Implementasi Tanda Tangan Digital pada Surat Keterangan Sakit

Karlsen Adiyasa Bachtiar - 13519001
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail (gmail): 13519001.std.stei.itb.ac.id

***Abstract*—Beredarnya surat keterangan sakit palsu membuat perusahaan rugi karena harus meliburkan pegawainya yang berpura-pura sakit. Selain itu, proses pengidentifikasian surat keterangan sakit tersebut biasanya dilakukan dengan menelpon nomor yang tertera. Namun cara ini tetap dapat dipalsukan karena belum tentu nomor yang tertera tersebut merupakan dokter asli. Oleh karena itu, untuk menjamin identitas dan keaslian surat tersebut, tanda tangan digital perlu diimplementasikan. Dengan diterapkannya tanda tangan digital pada surat keterangan sakit, diharapkan dapat membantu perusahaan menghindari surat keterangan sakit palsu dan membuat pegawai yang memiliki surat keterangan sakit asli tidak dicurigai.**

Keywords—Digital Signature,* tanda tangan digital, *RSA, SHA-256, hash, kriptografi, surat keterangan sakit

I. PENDAHULUAN

Pemalsuan surat keterangan sakit adalah masalah yang serius dan merugikan dalam konteks masyarakat dan tempat kerja. Pemalsuan surat keterangan sakit adalah praktik yang melibatkan pembuatan atau pengeditan surat keterangan sakit palsu untuk tujuan tertentu, seperti menghindari tanggung jawab atau memperoleh keuntungan pribadi.

Pemalsuan surat keterangan sakit memiliki dampak negatif yang luas. Pertama, itu merugikan para pekerja yang sebenarnya sakit dan membutuhkan waktu untuk pulih. Pemalsuan surat keterangan sakit dapat menghalangi akses mereka ke hak cuti sakit yang sah dan perlindungan yang sesuai dalam lingkungan kerja.

Selain itu, pemalsuan surat keterangan sakit juga berdampak negatif pada efisiensi dan produktivitas tempat kerja. Kehadiran yang rendah atau absen karena alasan palsu dapat menyebabkan penundaan proyek, ketidakseimbangan beban kerja, dan meningkatkan biaya penggantian karyawan.

Selanjutnya, pemalsuan surat keterangan sakit juga memiliki konsekuensi hukum. Tindakan ini melanggar undang-undang yang mengatur kegiatan kesehatan dan ketenagakerjaan. Individu yang terlibat dalam pemalsuan surat keterangan sakit dapat dihadapkan pada tuntutan hukum, sanksi di tempat kerja, atau bahkan kehilangan pekerjaan mereka.

Dalam hal ini, teknologi juga dapat digunakan sebagai alat untuk mengatasi pemalsuan surat keterangan sakit. Penggunaan tanda tangan digital dengan algoritma RSA untuk memverifikasi identitas penandatangan dan keaslian surat keterangan sakit dapat membantu dalam mendeteksi pemalsuan.

II. DASAR TEORI

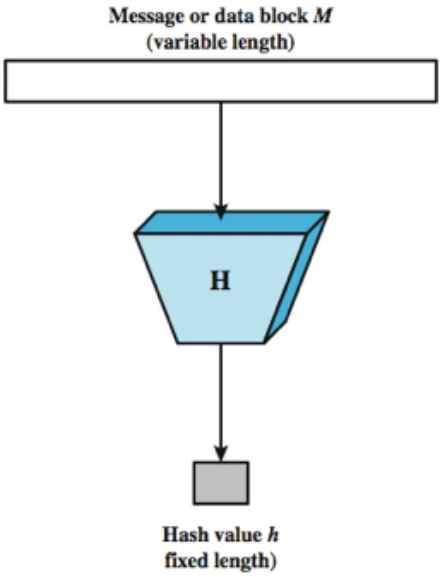
A. Kriptografi

Kriptografi adalah ilmu dan seni mengamankan komunikasi dengan mengubah teks atau data menjadi bentuk yang tidak dapat dimengerti (*ciphertext*) menggunakan teknik matematis dan algoritma tertentu. Tujuan utama kriptografi adalah menjaga *confidentiality*, *authentication*, data *integrity*, dan *nonrepudiation*.

Kriptografi melibatkan penggunaan kunci dan algoritma yang dirancang untuk mengenkripsi (mengacak) pesan asli ke dalam bentuk yang tidak dapat dibaca tanpa kunci yang sesuai. Proses ini disebut enkripsi. Pesan yang telah dienkripsi hanya dapat dibaca kembali menjadi pesan asli dengan menggunakan kunci yang tepat. Proses ini disebut dekripsi.

B. Hash

Hash adalah output unik, tetap, dan dengan panjang yang selalu sama yang dihasilkan oleh sebuah fungsi hash kriptografik. Fungsi hash kriptografik mengambil input data apa pun, seperti pesan teks, file, atau informasi lainnya, dan menghasilkan nilai hash yang merupakan representasi numerik dari data tersebut.



Gambar 1. Representasi Hash

Beberapa fungsi hash kriptografik yang umum digunakan adalah MD5 (sekarang dianggap tidak aman), SHA-1 (juga rentan terhadap serangan), SHA-256, dan SHA-3. Fungsi hash kriptografik yang aman harus menghasilkan nilai hash yang unik dan sangat tidak mungkin bagi dua input yang berbeda menghasilkan nilai hash yang sama (*hash collision*). Pada makalah ini, algoritma hash yang digunakan adalah SHA-256.

C. SHA-256

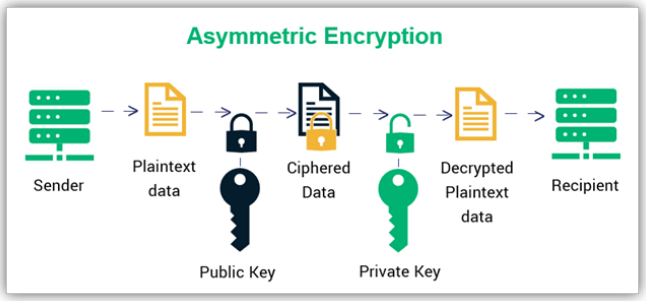
SHA-256 (Secure Hash Algorithm 256-bit) adalah salah satu fungsi hash kriptografik yang populer dan kuat. Fungsi hash ini mengambil input data apa pun dan menghasilkan output nilai hash yang terdiri dari 256 bit (32 byte). Berikut adalah cara kerja SHA-256:

1. Persiapan: Data yang akan di-*hash* dibagi menjadi blok-blok dengan ukuran yang tepat. Setiap blok biasanya berukuran 512 bit. Jika data tidak memenuhi panjang blok, *padding* (pengisian) dilakukan pada data untuk memastikan bahwa panjangnya sesuai.
2. Inisialisasi: SHA-256 menggunakan konstanta awal (*initial constants*) yang telah ditentukan sebelumnya. Nilai-nilai ini digunakan untuk menginisialisasi variabel-variabel internal yang akan digunakan selama proses *hash*.
3. Pengolahan Blok: Setiap blok data diolah secara berurutan. Pada setiap langkah, SHA-256 menggabungkan input data dengan nilai dari fungsi hash sebelumnya menggunakan operasi bitwise (operasi bit-per-bit) yang kompleks. Operasi ini melibatkan rotasi bit, operasi XOR (*exclusive* OR), dan operasi *bitwise* lainnya. Selain itu, SHA-256 juga menggunakan sejumlah besar tabel konstan dan fungsi non-linier yang disebut "*round functions*" untuk mengacak data.
4. Penggabungan Hash: Setelah semua blok data diolah, nilai *hash* dari setiap blok digabungkan menjadi satu nilai *hash* akhir. Proses ini melibatkan penggabungan (*concatenation*) dan manipulasi bit dari nilai *hash* setiap blok sesuai dengan aturan yang ditentukan.
5. Output: Hasil akhir dari proses *hash* adalah nilai *hash* SHA-256 yang terdiri dari 256 bit (32 byte). Nilai *hash* ini unik untuk setiap input data yang berbeda. Bahkan sedikit perubahan pada input data akan menghasilkan nilai *hash* yang sangat berbeda.

SHA-256 adalah fungsi *hash* kriptografik yang aman dan secara luas digunakan dalam berbagai aplikasi keamanan, termasuk pengesahan data, penyimpanan kata sandi, dan *mining cryptocurrency*. Keamanan SHA-256 didasarkan pada kerumitan operasi *bitwise* dan fungsi non-linier yang digunakan, serta panjang nilai *hash* yang besar (256 bit), yang membuatnya sangat sulit untuk dipecahkan atau diprediksi kembali menjadi input data aslinya.

D. Kriptografi Asimetris

Dalam kriptografi asimetris (atau kriptografi kunci publik), pasangan kunci digunakan, yaitu kunci publik dan kunci privat. Kunci publik dapat dibagikan secara bebas kepada siapa pun, sedangkan kunci privat harus dijaga dengan ketat oleh pemiliknya. Pesan dienkripsi dengan menggunakan kunci publik dan hanya dapat didekripsi dengan menggunakan kunci privat yang sesuai. Kriptografi asimetris sering digunakan untuk tujuan seperti pertukaran kunci rahasia, tanda tangan digital, dan pengesahan identitas. Algoritma yang umum digunakan dalam kriptografi asimetris adalah RSA, Diffie-Hellman, dan Elliptic Curve Cryptography (ECC).



Gambar 2. Alur Kriptografi Asimetrik

E. RSA

RSA (Rivest-Shamir-Adleman) adalah sebuah algoritma kriptografi asimetris yang menggunakan pasangan kunci, yaitu kunci publik (public key) dan kunci privat (private key).

Berikut adalah cara kerja RSA:

1. Generasi Kunci: Proses dimulai dengan generasi pasangan kunci RSA. Kunci publik terdiri dari modulus (n) dan eksponen publik (e), sementara kunci privat terdiri dari modulus (n) yang sama dan eksponen privat (d). Modulus (n) diperoleh dari perkalian dua bilangan prima besar yang acak (p dan q).

2. Enkripsi: Untuk mengenkripsi pesan, pengirim menggunakan kunci publik (n, e) penerima. Pesan

diubah menjadi representasi numerik dan dipangkatkan dengan eksponen publik (e) menggunakan operasi modulo n. Hasilnya adalah teks terenkripsi yang kemudian dikirim kepada penerima.

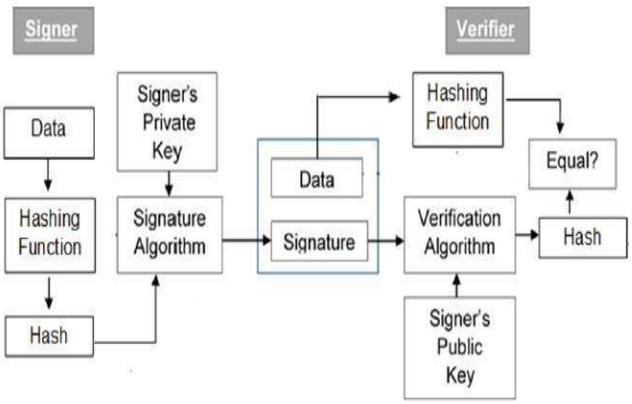
3. Dekripsi: Penerima menggunakan kunci privat (n, d) untuk mendekripsi pesan yang diterima. Teks terenkripsi dianggap sebagai pesan terenkripsi dan dipangkatkan dengan eksponen privat (d) menggunakan operasi modulo n. Hasilnya adalah pesan asli yang dapat dibaca.

4. Keamanan: Keamanan RSA didasarkan pada kesulitan memecahkan masalah faktorisasi, yaitu memperoleh faktor-faktor prima (p dan q) dari modulus (n). Meskipun enkripsi dan dekripsi dilakukan menggunakan eksponen yang besar, proses dekripsi hanya efisien dengan adanya faktor-faktor prima yang diketahui. Jika seseorang dapat memecahkan masalah faktorisasi dan menghitung eksponen privat (d), maka kunci privat dapat ditemukan dan keamanan RSA dapat dikompromikan.

5. Tanda Tangan Digital: RSA juga digunakan untuk tanda tangan digital. Pada proses ini, pesan di-hash terlebih dahulu menggunakan fungsi hash kriptografik, dan nilai hash tersebut dienkripsi dengan menggunakan kunci privat pengirim. Tanda tangan digital dapat diverifikasi oleh penerima dengan menggunakan kunci publik pengirim.

F. Digital Signature

Dalam kriptografi asimetris (atau kriptografi kunci publik), pasangan kunci digunakan, yaitu kunci publik dan kunci privat. Kunci publik dapat dibagikan secara bebas kepada siapa pun, sedangkan kunci privat harus dijaga dengan ketat oleh pemiliknya. Pesan dienkripsi dengan menggunakan kunci publik dan hanya dapat didekripsi dengan menggunakan kunci privat yang sesuai. Kriptografi asimetris sering digunakan untuk tujuan seperti pertukaran kunci rahasia, tanda tangan digital, dan pengesahan identitas. Algoritma yang umum digunakan dalam kriptografi asimetris adalah RSA, Diffie-Hellman, dan Elliptic Curve Cryptography (ECC). Pada makalah ini, algoritma kriptografi asimetris yang digunakan adalah RSA.



Gambar 3. Alur Proses Tanda Tangan Digital

Terdapat dua proses pada *digital signing*:

1. Proses Tanda Tangan (Signing Process)
   Untuk membuat digital signature, pemilik menggunakan kunci privatnya untuk mengenkripsi hash value pesan asli. Tanda tangan digital ini kemudian melekat pada pesan dan berfungsi sebagai bukti bahwa pesan tersebut berasal dari pemilik kunci privat yang sah. Hanya pemilik kunci privat yang dapat membuat tanda tangan digital yang valid untuk pesan tertentu.
2. Proses Verifikasi (Verification Process)

   Penerima pesan dapat memverifikasi keaslian dan integritas pesan dengan menggunakan kunci publik yang sesuai. Penerima menggunakan fungsi hash yang sama untuk menghasilkan nilai hash dari pesan yang diterima. Kemudian, dengan menggunakan kunci publik pengirim, tanda tangan digital di-dekripsi dan dibandingkan dengan nilai hash pesan yang dihasilkan. Jika kedua nilai cocok, maka pesan dianggap otentik dan tidak mengalami perubahan sejak dibuat.

III. IMPLEMENTASI

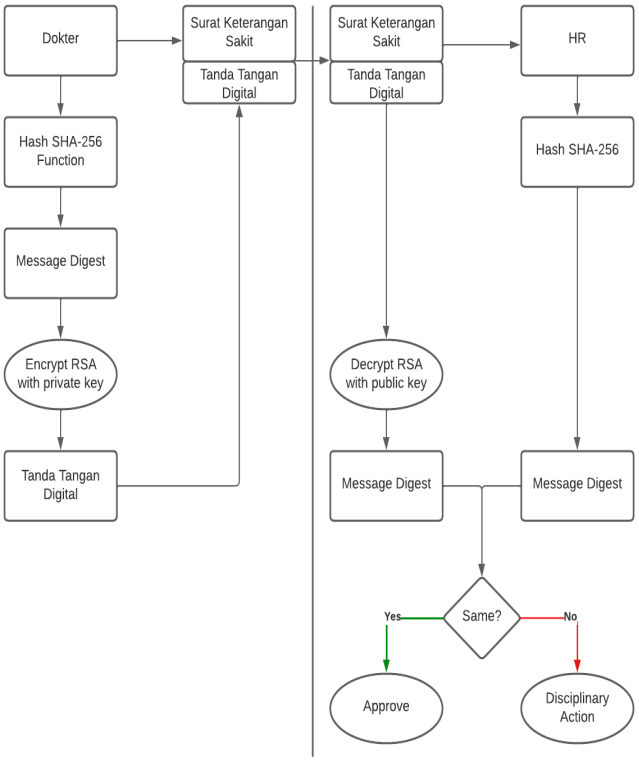
Untuk mengatasi masalah pemalsuan surat keterangan sakit, makalah ini akan menggunakan penerapan tanda tangan digital menggunakan kombinasi fungsi *hash* dan kriptografi kunci publik. Penggunaan tanda tangan digital dapat mengautentikasi, membuktikan integritas data, dan memberikan anti-penyangkalan.

Fungsi hash yang akan digunakan untuk mengatasi masalah ini adalah SHA-256 dengan pertimbangan bahwa karena SHA-256 tidak memiliki *vulnerabilities* yang diketahui yang membuatnya tidak aman dan belum "rusak" seperti beberapa algoritma *hashing* populer lainnya. Panjang pesan yang akan dicerna akan menjadi 256-bit setelah memperhitungkan kecepatan pembuatan.

Algoritma kunci publik yang akan digunakan adalah kunci publik RSA dengan pertimbangan keamanan dan popularitasnya. Panjang key yang akan digunakan adalah 256 byte dengan pertimbangan keamanan dan kecepatan prosesnya

A. Rancangan Solusi

Solusi yang dipilih terdiri dari dua pengguna, yaitu dokter dan HR (*Human Resource*). Arsitektur untuk solusi ini dapat dilihat pada gambar di bawah. Rancangan solusi dimulai dari dokter membuat surat keterangan sakit secara *online*, kemudian melakukan penandatanganan (*signing*) pada surat keterangan sakit, lalu mengirimkan surat tersebut kepada HR perusahaan pegawai tersebut bekerja. HR akan melakukan verifikasi surat tersebut dengan membandingkan hasil hash surat dan hasil dekripsi tanda tangan digital. Jika hasilnya sama, maka surat tersebut dapat dijamin identitas dan keasliannya. Sedangkan jika hasilnya tidak sama, maka surat tersebut tidak dapat dijamin identitas penulis ataupun keaslian isi surat.



Gambar 4. Arsitektur Solusi

B. Implementasi Solusi

Implementasi solusi dilakukan dengan membuat beberapa API (*application programming interface*) untuk mendukung seluruh kebutuhan proses *digital signature* ini. API dibuat dengan menggunakan desain arsitektur REST (*representational state transfer*) dan semua request payload dan response payload yang diterima hanya JSON (*JavaScript Object Notation*).

Ada tiga API yang akan dibuat, yaitu generateKey, generateSignature, verifySignature, sendEmail, dan getDoctorPublicKey

i. generateKey (POST)

API ini berfungsi untuk men-generate public key dan private key pengguna dengan menggunakan algoritma RSA.

Berikut contoh request payload dan response payload pada API generateKey

Request Payload
{}
Response Payload

{
“privateKey”: String,
“publicKey”: String
}

ii. generateSignature (POST)

API ini berfungsi untuk melakukan penandatanganan (*signing*) pada *plaintext*. Pertama-tama *plaintext* akan di-*hash* dengan menggunakan algoritma SHA-256. Kemudian hasil *message digest* tersebut akan dienkripsi dengan algoritma RSA menggunakan *private key* dokter sehingga menghasilkan tanda tangan digital.

Berikut contoh request payload dan response payload pada API generateSignature

Request Payload

{
“plaintext”: String
}

Response Payload

{
“signature”: String
}

iii. verifySignature (PUT)

API ini berfungsi untuk memverifikasi identitas pengirim dan menjamin keaslian dari surat keterangan kerja yang dikirim. Pertama-tama, HR akan mengirimkan *public key* dokter yang tertera pada surat dan surat keterangan sakit ke API ini. Kemudian API akan memvalidasi surat tersebut dengan membandingkan hasil *message digest* dari *hash plaintext* dan dekripsi tanda tangan digital.

Berikut contoh request payload dan response payload pada API verifySignature

Request Payload

{
“plaintext”: String,
“publicKey”: String
}

Response Payload

{
“isValid”: Boolean
}

iv. sendEmail (POST)

API ini berfungsi untuk membantu dokter mengirimkan surat keterangan sakit yang telah ditandatangani kepada HR tempat pegawai bekerja.

Berikut contoh request payload dan response payload pada API sendEmail

Request Payload

{
“plaintext”: String,
“signature”: String
}

Response Payload

{
“success”: Boolean
}

v. getDoctorPublicKey (GET)

API ini berfungsi untuk membantu HR mendapatkan *public key* dokter yang menandatangani surat keterangan sakit. Hasil dari API ini kemudian dipakai untuk memanggil API verifySignature.

Berikut contoh request payload dan response payload pada API getDoctorPublicKey

Path Param

{
“doctorName”: String
}

Response Payload

{
“publicKey”: String
}

IV. HASIL DAN ANALISA

Bagian ini menampilkan contoh pengujian keefektifan tanda tangan digital dan analisa berdasarkan hasil yang didapat.

A. Pengujian

Pengujian dilakukan dengan menggunakan skenario berikut:

1. Dokter membuat *private key* dan *public key* dengan memanggil API generateKey.

API generateKey

{
“privateKey”:
”MIICdwIBADANBgkqhkiG9w0BAQEFAASCAmEwggJdAgEAAoGBAJAv9sT510kGRhsO1uX1oJrdRbLsh3+JEl7k4FSyOWZr8ehHW2/jv4t+AX6oBuiyNczl/4TrbhSqlqoM0+Rq3iInz1MrGqIR2f83PQsgVqHjZEGlZhbnSkSiHEyugGpzzRVwW2cMhA3xDPU0waLML1UK8G+Jv3yosSRVFOG/NMIrAgMBAAECgYB27ikktlO+Vx9yaB4X1i/gLU21VOY66yws+6qSEePIuJbzmhAwzXGMNKY0+5GfVIeN99R4pdzJzR+zKhuJGCEaWvYcAR/8GN2r9Ceh79JRUG5DfuuzzKn/uWSpbl0BTKUSjWD4mhn+m22/QwHUUr49l/JFLnnQzJ1NQWpNqETTqQJBANIfsExQAy7xu0oQU+DitFd9JFsR8SyVYDrFDPNbVFubDc8jkiR0Kh/loLrb22IgGutzVch7t2DahMNPbv40Xj0CQQCvqu8MXkocz/K6buNNZMcTE7rtmOIdiuEtQU8gdsjAhVhDWmvvBoiRFQSxvHyI5pQdjDPIpdPdWtAM8Meq9FCHAkEAvpNflevC3TTmWkVqnHzfyZRQwgS5YzVqnfuNwouiKco7yYj9kwlqKhsDw44+9aCk+oXc65Bs+Z9Y2c4E/cLQQJBAKP3BYdi3wRDqMkRRYJwpUwOJ3UsNSnj/kfSZLvlg34HOJotZP9G9uOMXcRULV1O80MdIRyQg89huz3I5GWSAzMCQGY2s12Caxq5hDhOjpE7dvZqFcXpTGqyvLb+t3x3bBJ1yVxSbvR4jLkuSYZaErkIDGW49txnFvEhiQjEarhncYc=”,
“publicKey”:
“MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQCQL/bE+ddJBkYbDtbl9aCa3UWy7Id/iRJe5OBUsjlma/HoR1tv47+LfgF+qAbosjXM5f+E624UqpaqDNPkat4iJ89TKxqiEdn/Nz0LIFah42RBpWYW50pEohxMroBqc80VcFtnDIQN8Qz1NMGizC9VCvBvib98qLEkVRThvzTCKwIDAQAB”
}

2. Dokter membuat surat keterangan kerja dan menandatanganinya dengan memanggil API generateSignature.

API generateSignature
Request Payload

{
“plaintext”:
“RUMAH SAKIT UMUM BANJARNEGARA
Alamat : Jl. Pertempuran No. 123, Banjarnegara
 	
SURAT KETERANGAN DOKTER
 	
Yang bertanda tangan di bawah ini menerangkan bahwa:
 	
Nama : Wijaya Kusuma
Jenis Kelamin : Laki-Laki
Umur : 23 Tahun
Pekerjaan : Karyawan Swasta
Alamat : Jalan Perjuangan No. 122, Banjarnegara
 	
Berdasarkan hasil pemeriksaan yang telah dilakukan, pasien tersebut dalam keadaan sakit, sehingga perlu beristirahat selama 3 hari, dari tanggal 12 April 2021 s/d 15 April 2021.
 	
Diagnosa : Penyakit Demam Berdarah.
 	
Demikian surat keterangan ini diberikan, untuk diketahui dan dipergunakan sebagaimana mestinya.
 	
Banjarnegara, 12 April 2021
Dokter Pemeriksa,
 	
ttd
dr. Ahmad Subari”
}

Response Payload

{
“signature”:
“U3o/I4MRJf6kPq76IWh8jtYjneLrJZKOLJDSlIeb0Pcu/2vHZSo8flg5nYgOkH4Ox1p3Y1mFfAmcZbNwPtLSWrD1+G77zk6Y+y+wXnciNNhfGlOvMblAeBcbCDxjRwHn/zDjcjqVbzzpAGpE1w/sP8mLmyNnXtpaV1ZBlORKXHc=”
}

3. Dokter mengirim surat keterangan sakit beserta tanda tangan digital kepada HR.

API sendEmail
Request Payload
{ “plaintext”: “RUMAH SAKIT UMUM BANJARNEGARA Alamat : Jl. Pertempuran No. 123, Banjarnegara SURAT KETERANGAN DOKTER Yang bertanda tangan di bawah ini menerangkan bahwa: Nama : Wijaya Kusuma Jenis Kelamin : Laki-Laki Umur : 23 Tahun Pekerjaan : Karyawan Swasta Alamat : Jalan Perjuangan No. 122, Banjarnegara Berdasarkan hasil pemeriksaan yang telah dilakukan, pasien tersebut dalam keadaan sakit, sehingga perlu beristirahat selama 3 hari, dari tanggal 12 April 2021 s/d 15 April 2021. Diagnosa : Penyakit Demam Berdarah. Demikian surat keterangan ini diberikan, untuk diketahui dan dipergunakan sebagaimana mestinya. Banjarnegara, 12 April 2021 Dokter Pemeriksa,   ttd dr. Ahmad Subari”, “signature”: “U3o/I4MRJf6kPq76IWh8jtYjneLrJZKOLJDSlIeb0Pcu/2vHZSo8flg5nYgOkH4O

x1p3Y1mFfAmcZbNwPtLSWrD1+G77z k6Y+y+wXnciNNhfGlOvMblAeBcbCDx jRwHn/zDjcjqVbzzpAGpE1w/sP8mLmy NnXtpaV1ZBlORKXHc=” }
Response Payload

{
“success”: true
}

4. Surat yang dikirim akan ada dua, yaitu surat yang asli dan yang sudah dimodifikasi. Surat yang sudah dimodifikasi memiliki perbedaan pada durasi istirahat yang tadinya 3 hari menjadi 10 hari. Kemudian akan diuji juga untuk kasus penggunaan *public key* palsu

Surat yang sudah dimodifikasi
“RUMAH SAKIT UMUM BANJARNEGARA Alamat : Jl. Pertempuran No. 123, Banjarnegara SURAT KETERANGAN DOKTER Yang bertanda tangan di bawah ini menerangkan bahwa: Nama : Wijaya Kusuma Jenis Kelamin : Laki-Laki Umur : 23 Tahun Pekerjaan : Karyawan Swasta Alamat : Jalan Perjuangan No. 122, Banjarnegara Berdasarkan hasil pemeriksaan yang telah dilakukan, pasien tersebut dalam keadaan sakit, sehingga perlu beristirahat selama 10 hari, dari tanggal 12 April 2021 s/d 15 April 2021. Diagnosa : Penyakit Demam Berdarah. Demikian surat keterangan ini diberikan, untuk diketahui dan dipergunakan sebagaimana mestinya. Banjarnegara, 12 April 2021 Dokter Pemeriksa, ttd dr. Ahmad Subari”

5. HR menerima surat keterangan sakit kemudian melakukan verifikasi dengan cara memanggil API getDoctorPublicKey untuk mendapatkan *public key* dokter, lalu memanggil API verifySignature untuk memverifikasi tanda tangan digital.

API getDoctorPublicKey (dr. Ahmad Subari)
Response Payload

{
“publicKey”:
“MIGfMA0GCSqGSIb3DQEBAQUAA4
GNADCBiQKBgQCQL/bE+ddJBkYbDt
bl9aCa3UWy7Id/iRJe5OBUsjlma/HoR1t
v47+LfgF+qAbosjXM5f+E624UqpaqDN
Pkat4iJ89TKxqiEdn/Nz0LIFah42RBpW
YW50pEohxMroBqc80VcFtnDIQN8Qz1
NMGizC9VCvBvib98qLEkVRThvzTCK
wIDAQAB”
}

Untuk mempersingkat penulisan, request payload akan diisi menggunakan “surat asli” dan “surat palsu”. Surat yang sudah dimodifikasi memiliki perbedaan pada durasi istirahat yang tadinya 3 hari menjadi 10 hari.

API verifySignature
Request Payload (surat asli)

{
“plaintext”: “surat asli,
“publicKey”:
“MIGfMA0GCSqGSIb3DQEBAQUAA4
GNADCBiQKBgQCQL/bE+ddJBkYbDt
bl9aCa3UWy7Id/iRJe5OBUsjlma/HoR1t
v47+LfgF+qAbosjXM5f+E624UqpaqDN
Pkat4iJ89TKxqiEdn/Nz0LIFah42RBpW
YW50pEohxMroBqc80VcFtnDIQN8Qz1
NMGizC9VCvBvib98qLEkVRThvzTCK
wIDAQAB”
}

Response Payload
{

“isValid”: true }
Request Payload (surat palsu)

{
“plaintext”: “surat asli,
“publicKey”:
“MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQCQL/bE+ddJBkYbDtbl9aCa3UWy7Id/iRJe5OBUsjlma/HoR1tv47+LfgF+qAbosjXM5f+E624UqpaqDNPkat4iJ89TKxqiEdn/Nz0LIFah42RBpWYW50pEohxMroBqc80VcFtnDIQN8Qz1NMGizC9VCvBvib98qLEkVRThvzTCKwIDAQAB”
}

Response Payload

{
“isValid”: false
}

Request Payload (surat asli dan *public key* palsu)

{
“plaintext”: “surat asli,
“publicKey”:
“ZIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQCQL/bE+ddJBkYbDtbl9aCa3UWy7Id/iRJe5OBUsjlma/HoR1tv47+LfgF+qAbosjXM5f+E624UqpaqDNPkat4iJ89TKxqiEdn/Nz0LIFah42RBpWYW50pEohxMroBqc80VcFtnDIQN8Qz1NMGizC9VCvBvib98qLEkVRThvzTCKwIDAQAZ”
}

Response Payload

{
“isValid”: false
}

6. Jika API verifySignature mengembalikan *true,* maka HR akan meng-*approve* surat keterangan sakit. Sebaliknya, jika hasil balikkan adalah *false*, maka HR tidak akan meng-*approve* dan akan memberikan hukuman/sanksi kepada pegawai.
7. Berikut perbedaan *hash* antara surat keterangan sakit asli dan palsu

Message digest surat asli
36843D58EB3E095236B26A45E75A17AFDAF46AFAEF4ED55880AD26CEA8063D48
Message digest surat palsu
35C26288FE09362F00B1ABACDB8F73EF098D0FB8A525F176A978115BDD7F69C7

V. KESIMPULAN

Berdasarkan hasil pengujian, dapat disimpulkan bahwa implementasi *digital signature* pada surat keterangan sakit dapat menjamin keaslian data dan juga dapat menjamin bahwa pengirim data merupakan pengirim yang seharusnya. Secara aspek keamanan, dengan menggunakan algoritma RSA untuk proses enkripsi, kesulitan berada pada menemukan faktor prima dari bilangan bulat yang besar.

Dapat disimpulkan bahwa penerapan *digital signature* dalam upaya mencegah pemalsuan surat keterangan sakit merupakan hal yang perlu untuk dilakukan untuk menjamin identitas pengirim dan menjaga keaslian surat keterangan kerja.

UCAPAN TERIMA KASIH

Penulis ingin mengucapkan puji syukur kepada Tuhan Yang Maha Esa, karena telah diberikan kemudahan dalam menyelesaikan karya tulis ini. Selain itu, penulis juga ingin mengucapkan terima kasih kepada Bapak Dr. Ir. Rinaldi Munir, M.T. yang telah memberikan kesempatan dan wawasan mengenai ilmu kriptografi dengan jelas sehingga penulis dapat menyelesaikan penulisan karya tulis ini.

REFERENCES

[1] Gueron, S., Johnson, S., & Walker, J. (2011, April). SHA-512/256. In 2011 Eighth International Conference on Information Technology: New Generations (pp. 354-358). IEEE.

[2] Munir, Rinaldi. “Fungsi Hash”, https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2022-2023/25-Fungsi-hash-2023.pdf (terakhir diakses pada 10.49 WIB, 22 Mei 2023)

[3] Munir, Rinaldi. “Kriptografi Kunci Publik”, https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2022-2023/18-Kriptografi-Kunci-Publik-2023.pdf (terakhir diakses pada 11.28 WIB, 22 Mei 2023)

[4] Munir, Rinaldi. “Tanda Tangan Digital”, https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2022-2023/28-Tanda-tangan-digital-2023.pdf (terakhir diakses pada 15.25 WIB, 22 Mei 2023)

[5] Munir, Rinaldi. “Algoritma RSA”, https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2022-2023/19-Algoritma-RSA-2023.pdf (terakhir diakses pada 14.39 WIB, 22 Mei 2023)