Menjaga Integritas Data dan Mengidentifikasi Pengguna melalui Kombinasi Algoritma RSA dan SHA-256

Marvel Pangondian - 135220751

Program Studi Teknik Informatika

Sekolah Teknik Elektro dan Informatika

Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia

113522075@itb.ac.id

*Abstract*—Kriptografi adalah cabang utama dalam ilmu keamanan komputer yang melibatkan teknik-teknik matematis dan komputasi untuk mengamankan komunikasi dan informasi. Tujuan kriptografi adalah teknik-teknik matematis dan komputasi untuk mengamankan komunikasi dan informasi. Salah satu algoritma yang digunakan untuk enkripsi dan deskripsi teks adalah algoritma RSA, tetapi hanya menggunakan algoritma tersebut dapat menyebabkan berbagai masalah. Salah satunya adalah kunci privat/publik yang tidak sesuai ketika mencoba enkripsi/dekripsi teks, keamanan enkripsi/dekripsi yang tidak terlalu bervariasi, serta integritas data yang tidak dapat dideteksi. Dalam makalah ini, akan dibahas penggunaan algoritma *hashing* yakni algoritma SHA-256 dalam meningkatkan keamanan data dan identifikasi pengguna, serta memeriksa integritas kunci publik dan privat yang digunakan.

*Keywords*—Kriptografi, Algoritma RSA, Algoritma SHA-256

# I. PENDAHULUAN

Kriptografi ilmu yang menggunakan matematika untuk mengenkripsi dan mendekripsi data[1]. Sejarah kriptografi bermula ribuan tahun yang lalu, dan penciptaannya didorong oleh kebutuhan akan komunikasi yang aman dan perlindungan informasi yang sensitif. Salah satu metode enkripsi dan dekripsi yang terkenal pada peradaban kuno adalah Sandi Caesar, dinamai oleh penemunya Julius Caesar. Julius menciptakan Sandi Caesar karena dia tidak percaya kurir yang mengirimkan surat tersebut.

Kriptografi diciptakan karena kebutuhan manusia untuk menjaga kerahasiaan data yang mereka miliki. Salah satu algoritma yang efektif dalam melakukan hal ini adalah algoritma RSA(*Rivest-Shamir-Adleman*). Algoritma ini akan mengenkripsi dan mendekripsi teks menggunakan dua kunci berbeda yakni kunci privat dan kunci publik. Kunci publik merupakan kunci untuk melakukan enkripsi kepada sebuah teks, kunci ini bersifat publik dan dapat disebarkan. Kunci privat adalah kunci untuk dekripsi, kunci ini tidak dapat disebarkan. Kedua kunci tersebut akan mengenkripsi dan mendekripsi sebuah teks menggunakan teori bilangan yang akan dibahas lebih dalam pada bagian dasar teori

Algoritma RSA merupakan algoritma yang kuat, tetapi memiliki beberapa kelemahan. Apa yang terjadi jika pengguna mencoba mengakses *plain text* dengan kunci publik/privat yang tidak sesuai ? Atau apa yang terjadi jika enkripsi data diubah, bagaimana pengguna mengetahui hasil enkripsinya telah diubah oleh pihak ketiga ? Untuk menjaga keamanan dan integritas data, diperlukan konsep dan algoritma lain. Konsep yang akan digunakan untuk menangani adalah konsep *signature* .

*Signature* adalah konsep/cara untuk memverifikasi integritas data, pengguna, dan kunci. Dengan konsep *signature*, dapat dibuat sebuah program untuk mendeteksi integritas sebuah data dan mendeteksi apakah data tersebut sudah diubah sebelumnya atau tidak. *Signature* juga digunakan untuk memverifikasi identitas pengguna

Salah satu cara untuk menerapkan konsep *signature* adalah penerapan *signature* kepada sebuah *encrypted text*. Ketika sebuah teks dienkripsi, hasil enkripsi tersebut dapat diubah oleh pengguna lainnya. Untuk mengetahui integritas dari *encrypted text*, dapat digunakan konsep *signature* yang menyimpan *hash value text* sebelumnya, sehingga *text* dapat dideteksi integeritasnya menggunakan *signature* yang sudah disimpan. Jika hasil *hash text* berbeda dengna *signature*, maka *text* tersebut sudah diubah sebelumnya

Salah satu algoritma yang dapat digunakan untuk konsep *signature* ini adalah algoritma SHA-256 yang dapat mengubah sebuah string menjadi *hash value unique* sebesar 256 bits. Dengan algoritma ini, dapat dibentuknya program untuk mengecek integritas pengguna dan data yang ingin dideskripsi

# II. TEORI DASAR

1. *Bilangan Bulat*

Bilangan bulat adalah angka-angka yang tidak memiliki bagian desimal atau pecahan. Bilangan ini dapat berupa angka positif, negatif, atau nol. Bilangan bulat termasuk dalam kategori bilangan cacah (non-negatif) dan bilangan negatif. Bilangan bulat dapat berupa bilangan positif (1, 2, 3, ...), bilangan negatif (-1, -2, -3, …), dan nol (0).

1. *Bilangan Riil*

Bilangan riil adalah bilangan yang mencakup seluruh bilangan yang mencakup seluruh bilangan rasional dan irasional. Bilangan riil terdiri atas bilangan bulat (…,-3, -2, -1, 0, 1, 2, 3,…), bilangan rasional (1/2, -5/4, dll), dan bilangan irasional (√2, π, dll).

1. *Bilangan Prima*

Bilangan prima adalah bilangan bulat yang lebih dari satu dan hanya dapat dibagi oleh 1 dan bilangan tersebut sendiri. Contoh bilangan bulat adalah 7 karena 7 hanya dapat habis dibagi oleh 1 dan 7 saja. Bilangan selain bilangan prima disebut sebagai bilangan komposit

1. *Sifat Pembagian*

Sebuah bilangan bulat *a* habis membagi sebuah bilangan bulat *b* jika dan hanya jika terdapat sebuah bilangan *c* yang merupakan bilangan bulat, sedemikian hingga *ac = b* . Jika tidak ada *c*, maka *a* tidak habis membagi *b.* Berikut adalah notasi sifat pembagian :

1. *Teorema Euclidean*

Teorema Eculidean menyatakan bahwa untuk setiap dua integer, *m* (pembilang) dan *n* (penyebut), di mana n tidak sama dengan nol, terdapat dua integer *q* (hasil bagi)dan *r* (hasil sisah) sedemikian hingga,

1. *Pembagi Bersama Terbesar (gcd)*

Pembagi bersama terbesar (gcd) merupakan bilangan bulat terbesar *c* sedemikian hingga dapat habis membagi dua bilangan bulat *a* dan *b*.

1. *Algoritma Euclidean*

Algoritma Euclidean adalah algoritma untuk mencari gcd dari dua bilangan bulat. Algoritma ini menyatakan bahwa untuk setiap bilangan bulat non negatif *m* dan *n* di mana *m*  ≥ *n*, maka jika *m = r0*  dan *n = r1*berlaku :

Dengan aturan di atas, maka gcd *m* dan *n* dapat dicari menggunakan Teorema 2,

1. Relatif Prima

Dua bilangan, *m* dan *n*, dikatakan relatif prima jika dan hanya jika,

Relatif prima akan digunakan dalam implementasi enkripsi menggunakan algoritma RSA

1. *Aritmatika Modulo*

Untuk bilangan bulat *m* dan *n* di mana berlaku kombinasi linier sebagai berikut :

maka dapat ditulis ulang sebagai berikut :

notasi di atas dapat diinterpretasikan sebagai “*a* dibagi dengan *m* menghasilkan sisa *r* ”

1. *Kongruen dan Sifat – Sifatnya*

Untuk bilangan bulat *a* dan *b* dikatakan kongruen jika dan hanya jika terdapat modulus *c* yang membagi  *a*  dan *b* dan menghasilkan sisa *r* yang sama

Jika berlaku di atas, maka *a* kongruen dengan *b* dalam modulus *c*. Bilangan *a* dapat dikatakan kongruen dengan b dalam jika dan hanya jika (*a*-*b*) juga habis dibagi dengan *c*.

melalui kongruen, didapat beberapa sifat yakni,

1. Jika *a* ≡ *b* (mod *c*), dan *k* adalah sembarang bilangan bulat, maka berlaku
2. (*a + k*) ≡ (*b + k* ) (mod *c*)
3. *ak* ≡ *bk* (mod *c*)
4. *ak* ≡ *bk* (mod *c*) , *k* bilangan bulat tak-negatif
5. Jika *a* ≡ *b* (mod *c*) dan *k* ≡ *m* (mod *c*), maka
6. (*a + k*) ≡ (*b + m*) (mod *c*)
7. *ak* ≡ *bm* (mod *c*)
8. *Balikan Modulo (Modulo Inverse)*

Untuk mencari inverse dari *a* (mod b), maka harus berlaku gcd(*a, b*) = 1, jika tidak maka *a* (mod b) tidak memiliki inverse. Untuk gcd(*a, b*) = 1, maka berlaku :

melalu implikasi, didapat :

Perhatikan bahwa

maka,

Kekongruenan terakhir ini berarti m merupakan *inverse* dari a (mod b). Cara lain mencari inverse dari *a* (mod *b*) adalah dengan cara berikut :

*x* yang merupakan *inverse* dari *a* (mod *b*) dapat diperoleh dengan mencari nilai *k* sedemikian hingga hasil *x* di atas merupakan bilangan bulat.

1. *Kekongruenan Linier dan Sistem Kekongruenan linier*

Kekongruenan linier secara umum memiliki bentuk sebagai berikut:

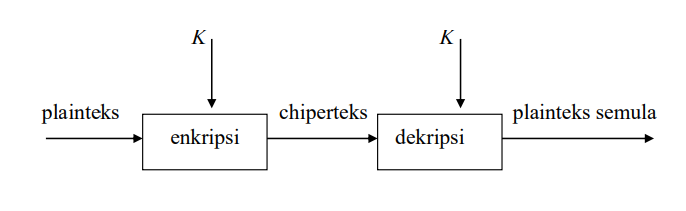
solusi ­­*x* dapat dicari dengan mengubah bentuk kongruen di atas menjadi

dengan memilih nilai k sedemikian hingga hasil *x* adalah bilangan bulat. Nilai *k* yang menghasilkan nilai *x* merupakan bilangan bulat.

Sistem kekongruenan linier berarti sistem kongruen dengan lebih dari satu kongruen, sebagai contoh :

Sistem ini dikenal sebagai sistem kekongruenan linier karena setiap persamaan adalah kongruen linier. Untuk memecahkan sistem kekongruenan linier, kita perlu mencari *x* yang memenuhi semua kongruen pada sistem kekongruenen linier. Dalam banyak kasus, sistem kekongruenan linier dapat dipecahkan menggunakan teorema Chinese Remainder Theorem (CRT)

1. *Kriptografi*

**Kriptografi adalah seni menyembunyikan data menggunakan ilmu matematika. Kriptografi berasal dari bahasa Yunani "kryptós" yang berarti tersembunyi/rahasia, serta "gráphein" yang berarti menulis. Ketika kedua kata tersebut digabung, didapat kata kriptografi yang berarti studi mempelajari teknik – Teknik untuk menyembunyikan data agar terlindungi dari pihak ketiga. Kriptografi memiliki tujuan yakni menjaga pesan/data agar tidak dapat dibaca oleh pihak ketiga. Kriptografi terdiri atas dua komponen yakni pesan (*plaintext*) yang berisi data/informasi yang ingin disembunyikan, serta *cipherteks* (*ciphertext*) yang berisi hasil setelah pesan disembunyikan. Terdapat dua proses utama dalam kriptografi yakni proses enkripsi dan dekripsi. Proses enkripsi adalah proses mengubah sebuah data, biasanya sebuah pesan, menjadi data yang tidak dapat dibaca oleh orang lain secara umum. Hasil proses enkripsi adalah text/data yang bersifat *unreadable/indecipherable.* Proses berikutnya dalam kriptiografi adalah proses dekripsi. Proses ini melibatkan *ciphertext* dan mengubah kembali menjadi *plain text*. Proses ini hanya dapat dilakukan oleh pengguna yang mengetahui jenis algoritma yang digunakan untuk mengenkripsi *plain text* serta mengetahui kunci – kunci yang digunakan.

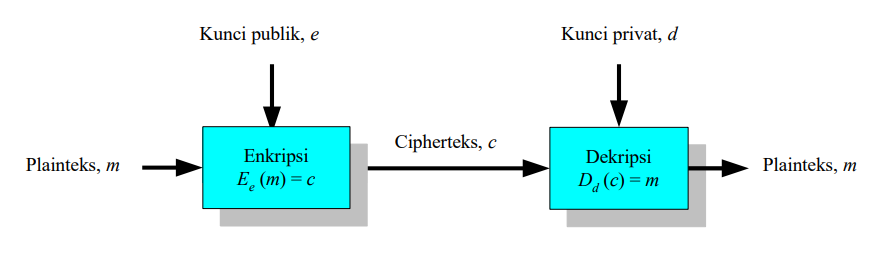
*Gambar 2.1 Alur Kriptografi*

*Sumber :* [*https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Teori-Bilangan-2020-Bagian3.pdf*](https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Teori-Bilangan-2020-Bagian3.pdf)

Kripografi terdiri atas dua tipe, yakni tipe pertama *Symmetric-Key Cryptography* dan tipe kedua *Public-Key Cryptography (Asymmetric Cryptography)*[2]*.* Untuk tipe pertama, kriptografi menggunakan satu kunci saja untuk enkripsi dan dekripsi, sedangkan tipe kedua menggunakan dua kunci yakni kunci publik untuk enkripsi, dan kunci privat untuk dekripsi. Di antara kedua tipe, tipe kedua memiliki keamanan yang lebih ketat dibandingkan tipe pertama. Salah satu algoritma implementasi kriptografi tipe *Asymmetric Cryptography* adalah algoritma RSA (*Rivest-Shamir-Adleman*).

1. *Algoritma RSA (Rivest-Shamir-Adleman)*

Algoritma RSA dinamakan berdasarkan penemunya yaitu Ron Rivest, Adi Shamir, dan Leonard Adleman. Algoritma ini merupakan penerapan jenis kriptografi yakni *Asymmetric Cryptography* yang menggunakan dua kunci[8]. Dua kunci tersebut adalah kunci publik untuk enkripsi dan kunci privat untuk dekrispi.



*Gambar 2.2 Alur Algoritma RSA*

*Sumber :* [*https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Teori-Bilangan-2020-Bagian3.pdf*](https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Teori-Bilangan-2020-Bagian3.pdf)

alur pembuatan kunci algoritma RSA adalah sebagai berikut :

1. Pertama, dipilihnya dua angka, *p* dan *q* yang merupakan angka prima yang *random* dan berbeda.
2. Menhitung *n* = *pq* yang merupakan modulus algoritma RSA (tidak perlu dirahasiakan)
3. Menghitung *ϕ*(*n*) = (*p - 1*) (*q - 1*)(perlu dirahasiakan)
4. Mencari kunci publik (e), kunci tersebut memiliki nilai 1 < *e* < *ϕ*(*n*) dan relatif prima dengan *n* dan *ϕ*(*n*). Kunci ini akan digunakan untuk enkripsi dan dapat disebarkan
5. Mencari kunci privat (*d*) yang memenuhi syarat *de ≡* 1 (mod *ϕ*(*n*)). Kunci privat (*d*) akan digunakan untuk mendekripsi *ciphertext*

alur enkripsi dan dekripsi dengan algoritma RSA:

1. Pertama, akan dibuat kunci publik (*e*) dan kunci privat (*d*). Pembentukan kunci biasanya dilakukan dengan memilih *p* dan *q* yang random
2. Untuk enkripsi, mengubah tiap character pada *plain text* menjadi unicode, setelah itu mengubah tiap value character menjadi value lain dengan rumus berkut :

Setelah itu hasilnya disimpan menjadi *cipher text*

1. Untuk dekripsi, pertama menerima *cipher text* dan mengubah setiap character value pada *cipher text* kembali menjadi value semula dengan rumus berikut :

hasilnya kemudian disimpan menjadi *plain text* yang semula

1. *Fungsi Hash*

Fungsi hash adalah algoritma matematis yang mengubah input data (pesan) menjadi nilai hash dengan panjang tetap digunakan untuk beragam keperluan, termasuk memastikan keutuhan data, memvalidasi kata sandi, dan menyimpan data dalam bentuk struktur data hash. Salah satu fungsi hash yang efektif adalah SHA-256 (*Secure Hash Algorithm 256-bit*)

1. *Fungsi Hash SHA-256 (Secure Hash Algorithm 256-bit)*

SHA-256, yang merupakan singkatan dari Secure Hash Algorithm 256-bit, adalah suatu algoritma hash kriptografis yang termasuk dalam serangkaian algoritma Secure Hash Algorithm (SHA) yang dirancang oleh National Institute of Standards and Technology (NIST) Amerika Serikat. Proses SHA-256 menghasilkan nilai hash yang memiliki panjang tetap sebesar 256 bit, setara dengan 32 byte.

1. *Digital Signature*

*Digital signature* adalah jenis tanda tangan elektronik khusus yang melibatkan penggunaan teknik kriptografi. *Digital signature* menyediakan cara yang aman dan tahan terhadap penyusup untuk mengonfirmasi asal dan otentikasi pesan atau dokumen digital. *Signature* ini bergantung pada kriptografi kunci publik, di mana kunci pribadi digunakan untuk menghasilkan tanda tangan, dan kunci publik yang sesuai digunakan untuk memverifikasinya.

# III. ANALISIS MASALAH

## A. Implementasi Konsep Digital Signature

Salah satu cara untuk memvalidasi integritas data adalah melalui *Digital signature*. *Digital signature* digunakan untuk memastikan data valid/tidak diubah oleh pihak luar. Implementasi *Digital signature* akan menggunakan fungsi hash SHA-256 untuk menghash data berupa *text* menjadi nilai hash yang memiliki panjang tetap sebesar 256 bit, setara dengan 32 byte. Alasan penulis memilih fungsi hash SHA-256 adalah agar memastikan besar *signature* selalu konsisten serta algoritma SHA-256 telah terbukti menghasilkan probabilitas *collision* yang rendah[3] . *Signature* akan dimasukan pada hasil enkripsi (*cipher text*) dan akan diambil kembali ketika ingin memvalidasi integritas data. *Signature* juga akan digunakan untuk mengecek integritas *private key* dan *public key* yang digunakan pengguna untuk enkripsi atau dekripsi.

## B. Validasi Integritas Data

Data akan divalidasi melalui *signature* yang ada dalam data tersebut. Pada proses enkripsi, *text* akan diubah menjadi *hash value* 256 bit menggunakan algoritma SHA-256. *Hash value* tersebut kemudian dijadikan *signature* dengan menggunakan *private key (d)*. Caranya adalah dengan mengenkripsi hash value tersebut menggunakan *d* dan *modulus (n)* . *Signature* kemudian dimasukan ke dalam *cipher text*. Ketika ingin validasi integritas data, maka *signature* dalam *cipher text* perlu dipsah dari *cipher text* terlebih dahulu. *Cipher text* perlu di dekripsi, setelah itu hasil dekripsi di hash menggunakan algoritma SHA-256. Signature awal text di ubah menjadi *value* semulanya (*value* sebelum di enkripsi menjadi signature) dengan menggunakan kunci publik (*e*) dan modulus (*n*) . Hasil *hash* semula kemudian di bandingkan dengan hasil hash dekripsi. Jika sama, maka data sama dengan data orijinal, jika beda maka data telah diubah/diedit oleh pihak ketiga (*data is compromised*).

## B. Validasi Pengguna

Ketika mencoba untuk mendekripsi/mengenkripsi data, perlu dipastikan integritas *private key* dan *public key* yang digunakan. Hal tersebut dapat dilakukan dengan melakukan validasi pengguna. Setiap pengguna akan memiliki *username* dan *password* pilihan mereka sendiri, kemudian program akan menggunakan *private key* , *username*, dan *password* untuk membuat *signature* unik yang akan digunakan untuk menvalidasi *publik key* dan *private key ,* *signature* tersebut kemudian disimpan di *database*. Ketika pengguna ingin mengenkripsi/mendekrispsi, program akan meminta *username, password, public key,* dan *private key* pengguna, kemudian program akan memvalidasi integritas *public key,* dan *private key* berdasarkan *signature* yang sudah disimpan di database.

## C. Implementasi Algoritma RSA

Algoritma enkripsi dan dekripsi yang akan digunakan adalah algoritma RSA (*Rivest-Shamir-Adleman*). Berikut adalah alur enkripsi dalam implementasi RSA menggunakan python :

## C. Database

Untuk menyimpan hasil *signature* untuk memvalidasi isi data dan memvalidasi pengguna, diperlukan *database* untuk menyimpan *signature* pengguna serta *cipher text* yang berisi *signature* data. Jenis data yang penulis gunakan untuk menyimpan hal-hal tersebut adalah .json file. Implementasi ini masih sederhana, dalam dunia nyata .json file jarang digunakan sebagai tempat penyimpanan data, tetapi dalam kasus simpel ini pengguna menggunakan .json file untuk memudahkan proses *debugging.* .json file akan berupa list of dictionary, dalam setiap dictionary akan terdiri atas “id”, “username”, “signature”, “data” sebagai keys nya. Value “id” berisi id dari dictionary, value “username” akan berisi username pengguna yang sudah di hash menggunakan algoritma SHA-256 (untuk menjaga keamanan dan menghemat data, ukuran string dalam python secara default 48 byte (jika string kosong), jauh lebih besar dibandingkan hasil hash SHA-256 yakni 32 byte[9]), value “siganature” adalah *signature* pengguna yang akan digunakan untuk validasi pengguna, value “data” adalah *cipher text* berserta *signature* data yang sudah disisipkan pada *cipher text* yang akan digunakan untuk memvalidasi integritas data

# III. Helpful Hints

## A. Figures and Tables

Large figures and tables may span both columns. Place figure captions below the figures; place table titles above the tables. If your figure has two parts, include the labels “(a)” and “(b)” as part of the artwork. Please verify that the figures and tables you mention in the text actually exist. Please do not include captions as part of the figures. Do not put captions in “text boxes” linked to the figures. Do not put borders around the outside of your figures. Use the abbreviation “Fig.” even at the beginning of a sentence. Do not abbreviate “Table.” Tables are numbered with Roman numerals.

Figure axis labels are often a source of confusion. Use words rather than symbols. As an example, write the quantity “Magnetization,” or “Magnetization *M*,” not just “*M*.” Put units in parentheses. Do not label axes only with units. As in Fig. 1, for example, write “Magnetization (A/m)” or “Magnetization (Am1),” not just “A/m.” Do not label axes with a ratio of quantities and units. For example, write “Temperature (K),” not “Temperature/K.”

Multipliers can be especially confusing. Write “Magnetization (kA/m)” or “Magnetization (103 A/m).” Do not write “Magnetization (A/m)  1000” because the reader would not know whether the top axis label in Fig. 1 meant 16000 A/m or 0.016 A/m. Figure labels should be legible, approximately 8 to 12 point type.

## B. References

Number citations consecutively in square brackets [1]. The sentence punctuation follows the brackets [2]. Multiple references [2], [3] are each numbered with separate brackets [1]–[3]. When citing a section in a book, please give the relevant page numbers [2]. In sentences, refer simply to the reference number, as in [3]. Do not use “Ref. [3]” or “reference [3]” except at the beginning of a sentence: “Reference [3] shows ... .”

Number footnotes separately in superscripts (Insert | Footnote)[[1]](#footnote-1). Place the actual footnote at the bottom of the column in which it is cited; do not put footnotes in the reference list (endnotes). Use letters for table footnotes.

Please note that the references at the end of this document are in the preferred referencing style. Give all authors’ names; do not use “*et al*.” unless there are six authors or more. Use a space after authors' initials. Papers that have not been published should be cited as “unpublished” [4]. Papers that have been submitted for publication should be cited as “submitted for publication” [5]. Papers that have been accepted for publication, but not yet specified for an issue should be cited as “to be published” [6]. Please give affiliations and addresses for private communications [7].

## C. Abbreviations and Acronyms

Define abbreviations and acronyms the first time they are used in the text, even after they have already been defined in the abstract. Abbreviations that incorporate periods should not have spaces: write “C.N.R.S.,” not “C. N. R. S.” Do not use abbreviations in the title unless they are unavoidable.

## D. Equations

Number equations consecutively with equation numbers in parentheses flush with the right margin, as in (1). First use the equation editor to create the equation. Then select the “Equation” markup style. Press the tab key and write the equation number in parentheses. To make your equations more compact, you may use the solidus ( / ), the exp function, or appropriate exponents. Use parentheses to avoid ambiguities in denominators. Punctuate equations when they are part of a sentence, as in

 (1)

Be sure that the symbols in your equation have been defined before the equation appears or immediately following. Italicize symbols (*T* might refer to temperature, but T is the unit tesla). Refer to “(1),” not “Eq. (1)” or “equation (1),” except at the beginning of a sentence: “Equation (1) is ... .”

## E. Other Recommendations

Use one space after periods and colons. Hyphenate complex modifiers: “zero-field-cooled magnetization.” Avoid dangling participles, such as, “Using (1), the potential was calculated.” [It is not clear who or what used (1).] Write instead, “The potential was calculated by using (1),” or “Using (1), we calculated the potential.”

Use a zero before decimal points: “0.25,” not “.25.” Use “cm3,” not “cc.” Indicate sample dimensions as “0.1 cm  0.2 cm,” not “0.1  0.2 cm2.” The abbreviation for “seconds” is “s,” not “sec.” Do not mix complete spellings and abbreviations of units: use “Wb/m2” or “webers per square meter,” not “webers/m2.” When expressing a range of values, write “7 to 9” or “7-9,” not “7~9.”

A parenthetical statement at the end of a sentence is punctuated outside of the closing parenthesis (like this). (A parenthetical sentence is punctuated within the parentheses.) In American English, periods and commas are within quotation marks, like “this period.” Other punctuation is “outside”! Avoid contractions; for example, write “do not” instead of “don’t.” The serial comma is preferred: “A, B, and C” instead of “A, B and C.”

If you wish, you may write in the first person singular or plural and use the active voice (“I observed that ...” or “We observed that ...” instead of “It was observed that ...”). Remember to check spelling. If your native language is not English, please get a native English-speaking colleague to proofread your paper.

# IV. Some Common Mistakes

The word “data” is plural, not singular. The subscript for the permeability of vacuum µ0 is zero, not a lowercase letter “o.” The term for residual magnetization is “remanence”; the adjective is “remanent”; do not write “remnance” or “remnant.” Use the word “micrometer” instead of “micron.” A graph within a graph is an “inset,” not an “insert.” The word “alternatively” is preferred to the word “alternately” (unless you really mean something that alternates). Use the word “whereas” instead of “while” (unless you are referring to simultaneous events). Do not use the word “essentially” to mean “approximately” or “effectively.” Do not use the word “issue” as a euphemism for “problem.” When compositions are not specified, separate chemical symbols by en-dashes; for example, “NiMn” indicates the intermetallic compound Ni0.5Mn0.5 whereas “Ni–Mn” indicates an alloy of some composition NixMn1-x.

Be aware of the different meanings of the homophones “affect” (usually a verb) and “effect” (usually a noun), “complement” and “compliment,” “discreet” and “discrete,” “principal” (e.g., “principal investigator”) and “principle” (e.g., “principle of measurement”). Do not confuse “imply” and “infer.”

Prefixes such as “non,” “sub,” “micro,” “multi,” and “"ultra” are not independent words; they should be joined to the words they modify, usually without a hyphen. There is no period after the “et” in the Latin abbreviation “*et al.*” (it is also italicized). The abbreviation “i.e.,” means “that is,” and the abbreviation “e.g.,” means “for example” (these abbreviations are not italicized).

# V. Conclusion

A conclusion section is not required. Although a conclusion may review the main points of the paper, do not replicate the abstract as the conclusion. A conclusion might elaborate on the importance of the work or suggest applications and extensions.

# VI. Appendix

Appendixes, if needed, appear before the acknowledgment.

# VII. Acknowledgment

The preferred spelling of the word “acknowledgment” in American English is without an “e” after the “g.” Use the singular heading even if you have many acknowledgments. Avoid expressions such as “One of us (S.B.A.) would like to thank ... .” Instead, write “F. A. Author thanks ... .” Sponsor and financial support acknowledgments are placed in the unnumbered footnote on the first page.

# References

1. Dale Liu, Caceres, M., Robichaux, T., Forte, D. v., Seagren, E. S., Ganger, D. L., Smith, B., Jayawickrama, W., Stokes, C., & Jan Kanclirz, Jr. (2009). Chapter 3 - An Introduction To Cryptography. In Next Generation SSH2 Implementation.
2. <https://www.shiksha.com/online-courses/articles/types-of-cryptography/> diakses pada 10 desember 2023
3. H Mendel, F., Pramstaller, N., Rechberger, C., Rijmen, V. (2006). Analysis of Step-Reduced SHA-256. In: Robshaw, M. (eds) Fast Software Encryption. FSE 2006. Lecture Notes in Computer Science, vol 4047. Springer, Berlin, Heidelberg. <https://doi.org/10.1007/11799313_9>
4. <https://informatika.stei.itb.ac.id/~rinaldi.munir//Matdis/2023-2024/14-Teori-Bilangan-Bagian1-2023.pdf> , diakses pada 10 Desember 2023
5. <https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/15-Teori-Bilangan-Bagian2-2023.pdf> , diakses pada 10 Desember 2023
6. <https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/16-Teori-Bilangan-Bagian3-2023.pdf> , diakses pada 10 Desember 2023
7. Ikbal, J. (2007). *An introduction to cryptography*. *Information Security Management Handbook, Sixth Edition*. <https://doi.org/10.2307/2695435>’
8. <https://www.encryptionconsulting.com/education-center/what-is-rsa/>
9. <https://stackoverflow.com/questions/53899931/why-does-an-empty-string-in-python-sometimes-take-up-49-bytes-and-sometimes-51> , diakses pada 10 Desember 2023

# 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, 3 Desember 2023

Ttd (scan atau foto ttd)

Nama dan NIM

1. It is recommended that footnotes be avoided (except for the unnumbered footnote with the receipt date on the first page). Instead, try to integrate the footnote information into the text. [↑](#footnote-ref-1)