sering diubah seperti kriptografi kunci-simetri.
3. Dapat digunakan untuk mengamankan kunci simetri.
4. Beberapa algoritma kunci-publik dapat digunakan untuk melakukan tanda tangan digital.

Sedangkan berikut adalah kelemahan dari kriptografi kunci-publik.

1. Proses enkripsi dan dekripsi yang lebih lambat dari kunci-simetri.
2. Ukuran cipherteks yang lebih besar daripada plainteks.
3. Cipherteks tidak dapat memberikan informasi pengirim karena kunci publik untuk melakukan enkripsi diketahui publik.
4. Tidak ada algoritma kunci-publik yang terbukti aman (sama seperti cipher blok).

C. Algoritma Rivest-Shamir-Adleman (RSA)

Sesuai dengan namanya, algoritma Rivest-Shamir-Adleman (RSA) adalah algoritma kunci-publik terkenal yang dibuat oleh Ronal Rivest, Adi Shamir, dan Leonard Adleman pada tahun 1976. Algoritma ini memiliki letak keamanan pada sulitnya memfaktorkan bilangan bulat besar yang menjadi faktor-faktor prima.

Berikut adalah properti-properti pada algoritma RSA.

1. p dan q bilangan prima (rahasia)
2. n = p . q
3. $\phi(n) = (p - 1)(q - 1)$ (rahasia)
4. e (kunci enkripsi)
5. d (kunci dekripsi) (rahasia)
6. m (plainteks) (rahasia)
7. c (cipherteks)

Berikut adalah rumus untuk melakukan enkripsi dan dekripsi pada algoritma ini.

$$c = E_e(m) = m^e mod\ n \quad (5)$$
$$m = D_e(c) = c^d mod\ n \quad (6)$$

Berikut adalah cara membangkitkan kunci publik (e, n) dan kunci privat (d, n).

1. Pilih dua bilangan prima, p dan q (sebaiknya p berbeda dengan q).
2. Hitung n = pq.
3. Hitung $\phi(n) = (p - 1)(q - 1)$.
4. Pilih sebuah bilangan bulat e sebagai kunci publik, e harus relatif prima terhadap $\phi(n)$.
5. Hitung kunci dekripsi d dengan persamaan

$$ed \equiv 1 \pmod{\phi(n)} \quad (7)$$

III. PEMBAHASAN

A. Perancangan Program

Program yang dibuat merupakan sebuah halaman web sederhana yang tidak memiliki koneksi ke server. Program tersebut memiliki fitur untuk membangkitkan kunci dan melakukan enkripsi serta dekripsi berdasarkan kunci yang pengguna masukkan. Tujuan dari dibuatnya program sebagai halaman web sederhana adalah agar pengguna dapat menyematkan halaman web ini ke dalam catatan miliknya di aplikasi Notion. Selain itu, program tidak boleh terkoneksi dengan server apapun untuk menghindari bocornya plainteks pada saat pengiriman maupun pada saat pemrosesan teks. Proses enkripsi dan dekripsi murni dilakukan di sisi klien.

Algoritma yang digunakan pada program enkripsi teks ini adalah algoritma RSA. Algoritma ini digunakan dengan mempertimbangkan keberjalanan fitur kolaborasi yang ada pada aplikasi Notion. Karena algoritma RSA merupakan algoritma kriptografi kunci-publik, program dapat memungkinkan pengguna lain untuk menuliskan pesan rahasia di dalam catatan yang dibuat bersama atau kolaborasi dengan mengenkripsinya terlebih dahulu menggunakan kunci publik pemilik catatan. Jika program menggunakan algoritma kriptografi kunci-simetris, maka semua pengguna yang menulis pesan harus menjaga kerahasiaan kunci yang dibuat oleh pemilik catatan sehingga celah keamanannya menjadi lebih lebar.

Walaupun algoritma kriptografi kunci-publik memiliki waktu pemrosesan yang lebih lambat dibandingkan algoritma kriptografi kunci-simetri, tetapi algoritma kunci-publik jauh lebih unggul di sisi keamanan dan kenyamanan pengguna nantinya. Selain itu, alasan pemilihan algoritma RSA dibandingkan algoritma kunci-publik lainnya adalah waktu komputasinya yang cukup cepat dan mudah diimplementasikan dibandingkan algoritma lain, seperti algoritma ElGamal dan protokol pertukaran kunci Diffie-Hellman.

Program web sederhana ini diimplementasikan menggunakan HTML, CSS, dan Javascript murni. Terdapat beberapa fungsi yang dibuat untuk mengimplementasikan algoritma RSA, yaitu fungsi generateKeys, fungsi encrypt, dan fungsi decrypt. Ketiga

fungsi ini merupakan fungsi *wrapper* untuk pemanggilan library algoritma RSA yang sudah dibuat.

```
const generateKeys = () => {
  rsa.generateKeyPair({ bits: 2048, workers: -1 }, function (err, keypair) {
    setPublicKey(keypair.publicKey);
    setPrivateKey(keypair.privateKey);
    setN(keypair.publicKey.n.toString(16));
    setKey(keypair.publicKey.e.toString(16));
    setIsGenerated(true);
  });
};
```

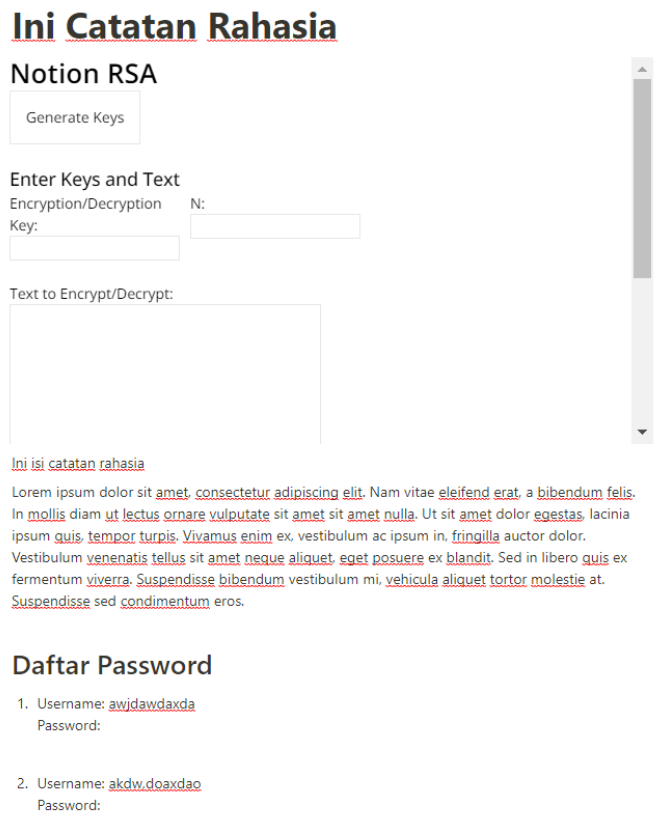
Gambar 2 Fungsi generateKeys

```
const encrypt = () => {
  const bytes = forge.util.encodeUtf8(text);
  const encrypted = publicKey.encrypt(bytes);
  const hex = forge.util.bytesToHex(encrypted);
  setResult(hex);
};

const decrypt = () => {
  const bytes = forge.util.hexToBytes(text);
  const decrypted = privateKey.decrypt(bytes);
  const utf8 = forge.util.decodeUtf8(decrypted);
  setResult(utf8);
};
```

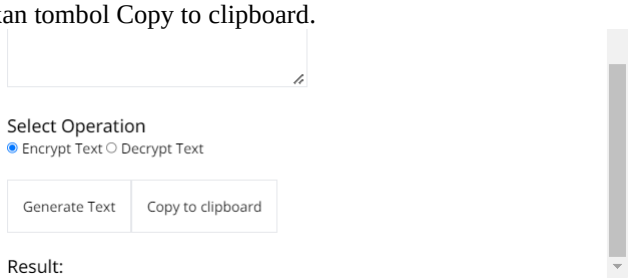
Gambar 3 Fungsi encrypt dan decrypt

Berikut adalah tampilan awal dari program enkripsi sederhana yang dibuat dengan menggunakan framework NextJS.



Gambar 9 Tampilan Awal Program Enkripsi Sederhana yang Sudah Disematkan pada Catatan Notion

Pada Gambar 10 dapat dilihat bahwa program memiliki fitur untuk membangkitkan kunci, memasukkan kunci, dan memasukkan teks yang ingin dienkripsi ataupun didekripsi. Ketika pengguna menekan tombol Generate Keys, akan muncul pasangan kunci publik dan kunci privat. Selain itu, jika pengguna memasukkan kunci, N, dan teks kemudian menekan tombol Generate Text, maka hasil enkripsi/dekripsi akan muncul di bawah tulisan "Result:". Pengguna juga dapat menyalin hasil enkripsi/dekripsinya langsung dengan cara menekan tombol Copy to clipboard.



Gambar 10 Tampilan Awal Program Bagian Bawah

B. Percobaan dan Analisis

Sebagai persiapan sebelum melakukan percobaan, dibuat terlebih dahulu kunci publik dan kunci privat yang berasal dari pembangkit kunci sebagai berikut.

```
KUNCI PUBLIK
Key: 10001

N:
d95df373a0fb13d983a15f3703914db7e1e44686
847cf206e4cca7d6a43801d5ac9358c517c0d947
002f3acff067d53233386048be64f70861f64823
d7b12ccbad126fe1b05af2dfd39c0e15b08926fe
58707b61069bc59bfcfbb801a4a31390ddb4edd0
b9be0bc136216d23f96d639ef0872543404392ac
c8fd089042b55d26460b88e28ea852e9e63d14dd
00864f99f8f16cdaa68f1e1112beb0e8ccc68f41
c684ae72d7f9a0eb6873b26aa8ff69e5deccec83
eadaa7d2b861c8beabe5d71212559bfaa722d521
390ae5e5f140714f49ba772d815e4bdcb9625c74
0e6fc8df73a53378bd7472c227a2c24c9f64052e
4cdb917d13a10dcbe806318af54a32c7
```

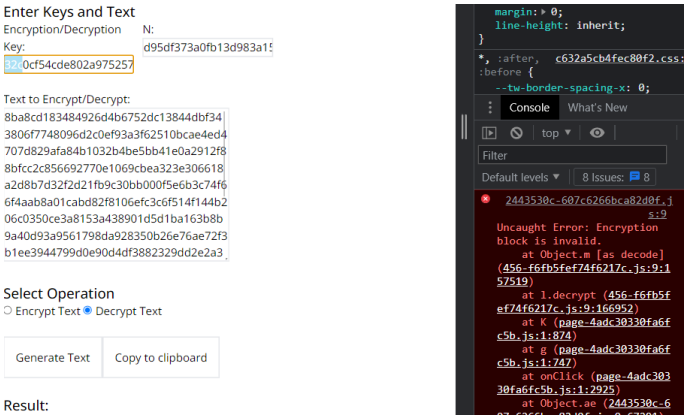
```
KUNCI PRIVAT
Key:
43b0cf54cde802a9752576da520118fbcf9be01e
99f9b30154952b1b5bae995baa4f849074323937
b4260121aa531b807394f5df1173adbe6d8b5db1
d3c4b817818141583e2c4480518cc2921c94c990
e187bdd45c537c3974920b1e6fdfdd02fb18120b
331cacda7179fd5bfb9c062f03e682432c2b1eed
31f62289924c9a81f2d5f38ddd60a8fd18a465dd
20a3ead77122ca0e5282904dee51d59c268019fb
a45e9fcf120d3cd8abe1da3cbb1d84a95792a730
117124b5b5bc48aef15d76c5cbcd5aab31148d5d
92b25793d163929466a11935315fd14e1a941a6c
efbda363868ed35a5152a0f758e49583d75995f6
2bc2ec86f5a8c60671f24f1c4bdfd151
```

```
N:
d95df373a0fb13d983a15f3703914db7e1e44686
847cf206e4cca7d6a43801d5ac9358c517c0d947
002f3acff067d53233386048be64f70861f64823
d7b12ccbad126fe1b05af2dfd39c0e15b08926fe
58707b61069bc59bfcfbb801a4a31390ddb4edd0
b9be0bc136216d23f96d639ef0872543404392ac
c8fd089042b55d26460b88e28ea852e9e63d14dd
00864f99f8f16cdaa68f1e1112beb0e8ccc68f41
c684ae72d7f9a0eb6873b26aa8ff69e5deccec83
eadaa7d2b861c8beabe5d71212559bfaa722d521
390ae5e5f140714f49ba772d815e4bdcb9625c74
0e6fc8df73a53378bd7472c227a2c24c9f64052e
4cdb917d13a10dcbe806318af54a32c7
```

Untuk percobaan pertama, program diuji dengan melakukan enkripsi dan dekripsi seperti biasa terlebih dahulu. Hal ini dilakukan untuk memastikan fungsi-fungsi dasar dari program berjalan dengan baik. Sesuai harapan, ketika melakukan enkripsi dengan kunci yang diberikan pembangkit di dalam program, pesan terenkripsi dengan baik. Begitu juga dengan dekripsi, jika kunci yang dimasukkan berasal dari pembangkit, maka plainteks akan didapatkan dengan benar.

Gambar 11 Percobaan Satu Proses Enkripsi dan Dekripsi Normal

Untuk percobaan kedua, program diuji dengan memasukkan kunci privat yang berbeda dari hasil pembangkitan. Hal ini dilakukan untuk membuktikan bahwa hanya orang yang mengetahui kunci privat yang dapat mendekripsi pesan yang ada. Ketika tiga karakter pertama dari kunci privat diubah, dapat dilihat pada gambar 12 bahwa proses dekripsi tidak terjadi. Hal ini membuktikan bahwa tidak sembarang orang dapat mendekripsi pesan rahasia jika tidak mengetahui kunci privatnya.



Gambar 12 Percobaan Dua Proses Dekripsi dengan Kunci yang Salah

IV. KESIMPULAN

Algoritma kriptografi kunci-publik RSA dapat digunakan untuk menambahkan lapisan keamanan di sisi *end-user* pada aplikasi Notion. Kelebihan dari penggunaan algoritma RSA adalah pengguna dapat menggunakan fitur kolaborasi untuk menuliskan pesan rahasia karena kunci untuk melakukan enkripsi dibuka ke publik. Namun, hanya pemilik kunci privat yang dapat mendekripsi cipherteksnya. Hal ini membuat para pengguna tidak merasa nyaman karena harus meminta tolong pemilik kunci untuk mendekripsi pesan yang sudah ditulis. Karena kekurangan ini, masih terdapat algoritma kriptografi kunci-publik lain yang dapat digunakan untuk menambahkan lapisan keamanan dengan tetap memperhatikan kenyamanan dalam menulis teks secara kolaborasi, seperti *Group Key Exchange System* yang implementasinya jauh lebih rumit dibandingkan RSA.

VI. UCAPAN TERIMA KASIH

Penulis ucapkan terima kasih kepada Allah Swt. karena berkat rahmat dan karunianya penulis mampu menyelesaikan makalah yang berjudul "Penambahan Lapisan Keamanan dengan Fitur Enkripsi Teks Menggunakan Algoritma RSA pada Aplikasi Notion". Kemudian, penulis ingin berterima kasih kepada Bapak Rinaldi Munir selaku dosen mata kuliah IF4020 Kriptografi atas segala ilmu dan pedoman yang telah diberikan. Tak lupa juga penulis berterima kasih kepada kedua orang tua serta teman-teman penulis yang selalu mendukung dalam menjalani perkuliahan di kampus.

REFERENSI

[1] *Security & privacy*. Notion Labs. Diakses pada 23 Mei 2023, dari https://www.notion.so/security

[2] Robert Scott. *We're SOC 2 Type 2 compliant—Here's what that means for you* (2 Agustus 2021). Notion Labs. Diakses pada 23 Mei 2023, dari https://www.notion.so/blog/notion-soc-2-compliant

[3] Munir, R. (2023). *Algoritma RSA*. Program Studi Teknik Informatika STEI-ITB. https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2022-

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 21 Mei 2023

Muhammad Helmi Hibatullah
13520014